Reference Information Extraction and Processing Using Conditional Random Fields

Tudor Groza, Gunnar AAstrand Grimnes, and Siegfried Handschuh

ABSTRACT

Fostering both the creation and the linking of data with the scope of supporting the growth of the Linked Data Web requires us to improve the acquisition and extraction mechanisms of the underlying semantic metadata. This is particularly important for the scientific publishing domain, where currently most of the datasets are being created in an author-driven, manual manner. In addition, such datasets capture only fragments of the complete metadata, omitting usually, important elements such as the references, although they represent valuable information. In this paper we present an approach that aims at dealing with this aspect of extraction and processing of reference information. The experimental evaluation shows that, currently, our solution handles very well diverse types of reference format, thus making it usable for, or adaptable to, any area of scientific publishing.

1. INTRODUCTION

The progressive adoption of Semantic Web[1] techniques resulted in the creation of a series of datasets connected by the Linked Data[2] initiative, and via the Linked Data principles, into a universal Web of Linked Data. In order to foster the continuous growth of this Linked Data Web, we need to improve the acquisition and extraction mechanisms of the underlying semantic metadata.

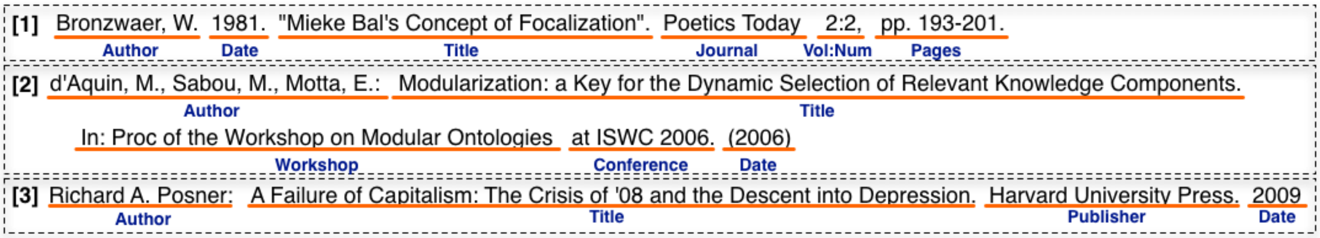
Unfortunately, the scientific publishing domain, a domain with an enormous potential for generating large amounts of Linked Data, still promotes trivial mechanisms for producing semantic metadata.[3] As an illustration, the metadata acquisition process of the Semantic Web Dog Food Server,[4] the main Linked Data publication repository available on the Web, consists of two steps:

- the authors manually fill-in submission forms corresponding to different publishing venues (e.g., conferences or workshops), with the resulting (usually XML) information being transformed via scripts into semantic metadata, and
- the entity URIs (i.e., authors and publications) present in this semantic metadata are then *manually* mapped to existing Web URIs for linking/consolidation purposes.

Tudor Groza (tudor.groza@uq.edu.au) is Postdoctoral Research Fellow, School of Information Technology and Electrical Engineering, University of Queensland, ***Gunnar AAstrand Grimnes*** (grimnes@dfki.uni-kl.de) is Researcher, German Research Center for Artificial Intelligence (DFKI) GmbH, Kaiserslautern, Germany, ***Siegfried Handschuh*** (msiegfried.handschuh@deri.org) is Senior Lecturer/Associate Professor, National University of Ireland, Galway, Ireland.

Moreover, independent of the creation/acquisition process, one particular component of the publication metadata, i.e., the reference information, is almost constantly neglected. The reason is mainly the amount of work required to manually create it, or the complexity of the task, in the case of automatic extraction. As a result, currently, there are no datasets in the Linked Data Web exposing reference information, while the number of digital libraries providing search and link functionality over references is rather limited. This is quite a problematic gap if we consider the amount of information provided by references and their foundational support for other application techniques that bring value to researchers and librarians, such as citation analysis and citation metrics, tracking temporal author-topic evolution[5] or co-authorship graph analysis.[6,7]

In this paper we focus on the first of the above-mentioned steps, i.e., providing the underlying mechanisms for automatic extraction of reference metadata. We devise a solution that enables extraction and chunking of references using Conditional Random Fields (CRF).[8] The resulting metadata can then be easily transformed into semantic metadata adhering to particular schemas via scripts, the added value being the exclusion of the manual author-driven creation step from the process. From the domain perspective, we focus on computer science and health sciences only because these domains have representative datasets that can be used for evaluation and hence enable comparison against similar approaches. However, we believe that our model can be applied also in domains such as digital humanities or social sciences, and we intend, in the near future, to build a corresponding corpus that would allow us to test and adapt (if necessary) our solution to these domains.



[1] Bronzwaer, W. 1981. "Mieke Bal's Concept of Focalization". Poetics Today 2:2, pp. 193-201.

[2] d'Aquin, M., Sabou, M., Motta, E.: Modularization: a Key for the Dynamic Selection of Relevant Knowledge Components. In: Proc of the Workshop on Modular Ontologies at ISWC 2006. (2006)

[3] Richard A. Posner: A Failure of Capitalism: The Crisis of '08 and the Descent into Depression. Harvard University Press. 2009

Figure 1. Examples of Chunked and Labeled Reference Strings

Reference chunking represents the process of label sequencing a reference string, i.e., tagging the parts of the reference containing the authors, the title, the publication venue, etc. The main issue associated with this task is the lack of uniformity in the reference representation. Figure 1 presents three examples of chunked and labeled reference strings. One cannot infer generic patterns for all types of references. For example, the year (or date) of some of the references of this paper are similar to example 2 from the figure, i.e., they are located at the very end of the reference string. Unfortunately, this does not hold for some journal reference formats, such as the one presented in example 1. And at the same time, the actual date might not comprise only the year, but also the month (and even day).

In addition to the placement of the particular types of tokens within the reference string, one of the major concerns when labeling these types of tokens is disambiguation. Generally, there are three categories of ambiguous elements:

- names—can act as authors, editors, or even part of organization names (e.g., Max Planck Institute); in example 1 a name is used as part of the title;
- numbers—can act as pages, years, days, volume numbers, or just numbers within the title;
- locations—can act as actual locations or part of organization names (e.g., Univ. of Wisconsin)

To help the chunker in performing disambiguation, one can use a series of markers, such as, *pp.* for pages, *TR* for technical reports, *Univ.* or *Institute* for organization. However, there are cases where such markers help in detecting the general category of the token, e.g., publication venue, but a more detailed disambiguation is required. For example, the *Proc.* marker generally signals the publication venue of the reference, without knowing exactly whether it represents a workshop, conference or even journal (as in the case of Proc. Natl. Acad. Sci.—Proceedings of the National Academy of Sciences).

The solution we have devised was built to properly handle such disambiguation issues and the intrinsic heterogeneous nature of references. The features of the CRF chunker model were chosen to provide a representative discrimination between the different fields of the reference string. Consequently, as the experimental results show, the resulting chunker has a superior efficiency, while at the same time maintaining an increased versatility.

The rest of the paper is structured as follows: in section 2 we briefly describe Conditional Random Fields and analyze the existing related work. Section 3 details the CRF-based reference chunker and before concluding in section 5, section 4 presents our experimental results.

2. BACKGROUND

2.1 Conditional Random Fields

To have a better understanding of the Machine Learning technique used by our solution, in the following we give a brief description of the Conditional Random Fields paradigm.

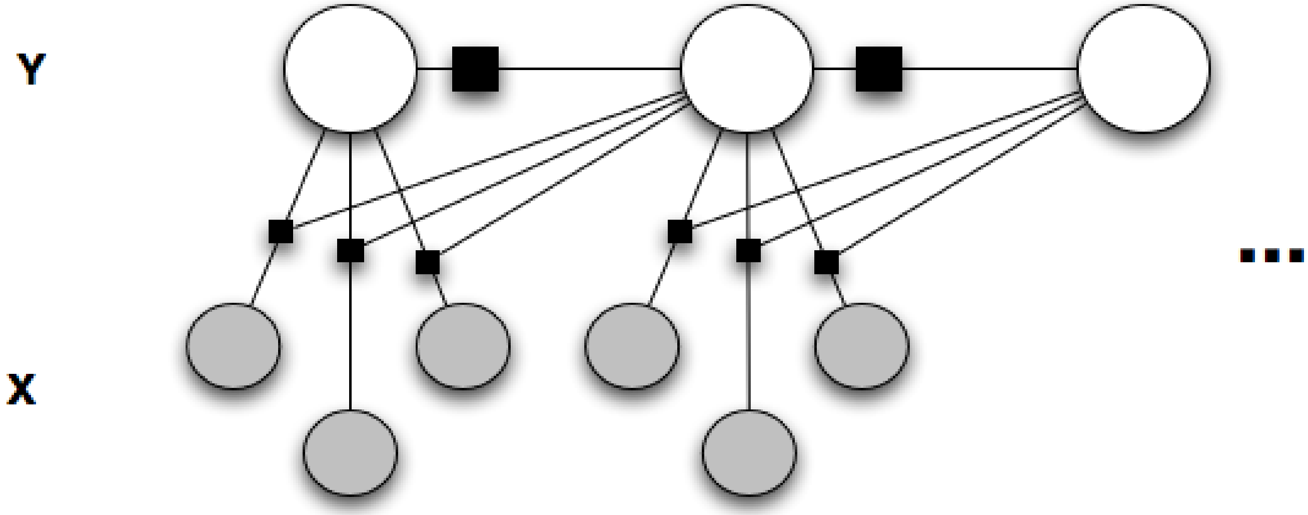


Figure 2. Example Linear CRF—Showing Dependencies Between Features X and Classes Y

Conditional Random Fields (CRF) is a probabilistic graphical model for classification. CRF, in general, can represent many different types of graphical models, however in the scope of this paper, we use the so-called linear-chain CRFs. A simple example of a linear dependency graph is shown in Figure 2, here only the features X of the previous item influences the class of the current item Y. The conditional probability is defined as:

$$p_\theta(y|x) = \frac{1}{Z(x)} \exp(\sum_j \theta_j F_j(x, y))$$

where $F_j(x,y) = \sum_{i=1}^{n} f_j\,(x, y_i, y_{i+1}, i)$ and $Z(x) = \sum_y \exp(\sum_j \theta_j F_j(x,y))$.

The model is usually trained by maximizing the log-likelihood of the training data by gradient methods. A dynamic algorithm is used to compute all the required probabilities $p_\theta(y_i, y_{i+1})$ for calculating the gradient of the likelihood.

This means that in contrast to *traditional* classification algorithms in Machine Learning (e.g., Support Vector Machines[9]), it not only considers the attributes of the current element when determining the class, but also attributes of preceding and succeeding items. This makes it ideal for tagging sequences, such as chunking of parts of speech or parts of references, which is what we require for our chunking task.

2.2 Related Work

In recent years, extensive research has been performed in the area of automatic metadata extraction from scientific publications. Most of the approaches focus on one of the two main metadata components, i.e., on the heading/bibliographic metadata or on the reference metadata, but there are also cases when the entire set is targeted. As this paper focuses only on the second component, within this section we present and discuss those applications that deal strictly with reference chunking.

The ParsCit framework is the closest technique mapping to our goals and methodology. [10] ParsCit is an open-source reference-parsing package. While its first version used a Maximum Entropy model to perform reference chunking, [11] currently, inspired by the work of Peng et al. ,[12] it uses a trained CRF model for label sequencing. The model was obtained based on a set of twenty-three token-oriented features tailored towards correcting the errors that Peng's CRF model produced. Our CRF chunker builds on the work of ParsCit. However, as we aimed at improving the chunking performance, we altered some of the existing features and introduced additional ones. Moreover, we have compiled significantly larger gazetteers required for detecting different aspects, such as names, places, organizations, journals, or publishers.

One of the first attempts to extract and index reference information led to the currently well-known system, CiteSeer.[13] Around the same period, Seymore et al. developed one of the first reference chunking approaches that used Machine Learning techniques.[14] The authors trained a Hidden Markov Model (HMM) to build a reference sequence labeler using internal states for different parts of the fields. As it represented pioneering work, it also resulted in the first gold standard set, the CORA dataset. At a later stage, the same group applied CRF for the first time to perform reference chunking, which later inspired ParsCit. [15]

In the same learning-driven category is the work of Han et al.[16] The authors proposed an effective word clustering approach with the goal of reducing feature dimensionality when compared to HMM, while at the same time improving the overall chunking performance. The resultant domain, rule-based word clustering method for cluster feature representation used clusters formed from various domain databases and word orthographic properties. Consequently, they achieved an 8.5 percent improvement on the overall accuracy of reference fields classification combined with a significant dimensionality reduction.

FLUX-CIM[17] is the only unsupervised[18] approach that targets reference chunking. The system uses automatically constructed knowledge bases from an existing set of sample references for recognizing the component fields of a reference. The chunking process features two steps:

- a probability estimation of a given term within a reference which is a value for a given reference field based on the information encoded in their knowledge bases, and
- the use of generic structural properties of references.

Similarly to Seymore et al.,[19] the authors have also created two datasets (specifically for the computer science and health science areas) to be used for comparing the achieved accuracies.

A completely different, and novel, direction was developed by Poon and Domingos.[20] Unlike all the other approaches, they propose a solution where the segmentation (chunking) of the reference fields is performed together with the entity resolution in a single integrated inference process. They, thus, help in disambiguating the boundaries of less-clear chunked fields, using the already well-segmented ones. Although the results achieved are similar to, and even better than some of, the above-mentioned approaches, this is suboptimal from the computational perspective: the chunking/resolution time reported by the authors measured around thirty minutes.

In addition to the previously described works, which were specifically tailored for bibliographic metadata extraction, there are a series of other approaches that could be used for the same purpose. For example, Cesario et al. propose an innovative *recursive boosting* strategy, with progressive classification, to reconcile textual elements to an existing attribute schema.[21] In the case of bibliographic metadata segmentation, the metadata fields would correspond to the textual elements, while an ontology describing them (e.g., DublinCore[22] or SWRC[23]) would have the schema role. The authors even describe an evaluation of the method using the DBLP citation dataset, however, without giving precise details on the fields considered for segmentation. Some other approaches include, in general, any sequence labeling techniques, e.g., SLF, [24] named entity recognition techniques, [25] or even Field Association (FA) terms extraction,[26] the latter working on bibliographic metadata fields in a quasi-similar manner as the *recursive boosting* strategy.

In conclusion, it is worth mentioning that retrieving citation contexts is an interesting research area especially in the context of digital libraries. Our current work does not feature this aspect, but we regard it as one of the key next steps to be tackled. Consequently, we mention the research performed by Schwartz et al.[27] Teufel et al.,[28] or Wu et al.[29] that deal with using citation contexts for discerning a citation's function and analyzing how this influences or is influenced by the work it points to.

3. METHOD

This section presents the CRF chunker model. We start by defining the preprocessing steps that deal with the extraction of the references block, dividing the block into actual reference entries and cleaning the reference strings, and then detail the CRF reference chunker features.

3.1 Prerequisites

Most of the features used by the CRF chunker require some forms of vocabulary entries. Therefore, we have manually compiled a comprehensive list of gazetteers (only for English, except for the names), explained as follows:

- *FirstName*—25,155 entries gazetteer of the most common first names (independent of gender);
- *LastName*—48,378 entries list of the most common surnames;
- *Month*—month names gazetteer and associated abbreviations;
- *VenueType*—a structured gazetteer with five categories: *Conference*, *Workshop*, *Journal*, *TechReport*, and *Website*. Each category has attached its own gazetteer, containing specific keywords and not actual titles. For example, the *Conference* gazetteerfeatures ten unigrams signaling conferences, such as *Conference*, *Conf,* or *Symposium*;
- *Location*—places, cities, and countries gazetteer comprising 17,336 entries;
- *Organization*—150 entries gazetteer listing organization prefixes and suffixes (e.g., *e.V.* or *KGaA*);
- *Proceedings*—simple list of all possible appearances of the *Proceedings* marker;
- *Publisher*—564 entries gazetteer comprising publisher unigrams (produced from around 150 publisher names);
- *JTitle*—12,101 entries list of journal title unigrams (produced from around 1600 journal titles);
- *Connection*—a 42 entries stop-word gazetteer (e.g., *to*, *and*, *as*).

3.2 Preprocessing

In the preprocessing stage we deal with three aspects:

- cleaning the provided input,
- extracting the reference block, and
- the division of the reference block into reference entries.

The first step aims to clean the raw textual input received by the chunker of unwanted spacing characters while at the same time ensuring proper spacing where necessary. Since the source of the textual input is unknown to the chunker, we make no assumptions with regard to its structure or content.[30] Thus, in order to avoid inherent errors that might appear as a result of extracting the raw text from the original document, we perform the following cleaning steps:

- we compress the text by eliminating unnecessary carriage returns, such that the lines containing less than 15 characters are merged with previous ones,[31]
- we introduce spaces after some punctuation characters, such as “,” “.” or “-”, and finally,
- we split the camel-cased strings, such as JohnDoe.

The result will be a compact and clean version of the input. Also, if the raw input is already compact and clean, this preprocessing step will not affect it.

The extraction of the reference block is done using regular expressions. Generally, we search in the compacted and cleaned input for specific markers, like *References* or *Bibliography*, located mainly at the beginning of a line. If these are not directly found, we try different variations, such as, looking for the markers at the end of a line, or looking for split markers onto two lines (e.g., *Ref – erences*, or *Refer – ences*). This latter case is a typical consequence of the above-described compacting step if the initial input was erroneously extracted. The text following the markers is considered for division, although it may contain unwanted parts such as appendices or tables.

The division into individual reference entries is performed on a case basis. After splitting the reference block based on new lines, we look for prefix patterns at the beginning of each line. As an example, we analyze which lines start with "[", "(", or a number followed by "." or space, and we record the positions of these lines in the list of all lines. To ensure that we don't consider any false positives when merging the adjacent lines into a reference entry, we compute a global average of the differences between positions. Assuming that a reference does not span on more than four lines, if this average is between one and four, a reference entry is created. The same average is also used to extract the last reference in the list, thus detaching it from eventual appendices or tables.

3.3 The reference chunking model

We have built the CRF learning model based on a series of features used in principle also by the other CRF reference chunking approaches such as ParsCit[32] or Peng and McCallum[33]. A set of feature values is used to characterize each token present in the reference string, where the reference's token list is obtained by dividing the reference string into space-separated pieces. The complete list of features is detailed as follows. We use example 1 from figure 1 toexemplify the feature values.

- Token—the original reference token: *Bronzwaer,*
- Clean token—the original token, stripped of any punctuation and lower cased: *bronzwaer*
- Token ending—a flag signaling the type of ending (possible values: lower cap – *c* / upper cap – *C* / digit – *0* / punctuation character: *,*
- Token decomposition–start—five individual values corresponding to token's first five characters, taken gradually: *B, Br, Bro, Bron, Bronz*
- Token decomposition–end—five individual values corresponding to the token's last five characters, taken gradually: *r, er, aer, waer, zwaer,*
- POS Tag—the token's part of speech tag (possible values: proper noun phrase – *NNP* ,
- noun phrase – *NP*, adjective – *JJ*, cardinal number – *CD*, etc): *NNP*
- Orthographic case—a flag signaling the token's orthographic case (possible values:
- *initialCap*, *singleCap*, *lowercase*, *mixedCaps*, *allCaps*): *singleCap*
- Punctuation type—a flag signaling the presence and type of a trailing punctuation character (possible values: *cont*, *stop*, *other*): *cont*
- Number type—a flag signaling the presence and type of a number in the token (possible values: *year, ordinal, 1dig, 2dig, 3dig, 4dig, 4dig+, noNumber*): *noNumber*

- Dictionary entries—a set of ten flags signaling the presence of the token in the set of individual gazetteers listed in Sect. 3.1. For our example the dictionary feature set would be: *no LastName no no no no no no no no*
- Date check—a flag checking whether the token may contain a date in form of a period of days, e.g., 12-14 (possible values: *possDate, no*): *no*
- Pages check—a flag checking whether the token may contain pages, e.g., 234–238 (possible values: *possPages, no*): *no*
- Token placement—the token placement in the reference string, based on its division into nine equal consecutive buckets. This feature indicates the bucket number: *0*

For training purposes we compiled and manually tagged a set of 830 randomly chosen references. These were extracted from random publications from diverse conferences and journals from the computer science field (collected from IEEE Explorer, Springer Link or the ACM Portal), manually cleaned, tagged, and categorized according to their type of publication venue.[34] To achieve an increased versatility, instead of performing cross- validation,[35] which would result in a dataset-tailored model with limited or no versatility, we opted for sampling the test data. Hence, we included in the training corpus some samples from the testing datasets as follows: 10 percent of the CORA dataset (i.e., 20 entries),[36] 10 percent of the FLUX-CIM CS dataset (i.e., 30 entries),[37] and 1% of the FLUX-CIM HS dataset (i.e., 20 entries). Consequently, the final training corpus consisted of a total of 900 reference strings. To clarify, this is, to some extent, similar to the dataset-specific cross-validation, but instead of considering, for example, a 60–40 ratio for training/testing, we used only 10 percent for training, while the testing (described in section 4) was performed as a direct application of the chunker on the entire dataset.

As already mentioned, our focus on computer science and health sciences is strictly due to evaluation purposes. Our proposed model is domain-agnostic, and hence, the steps described here can be easily performed on datasets emerged from other domains, if at all necessary. In reality, the chunker's performance on references from a domain not covered above can be easily boosted simply by including a sample of references in the training set and then retraining the chunker.

The list of labels used for training and then testing consists of *Author*, *Title*, *Journal*, *Conference*, *Workshop*, *Website*, *Technicalrep*, *Date*, *Publisher*, *Location*, *Volnum*, *Pages*, *Etal*, *Note*, *Editors*, *Organization*. As we will see in the evaluation, not all labels were actually used for testing (e.g., *Note* or *Editors*), some of them being present in the model for the sake of disambiguation. Also, as opposed to the other approaches, we made a clear distinction between *Workshop* and *Conference*, which adds an extra degree to the complexity of the disambiguation. The CRF model was trained using the MALLET (A Machine Learning for Language Toolkit) implementation.[38]

The output of the chunker is post-processed to expose a series of fine-grained details. As shown in figure 1 in all the examples, the chunking provides a blocked partition of the reference string, but we require for the *Author* field an even deeper partition. Consequently, following a rule-based approach we extract the individual author names from the *Author* block making use of the punctuation marks, the orthographic case, and the alternation between initials and actual names. When no initials, subject to the existing punctuation marks, we consider as a rule-of-thumb that each name generally comprises one first name and one surname (in this order, i.e., John Doe). The result of the post-processing is used in the linking process.

4. EXPERIMENTAL RESULTS

We have performed an extensive evaluation of the proposed reference chunking approach. In general, all the previous work in reference chunking focuses on raw reference chunking, i.e., label sequencing at the macro level. More concretely, the other approaches split and tag the reference strings using blocks of complete references, without going into details such as chunking individual authors. The only exception is the ParsCit package that does perform complete reference chunking in a similar fashion as we do. The evaluation results presented in this section, will feature complete chunking only for our solution and for ParsCit, and raw chunking for the rest of the approaches.

Field	ParsCit			Peng	Han et al.			Our approach		
	P	R	F$_1$	F$_1$	P	R	F$_1$	P	R	F$_1$
Author	98.7	99.3	98.99	99.4	92.6	99.1	97.6	99.08	99.6	**99.30**
Title	96.0	98.4	97.18	**98.3**	92.2	93.0	92.6	95.64	95.64	95.64
Date	100	98.4	**99.19**	98.9	98.5	95.9	97.2	99.33	*98.67*	98.99
Pages	97.7	98.4	98.04	98.6	95.6	96.9	96.2	99.28	99.22	**99.24**
Location	95.6	90.0	92.71	87.2	77.7	71.5	74.5	93.45	92.59	**93.01**
Organization	90.9	87.9	89.37	**94.0**	76.5	77.3	76.9	*100*	87.87	93.54
Journal	90.8	91.2	90.99	91.3	77.1	78.7	77.9	94.02	97.42	**95.68**
Booktitle	92.7	94.2	93.44	93.7	88.7	88.9	88.88	97.77	98.44	**98.10**
Publisher	95.2	88.7	91.83	76.1	56.0	64.1	59.9	94.84	95.83	**95.33**
Tech. rep.	94.0	79.6	86.2	86.7	56.2	64.1	59.9	100	90.90	**95.23**
Website	-	-	-	-	-	-	-	100	100	**100**

Table 1. Evaluation Results on the CORA Dataset

An additional observation we need to make is related to the reference fields taken into account. Most of the fields we have focused on coincide with the fields considered by all the existing relevant approaches. Nevertheless, there are also some discrepancies, listed as follows:

- the fields: *Volume*, *Number*, *Editors*, or *Note* were used in the chunking process but are not considered for evaluation
- unlike all the other approaches, we make the distinction between *Conference* and *Workshop* as publication venues. However, for alignment purposes (i.e., to be able to compare our results with the other approaches), in the evaluation results these are merged into the *Booktitle* field.

The actual tests were performed on four different datasets, three of them used also for evaluating the other approaches, and a fourth one compiled by us. In the case of the three existing datasets, during the experimental evaluation we did not make use of the preprocessing step as they were already clean.

As evaluation metric, we used the F_1 score,[39] i.e., the harmonic mean of precision and recall, using the following formula:

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

In the following, we iterate over each dataset, by providing a short description and the experimental results. It is worth mentioning that our CRF reference chunker was trained only once, as described earlier, and not specifically for each dataset.

4.1 Dataset: CORA

The CORA dataset is the first gold standard created for automatic reference chunking.[40] It comprises two hundred reference strings and focuses on the computer science area. Each entry is segmented into thirteen different fields: *Author*, *Editor*, *Title*, *Booktitle*, *Journal*, *Volume*, *Publisher*, *Date*, *Pages*, *Location*, *Tech*, *Institution* and *Note*.

Table 1 shows the comparative evaluation results on the CORA dataset of ParsCit, Peng et al.,[41] Han et al.,[42] and our approach. We observe that our chunker outperforms the other chunkers on most of the fields, with some of them presenting a significant increase in performance (looking at the F1 score): *Journal* from 91.3 percent to 95.68 percent, *Booktitle* from 93.44 percent to 98.10 percent, *Publisher* from 91.83 percent to 95.33 percent, and especially *Tech. rep.* from 86.7 percent to 95.23 percent. In the case of the fields where our chunker was outperformed, the F1 score is very close to the best of the approaches and includes an increase in one of its two components (i.e., precision or recall). For example, on the *Organization* field, we scored 93.54percent, the best being Peng's 94 percent. However, we achieved a gain of almost 10 percent in precision when compared with ParsCit (100 percent vs. 90.9 percent precision). Similarly, on the *Date* field, our F1 was 98.99 percent, opposed to ParsCit's 99.19 percent, but with a better recall of 98.67 percent.

Field	ParsCit			FLUX-CIM			Our approach		
	P	R	F_1	P	R	F_1	P	R	F_1
Author	98.8	99.0	98.89	93.59	95.58	94.57	99.08	99.08	**99.08**
Title	98.8	98.3	98.54	93.0	93.0	93.0	99.65	99.65	**99.65**
Date	99.8	94.5	97.07	97.75	97.44	97.59	98.55	98.19	**98.36**
Pages	94.7	99.3	96.94	97.0	97.84	97.41	97.28	97.72	**97.49**
Location	96.9	88.4	92.45	96.83	97.6	**97.21**	95.55	94.5	95.02
Journal	97.1	82.9	89.43	95.71	97.81	**96.75**	94.0	*97.91*	95.91
Booktitle	95.7	99.3	97.46	97.47	95.45	96.45	99.13	99.13	**99.13**
Publisher	98.8	75.9	85.84	100	100	**100**	98.59	98.59	98.59

Table 2. Evaluation Results on the FLUX-CIM Dataset—CS Domain

Field	FLUX-CIM			Our approach		
	p	R	F_1	p	R	F_1
Author	98.57	99.04	98.81	99.8	*99.36*	99.57
Title	84.88	85.14	85.01	*91.39*	*91.39*	97.39
Date	*99.85*	*99.5*	*99.61*	99.89	*99.69*	99.78
Pages	*99.1*	99.2	*99.45*	*99.94*	*99.59*	99.76
Journal	97.23	89.35	93.13	99.42	*99.16*	99.28

Table 3. Evaluation Results on the FLUX-CIM Dataset—HS Domain

4.1 Dataset: FLUX-CIM

FLUX-CIM[43] is an unsupervised[44] reference extraction and chunking system. In order to evaluate its performance, the authors of FLUX-CIM created two separate datasets:

- the FLUX-CIM CS dataset, composed on a collection of heterogeneous references from the Computer Science field, and
- the FLUX-CIM HS dataset is comprised of an organized and controlled collection of references from PubMed.

The FLUX-CIM CS dataset contains three hundred reference strings randomly selected from the ACM Digital Library. Each string is segmented into ten fields: *Author*, *Title*, *Conf*, *Journal*, *Volume*, *Number*, *Pub*, *Date*, *Pages* and *Place*. The FLUX-CIM HS dataset contains 2000 entries, with each entry segmented into six fields: *Author*, *Title, Journal*, *Volume*, *Date* and *Pages*.

Table 2 presents the comparative test results achieved by ParsCit, FLUX-CIM, and our approach on the CS dataset. Similar to the CORA dataset, our chunker outperformed the other chunkers on the majority of the fields, exceptions being the *Location*, *Journal,* and *Publisher* fields.

The test results on the HS dataset are presented in table 3. Here we can observe a clear performance improvement on all fields, in some cases the difference being significant, e.g., the *Title* field, from 85.01 percent to 97.39 percent, or the *Journal* field, from 93.12 percent to 99.28 percent. This increase is even more relevant considering the size of the dataset, each 1percent representing twenty references.

4.3 Dataset: CS-SW

While the CORA and FLUX-CIM CS datasets do focus on the computer science field, they do not cover the slight differences in reference format that can be found nowadays in the Semantic Web community. Consequently, to show the even broader application of our approach, we have compiled a dataset named CS-SW comprising 576 reference strings randomly selected from publications in the Semantic Web area, from conferences such as International Semantic Web Conference (ISWC), the European Semantic Web Conference (ESWC), the World Wide Web Conference (WWW), or the European Conference on Knowledge Acquisition (and co-located workshops).[45] Each reference entry is segmented into twelve fields: *Author*, *Title*, *Conference*, *Workshop*, *Journal*, *Techrep*, *Organization*, *Publisher*, *Date*, *Pages*, *Website* and *Location*.

Table 4 shows the results of the tests carried out on this dataset. One can easily observe that the chunker performed in a similar manner as on the CORA dataset, with emphasis on the *Author*, *Date*, *Pages* and *Publisher* fields.

Field	Our approach		
	P	R	F_1
Author	98.61	99.27	98.93
Title	94.91	93.29	94.09
Date	98.89	98.34	98.61
Pages	98.94	97.24	98.08
Location	93.9	92.77	93.33
Organization	85.71	80	82.75
Journal	94.59	93.33	93.95

Conference	96.66	95.08	95.86
Workshop	83.33	88.23	85.71
Publisher	96.61	97.43	97.01
Tech. rep.	100	80	88.88
Website	98.14	94.64	96.35

Table 4. Evaluation Results on the CS-SW Dataset

5. CONCLUSION

In this paper we presented a novel approach for extracting and chunking reference information from scientific publications. The solution, realized using a CRF trained chunker, achieved good results in the experimental evaluation, in addition to an increased versatility shown by applying the one-time trained chunker on multiple testing datasets. This enables a straightforward adoption and reuse of our solution for generating semantic metadata in any digital library or publication repository focused on scientific publishing.

As next steps, we plan to create a comprehensive dataset covering multiple heterogeneous domains (e.g., social sciences or digital humanities) and evaluate the chunker's performance on it. Then we will focus on developing an accurate reference consolidation and linking technique, to address the second step mentioned in section 1, i.e., aligning the resulting metadata to the existing Linked Data on the Web. We plan to develop a flexible consolidation mechanism by dynamically generating and executing SPARQL queries from chunked reference fields and filtering the results via two string approximation metrics (a combination of Monge-Elkan and Chapman Soundex algorithms). The SPARQL queries generation will be implemented in an extensible manner, via customizable query modules, to accommodate the heterogeneous nature of the diverse Linked Data sources. Finally, we intend to develop an overlay interface for arbitrary online publication repositories, to enable on-the-fly creation, visualization, and linking of semantic metadata from repositories that currently do not expose their datasets in a semantic / linked manner.

ACKNOWLEDGEMENTS

The work presented in this paper has been funded by Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lion-2).

REFERENCES AND NOTES

[1] Tim Berners-Lee et al., "The Semantic Web," *Scientific American* 284 (2001): 35–43.

[2] Christian Bizer et al., "Linked Data—The Story So Far," *International Journal on Semantic Web and Information Systems* 5 (2009): 1–22.

[3] Generating computer-understandable metadata represents an issue, in general, in the publishing domain, and not necessarily only in its scientific area. However, the relevant literature dealing with metadata extraction/generation has focused on scientific publishing, because of its accelerated growing rate, especially with the increasing use of the World Wide Web as a dissemination mechanism.

[4] Knud Moeller et al., “Recipes for Semantic Web Dog Food – The ESWC and ISWC Metadata Projects,” *Proceedings of the 6th International Semantic Web Conference* (Busan, Korea, 2007).

[5] Wei Peng and Tao Li, “Temporal relation co-clustering on directional social network and author-topic evolution,” *Knowledge and Information Systems* 26 (2011): 467–86.

[6] Laszlo Barabasi et al., “Evolution of the social network of scientific collaborations,” *Physica A: Statistical Mechanics and its Applications* 311 (2002): 590–614.

[7] Xiaoming Liu et al., “Co-authorship networks in the digital library research community,” *Information Processing & Management* 41 (2005): 1462–80.

[8] John D. Lafferty et al., “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data,” *Proceedings of the 18th International Conference on Machine Learning* (San Francisco, CA, USA, 2001): 282–89.

[9] Vladimir Vapnik, *The Nature of Statistical Learning Theory* (New York: Springer, 1995).

[10] Isaac G. Councill et al., “ParsCit: An Open-source CRF Reference String Parsing Package,” *Proceedings of the Sixth International Language Resources and Evaluation* (Marrakech, Morocco, 2008).

[11] Yong Kiat Ng, “Citation Parsing Using Maximum Entropy and Repairs” (master's thesis, National University of Singapore, 2004).

[12] Fuchun Peng and Andrew McCallum, “Information Extraction from Research Papers Using Conditional Random Fields,” *Information Processing & Management* 42 (2006): 963–79.

[13] C. Lee Giles et al., “CiteSeer: An Automatic Citation Indexing System,” *Proceedings of the Third AMC Conference on Digital Libraries* (Pittsburgh, PA, 1998): 89–98.

[14] Kristie Seymore et al., “Learning Hidden Markov Model Structure for Information Extraction,” *Proceedings of the AAAI Workshop on Machine Learning for Information Extraction* (1999): 37–42.

[15] Isaac G. Councill et al., “ParsCit: An Open-source CRF Reference String Parsing Package,” *Proceedings of the Sixth International Language Resources and Evaluation* (Marrakech, Morocco, 2008).

[16] Hui Han et al., “Rule-based Word Clustering for Document Metadata Extraction,” *Proceedings of the Symposium on Applied Computing* (Santa Fe, New Mexico, 2005).

[17] Eli Cortez et al., “FLUX-CIM: Flexible Unsupervised Extraction of Citation Metadata,” *Proceedings of the 2007 Conference on Digital Libraries* (New York, 2007): 215–24.

[18] Machine Learning methods can be broadly classified into two categories: supervised and unsupervised. Supervised methods require training on specific datasets that exhibit the characteristics of the target domain. To achieve high accuracy levels, the training dataset needs to be reasonably large, and more importantly, it has to cover most of the possible

exceptions from the intrinsic data patterns. Unlike supervised methods, unsupervised methods do not require training, and in principle, use generic rules to encode both the expected patterns and the possible exceptions of the target data.

[19] Peng and McCallum, "Information Extraction from Research Papers Using Conditional Random Fields."

[20] Hoifung Poon and Pedro Domingos, "Joint inference in information extraction," *Proceedings of the 22nd National Conference on Artificial Intelligence* (Vancouver, British Columbia, Canada, 2007): 913–18.

[21] Ariel Schwartz et al., "Multiple Alignment of Citation Sentences with Conditional Random Fields and Posterior Decoding," *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (Prague, Czech Republic, 2007): 847–57.

[22] Simone Teufel et al., "Automatic Classification of Citation Function," *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing* (Sydney, Australia, 2006): 103–10.

[23] Jien-Chen Wu et al., "Computational Analysis of Move Structures in Academic Abstracts," *COLING/ACL Interactive Presentation Sessions* (Sydney, Australia, 2006): 41–44.

[24] Eugenio Cesario et al., "Boosting text segmentation via progressive classification," *Knowledge and Information Systems* 15 (2008): 285–320.

[25] Dublin Core website, http://dublincore.org (accessed May 4, 2011).

[26] York Sure et al., "The SWRC ontology – Semantic Web for research communities," *Proceedings of the 12th Portuguese Conference on Artificial Intelligence* (Covilha, Portugal, 2005).

[27] Yanjun Qi et al., "Semi-Supervised Sequence Labeling with Self-Learned Features," *Proceedings of IEEE International Conference on Data Mining* (Miami, FL, USA, 2009).

[28] David Sanchez et al., "Content Annotation for the Semantic Web: An Automatic Web-Based Approach," *Knowledge and Information Systems* 27 (2011): 393-418.

[29] Tshering Cigay Dorji et al., "Extraction, selection and ranking of Field Association (FA) Terms from domain-specific corpora for building a comprehensive FA terms dictionary," *Knowledge and Information Systems* 27 (2011): 141–61.

[30] Please note that the chunker is document-format agnostic and takes as input only raw text. The actual extraction of this raw text from the original document (PDF, DOC or some other format) is the user's responsibility.

[31] As a note, we chose this length of fifteen characters empirically, and based on the assumption that in any format the publication content lines usually have more than fifteen characters.

[32] Lafferty et al., “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data.”

[33] Councill et al., “ParsCit: An Open-source CRF Reference String Parsing Package.”

[34] The manual tagging was performed by a single person and since the reference chunks have no ambiguity attached, we did not see the need for running any data reliability tests.

[35] Ron Kohavi, “A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection,” *Proceedings of the 14th International Joint Conference on Artificial Intelligence* (Montreal, Quebec, 1995): 1137–43.

[36] Peng and McCallum, “Information Extraction from Research Papers Using Conditional Random Fields.”

[37] Councill et al., “ParsCit: An Open-source CRF Reference String Parsing Package.”

[38] Mallet: MAchine Learning for LanguagE Toolkit, http://mallet.cs.umass.edu (accessed May 4, 2011).

[39] William M. Shaw et al., “Performance standards and evaluations in IR test collections: Cluster-based retrieval models,” *Information Processing & Management* 33 (1997): 1–14.

[40] Peng and McCallum, “Information Extraction from Research Papers Using Conditional Random Fields.”

[41] Councill et al., “ParsCit: An Open-source CRF Reference String Parsing Package.”

[42] Seymore et al., “Learning Hidden Markov Model Structure for Information Extraction.”

[43] Han et al., “Rule-based Word Clustering for Document Metadata Extraction.”

[44] Cortez et al., “FLUX-CIM: Flexible Unsupervised Extraction of Citation Metadata.”

[45] The CS-SW dataset is available at http://resources.smile.deri.ie/corpora/cs-sw (accessed May 4, 2011).