Methodology for evaluating citation parsing and matching

Mateusz Fedoryszak, Łukasz Bolikowski, Dominika Tkaczyk, and Krzyś Wojciechowski

Interdisciplinary Centre for Mathematical and Computational Modelling, Warsaw University
{m.fedoryszak, l.bolikowski, d.tkaczyk, k.wojciechowski}@icm.edu.pl

Abstract. Bibliographic references between scholarly publications contain valuable information for researchers and developers involved with digital repositories. They are indicators of topical similarity between linked texts, impact of the referenced document, and improve navigation in user interfaces of digital libraries. Consequently, several approaches to extraction, parsing and resolving said references have been proposed to date. In this paper we develop a methodology for evaluating parsing and matching algorithms and choosing the most appropriate one for a document collection at hand. We apply the methodology for evaluating reference parsing and matching module of the YADDA2 software platform.

Keywords: citation parsing, citation matching, evaluation, test set, YADDA2 software platform

1 Introduction and related work

This paper discusses methods of evaluating algorithms for matching scholarly citations. Citation matching attempts to cluster bibliographic references to the same document, and possibly link them with the referenced document (provided that it is present in a collection). This can be seen as an instance of a broader problem of record linkage in databases [5, 4]. Citation matching is a fundamental step in creating a digital library of scholarly publications. Links between documents conveying the fact that document A references document B (represented e.g. by the `references` term in the Dublin Core standard) are needed for a number of reasons:

- more user-friendly interfaces – similarly to hypertext links, citation links allow a user to navigate between documents [11].
- scientometrics – number of citations received by documents is an established measure of impact of individual researchers, journals, institutes and counties. For example, Impact Factor [7] and Hirsch Index [10] depend critically on availability of a citation graph.
- link-based classification – bibliographic references provide an excellent context for classifying documents or determining author identities [3, 14].

Historically, citation matching was performed manually [6]. Hitchcock et al. [11] demonstrated a proof-of-concept system that performed autonomous linking within Cognitive Science Open Journal. Citeseer was one of the first large-scale systems [8] that autonomously indexed scholarly citations. Pasula et al. [16] proposed a probabilistic model for citation matching.

Citation matching is sometimes divided into two phases: segmentation and entity resolution. Segmentation, or citation parsing, aims to deconstruct a bibliographic reference into functional pieces such as author names, title, year of publication, etc. Entity resolution clusters records representing the same document. However, several authors depart from such a division, most notably Wellner et al. [20], Poon and Domingos [17], Liao and Zhang [13], or Goutorbe [9].

Introduction of autonomous citation matching solutions created a need for their evaluation. Lawrence et al. [12] created a training set by hand and evaluated performance of their citation matching approaches by calculating the percentage of fully correct groups of citations. Their set is often referred to as "CiteSeer set." McCallum et al. [15] demonstrated how to employ machine learning techniques in automating construction of digital libraries. They created a number of training data sets[1], including one for evaluating citation matching. This set is often referred to as "Cora set." Most other authors train and evaluate their algorithms on one or both of these sets.

Our paper proposes a different take on evaluating performance of citation matching algorithms. In particular, we propose a method of *autonomous generation* of training and test sets and a *wider range of metrics* for evaluation of citation matching solutions.

2 Methodology

In this section we shall present a method of evaluating a citation matcher. We shall demonstrate how we create a test set and then propose a set of metrics used to measure matcher correctness.

We define citation matching problem in a slightly different way than usual: we assume we have a set of citations and a set of documents' metadata in a database. We want to assign to each citation a database record or information that the store does not contain the appropriate document.

2.1 Test set preparation

To generate the test set we have used the metadata of 1400 randomly selected publications from the Spanish Digital Mathematics Library (DML-E) aggregated in the European Digital Mathematics Library (EuDML, [18]). The library contains mathematical publications written in either English or Spanish, of which the oldest date back to late 50's of the previous century. The metadata was available as XML files in NLM format, each describing one publication.

[1] See: `http://people.cs.umass.edu/~mccallum/data.html`

We have divided our document set into 3 subsets containing 1000, 200 and 200 documents respectively. We have used metadata of documents in 1st and 2nd subset to generate citations and put those from subsets 1st and 3rd into the database (see Fig. 1).

We wanted to generate many citation strings using various popular bibliographical styles. The easiest way to achieve that was to use BibTeX. We only needed to create a database file with the metadata of all the documents we wanted to create citations for and supply bibliographical styles [2]. BibTeX generated files containing LaTeX bibliography. From these files we extracted citation strings and converted them to plain text by removing all the LaTeXcommands. Each style constitutes one test set, metrics described in the following section are defined per a set.

Finally, we asked the matcher under evaluation to match citations to the database records.

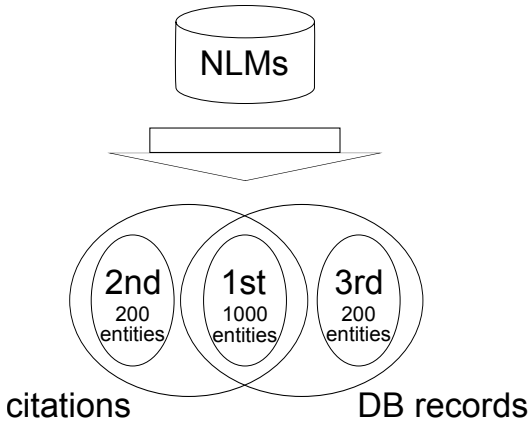

Fig. 1. Test set generation

2.2 Metrics

We have designed a set of metrics which we used in our experiments. They can be divided into several groups according to the aspect of the matching they deal with. In the following paragraphs we shall present these groups and define the metrics they contain.

Grouping correctness metric As we have mentioned, citation matching task is often defined as finding among the set of citations those that refer to the same

[2] We have used `abbrv`, `acm`, `alpha`, `apalike`, `ieeetr`, `jpc`, `pccp`, `plain`, `ppcf` and `revcompchem`

paper. Such a formulation of the problem appears, among others, in the classic paper by Lawrence et al. [12]. Let there S be the set of correct citation groups and R the set of groups returned by a matcher. *Grouping correctness* is then defined as

$$G = \frac{|S \cap R|}{|S|}$$

To make our results comparable with this approach, we shall define a similar metric in our evaluation framework. We can assume we have two types of group elements: citations and database records.

Reference group set S, representing the correct clustering that we hope to reconstruct, will consist of (see Fig. 2):

- 1000 2-element groups each containing a citation and an appropriate record,
- 200 1-element groups each containing a citation only,
- 200 1-element groups each containing a database record only.

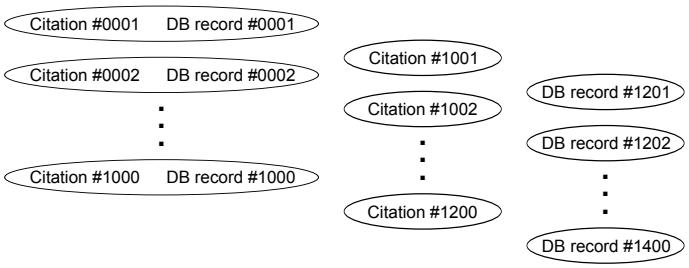

Fig. 2. Groups in the test set

As for result set, it will be defined as follows. Let there P be the set of (citation, database record) pairs returned by a matcher. Two elements x and y are in the same group if $(x, y) \in P \vee (y, x) \in P$. Result set R is a set of such groups.

Now we can introduce grouping correctness metric into our framework in an analogous way.

References-based metrics We can also look at matching task as finding (citation, database record) pairs. Let there C be the set of such pairs that exist in test set and P be the answers of an algorithm such that their database records are not empty. We can now define the following metrics:

- reference precision $P_r = \frac{|C \cap P|}{|P|}$
- reference recall $R_r = \frac{|C \cap P|}{|C|}$
- reference F-measure $F_r = 2\frac{P_r P_r}{P_r + P_r}$

Nonexistent record metrics There is a number of citations that do not have a corresponding record in the database. The metrics defined in this section are to cover them. M is the set of citations that do not have a database record and N is the set of citations for which an algorithm did not match any record. We define:

- nonexistent precision $P_\perp = \frac{|M \cap N|}{|N|}$
- nonexistent recall $R_\perp = \frac{|M \cap N|}{|M|}$
- nonexistent F-measure $F_\perp = 2\frac{P_\perp P_\perp}{P_\perp + P_\perp}$

Miscellaneous metrics Let E be the set of all correct (citation, database record) pairs where citation is not empty and P the set of pairs returned by a matcher We define *accuracy* as

$$Acc = \frac{|E \cap P|}{|E|}$$

3 Evaluated bibliographic reference matcher

In order to demonstrate the methodology described in Section 2, we will apply it to evaluate our in-house citation matcher implemented in the YADDA2 platform. In this section we shall briefly describe our matcher, while the next section (Section 4) shall be devoted to presentation of its results.

Bibliographic reference matcher often has to deal with a collection of documents containing bibliographic references. References can exist in different forms, from raw text strings to hierarchical structures with tagged metadata information. Our requirements for bibliographic reference matcher include:

- identifying all pairs *reference — referenced document* in the collection,
- finding all documents referenced by a new document added to the collection,
- finding all documents referencing a new document added to the collection.

The whole implementation of bibliographic references matcher we have evaluated consists of three parts:

- bibliographic references parser used to extract valuable metadata information from bibliographic reference strings,
- metadata store that indexes and allows to search the metadata information of both the documents and the references from the collection,
- the bibliographic reference matcher being able to identify documents referenced by a given document and documents referencing a given document, based on comparing various metadata information of documents and references.

Figure 3 shows the process of matching references in the collection. First all reference strings are parsed and reference metadata is extracted. In the second step the metadata of both the documents and the references included in the collection is added to the metadata store and indexed. After this the matcher is ready to match references with documents by comparing various metadata fragments found in the metadata store.

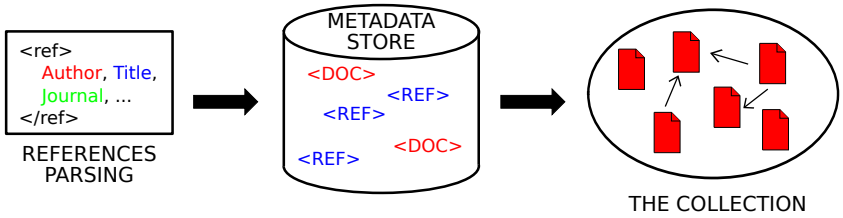

Fig. 3. The process of references matching

3.1 Bibliographic reference parser

One should not assume that the references in the collection are in the form of parsed structures with tagged metadata information. In many cases the matcher has to deal with references in the form of raw text strings. As a result the matcher requires a method for extracting metadata information from reference strings.

The goal of parsing the bibliographic reference strings is to identify fragments of the strings containing meaningful pieces of metadata information. The information our parser extracts include: *author*, *title*, *journal*, *volume*, *issue*, *pages*, *publisher*, *location* and *year*. Extracted metadata information fragments are indexed in the next step, which allows us to match them with the metadata of the documents in the collection.

The implementation of bibliographic reference parser is based on a Hidden Markov Model. First the reference string is tokenized into substrings containing only letters and digits or a single character of another type. In our model HMM sequence is composed of reference's tokens, labels of tokens are treated as unknown states and vectors of features computed for every token are visible observations. The Viterbi algorithm is used to determine the most probable sequence of token labels based on initial, transition and emission probability obtained from a training set. More details of the parser implementation can be found in [19].

The citation sample we have used to train the parser contained 100 references, each in 10 different bibliographic styles (described by BibTeX styles `abbrv`, `acm`, `alpha`, `apalike`, `ieeetr`, `jpc`, `pccp`, `plain`, `ppcf` and `revcompchem`), 1,000 references in total. The citations were generated in a similar way we did it in test set building. The metadata information extracted from the bibliographic references has been used in further matching steps.

3.2 Metadata store

The metadata store indexes the metadata information of both the documents from the collection and bibliographic references contained by them. The reference matcher uses the metadata store to search for documents and references based on various metadata information.

The implementation of the metadata store can be based on any software able to index and search data. Our first implementation used Apache Solr search platform [1], for reference matching evaluation we used PostgreSQL database [2].

3.3 Bibliographic reference matching

Bibliographic reference matcher is based on comparing various metadata information of documents and references. The matcher allows to:

- find the document referenced by a given bibliographic reference,
- identify all the documents referencing a given document, that is the documents containing references that reference a given document.

In both cases the matching process consists of several matching steps executed in a certain order. The result of each step is a set of matched objects. If the matcher tries to find the document referenced by a given bibliographic reference, the first matched object is returned and the whole process exits. If the matcher attempts to identify all the references referencing a given document, all steps are executed and the results are combined into one set of matched objects.

Each matching step consists of two phases:

1. selecting candidates from the metadata store according to a specific criterion,
2. evaluating the candidates by comparing corresponding metadata information.

In our evaluation process we have used two matching steps.

During the first step the candidates have been selected based on the following metadata information: authors' surnames, year of publication and hash of journal name. Then the candidates have been evaluated by comparing: authors' full name, journal name, volume, issue and year of publication.

During the second step the candidates have been selected based on only authors' surnames and year of publication. Then the candidates have been evaluated by comparing: authors' full name, journal name, volume, issue, year of publication and title.

Comparing various metadata information is not a trivial task due to different formats, abbreviations, typos, etc. In our implementation we use different methods of comparing for different metadata information. For example author full name, volume, issue and year of publication are considered equal if the corresponding strings are identical. Journal names are considered equal if one string is a subsequence of the other. In the case of title we make use of Levenshtein distance: two titles are equal if one of them is a subsequence of the other or if both are long and Levenshtein distance between them is less than a small fixed number.

4 Results

We have followed described evaluation path for our in-house citation matcher. Numerical values achieved are presented in Table 1.

Table 1. Matcher evaluation. Each column shows numerical values of metrics defined in Section 2.2. Each row represents a single test set generated using one bibliographic style.

Style	*Acc*	P_r	R_r	F_r	$P_\perp$	$R_\perp$	$F_\perp$	G
alpha	0.70	0.98	0.64	0.78	0.36	0.98	0.52	0.74
abbrv	0.87	0.98	0.85	0.91	0.57	0.98	0.72	0.88
ieeetr	0.87	0.99	0.85	0.91	0.57	0.98	0.72	0.89
plain	0.41	0.99	0.29	0.45	0.22	0.99	0.36	0.49
apalike	0.71	1.00	0.65	0.79	0.36	1.00	0.53	0.75
acm	0.87	0.99	0.84	0.91	0.57	0.98	0.72	0.88
jpc	0.94	0.99	0.93	0.96	0.77	0.98	0.86	0.95
pccp	0.94	0.99	0.93	0.96	0.76	0.98	0.85	0.95
ppcf	0.71	1.00	0.66	0.79	0.37	1.00	0.54	0.76
revcompchem	0.93	0.98	0.92	0.95	0.73	0.98	0.83	0.94
Average	0.79	0.99	0.75	0.84	0.53	0.98	0.67	0.82

Accuracy, being the most basic metric, can be treated as a single-valued benchmark of the overall performance. It tells us that in general our matcher does fairly well. From references-based metrics we conclude that if our algorithm matches a citation to the database record it almost always does that correctly, but there are many more entities that should have been linked (i.e. high precision P_r, relatively low recall R_r). Similar information can be drawn from nonexistent record metrics: almost all citations that have no database record are correctly classified as such (high recall $R_\perp$), but some citations identified as not having a corresponding database record in fact do have one (low precision $P_\perp$).

5 Conclusions

We have presented a methodology for evaluating a reference matcher. We have shown how a test set can be automatically built using existing publication metadata and BibTeX, removing the need for a laborious construction by hand. We have also proposed some metrics which should be used to generate numerical values reflecting algorithm performance. Finally, we have described and evaluated our in-house matcher using presented methodology.

However, one can point out some weaknesses of proposed test set creation method: generated citations are very consistent in terms of formatting and contain no punctuation errors. Moreover, we use the same metadata for citation generation and matching. That means that we do not deal with some matching issues, e.g. different ways of abbreviating journal names. This issues can be at least partially addressed by introducing some arbitrary errors. To simulate them we could substitute a number of random characters in a citation string for different ones.

Nevertheless this small flaw should not make us forget about obvious advantages of a proposed method, among them the huge scalability as we are able to create unlimited number of citations using arbitrarily many bibliographic styles.

6 Acknowledgements

This work is supported by the National Centre for Research and Development (NCBiR) under Grant No. SP/I/1/77065/10 by the Strategic scientific research and experimental development program: "Interdisciplinary System for Interactive Scientific and Scientific-Technical Information."

References

1. Apache Solr, http://lucene.apache.org/solr/
2. PostgreSQL, http://www.postgresql.org/
3. Bolelli, L., Ertekin, S., Giles, C.L.: LNAI 4213 - Clustering Scientific Literature Using Sparse Citation Graph Analysis. Information Sciences pp. 30–41 (2006)
4. Christen, P.: A survey of indexing techniques for scalable record linkage and deduplication. IEEE Transactions on Knowledge and Data Engineering (2011)
5. Elmagarmid, A., Ipeirotis, P., Verykios, V.: Duplicate Record Detection: A Survey. IEEE Transactions on Knowledge and Data Engineering 19(1), 1–16 (Jan 2007)
6. Garfield, E.: Citation Indexing: Its Theory and Application in Science, Technology, and Humanities. John Wiley & Sons, New York (1979)
7. Garfield, E.: The history and meaning of the journal impact factor. Journal of the American Medical Association 295(1), 90–93 (2006)
8. Giles, C., Bollacker, K., Lawrence, S.: CiteSeer: An automatic citation indexing system. In: Proceedings of the third ACM conference on Digital libraries. pp. 89–98. ACM (1998)
9. Goutorbe, C.: Document Interlinking in a Digital Math Library. In: Towards a Digital Mathematics Library. pp. 85–94 (2009)
10. Hirsch, J.E.: An index to quantify an individual's scientific research output. Proceedings of the National Academy of Sciences of the United States of America 102(46) (2005)
11. Hitchcock, S.M., Carr, L.A., Harris, S.W., Hey, J.M.N., Hall, W.: Citation Linking: Improving Access to Online Journals. Proceedings of Digital Libraries 97 pp. 115–122 (1997)
12. Lawrence, S., Giles, C.L., Bollacker, K.D.: Autonomous citation matching. In: Etzioni, O., Müller, J.P., Bradshaw, J.M. (eds.) Proceedings of the third annual conference on Autonomous Agents AGENTS 99. vol. 1, pp. 392–393. ACM, ACM Press (1999)
13. Liao, Z., Zhang, Z.: A Generalized Joint Inference Approach for. In: Lecture Notes in Artificial Intelligence 5360, pp. 601–607 (2008)
14. Macskassy, S.A., Provost, F.: Classification in Networked Data : A Toolkit and a Univariate Case Study. Journal of Machine Learning Research 8, 935–983 (2007)
15. McCallum, A., Nigam, K., Rennie, J.: Automating the construction of internet portals with machine learning. Information Retrieval pp. 127–163 (2000)
16. Pasula, H., Marthi, B., Milch, B., Russell, S., Shpitser, I.: Identity uncertainty and citation matching. In: Proceedings of NIPS 2002. MIT Press (2002)

17. Poon, H., Domingos, P.: Joint Inference in Information Extraction. In: Artificial Intelligence. vol. 22, pp. 913–918. AAAI Press (2007)
18. Sylwestrzak, W., Borbinha, J., Bouche, T., Nowiski, A., Sojka, P.: EuDMLTowards the European Digital Mathematics Library. In: Towards a Digital Mathematics Library. pp. 11–26 (2010), http://www.eudml.eu/
19. Tkaczyk, D., Bolikowski, L., Czeczko, A., Rusek, K.: A modular metadata extraction system for born-digital articles. In: 10th IAPR International Workshop on Document Analysis Systems. pp. 11–16 (2012)
20. Wellner, B., McCallum, A., Peng, F., Hay, M.: An integrated, conditional model of information extraction and coreference with application to citation matching. Proc. UAI pp. 593–601 (2004)