DASyR(IR) - Document Analysis System for Systematic Reviews (in Information Retrieval)

Florina Piroi, Aldo Lipani, Mihai Lupu, and Allan Hanbury
Institute of Software Engineering and Information Systems
Vienna University of Technology
Favoriten Strasse 9-11/188, Vienna 1040, Austria
Email: (last name)@ifs.tuwien.ac.at

***Abstract*—Creating systematic reviews is a painstaking task undertaken especially in domains where experimental results are the primary method to knowledge creation. For the review authors, analysing documents to extract relevant data is a demanding activity. To support the creation of systematic reviews, we have created DASyR—a semi-automatic document analysis system. DASyR is our solution to annotating published papers for the purpose of ontology population. For domains where dictionaries are not existing or inadequate, DASyR relies on a semi-automatic annotation bootstrapping method based on positional Random Indexing, followed by traditional Machine Learning algorithms to extend the annotation set. We provide an example of the method application to a subdomain of Computer Science, the Information Retrieval evaluation domain. The reliance of this domain on large scale experimental studies makes it a perfect domain to test on. We show the utility of DASyR through experimental results for different parameter values for the bootstrap procedure, evaluated in terms of annotator agreement, error rate, precision and recall.**

I. Introduction

In knowledge communication, systematic reviews are an essential instrument, most prominent in the medical domain [1]. Here, for realising a systematic review thousands and even hundreds of thousands [2] of documents have to be analysed in order to create a comprehensive view about the state of the evidence on a specific topic of interest. Software tools to assist with this process have been developed already since the mid 1990s [3]. Such tools have, however, slow acceptance gain [4] and are mostly limited to assisting with the logistics of the work (like note taking and document management) [5]. Empirical evidence is increasingly larger in fields other than medical sciences, so the practice of systematic reviews becomes attractive in areas like software engineering [6] or computer science [7]. We are then confronted with the need to support a wider range of users and domains, for which, in the vast majority of cases, there is no commonly agreed upon vocabulary, even less so dictionaries or thesauri. In this context we have developed DASyR—a Document Analysis System for Systematic Reviews—and applied it to the study of information retrieval experimental results. Information Retrieval (IR) is essentially an empirical science, where experimentation and benchmarking is paramount to the successful adoption of a retrieval model or system. DASyR is the first such system that allows the user to dynamically create a domain dictionary (or populate a domain ontology) based on a latent semantic analysis of the corpus at hand, and DASyR(IR) is the first application on information retrieval documents.

A. Motivation

A number of campaigns for the evaluation of IR methods are running on a yearly cycle. These campaigns provide testing tools (queries, document collections and relevance judgements) to all researchers. The result is a rich body of knowledge, distributed across numerous scientific reports. For instance, the CLEF Association has published over 1,500 pages of evaluation reports from all CLEF conferences since 2000[1]. TREC[2] keeps a well maintained archive of evaluation tracks and participants' reports since its beginnings, consisting currently of over 1,420 papers. NTCIR[3], too, has about 2000 reports on its website. Adding to these the new evaluation campaigns, such as FIRE[4] or MediaEval[5], we can say that 1) there is considerable literature about the performance of IR systems in a wide variety of domains; and 2) there are very few, if any, comprehensive horizontal studies across the existing resources [8], [9]. The reason for such a limited number of horizontal studies is that putting this existing body of experience to work—in academic research or industrial practice—takes a significant effort and is generally done through the experience of various participants. A novice will need to quickly get up to date with a large amount of literature.

In general, it has been observed that when the information need is to be satisfied by pieces of information in various documents, as is the case for systematic reviews, current information access systems fail [10].

DASyR addresses this issue and, in this report, we show its instantiation for the IR domain.

B. Related Work

To automate document analysis, we must understand the data we have and annotate it accordingly. Setting on this path the following questions are the most immediate to answer:

1) what kind of and which entities should be annotated?
2) how can this be done efficiently and effectively?

To the best of our knowledge, the system most similar to DASyR was recently presented by Schuster and colleagues [11]. Like other document analysis systems [12], [13]

[1] http://www.clef-initiative.eu
[2] http://trec.nist.gov/
[3] http://research.nii.ac.jp/ntcir/
[4] http://www.isical.ac.in/~clia/
[5] http://www.multimediaeval.org/

978-1-4799-1805-8/15/$31.00 ©2015 IEEE

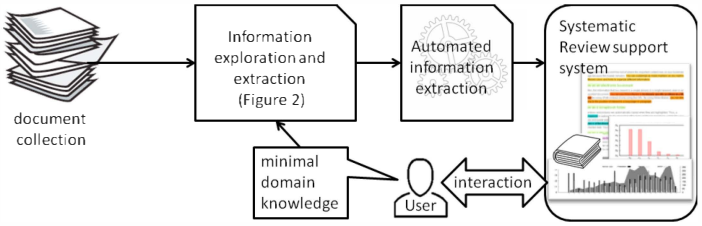

Fig. 1. DASyR system outline

however, it focuses on a specific kind of documents (i.e. invoices, receipts) and extracts a predefined set of metadata (e.g. document type, sender, received, date, amount, customer ID). The systematic reviews problem does not benefit from a predefined set of fields to extract, and, even more problematic, there is no dictionary agreed upon by the research community. We can make use of an ontology of concepts particular to a domain, but using such an ontology is only partly solving the problem. The more challenging part, which is not yet addressed and we do so here, is which entities should be annotated.

To scale up the annotation there is a substantial amount of previous work described in the Information Extraction (IE) literature. In particular, Named Entity Recognition (NER) is done following two main paradigms [14], both making use of machine learning techniques to extend the set of extracted terms. The first paradigm is based on the distributional properties of the concept terms (e.g. term frequency, tf-idf). Here Support Vector Machines (SVM) approaches are still efficient and effective: Lee et al. [15] compare two NER algorithm families, modified structural SVMs and Conditional Random Fields (CRF), finding that the former outperforms CRF and has reduced training times. The second paradigm is based on contextual information associated with the selected terms, as used by Gildea and Jurafsky for semantic roles annotation [16], or, more recently, by Yimam et al [17]. DASyR uses both of these paradigms into one coherent document analysis framework. The first paradigm is exemplified by an SVM classification where the manual annotations are the training set (Section II-B), while the second is concretised by the use of a semantic vector space and decision trees (Sections II-A, II-B).

In what follows, Section II describes our system architecture, Section III shows the results of applying the system on the IR data collection, and Section IV summarises, providing an outlook for the system.

II. System Description

DASyR has two parts (Figure 1): an information exploration and extraction component with which the user interacts (Section II-A, Figure 2), and an automated information extraction component (Section II-B). Similarly to the Intellix system mentioned above [11], DASyR relies on existing tools to perform OCR (step 1 in Figure 2). The output of this step is stored as XML data and in what follows we shall refer to this XML set as the "data collection" or the "data set".

A. Bootstrapped manual annotations

In any information exploration and extraction system the main design challenge is giving users the possibility to annotate as many term instances as possible in a short time.

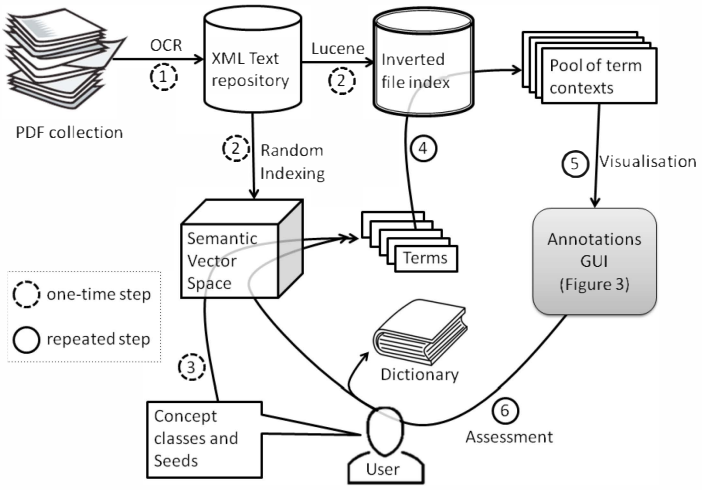

Fig. 2. Annotation system workflow

Therefore, before a user can interact with the system, the data collection is used in step 2 of Figure 2 to create:

1) a semantic vector space, based on positional Random Indexing (pRI) [18]. The pRI engine allows to identify terms related to a given term according to their semantic similarity (e.g. different types of diseases: given 'flu' identify 'cold').
2) an inverted file search engine, based on Lucene [19], to provide the user with a surrounding textual context for the automatically identified related terms.

Asking users to freely annotate documents is problematic because of the huge time requirements imposed by an open-end annotation procedure on unstructured documents. Instead, a closed-set approach (one where a user already sees the text annotated and only approves or rejects annotations) is significantly faster. However, the closed-set approach requires an existing dictionary and is limited to that dictionary.

To overcome the lack of domain dictionaries, DASyR requires some minimal input from the user. This is a small set of examples of the concepts to be extracted from the data collection. In DASyR we refer to these examples as *seeds*. The semantic vector space created by the pRI component is used to create a dictionary that is continuously updated based on the ongoing use of the system.

For the terms annotated by the user in a class, in addition to storing them in a domain specific dictionary (step 6 in Figure 2, the semantic vector space engine returns a number, N, of new terms. These are candidates to be further representatives of the class (step 3 in Figure 2). All instances of these terms are then identified in the indexed dataset by the search engine which, in turn, sends its output to a pool of terms and their contexts (step 4 in Figure 2). The GUI will present to the user terms out of this pool, one at a time, together with their context (step 5), for the user to annotate (step 6). The granularity of the term context presented to the user is at phrase level, where phrases are identified using a simple sentence splitter. The term preprocessing is conservative, using only the S-Stemmer [20], in order to maintain the flexibility of the system as a whole.

In addition to the classes of interest initially selected by the user the system offers the annotators two further special

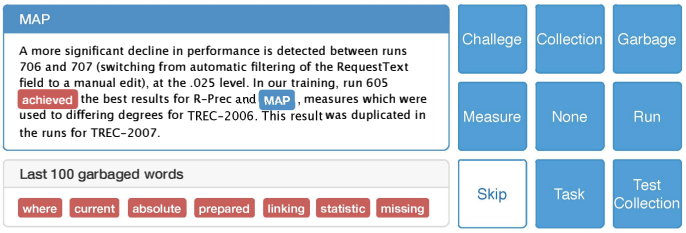

Fig. 3. Annotation system graphical user interface

concept classes: `None` and `Garbage`. The first one is used when none of the user given concept classes can be chosen to annotate a presented instance. A term annotated with the `Garbage` class is removed from the annotation pipeline. `Garbage` is used to speedup the annotation process by exploiting the annotator's expertise. That is, when the annotator is confident that the presented term can never be an instance of any of the initial concept classes, he or she 'garbages' the term. If new term instances lead to reconsidering a garbaged term, annotators do have the possibility to change the annotation to another concept class. This can be done in two ways: extemporaneously, via the list of the last 100 garbaged tokens presented in the interface (Figure 3), or contemporaneously, via annotation highlights of all the garbaged words in the shown context of a target term. This latter visualisation feature helps the user realise whether a garbaged term is actually not `Garbage` in that particular context. To further speed up the annotation process, all words that have been marked as `Garbage` by at least two users are not shown to the other users anymore, thus improving the user experience.

B. Automatic annotation

Starting from the manually created annotations, DASyR expands to the entire set of possible entities using Machine Learning (ML). As discussed in Section I-B, the ML task is approached from two angles: 1) classifying each token in a specific class following the first paradigm mentioned by Maynard [14], using SVM; and 2) for each term previously annotated as something other than `None` or `Garbage` by one of the annotators, classifying all of its occurrences as either `None` or as an ontology class, using the contextual paradigm, implemented by decision trees.

To generate training features, DASyR relies on a popular text engineering toolkit, GATE [21], to extract the token form (i.e. the term itself), type (e.g. number, lowercased, initialUpperCased), and part-of-speech tag, for a window of 10 tokens around each token.

For the first approach (classification of all tokens) DASyR uses a radial basis function kernel SVM (RBF-SVM) [22] for which it first performs a parameter scan to identify the γ parameter of the RBF, as well as the tradeoff between training error and margin, c, and the cost factor, j, by which training errors on positive examples outweigh errors on negative examples. To speed up the parameter scan, it uses the $\xi alpha$ estimates generated by the *SVMLight*[6] implementation. Linear and polynomial kernels were also explored, as well as the Perceptron with Uneven Margins method, with worse results. When multiple classes are to be detected, as it would generally be the case, the multi-classification problem is split into binary classification tasks using a one-vs-all approach.

[6] http://svmlight.joachims.org/

TABLE I. CLASSES IN THE CURRENT INSTANTIATION OF DASYR

Collection	a set of documents
Test Collection	a `Collection` enriched with queries and relevance judgements
Task	a set of queries modelling a specific information seeking task
Challenge	a set of related tasks
Run	the answer of a system to a `Task`
Measure	a function to give a `run` a numeric value indicative of quality

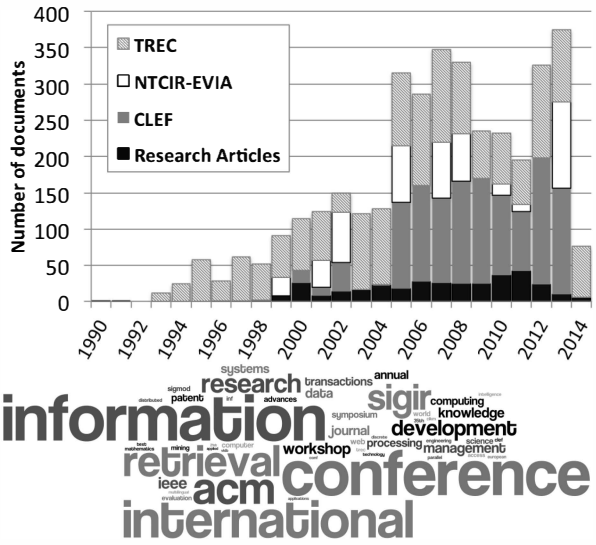

Fig. 4. Dataset size and publication venues of the *Research Articles* sub collection

For the second approach (classification of specific terms) DASyR uses the J48 [23] implementation of the decision tree classifier C4.5, which has the benefit of giving insights on the data by legible trees, whence it is possible to produce rules.

III. EVALUATION AND USE OF ANNOTATIONS

To demonstrate DASyR, we instantiated it for a horizontal survey of evaluation metrics used in information retrieval research reports. We refer to it as DASyR(IR). We created a dataset based on the TREC, CLEF and NTCIR/EVIA reports available on their respective websites. We also added a set of 687 research articles in the IR field from a private set (i.e. without crawling publishers' websites). Figure 4 shows the document distribution across years and the most common terms in the publication venues of the research articles. The dataset uses approximately 17GBs hard disk space.

The set of concept classes in the annotation process were chosen from the IR ontology proposed by Lipani et al. [24]. For this pilot evaluation we restricted the set to six classes (Table I), all part of the Evaluation category in the ontology. We observe, next, the performance of the information exploration and extraction task as a whole as well as that of the individual components. We analyse first how the pRI bootstrapping method behaves depending on the choice of N, the number of terms generated based on each newly classified term. We then look at annotation agreements, final annotation results, and show some statistics enabled by the use of DASyR(IR).

The annotation was performed by five users, who generated 29 585 annotations. The average annotation time was 2.71s per

term. Considering the computer latency, which was of 1.3s, this resulted in around 35 person-hours of effort. Using these annotations and based on the existing pool of documents, the GATE component generated 9 008 455 features to be used by the automated information extraction engine.

A. Random Indexing bootstraping

The question we need to answer here is whether pRI should generate fewer or more similar terms every time a new class instance is annotated. We can think of this as a depth-first (few terms) vs. breadth-first (many terms) search. An *a posteriori* analysis, where the *valid* terms are those annotated as instances of a class by a user, shows that for small candidates sets, ($2 \leq N \leq 5$), pRI stops generating new terms very soon, and would require a restart with new, externally introduced seeds. At the same time, the more candidates we introduce at each step (larger N), the lower the proportion of valid terms is. This behaviour is illustrated in figures 5 and 6. Figure 5 shows the cumulative number of valid suggestions for $1 \leq N \leq 20$, with highlighted lines for N= 2, 5, 10, and 20. The x-axis has terms filed to the pRI, in chronological order of their annotation. The y-axis shows the cumulative sum of new valid terms found by pRI. Figure 6 shows the proportion of valid to garbaged terms, for each N.

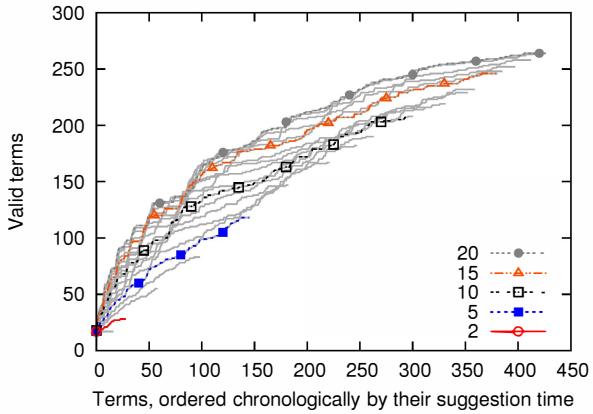

Fig. 5. Good suggestions made by the semantic vector space engine

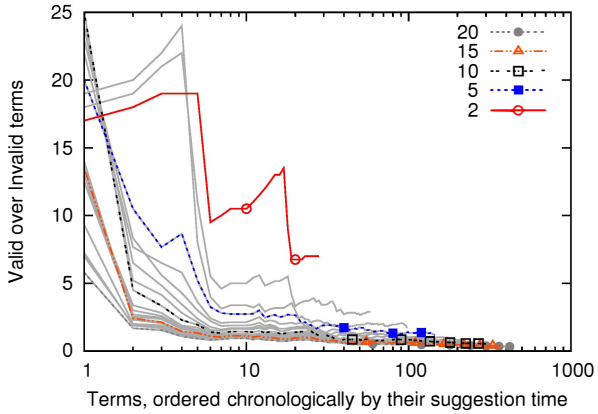

Fig. 6. Proportion of good suggestions

B. Annotation agreement

For each term highlighted by the system, the annotators select from a subset of classes of the IR Ontology[7]. To test the quality of the analysis, we observe how often two users agree on the terms provided by DASyR. Table II shows their agreement scores. The two sets of numbers presented are for normal and *Lenient* agreement. The leniency comes from the observation that annotators often disagreed when two concepts were very similar to each other. For instance, between `TestCollection` and `Collection` or between `Challenge` and `Task`. The *Lenient* agreement ignores disagreements between these pairs. Column **general** shows the general agreement, between all annotations, including `None`. Column **no None** removes from consideration the `None` annotations, in order to observe agreement or disagreement between the classes of interest. It is in this case that we observe both the highest and the lowest disagreement, indicating that it is difficult, even for the users, to clearly distinguish between similar classes.

[7] http://ifs.tuwien.ac.at/~admire/ir_ontology/ir

TABLE II. STRICT/LENIENT MANUAL ANNOTATION AGREEMENT

	general	no None
agreed	1584/1661	266/343
disagreed	269/192	82/5
agreement	85%/90%	76%/99%

C. Machine learning

For the purposes of this evaluation, we focus on the classification for the `Measure` class. There are 29 585 training vectors and ∼27 mil. candidate tokens. Leave-one-out cross-validation of the radial basis function kernel SVM estimated 2.5% error rate, 77.8% recall,and 86.4% precision

As mentioned in Section II-B, the user also has the option to gain insight into the ML results by observing the output of the decision tree classifier. As an example, we present the result of the classifier for the term 'Recall'. There are 208 training vectors and 4 500 candidate tokens. Leave-one-out validation estimated 7.1% mean absolute error, 94.2% recall, and 93.1% precision. The generated rule, predicting whether an instance of the term 'Recall' is a `Measure` is:

R1 $IN(1) \wedge \neg that(1) \wedge \neg the(1) \rightarrow Measure$
R2 $IN(1) \wedge that(1) \rightarrow None$

This essentially says that *Recall* is a `Measure` if it is not followed by the term *that*.

D. Use for systematic reviews

Finally, Figure 7 shows the proportion of documents in each year mentioning one of five popular IR evaluation metrics, plotted against the background of the number of documents in the year. Precision and Recall are obviously frequent, while Mean Average Precision (MAP) is predominantly used since 2000. On viewing this data, we might conjecture that the very low mentions of the normalized Discounted Cummulated Gain (NDCG) metric is related to the low number of test collections with graded relevance judgements (required when using the NDCG metric), and decide to look more closely to the evaluation campaigns that used these test collections and/or the NDCG measure.

IV. CONCLUSIONS AND FUTURE WORK

This paper has shown the first steps towards populating an IR ontology to support the kind of horizontal studies we

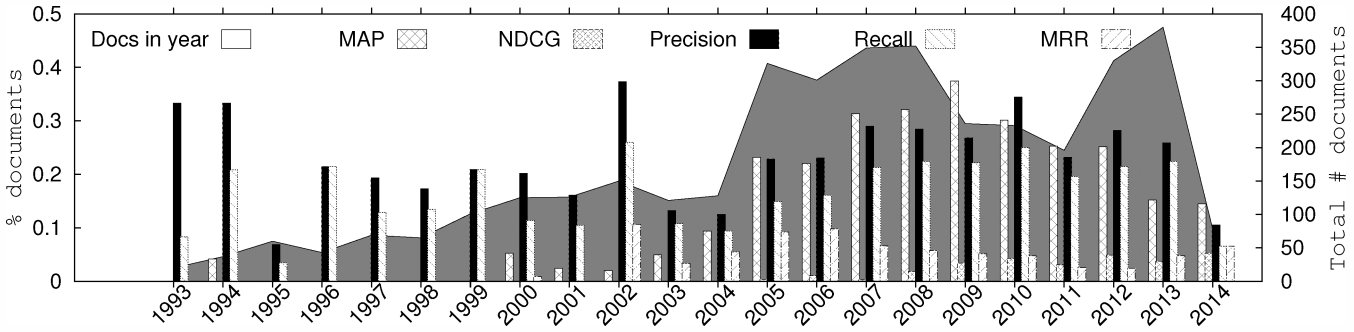

Fig. 7. Distribution of five metric mentions over years

now see in the medical domain. Having experimented, first, with open-end annotation we observed that annotators consider it too time-consuming, especially in the absence of domain-specific dictionaries. We proposed, therefore, a bootstrapping method using positional Random Indexing to introduce annotation candidates. Five experts added approximately 30 000 annotations at a speed of 4s/annotation on average. Two ML approaches were then used to scale from 30k annotations to 27 million tokens. Future work consists primarily in a better pruning of the search space and the introduction of a domain-specific chunking step in the NLP pipeline. Initial steps in this direction, using generic chunkers [25], have shown relatively poor performance. Finally, we showed the kind of analysis that can now be done on IR research with the potential to provide very useful insights. The simplicity of the user interface is an important factor in reducing the annotation time per term. The expert annotators also reported a certain addictive feeling in using the system to annotate terms. In the next iteration there will be a web interface for our peers to make their own observations on practices and trends in IR.

Acknowledgements

This research was supported by the Austrian Science Fund (FWF) project number P25905-N23 (ADmIRE) and project number I 1094-N23 (MUCKE, under the CHIST-ERA Program for Transnational Research Projects).

References

[1] D. Gaugh, S. Oliver, and J. Thomas, "An Introduction to Systematic Reviews," *Sage*, 2012.

[2] I. Shemilt, A. Simon, G. Hollands, T. Marteau, D. Ogilvie, and A. O'Mara-Eves, "Pinpointing needles in giant haystacks: use of text mining to reduce impractical screening workload in extremely large scoping reviews," *Res. Synth. Methods.*, vol. 13, no. 1218, 2013.

[3] W. R. Hersh and D. Hickam, "How Well Do Physicians Use Electronic Information Retrieval Systems? A framework for Investigation and Systematic Review," *JAMA: The Journal of the American Medical Association*, vol. 280, 1998.

[4] J. Thomas, "Diffusion of information in systematic review methododology. Why is study selection not yet assisted by automation?" *OA Evid. Based Med.*, vol. 1, no. 2, 2013.

[5] J. Thomas, J. Brunton, and S. Gaziosi, "EPPI-Reviewer 4.0: Software for Research Synthesis," *London: EPPI-Centre Software, Social Science Research Unit, Institute of Education*, 2010.

[6] S. Biffl, M. Kalinowski, F. Ekaputra, E. S. Asensio, and D. Winkler, "Building Empirical Software Engineering Bodies of Knowledge with Systematic Knowledge Engineering," in *Proc. of SEKE*, 2014.

[7] A. O'Mara-Eves, J. Thomas, J. McNaught, M. Miwa, and S. Ananiadou, "Using text mining for study identification in systematic reviews: a systematic review of current approaches," *Systematic Reviews*, vol. 4, no. 5, 2015.

[8] N. Ferro and G. Silvello, "CLEF 15th Birthday: What can we Learn From Ad Hoc Retrieval?" in *Proc. of CLEF*, 2014.

[9] T. Tsikrika, A. García Seco de Herrera, and H. Müller, "Assessing the Scholarly Impact of ImageCLEF," in *Proc. CLEF Conference*, 2011.

[10] H. de Ribaupierre and G. Falquet, "A User-centric Model to Semantically Annotate and Retrieve Scientific Documents," in *Proc. of ESAIR*, 2013.

[11] D. Schuster, K. Muthmann, M. Berger, C. Weidling, K. Aliyev, and A. Hofmeier, "Intellix - End-User Trained Information Extraction for Document Archiving," in *Proc. of ICDAR*, 2013.

[12] B. Janssen, E. Saund, E. Bier, P. Wall, and M. Sprague, "Receipts2go: the big world of small documents," in *Proc. of DocEng*, 2012.

[13] E. Medvet, A. Bartoli, and G. Davanzo, "A probabilistic approach to printed document understanding," *Int. J. Doc. Anal. Recognit.*, vol. 13, no. 4, 2011.

[14] D. Maynard, Y. Li, and W. Peters, "NLP Techniques for Term Extraction and Ontology Population," in *Proc. of Conf. on Ontology Learning & Population*, 2008.

[15] C. Lee, P.-M. Ryu, and H. Kim, "Named Entity Recognition Using a Modified Pegasos Algorithm," in *Proc. of CIKM*, 2011.

[16] D. Gildea and D. Jurafsky, "Automatic Labeling of Semantic Roles," *Comput. Linguist.*, vol. 28, no. 3, 2002.

[17] S. Yimam, R. de Castilho, I. Gurevych, and C. Biemann, "Automatic Annotation Suggestions and Custom Annotation Layers in WebAnno," in *Proc. of ACL*, 2014.

[18] T. Cohen, R. Schvaneveldt, and D. Widdows, "Reflective random indexing and indirect inference: A scalable method for discovery of implicit connections," *Journal of Biomedical Informatics*, vol. 43, no. 2, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1532046409001208

[19] [Online]. Available: http://lucene.apache.org/

[20] D. Harman, "How effective is suffixing?" *JASIS*, vol. 42, 1991.

[21] H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, N. Aswani, I. Roberts, G. Gorrell, A. Funk, A. Roberts, D. Damljanovic, T. Heitz, M. A. Greenwood, H. Saggion, J. Petrak, Y. Li, and W. Peters, *Text Processing with GATE (Version 6)*, 2011. [Online]. Available: http://tinyurl.com/gatebook

[22] T. Joachims, *Learning to Classify Text using Support Vector Machines*. Kluwer Academic Publishers, 2002. [Online]. Available: http://www.cs.cornell.edu/People/tj/svmtcatbook/

[23] R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[24] A. Lipani, F. Piroi, L. Andersson, and A. Hanbury, "An Information Retrieval Ontology for Information Retrieval Nanopublications," in *Proc. of CLEF*, 2014.

[25] L. Ramshaw and M. Marcus, "Text Chunking Using Transformation-Based Learning," in *Natural Language Processing Using Very Large Corpora*. Springer Netherlands, 1999, vol. 11.