Knowl Inf Syst (2013) 34:1–21
DOI 10.1007/s10115-011-0473-6

REGULAR PAPER

From raw publications to Linked Data

Tudor Groza · Gunnar AAstrand Grimnes · Siegfried Handschuh · Stefan Decker

Received: 7 September 2010 / Revised: 10 June 2011 / Accepted: 13 December 2011 /
Published online: 29 December 2011

Abstract The continuous development of the Linked Data Web depends on the advancement of the underlying extraction mechanisms. This is of particular interest for the scientific publishing domain, where currently most of the data sets are being created manually. In this article, we present a Machine Learning pipeline that enables the automatic extraction of heading metadata (i.e., title, authors, etc) from scientific publications. The experimental evaluation shows that our solution handles very well any type of publication format and improves the average extraction performance of the state of the art with around 4%, in addition to showing an increased versatility. Finally, we propose a flexible Linked Data-driven mechanism to be used both for refining and linking the automatically extracted metadata.

Keywords Metadata extraction · Support vector machines · Conditional random fields · Linked data

1 Introduction

The progressive adoption of the Semantic Web [3] resulted in the creation and publishing of an important number of scattered data sets on the Web. The aim of the Linked Data

T. Groza (✉) · S. Handschuh · S. Decker
DERI, National University of Ireland, Galway, Ireland
e-mail: tudor.groza@uq.edu.au

T. Groza · S. Handschuh · S. Decker
IDA Business Park, Lower Dangan, Galway, Ireland

Present Address:
T. Groza
School of ITEE, The University of Queensland, R. 78-709, Level 7,
GP South (#78), St. Lucia campus, 4072 Queensland, Australia

G. A. Grimnes
DFKI GmbH, Trippstadter Strasse 122, 67663 Kaiserslautern, Germany

initiative [4] is to establish bridges between these silos of semantic data, thus bringing into existence a Web of Linked Data. In order to foster both the creation and the linking of data, we need to continuously improve the underlying acquisition and extraction mechanisms, as well as the means for entity co-reference resolution (with the scope of finding and linking instances of the same entity on the Web). For example, the scientific publishing domain, a domain that can generate large amounts of semantic metadata, needs particular attention, as the acquisition process is still done, to a large extent, manually. At the same time, this metadata is required to enable the application of other techniques that bring value to researchers, such as tracking temporal author-topic evolution [20] or coauthorship graph analysis [2,15].

Currently, the authors of publications still fill in (manually) submission forms, with the resulted information being then transformed, via a series of scripts, into semantic metadata (see, for example, the metadata acquisition process for the Semantic Web Dog Food Server [16]). The linking part, that is, the creation/generation of `owl:sameAs` or `rdfs:seeAlso` relations between the data set-local URIs and existing Web URIs of the instances of the same entity, is realized in two ways: (i) either at transformation time, when the data set creator manually picks already existing URIs for the instances present in the data set, or (ii) after the publication of the data set, via different algorithmic approaches.

Our goal is to provide a two-step, complete solution that enables both the automatic extraction, as well as the linking of semantic metadata from scientific publications in a straightforward and transparent manner. Within the context of this article, we focus on the heading metadata and specifically on six of its fields: *Title*, *Authors*, *Affiliation*, *Address*, *Email* and *Abstract*.

Extensive research has been performed in the area of metadata extraction, with a number of solutions both heuristic and machine learning (ML)-powered being proposed. Our contribution, discussed in this article, is twofold. Firstly, we propose a novel Machine Learning-based extraction pipeline that uses: (i) a combination of a cascade of standard (dual-class) support vector machines (SVM) [29] classifiers, (ii) a rule-based convergence procedure and (iii) two conditional random fields (CRF) [14] chunkers for extracting the heading metadata. The pipeline is agnostic of the document format (i.e., PDF, DOC or Web pages) and document purpose (i.e., conference/Journal publications or technical reports) as it works directly on the raw text and thus ignoring the underlying representation or style. Secondly, we devise a flexible entity co-reference resolution mechanism, using dynamic SPARQL queries generation and execution, that can be applied for refining and linking the automatically extracted metadata, for example, with the goal of generating linked publication data sets.

The evaluation results, focused only on the extraction step, show that our pipeline performs as good as the state of the art solutions (when compared via tenfold cross-validation on the gold standard data set) and improves the average extraction effectiveness, in addition to showing an increased versatility, when trained and applied in setting similar to a real-world scenario. The versatility is shown by testing the pipeline on three data sets comprising a mixture of types of scientific publications (i.e., conference/workshop/journal publications or technical reports), resulted from different document formats, for example, PDF or Web pages.

The remainder of the article is structured as follows: we start in Sect. 2 by briefly describing the two Machine Learning paradigms used to develop our solution. Section 3 details the metadata extraction pipeline, while Sect. 4 presents the refinement and linking approach. In Sect. 5, we discuss an extensive evaluation of the metadata extraction. Finally, before concluding in Sect. 7, in Sect. 6, we present a comprehensive overview of the related approaches.

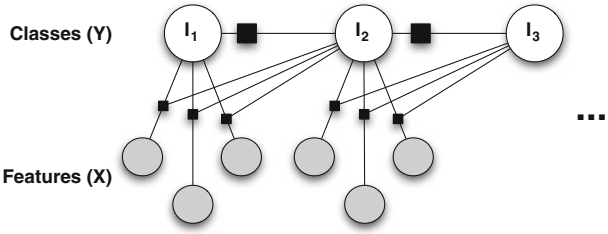

Fig. 1 Example linear CRF—showing dependencies between features X and classes Y

2 Used machine learning techniques

We start by providing a brief description of the two ML techniques used to achieve the automatic metadata extraction: conditional random fields (CRF) and support vector machines (SVM).

2.1 Conditional random fields

Conditional random fields (CRF) [14] are a discriminative probabilistic graphical model for classification. CRF, in general, can represent many different types of graphical models; however, in the scope of this article, we use the so-called linear-chain CRFs. A simple example of a linear dependency graph is shown in Fig. 1, here only the features (X) of the previous items influence the class of the current item (Y). The conditional probability of a label sequence y (of Y) given an observation of a feature x (of X) is defined as:

$$p(y|x, \lambda) = \frac{1}{Z(x)} \exp\left(\sum_j \lambda_j F_j(x, y)\right) \tag{1}$$

where $F_j(y, x) = \sum_{i=1}^{n} f_j(y_{i-1}, y_i, x, i)$, λ_j is a parameter to be estimated from training data, and $Z(x)$ is a normalization factor. The model is usually trained by maximizing the log-likelihood of the training data by gradient methods.

In contrast to *traditional* classification algorithms in Machine Learning, CRF 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, parts of references or a generic string which is what we require for our classification and chunking tasks.

2.2 Support vector machines

The support vector machine [29] is a method used for finding the optimal separating hyperplane between sets of points in some N-dimensional space. A set of data points close to the decision boundary is chosen (the support-vectors in the name), and the optimal hyperplane is calculated from these. The true power of SVMs comes from the so-called *kernel trick*, where a transformation function is used to transform the data to higher dimension when a linear separation of the classes is not possible.

Figure 2 shows on the left how some data are not linearly separable in 2D space, but it is easily achieved when transformed to a 3D space. The *trick* refers to the fact that the exact locations in the higher dimensional space (kernel-space) do not have to be computed,

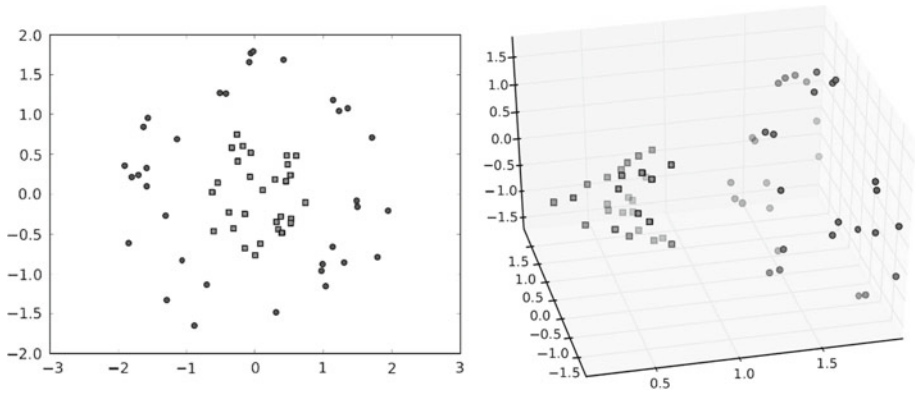

Fig. 2 Example of SVM transformation allowing linear separation in higher dimension space

the algorithm only makes use of the dot product between vectors, which are created in the transformed space by the kernel function. This means that a very high or even an infinite dimensional spaces may be used. Various well-known kernels for SVMs are known, for instance, the Gaussian RBF (radial basis function) kernel (for elliptical decision boundaries of the linear space) or the polynomial kernel. These are also used by our line-oriented classification mechanism (detailed in Sect. 3).

The polynomial kernel function of degree d has an output dependent on the direction of the two vectors in lower dimensional space, due to the dot product in the kernel (Eq. 2). All vectors with the same direction will have a high output from the kernel, with the magnitude of the output being dependent on the magnitude of the vector. As a result, polynomial kernels are suited for problems where all the training data are normalized.

$$K(x_i, x_j) = (x_i \cdot x_j + 1)^d \tag{2}$$

$$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right) \tag{3}$$

In the case of the Gaussian RBF (Eq. 3), the output is dependent on the Euclidean distance of x_j from x_i (one of these will be the support vector and the other will be the testing data point). The support vector will be the center of the RBF and σ will determine the area of influence this support vector has over the data space. A larger value of σ will give a smoother decision surface and more regular decision boundary. This is because an RBF with large σ will allow a support vector to have a strong influence over a larger area.

3 Metadata extraction pipeline

This section presents the metadata extraction pipeline we have developed. We start by having a look at the extraction task (or the problem we try to solve) and then introduce the pipeline structure, in addition to detailing the preprocessing stage, the classifiers, the rule-based convergence and finally the disambiguation chunkers.

3.1 Metadata extraction task

To have a better understanding of the problem we address, Fig. 3 depicts an example of a raw textual counterpart of a publication heading. The actual structure of the heading depends on

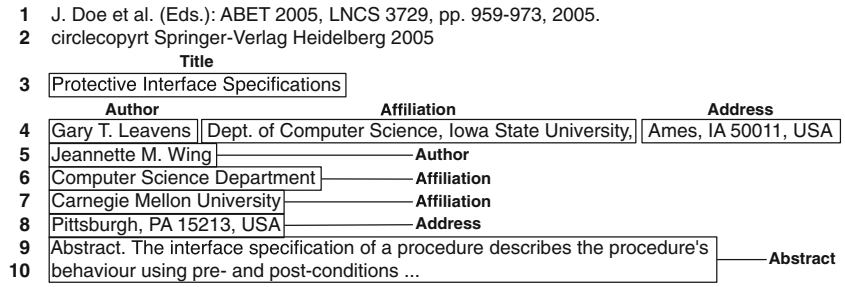

Fig. 3 Example of a raw textual counterpart of a publication heading, tagged with metadata fields

a series of factors, such as (i) the formatting style of the publication, (ii) the output of the text extractor from the original document, or (iii) the presence of certain characters that may influence the extractor's output. Thus, this example shows merely one of the many possible heading structures.

Our goal is to extract the bibliographic metadata fields (as shown in Fig. 3), initially by classifying each line of the heading into one of the corresponding classes: *Title*, *Authors*, *Affiliation*, *Address* and *Email*, and then by chunking the lines with multiple classes assigned, into the corresponding fields. All lines outside these categories should be discarded. There is also a sixth class, the *Abstract* class; however, the extraction of the abstract is performed as a preprocessing step before the actual classification. Based on Fig. 3, both the classification and final extraction processes, respectively, are not straightforward, due to the absence of patterns, specifically:

- the heading does not start directly with the title (in Fig. 3 it starts with proceedings information on lines 1 and 2, i.e., J. Doe et al. (Eds.). . .),
- the line containing the first author also contains his affiliation and address (see Line 4), and
- the information about the second author is spread over four lines (Lines 5–8).

The solution we propose and detail in the following sections, in the form of an extraction pipeline, is designed specifically to handle such ambiguous cases.

A typical real-world scenario in which the pipeline can be used is for the generation/production of linked metadata from conference proceedings. Similarly, it can also be used for compiling personal digital libraries of publications coming from heterogeneous sources, such as journal, conference proceedings, or Web pages. In practice, we have already deployed and tested successfully our approach to produce the Linked Data data set from the ISWC 2010 conference proceedings, following a process similar to the metadata creation process of the Semantic Web Dog Food Server, but executed automatically. The pipeline takes as input the raw textual content of scientific publications (being document format agnostic) and outputs the corresponding metadata. To achieve an increased accuracy and to focus on the actual extraction, we restrict the input to those parts of the publication that have the highest chances of containing valuable information. Therefore, instead of considering the entire publication content, we focus only on its first page for extracting the heading metadata. The way in which the content of the pages is being delimited, is out of our scope, and thus, we are shifting this responsibility to the specific text extractor used for the different document formats. Nevertheless, to enable a direct comparison between our approach and the relevant related ones, we have also performed tenfold cross-validation on a gold standard data set, although this is not aligned entirely to the application scenarios we have envisioned.

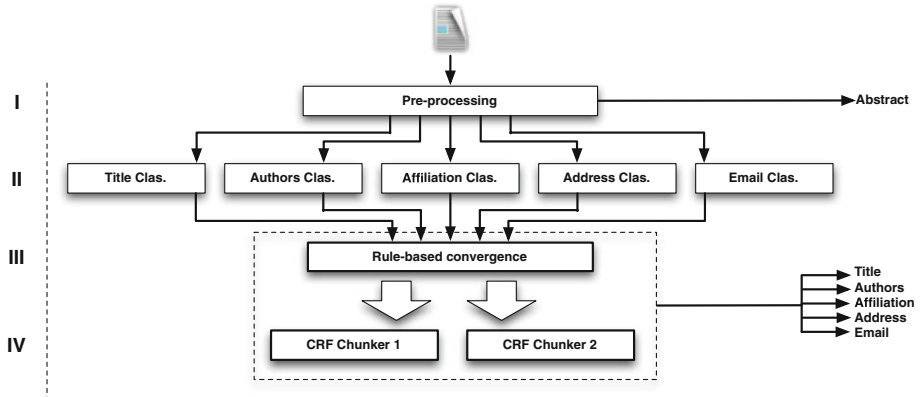

Fig. 4 Heading metadata extraction component structure

3.2 Heading metadata extraction

Figure 4 presents the configuration of the heading metadata extraction component. The raw textual input, that is, the first page of the publication (see the observation mentioned above), goes through a list of preprocessing steps (phase I) and is split into two parts: the first comprising the complete, cleaned and preprocessed input and a second part containing strictly the heading structure (cleaned as well). This second segment is delimited based on the Abstract block that is extracted in a specific preprocessing step before the line-oriented classification, as described in the following section. The two resulting segments are then used as input for the cascade of dedicated SVM classifiers (phase II). The merged line-oriented classification results create an enriched heading structure that for each line, has one of our five classes attached (i.e., *Title*, *Authors*, *Affiliation*, *Address* and *Email*). Subject to the classification, some lines will have multiple classes assigned, while others may have none. The final extraction result is decided in the last phase of the pipeline, where: (i) we apply a rule-based convergence procedure that takes into account classes assigned to the neighboring lines (phase III), and then (ii) we finalize the results by disambiguating the lines that have multiple classes assigned via a set of CRF chunkers (phase IV). The following sections detail each phase of the extraction, in the aforementioned order.

3.2.1 Preprocessing (I)

The preprocessing phase has two goals: (i) it aims to clean the raw textual input of unwanted spacing characters, while at the same time ensuring proper spacing where necessary, and (ii) it extracts the abstract of the publication, thus delimiting the two segments required by the actual pipeline execution.

Since the source of the textual input is unknown to the extraction pipeline, we make no assumptions with regard to its structure or content. 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 that contain less than 15 characters are merged with previous ones,
– we introduce spaces after some punctuation characters, such as “,”, “.” or “-”, and finally,

– we split into two camel-cased strings, such as JohnDoe.

As a note, for the first step, we chose the length of 15 characters empirically, based on the assumption that one cannot really express a real sentence in such a short text span.

In addition, as most of the features used throughout by the pipeline require some forms of vocabulary entries, we have compiled a list of gazetteers, explained as follows:

– *FirstName*—25,155 entries gazetteer of the most common first names (independent of the gender);
– *LastName*—48,378 entries list of the most common surnames;
– *Location*—places/cities/countries gazetteer comprising 17,336 entries;
– *Organization*—150 entries gazetteer listing organization prefixes and suffixes (e.g., *e.V.* or *KGaA*);
– *Address*—list of 98 tokens that signal an address (e.g., *street*);
– *Publisher*—564 entries gazetteer comprising publisher unigrams (produced from around 150 publisher names);
– *PubVenue*—structured gazetteer consisting of different variations for conference, journals and workshop terms (the actual terms, not instances of conference or workshop names);
– some auxiliary gazetteers: *Connection*—stop-word gazetteer; *Pages* and *Editors*—different variations of the tokens *pages*, and respectively, *editors*.

The second role of this phase is to extract the abstract of the publication (see Fig. 3, line 9–10). This is done using a series of regular expressions (e.g., searching for the *Abstract* token), combined with the placement of the abstract beginning in the page. If the abstract block is found, the step is repeated (using different regular expressions) to find the end of it or the beginning of the actual publication content. However, there are cases in which the beginning of the abstract block cannot be found via regular expressions. In such cases, we also apply jointly two of the features used by the SVM classifiers, detailed later in this section: the average inverse line frequency (ILF) and the average term frequency (TF). As we will show in the evaluation, this approach is enough to ensure high extraction performance, that is, 99.74% F_1, where F_1 [27] is the harmonic mean of precision and recall defined as:

$$F_1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}} \tag{4}$$

3.2.2 Line-oriented classification (II)

The line-oriented classification is realized via the cascade of five dedicated SVM classifiers, one for each of the target classes: *Title*, *Author*, *Affiliation*, *Address* and *Email*. More concretely, there is a standard (dual-class) *Title* classifier that classifies a line into *Title* or *NoTitle*, an *Author* classifier that performs the same task but focuses on authors, and so forth. The feature vector values are built from the two segments of cleaned initial input: the short segment comprising only the heading structure and the long segment comprising the entire input. We used two types of features for classification: (i) classifier-agnostic features—their values are computed using the same formula for all classifiers—and (ii) classifier-dependent features—discriminating features that take different values for different classifiers.

The use of an actual full page of the publication's content enables the derivation of a set of seven features, listed in Table 1 which guide (in particular with regard to *Title*) the classification. At the same time, we emulate sequencing to improve the overall accuracy, although the actual classification task is performed per line basis and considers only the small set of

Table 1 Classifier-agnostic "special" features

Feature	Value
Line pos. (short)	$\frac{i_S}{T_S}$, T_S—total number of lines in the short segment, and i_S – line number in the short segment
Line pos. (long)	$\frac{i_L}{T_L}$, T_L—total number of lines in the long segment, and i_L – line number in the long segment
Norm. line length (short)	$\frac{n}{N_S}$, n—number of tokens in the line, and N_S – total number of tokens in the short segment
Norm. line length (long)	$\frac{n}{N_L}$, N_L—total number of tokens in the long segment
Average TF (long)	$\frac{1}{n}\sum_{i=1}^{n}\frac{\text{TF}(token_i)}{N_L}$, $\text{TF}(token_i)$—term frequency of $token_i$
Average BF (long)	$\frac{1}{n}\sum_{i=1}^{n-1}\frac{\text{BF}(bigram_i)}{N_L-1}$, $\text{BF}(bigram_i)$—bigram frequency of $bigram_i$
Average ILF (long)	$\frac{1}{n}\sum_{i=1}^{n}\frac{1-\text{ILF}_i}{\max_{\text{ILF}}}$, where $\text{ILF}_i = \text{Log}_{10}\frac{T_L}{\text{TF}(token_i)}$

features used. This means that some of the dimensions of the feature vector of a certain line will contain dimensions inherited from a previous or succeeding line, thus, providing the above-mentioned features with an increased weight.

The set of 14 classifier-agnostic features (including the seven "special" ones) are listed in Tables 1 and 2. Each feature value is line-oriented and is calculated based on different aspects of the individual tokens present in the line. The list of tokens is obtained by splitting the line based on space characters, with n representing the length of the token list. This set is enriched with three additional classifier-dependent features (all listed in Table 3), bringing the feature vectors dimensions to a total of only 17 per line (under scrutiny).

The actual configuration of each classifier is listed below, including their specific features according to Table 3. We chose the type of kernel for the individual classifiers in an experimental manner (i.e., by running tests and tuning the different parameters until we achieved a satisfactory result). We were biased toward kernels that provided a higher recall. Details about the training and testing data sets are presented in Sect. 5, while background information about the two types of kernels we have used can be found in Sect. 2.

Title classifier: Polynomial kernel, 68 features per line (17 × 4 lines) including the current line (l), the previous line ($l-1$) and the following succeeding lines ($l+1$ and $l+2$), using the Address prob. (v1), Email prob. (v1) and Org. prob. (v1), in addition to Name prob. (v2) from Table 3.

Author classifier: RBF kernel, 68 features values per line (17 × 4 lines) including the current line (l), the previous line ($l-1$) and the following succeeding lines ($l+1$ and $l+2$), using the Address prob. (v1), Email prob. (v1) and Org. prob. (v1).

Affiliation classifier: RBF kernel, 51 features per line (17 × 3 lines) including the current line (l), the previous line ($l-1$) and the succeeding line ($l+1$), using the Address prob. (v2), Email prob. (v1) and Org. prob. (v2).

Table 2 Classifier-agnostic features

Feature	Value
Date prob.	$\frac{1}{n}\sum_{i=1}^{n} dateToken_i$, where $dateToken_i = 1$ if $token_i$ is a month, ordinal, or an year.
Editors prob.	$\frac{1}{n}\sum_{i=1}^{n} editToken_i$, where $editToken_i = 1$ if $token_i$ in *Editors*, *FirstName* or *LastName* gaz.
Number prob.	$\frac{1}{n}\sum_{i=1}^{n} numToken_i$, where $numToken_i = 1$ if $token_i$ is a number.
Pages prob.	$\frac{1}{n}\sum_{i=1}^{n} pagToken_i$, where $pagToken_i = 1$ if $token_i$ in *Pages* gaz.
Venue prob.	$\frac{1}{n}\sum_{i=1}^{n} pvToken_i$, where $pvToken_i = 1$ if $token_i$ in *Publisher*, *Location* or *PubVenue* gaz.
Web site prob.	$\frac{1}{n}\sum_{i=1}^{n} webToken_i$, where $webToken_i = 1$ if $token_i$ is a web site.
Name prob.	$\frac{\sum_{i=1}^{n}(fToken_i + lToken_i)}{2n^2} - \frac{1}{n}\sum_{i=1}^{n} addToken_i$, where $fToken_i = 1$ if $token_i$ in *FirstName*, $lToken_i = 1$ if $token_i$ in *LastName*. and $addToken_i = 1$ if $token_i$ in *Address* and *Location* gaz.,

Table 3 Classifier-dependent features

Feature	Value
Org. prob. (v1)	$\frac{1}{n}\sum_{i=1}^{n}(orgToken_i + locToken_i)$, where $orgToken_i = 1$ if $token_i$ in *Organization* and $locToken_i = 1$ if $token_i$ in *Location* gazetteers.
Org. prob. (v2)	$1.25 * \frac{1}{n}\sum_{i=1}^{n}(orgToken_i + locToken_i)$, if $\sum_{i}^{n} orgToken_i >$ 0, $\frac{1}{2.5n}\sum_{i=1}^{n} locToken_i$, otherwise
Address. prob. (v1)	$\frac{1}{n}\sum_{i=1}^{n}(addToken_i + locToken_i)$, where $addToken_i = 1$ if $token_i$ in *Address*
Address prob. (v2)	$1.5 * \frac{\sum_{i=1}^{n}(addToken_i + locToken_i + numToken_i)}{\sum_{i=1}^{n} orgToken_i}$, if $\sum_{i=1}^{n} orgToken_i >$ 0, $1.5 * \frac{1}{n}\sum_{i=1}^{n}(addToken_i + locToken_i + numToken_i)$, otherwise
Email. prob. (v1)	$\frac{1}{n}\sum_{i=1}^{n} emailToken_i$, $emailToken_i = 1$ if it contains an "@"
Email prob. (v2)	0.5 if the count of $emailToken$ is 1, otherwise, Email prob. (v1)
Name prob. (v2)	$\frac{1}{2n}\sum_{i=1}^{n}(fToken_i + lToken_i)$, with $fToken_i$ and $lToken_i$ as in Tab. 2.

Address classifier: Polynomial kernel, 51 features per line (17 × 3 lines) including the current line (l), the previous line ($l - 1$) and the succeeding line ($l + 1$), using the Address prob. (v2), Email prob. (v1) and Org. prob. (v2).

Email classifier: Polynomial kernel, 68 features per line (17 × 4 lines) including the current line (l), the previous line ($l - 1$) and the following succeeding lines ($l + 1$ and $l + 2$), using the Address prob. (v2), Email prob. (v2) and Org. prob. (v2).

As a final remark, in the cases in which the previous ($l - 1$) or the succeeding lines ($l + 1$ and $l + 2$) did not exist (e.g., first or last lines of the heading), we used an empty features line, having -1 as value on all vector dimensions.

Returning to our example depicted in Fig. 3, the result of performing the line-oriented classification is the following: (i) Line 3 will be assigned the *Title* tag, (ii) Line 4 will be assigned the *Author*, *Affiliation* and *Address* tags, (iii) Line 5 will be assigned the *Author* tag, (iv) Lines 6 and 7 will be assigned the *Affiliation* tag, and (v) Line 8 will be assigned the *Address* tag.

3.2.3 Rule-based convergence (III)

The individual testing of the SVM classifiers led us to a series of observations. For example, in some cases, for example, the *Authors*, *Address* or *Email* classifiers, we observed that we could have achieved a better individual performance (in terms of precision). However, we intentionally considered the configurations that provided a higher recall, at the expense of the precision, for the sake of obtaining better overall results. In order to achieve these results, we have introduced a rule-based convergence procedure and a disambiguation phase. More concretely, we first detect structural patterns in the heading (i.e., repeated sequences author—affiliation—address) and then apply iteratively to each detected block the following rules and actions:

- *Email* lines were rechecked for the presence of actual emails. If the lines are assigned multiple classes (e.g., *Authors* and *Email*, which is a quite common case), the email is the first of all extracted, and then the class is subsequently removed from the line. The line is then used in the other convergence steps.
- adjacent lines classified as *Authors* and *Affiliation* are merged and the authors vs. affiliation CRF is applied. For lines that were classified with both classes, the CRF chunker is applied, the *Authors* class is removed, and the line is processed further.
- adjacent lines classified as *Affiliation* and *Address* are merged and the affiliation vs. address CRF is applied. The same rule applies also for lines classified with both classes.
- lines that are classified both as *Title* and *Address* are considered only as *Title*.

In the case of a multiple class assignment involving the *Title* class, this last rule is in fact applied in a more general way, as we always give credit to the Title class over all the other classes (except for the Email class, covered by the first rule). This can, however, harm the classification results by affecting negatively the precision of Title classifier and the recall of classifiers associated with the rest of the assigned classes. The above procedure ends when all the detected blocks were analyzed.

3.2.4 Disambiguation chunkers (IV)

As mentioned earlier, there are cases in which some of the lines of the heading structure may be ambiguous—they may contain multiple metadata fields. Line 4 of the example presented in Fig. 3 represents such a case. To solve this issue, we trained two CRF chunkers, one for disambiguating names from affiliations (with or without addresses) and one for disambiguating affiliations from addresses. Applying the two chunkers, in serial, on Line 4 of the example (previously tagged with the *Author*, *Affiliation* and *Address* tags during the line-oriented classification), would result, first in splitting the name (*Gary T. Leavens*) from the rest of the text, and then in splitting the affiliation, *Department of Computer Science, Iowa State University* from the address *Ames, IA 50011 USA*.

Both chunkers were trained using the same set of features, but on different data sets. The authors-affiliation chunker was trained on 500 manually labeled, randomly gathered samples, while the affiliation-address chunker was trained on a data set consisting of 650 manually labeled, randomly chosen samples, both compiled from the generic classification training corpus (see Sect. 5). To test their efficiency, we ran an experimental evaluation, using a 200 sample and, respectively, a 400 sample data sets. These data sets are different from the ones used in training, but are derived from the same training corpus. Considering that the chunking task was simple, revolving only around finding the borderline between the two targeted categories, we achieved a 100% F_1 score for both chunkers. To clarify, as opposed to [19], we apply the two chunkers for disambiguation purposes only (i.e., as integrated components of the extraction pipeline). Therefore, their individual (stand-alone) value is irrelevant and consequently, presents no importance for specific evaluation.

Below, we list the features used for the CRF chunkers and exemplify them using the first token of Line 4 from Fig. 3. As a remark, a set of feature values is used to characterize each token present in a string (in this case the entire line), where the token list is obtained by dividing the string into space-separated pieces.

Original Token: *Gary*
Stripped Token: the original token, stripped of any punctuation and lower cased: *gary*
Punctuation: trailing punctuation character (values: *cont* / *stop*/*other*): *other*
Postfix (last character type): (possible values: lower cap—*c*, upper cap—*C*, digit—*0* or an actual punctuation character): *c*
Orthographic case: (values: *initialCap*/*singleCap* / *lowercase*/*mixedCaps*/*allCaps*): *singleCap*
Token description: 10 individual values, 5 corresponding to the first 5 characters of the token and 5 to the last 5 characters: *G Ga Gar Gary Gary y ry ary Gary Gary*
Number: (possible values: *year*, *ordinal*, *1dig*, *2dig*, *3dig*, *4dig*, *4dig+*, *noNumber*): *noNumber*
Gazetteer entries: 6 flags marking the presence of the token in 6 different gazetteers (listed in Sect. 3.2.1): the *FirstName*, *LastName*, *Location*, *Organization*, *Address* and *Connection* gazetteers. In the case of our example, the value of this feature would be: *FirstName no Location no no no*.
Position: token position in the string, with the string divided into 9 equal parts. This division into 9 equal parts is the result of a series of trials that had the goal of clustering as close as possible together the tokens that have the same type, for example, title or authors: *0*

This last phase completes the pipeline and produces the final metadata fields.

Listing 1 Mappings from reference fields to classes and properties

```
"author"       : {
  "name"       : ["http://xmlns.com/foaf/0.1/name"]
  "type"       : ["http://xmlns.com/foaf/0.1/Person", http://xmlns.com/foaf
       /0.1/Agent"]
}

"publication" : {
  "title"      : ["http://purl.org/dc/elements/1.1/title", "http://swrc.
       ontoware.org/ontology#title"]
  "type"       : ["http://swrc.ontoware.org/ontology#InProceedings"]
  "authors"    : ["http://purl.org/dc/elements/1.1/creator"]
  "pubVenue"   : ["http://purl.org/dc/terms/partOf", "http://data.semanticweb.
       org/ns/swc/ontology#isPartOf"]
}

"venue" : {
  "title"      : ["http://swrc.ontoware.org/ontology#booktitle"]
  "type"       : ["http://swrc.ontoware.org/ontology#Proceedings"]
  "pubList"    : ["http://data.semanticweb.org/ns/swc/ontology#hasPart"]
}
```

4 Metadata refinement and linking

The result of the extraction process is a set of metadata that may contain errors, such as missing parts of title or incomplete author names. However, as a possible application scenario would be the automatic creation of linked publication data sets, we require an additional step that aims to clean (or refine) the extracted metadata and link it as part of the Linked Data Web.

The Linked Data Web contains a large number of data sets, accessible via different methods. Some are exposed via SPARQL endpoints, and others provide specific RESTful APIs. In order to cater for this variety of methods, we designed a flexible entity co-reference resolution and linking process. Using a simple JSON-formatted configuration file, one can "plug-in" different query modules that will act as access points to the different data sets. For each component, one needs to specify its type and its voiD [1] description. The type signals a specific way of handling the Linked Data, while the voiD description provides information about the actual entry point to the data set, in addition to the classes and properties exposed by the respective data set. As we will see, this information is particularly important when mapping the extracted metadata fields to the queried linked resources.

The entity resolution and linking process uses an arbitrary number of SPARQL query modules. These enable dynamic querying of any data set exposed as a SPARQL endpoint that has an associated voiD description. The challenge here is to bridge the gap between the raw text resulted from the metadata extraction process and the SPARQL queries required for accessing the SPARQL endpoint. We have addressed this challenge by using a series of mappings and a transformation stage.

The mappings from the metadata fields to ontological resources (classes or properties) are done based on a JSON-formatted description. At this point, the process supports three types of mappings:

- author-related mappings—with the *name* and *type* properties
- publication-related mappings—with the *title*, *type*, *authors* and *pubVenue* properties, and
- publication venue-related mappings—with the *title* and *pubList* properties.

Listing 1 presents an example of a set of mappings that can be used as input for the process.

Listing 2 SPARQL query generated from the given example

```
SELECT DISTINCT ?x ?title WHERE
  {
    ?x rdf:type <http://swrc.ontoware.org/ontology#InProceedings> ;
       <http://swrc.ontoware.org/ontology#title> ?title
       FILTER regex(?title, "Recipes for Semantic Web Dog Food - The ESWC*",
            "si") .
  }
```

The transformation stage takes as input a string (metadata field) produced by the extraction pipeline and, subject to its field type, transforms it into a SPARQL query, for a specific SPARQL endpoint (identified via the voiD description). Consequently, if we aim at querying for publications, by using the publication title, the result will be a SPARQL publication query. Similarly, if we query for authors, we use the author names present in the reference and construct a SPARQL author query.

To have a better understanding of this transformation phase, we provide the following example: (i) we want to execute a publication query; (ii) on the Semantic Web Dog Food Server (SWDFS) using our SPARQL query component; (iii) considering an incorrectly extracted publication title, for example, *Recipes for Semantic Web Dog Food—The ESWC*. The correct and complete title of the publication is: *Recipes for Semantic Web Dog Food—The ESWC and ISWC Metadata Projects*. An error such as this one could appear due to an unidentified *Title* line.

This particular instance of the SPARQL query component has in its associated voiD description, corresponding to the SWDFS, among other properties, also:

sparqlEndpoint: `http://data.semanticweb.org/sparql`
hasInstancesOf:

- `http://swrc.ontoware.org/ontology#InProceedings`
- `http://xmlns.com/foaf/0.1/Person`

hasProperty:

- `http://swrc.ontoware.org/ontology#title`
- `http://purl.org/dc/elements/1.1/creator`

Taking into account all of the above, the transformation step will use the provided publication mappings and will construct and execute (using the given SPARQL endpoint) the SPARQL query presented in Listing 2. Querying for authors works in exactly the same manner, but this the corresponding author mappings in place.

The actual Linked Data-based entity resolution and linking mechanism is straightforward. For each metadata entry resulted from the previous step, one can use this linking process to query for the extracted publication title. Where possible, prior to querying, the title is transformed into a regular expression, to avoid fixed queries that might give empty results due to the errors of the extraction process. The query result will be a list of linked resources that may contain also the desired publication.

To mask possibly existing discrepancies in the title, we use string similarity measures. An empirical analysis led us to using a combination of the Monge–Elkan [17] and Soundex [18] algorithms, with fixed thresholds. The first one analyzes fine-grained sub-string details, while the second looks at coarse-grained phonetic aspects. The titles that pass the imposed thresholds (0.75 and 0.9, respectively) advance to the next step. Subsequently, we consider the

initially extracted author names and compare them with the ones associated with the publications that pass over the above mentioned thresholds. The comparison is done using the same similarity measures, but with different thresholds (0.85 and 0.95). The linked publication satisfying both conditions can be used as candidate to clean the originally extracted metadata fields. Subject to the amount of information existent in the publication's graph, one can do more than just correcting the title and authors, it can also enrich the extracted metadata with, for example, the publication venue.

In the case in which the title-based querying fails, the linking step can be repeated by querying for the extracted author names. The filtering process is the same, the difference being in the result, as eventually one might correct only the individual author names and not the entire publication.

5 Evaluation

As mentioned in Sect. 1, we aim at deploying (and using) the extraction pipeline in real-world settings. Therefore, in addition to a high accuracy, the pipeline should also exhibit an increased versatility. This, unfortunately, cannot be achieved by performing data set-based cross-validation, as the result will always be a data set-tailored model that encapsulates strictly the characteristics of the data set under scrutiny. Consequently, the resulting model will not have a uniform accuracy if applied on a data set with slightly different characteristics. In order to deal with these aspects, we followed a less common approach to training the extraction pipeline (or more specifically the SVM classifiers), by using two exclusive data sets for training and testing. However, for correctness purposes, and to create the proper environment for comparing our approach to the already existing ones, we have also performed tenfold cross-validation [13] on a gold standard data set. As a note, tenfold cross-validation represents the iterative evaluation process of randomly splitting a data set into ten equal disjoint parts, and using nine parts for training and one part for validation. Each iteration captures the local validation results, with the final validation result representing an average of the intermediary ones.

Before detailing the actual experimental results, we describe the generic training data set used to achieve an increased versatility and an uniform effectiveness over multiple test data sets. This comprised 650 publications, randomly chosen from different conferences and from different publication repositories (e.g., Springer Link or IEEE Explorer). As mentioned, this training corpus was used exclusively for training the SVM classifiers that, at a later stage, were tested based on the test corpora described in the rest of this section. To provide the training examples, we labeled the heading structure of each publication manually. More specifically, the labeling process was performed by a single person (one of the authors), by going manually through each training entry. In a typical ambiguous setting (e.g., word sense disambiguation), the labeling should have been done by multiple participants and the data reliability should have been shown via inter-annotator agreements. However, the heading metadata use-case has no real ambiguity attached, and consequently we did not see the need for multiple labeling. At the same time, the evaluation results on the CORA data set [26] (available as an already tagged data set) prove to some extent the reliability of our labeled training data.

We tested the extraction pipeline in a series of experiments on three different data sets:

– the CORA data set, both via a tenfold cross-validation on a 53–47% split (500–435) as in the case of the other approaches, and by using the generic training corpus,

Listing 3 Evaluation results on the CORA data set—other approaches

Field	Peng (CRF)	Peng (HMM)	Han (clusters)	Han (SVM)		
	F_1	F_1	F_1	P	R	F_1
Title	97.10	82.20	92.20	93.90	95.00	94.50
Authors	**97.50**	81.00	92.30	97.30	91.40	94.20
Affiliation	97.00	85.10	91.60	96.40	90.30	93.30
Address	95.80	84.80	92.30	93.60	86.70	90.00
Email	95.30	92.50	98.10	98.90	94.00	96.40
Abstract	99.70	98.00	97.5	98.50	99.20	98.80

Listing 4 Evaluation results on the CORA data set—our approach

Field	Our approach (generic training)			Our approach (tenfold x-validation)		
	P	R	F_1	P	R	F_1
Title	97.77	98.42	**98.09**	97.91	98.94	**98.42**
Authors	97.78	97.15	97.46	97.92	91.31	94.50
Affiliation	99.36	98.10	**98.72**	99.02	96.99	**97.99**
Address	98.67	98.12	**98.39**	97.55	96.37	**96.96**
Email	98.96	99.30	**99.13**	100	97.51	**98.74**
Abstract	100	99.5	**99.74**	100	99.5	**99.74**

– a data set containing 201 randomly selected publications from the ACM Portal, IEEE Explorer and Springer Link, and
– a data set comprising 200 PUBMED [21] publications.

As evaluation metric, we used the F_1 score, as defined in Eq. (4) on p 8. Below, we present the evaluation results, in the data set order listed above.

5.1 Data set: CORA

The CORA data set is the first gold standard created for the evaluation of heading metadata extraction. It comprises 935 publication headers and focuses on the Computer Science area. The headers were compiled from different types of publications, that is, conference/workshop/journal publications and technical reports. Each entry is chunked into thirteen fields: *Title*, *Author*, *Affiliation*, *Address*, *Note*, *Email*, *Date*, *Abstract*, *Phone*, *Keyword*, *Web*, *Degree* and *Pubnum*.

Table 3 presents the extraction performance achieved by the relevant related approaches (on the fields we have focused on), while Table 4 shows the results we have achieved, both via tenfold cross-validation and by using the generic training corpus. The results in both tables are represented according to the precision (P), recall (R) and F_1. The best results, according to the F_1 score, are presented in bold.

The cross-validation test shows similar results to Peng (CRF), being slightly better on all fields, except for the *Authors* field where the performance decrease is of 3%. This decrease is, in principle, due to the combination of features used in the author extraction, which was built to handle as many cases as possible, as shown also by the second test where the approaches achieve almost the same performance. On the other hand, we can observe a consistently better

Listing 5 Evaluation results on the Random and PUBMED data sets

Field	Random data set			PUBMED data set		
	P	*R*	F_1	*P*	*R*	F_1
Title	99.30	99.30	**99.30**	99.50	100	**99.74**
Authors	99.66	99.34	**99.49**	99.00	100	**99.49**
Affiliation	99.35	98.08	**98.71**	100	97.97	**98.97**
Address	99.05	98.12	**98.58**	99.00	100	**99.49**
Email	99.23	99.61	**99.41**	100	95.23	**97.55**
Abstract	98.75	98.75	**98.75**	100	100	**100**

performance on all fields when compared to Han's approach (which is, from the technical perspective, the closest to ours). However, our results were achieved using a set of features vectors with dimensions lower with two orders of magnitude than the SVM approach of [11].

The second test, using the generic training corpus, shows an even better, yet consistent, performance. While on the *Title*, *Affiliation* and *Abstract* fields, the performance increase is of around 1%, the *Address* and *Email* fields bring an improvement of around 4%. However, on the *Authors* field, the pipeline is still outperformed, this time by only 0.04%. Subject to the considered fields (and related approach), instead of the improvement in percentage, we can look at the decrease in the remaining error. Consequently, an effectiveness increase from 95 to 99% can be seen as a 80% decrease in remaining error from 5 to 1%.

5.2 Data set: random & PubMed

To show the versatility of the extraction pipeline, we compiled two other data sets with slightly different characteristics: (i) a data set comprising 201 PDF publications randomly selected from the ACM Portal, IEEE Explorer and Springer Link (not restricted to the Computer Science field), and (ii) a data set consisting of 200 PUBMED publication abstracts. Both data sets are available at http://resources.smile.deri.ie/corpora.

The publications in the first data set have different formatting styles, dictated by the corresponding publishing conferences (or journals). To keep the experiments as close as possible to a real-world scenario, the extraction pipeline was applied on the raw textual counterpart of the first page of each publication, as extracted by PDFBox [22]. Consequently, we maintain any extraction errors that appear as a result of applying our solution on any random PDF publication. The same experiment can also be performed on publications written in MS Word, since the extraction is independent of the underlying document format.

The PUBMED data set, on the other hand, contains the information provided openly by the PubMed repository. Each publication is presented in a simple textual format and contains the heading metadata and the abstract. We chose to experiment of such a data set to show that our approach is indeed document format agnostic, and the minimum information required to achieve the best results is the presence of the publication's abstract.

The evaluation results for both data sets, using the generic training corpus, are presented in Table 5. It can be observed that the extraction pipeline performs consistently, as on the CORA data set, in some cases achieving an improved performance (e.g., the *Authors* field, with an F_1 score of 99.49% on both data sets). At the same time, the extraction has very good results when applied both on full publications (i.e., on the first page) and on publications where only the abstract is present.

6 Related work

The work described in this paper has relevance, in principle, in two complementing research areas: (i) bibliographic metadata extraction and (ii) entity co-reference resolution. In the following, we detail and analyze the related work in both topics.

Several methods exist for automatic extraction of heading (bibliographic) metadata from scientific publications, such as regular expressions, rule-based parsers or Machine Learning. With respect to the two trends, regular expressions and rule-based systems have the advantage that they do not require any training. However, they depend on the application domain and require the presence of an expert to manually craft the rules or regular expressions. On the other hand, ML methods are robust and adaptable and, theoretically, can be used on any document format. Their main disadvantage is the rather expensive price to be paid for labeling or annotating training data. ML techniques used, so far, for bibliographic metadata extraction have included hidden Markov models (HMM), support vector machines (SVM) and conditional random fields (CRF).

One of the first approaches to perform automatic metadata extraction from scientific publications was the one of [26]. They explored the use of HMMs for the extraction task, specifically focusing on learning the model structure from data and maximizing the use of labeled and unlabeled data. A side result of their pioneering work was the creation of the first gold standard data set for bibliographic metadata, the CORA data set. At a later stage, the same group employed CRFs for the extraction task. In [19], they perform an empirical exploration of several CRF factors, such as variations on Gaussian or exponential and hyperbolic-L_1 priors for improved regularization. As opposed to their approach, we use CRF strictly for disambiguation and chunking purposes, in two particular cases: authors vs. affiliation and affiliation vs. address, and not with the goal of chunking the entire heading metadata.

The most closely related approach is the one of [11]. The authors trained a line-oriented multi-class SVM and then performed an iterative convergence procedure to improve the line classification by using the predicted class labels of the neighboring lines. There are two similarities between our approach and the one of Han: (i) we, as well, use line-oriented SVM classification, and (ii) we also perform an iterative convergence procedure, by discriminating the final line class based on the neighboring lines classes and a set of rules. However, our approach is novel with respect to the structure of the extraction pipeline, where we opted to cascade a set of line-specific, class focused, standard (dual-class) SVM classifiers, as opposed to a single multi-class classifier. More concretely, instead of one classifier, we connect five specific classifiers, each focused on a particular metadata field. This enable us to reduce the feature vector dimensions to a handful, thus making the overall model much easier to grasp and manage. Finally, the novelty of our solution comes also from improving the overall extraction performance by mixing the SVM classification with CRF chunking (strictly as a disambiguation mechanism) and a simple set of rules that guide the final classification results subject to particular cases.

Similar to [19], the authors attempted a different learning scheme. In [12], they propose an effective word clustering approach with the goal of reducing feature dimensionality and sparseness, while at the same time improving the overall classification performance. The resulted domain rule-based word clustering method for cluster feature representation used clusters formed from various domain databases and word orthographic properties. The experiments results were, however, less satisfactory than in the SVM case.

If the above mentioned methods consider only the textual content of the publication for processing, the non-ML approaches take into account also the content's environment, that is, the document format properties. Successful work has been reported in automatic metadata

extraction from HTML documents using natural language processing methods in [32], from Postscript documents using rules in [8] or from PDF documents via a visual/spatial approach in [9]. Unfortunately, they are restricted to the specific document format they are built for. Thus, they are not directly comparable to the ML-powered approaches, since they are document format agnostic.

In addition to the previously described works, which were specifically tailored for bibliographic metadata extraction, there are a series of other possibly relevant approaches, developed in different areas, that could be used for the same purpose. For example, [6] propose a innovative "recursive boosting" strategy, with progressive classification, to reconcile textual elements to an existing attribute schema. In the case of bibliographic metadata segmentation, the metadata fields would correspond to the textual elements, while an ontology describing them (e.g., DublinCore or SWRC [28]) would have the schema role. The authors even describe an evaluation of the method using the DBLP citation data set, however, without giving precise details on the fields considered for segmentation. Some other approaches include, in general, any sequence labeling techniques, for example, SLF [23], named entity recognition techniques, for example, [24], or even the newly emerging area of field association (FA) terms extraction [7], the latter working on bibliographic metadata fields in a quasi-similar manner as the "recursive boosting" strategy.

The second relevant area for our research is entity co-reference resolution. As described in Sect. 4, the mechanism we propose with respect to this topic is straightforward. It considers the mapping between the corresponding metadata field and the Linked Data resource, by following a strictly localized (micro-perspective) direction (i.e., it takes into account only the information attached to that particular field). This is, naturally, less expressive than most of the existing approaches which look at the entity resolution mechanism from a macro perspective, that is, by considering the entire information space available for analysis. For example, [10] follow a generative model-based approach which considers the syntactic, semantic and discourse constraints, attached to this process, as modules learned in an unsupervised manner. A different direction is proposed by [25] which takes into account also the possible relations between the entities to be consolidated via random walks models. Wick et al. [31] focused on entities, rather than their mentions in the textual input, and developed a discriminatively trained model to perform co-reference resolution, jointly with canonicalization (i.e., inclusion of explicit canonical values for the entities attributes, such as name, venue, title). Such an approach could be used as a foundational building block in the vision of the OKKAM project that aims at supporting the reuse of existing URIs for any type of entity [5], by providing a centralized Entity Name System (ENS).

Finally, an approach that does not fall into any of the previously described areas, however, which is worth mentioning, is Silk [30]. Silk is a framework for discovering and maintaining links between various data sets on the Web. The discovery step described by the Silk Link Specification Language (Silk LSL) requires that the datasources are accessible via SPARQL endpoints. The actual linking is done using a manually tailored specification file, which contains the mapping between different types of entities present in the data sets under scrutiny and the values to be used as thresholds for matching. To a certain extent, Silk can be seen as a more generic version of our linking method and applied at data set level, rather than metadata field level.

7 Conclusions

In this article, we presented a comprehensive approach for extracting metadata from scientific publications. Our metadata extraction pipeline consists of: (i) a cascade of dedicated SVM

classifiers, (ii) an iterative rule-based convergence procedure and (iii) two CRF disambiguation chunkers, for extracting the heading metadata. As shown, this structure enables the use of a very small set of features for classification, while at the same time, improving the current state of the art extraction effectiveness. The good results achieved in the evaluation, in addition to the flexibility of the linking approach, enables a straightforward adoption and reuse of our solution in any digital library or publication repository.

For future developments, we plan to release openly the flexible Linked data set generator we have implemented, using the extraction mechanisms described so far, that supports our goals of enhancing the extraction and acquisition processes with the scope of producing Linked Data. Also, to bring the Linked Data "feeling" into repositories that currently do not expose their data in a semantic manner, we plan to develop a Linked Data-powered overlay interface. The interface will be applicable on the ACM Portal or IEEE Explorer and will use the extraction pipeline and the linking process from the data set generator to enable a rich browsing experience for all users.

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

References

1. Alexander K, Cyganiak R, Hausenblas M, Zhao J (2011) Describing linked datasets with the VoID vocabulary. W3C Interest Group Note. http://www.w3.org/TR/void/ (March 2011)
2. Barabasi AL, Jeong H, Neda Z, Ravasz E, Schubert A, Vicsek T (2002) Evolution of the social network of scientific collaborations. Physica A Stat Mech Appl 311(3–4):590–614
3. Berners-Lee T, Hendler J, Lassila O (2001) The semantic web. Sci Am 284(5):35–43
4. Bizer C, Heath T, Berners-Lee T (2009) Linked data—the story so far. Int J Semant Web Inf Syst 5(3): 1–22
5. Bouquet P, Stoermer H, Niederee C, Mana A (2008) Entity name system: the backbone of an open and scalable web of data. In: Proceedings of the IEEE international conference on semantic computing (ICSC 2008), IEEE Computer Society, pp 554–561
6. Cesario E, Folino F, Locane A, Manco G, Ortale R (2008) Boosting text segmentation via progressive classification. Knowl Inf Syst 15(3):285–320
7. Dorji TC, sayed Atlam E, Yata S, Fuketa M, Morita K, ichi Aoe J (2010) Extraction, selection and ranking of Field Association (FA) Terms from domain-specific corpora for building a comprehensive FA terms dictionary. Knowl Inf Syst 27(1):141–161
8. Giuffrida G, Shek EC, Yang J (2000) Knowledge-based metadata extraction from Postscript files. In: JCDL '05, San Antonio, Texas, USA, pp 77–84
9. Groza T, Handschuh S, Hulpus I (2009) A document engineering approach to automatic extraction of shallow metadata from scientific publications. Tech. Rep. 2009–06–01. Digital Enterprise Research Institute
10. Haghighi A, Klein D (2010) Coreference resolution in a modular, entity-centered model. In: Proceedings of the 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics
11. Han H, Giles CL, Manavoglu E, Zha H, Zhang Z, Fox EA (2003) Automatic document metadata extraction using support vector machines. In: JCDL '03, Houston, pp 37–48
12. Han H, Manavoglu E, Zha H, Tsioutsiouliklis K, Giles CL, Zhang X (2005) Rule-based word clustering for document metadata extraction. In: Proceedings of the 2005 ACM symposium on applied computing, Santa Fe, New Mexico
13. Kohavi R (1995) A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Proceedings of the fourteenth international joint conference on artificial intelligence, pp 1137–1143
14. Lafferty JD, McCallum A, Pereira FCN (2001) Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: ICML'01, Francisco, pp 282–289
15. Liu X, Bollen J, Nelson ML, de Sompel HV (2005) Co-authorship networks in the digital library research community. Inf Process Manage 41(6):1462–1480

16. Möller K, Heath T, Handschuh S, Domingue J (2007) Recipes for semantic web dog food—The ESWC and ISWC metadata projects. In: Proceedings of ISWC 2007, Busan, Korea
17. Monge A, Elkan C (1997) An efficient domain-independent algorithm for detecting approximately duplicate database records. In: Proceedings of the SIGMOD workshop on data mining and knowledge discovery
18. National Archives and Records Administration (2007) The soundex indexing system. Technical report, May 2007
19. Peng F, McCallum A (2006) Accurate information extraction from research papers using conditional random fields. Inf Process Manage Int J 42(4):963–979
20. Peng W, Li T (2011) Temporal relation co-clustering on directional social network and author-topic evolution. Knowl Inf Syst 26(3):467–486
21. PubMed http://www.ncbi.nlm.nih.gov/pubmed
22. PDFBox http://pdfbox.apache.org/
23. Qi Y, Kuksa P, Collobert R, Sadamasa K, Kavukcuoglu K, Weston J (2009) Semi-supervised sequence labeling with self-learned features. In: Proceedings of IEEE international conference on data mining (ICDM)
24. Sanchez D, Isern D, Millan M (2010) Content annotation for the semantic web: an automatic web-based approach. Knowl Inf Syst 27(3):393–418
25. Sandler T, Ungar LH, Crammer K (2009) Resolving identity uncertainty with learned random walks. In: Proceedings of IEEE international conference on data mining (ICDM)
26. Seymore K, McCallum A, Rosenfeld R (1999) Learning hidden Markov model structure for information extraction. In: Proceedings of the AAAI'99 workshop on machine learning for information extraction, pp 37–42
27. Shaw WM, Burgin R, Howell P (1997) Performance standards and evaluations in IR test collections: cluster-based retrieval models. Inf Process Manage 33(1):1–14
28. Sure Y, Bloehdorn S, Haase P, Hartmann J, Oberle D (2005) The SWRC ontology—semantic web for research communities. In: Proceedings of the 12th Portuguese conference on artificial intelligence (EPIA 2005), Covilha, Portugal
29. Vapnik V (1995) The nature of statistical learning theory. Springer, Berlin
30. Volz J, Bizer C, Gaedke M, Kobilarov G (2009) Discovering and Maintaining Links on the Web of Data. In: Proceedings of the international semantic web conference (ISWC 2009)
31. Wick M, Culotta A, Rohanimanesh K, McCallum A (2009) An entity based model for coreference resolution. In: Proceedings of the nineth SIAM international conference on data mining, pp 365–377
32. Yilmazel O, Finneran CM, Liddy ED (2004) Metaextract: an nlp system to automatically assign metadata. In: JCDL '04, pp 241–242

Author Biographies

Tudor Groza is a Postdoctoral Research Fellow in the e-Research group at the University of Queensland, Australia. Previous to joining UQ, Tudor did his PhD studies at the Digital Enterprise Research Institute (DERI) Galway, National University of Ireland. His PhD topic was centered around modeling and extracting rhetorical and argumentation aspects, as well as shallow metadata from scientific publications with the goal of exposing them as Linked Data. Tudor is the author of twenty some publications focused mostly on using Semantic Web technologies in the scientific publishing area or for personal information management.

Gunnar AAstrand Grimnes received his Bachelor Degree from the University of Aberdeen, Scotland, where he also completed his PhD titled "A goal-directed learning agent for the Semantic Web". From 2005, he works at DFKI in Kaiserslautern, Germany, where he has been involved in a range of EU funded projects, including the FP6 NEPOMUK project where he served as technical director. NEPOMUK was concerned with specifying and implementing the Social Semantic Desktop. From 2011 onwards, he is the technical director of the iGreen project, working on bridging public and private knowledge management in the agricultural sector.

Siegfried Handschuh is a Senior Lecturer at the National University of Ireland, Galway (NUIG) and leader of the Semantic Collaboration research stream, as well as of the Semantic Information System and Language Engineering Group (SmILE), at the Digital Enterprise Research Institute (DERI). Siegfried holds Honours Degrees in both Computer Science and Information Science and a PhD from the University of Karlsruhe. He published over 100 papers as books and journal, book chapters, conference and workshop contributions, mainly in the areas of Annotation and Authoring for the Semantic Web, Knowledge Acquisition, Information Visualization and Social Semantic Collaboration. His publication record includes prestigious journal and conferences such as Computer Networks, IEEE Transactions, Journal of Web Semantics, ACM SIGIR, WWW, ISWC and ESWC. He is well cited, with an h-index over 20. Since 2000 he has initiated, participated and/or coordinated several R&D projects at an international level, such as DAML (US DARPA), aceMedia (EU IP), HALO 2 (Vulcan Inc.), Knowledge Web (EU NOE) and Nepomuk (EU IP), FAST (EU STREP), NET2 (EU IRSES) and Digital.me (EU STREP).

Stefan Decker is a professor at the National University of Ireland, Galway, and director of the Digital Enterprise Research Institute. Previously he worked at ISI, University of Southern California (2 years, Research Assistant Professor and Computer Scientist), Stanford University, Computer Science Department (Database Group) (3 Years, PostDoc and Research Associate), and Institute AIFB, University of Karlsruhe (4 years, PhD Student and Junior Researcher). He is one of the most widely cited Semantic Web scientists, and his current research interests include semantics in collaborative systems, Web 2.0, and distributed systems.