An End-to-end Approach for Extracting and Segmenting High-Variance References from PDF Documents

Zeyd Boukhers Shriharsh Ambhore and Steffen Staab
Institute for Web Science and Technologies (WeST)
University of Koblenz-Landau
Koblenz, Germany
{boukhers,ashriharsh,staab}@uni-koblenz.de

ABSTRACT

This paper addresses the problem of extracting and segmenting references from PDF documents. The novelty of the presented approach lies in its capability to discover highly varying references mainly in terms of content, length and location in the document. Unlike existing works, the proposed method does not follow the classical pipeline that consists of sequential phases. It rather learns the different characteristics of references to be used in a coherent scheme that reduces the error accumulation by following a probabilistic approach. Contrary to conventional references, mentioning the sources of information in some publications, such as those of social science, is not subject to the same specifications such as being located in a unique reference section. Therefore, the proposed method aims to extract references of highly varying reference characteristics by relaxing the restrictions of existing methods. Additionally, we present in this paper a new challenging dataset of annotated references in German social science publications. The main purpose of this work is to serve the indexation of missing references by extracting them from challenging publications such as those of German social science. The effectiveness of the presented methods in terms of both extraction and segmentation is evaluated on different datasets, including the German social science set.

CCS CONCEPTS

• **Computing methodologies** → *Information extraction*; *Supervised learning by classification.*

KEYWORDS

Reference Extraction, Reference Segmentation, Conditional Random Fields, Random Forest

1 INTRODUCTION

Acknowledging the scientific contribution of previous research work is necessary to ensure a smooth evolution in scientific fields. Over time, authors have adopted a manner to indicate the sources and the contributions of their fellows by mentioning their names, titles of their publications, etc. Several reasons, among which literature search and recommendation, necessitates making these references available and linked to their citations in a network. Therefore, different techniques have been developed to automatically detect, extract and segment these references[13, 16, 19].

In general cases, reference extraction goes through three steps: 1) *section identification*, 2) *reference extraction* and 3) *reference segmentation*. Section identification is the process of recognizing the section containing all and only references, where reference extraction is dedicated to extracting individual reference strings from the formerly identified reference section. Reference segmentation is used afterwards to segment these references into components (e.g. author, title, volume, etc.). To the extent of our knowledge, researchers have addressed these problems separately by using probabilistic approaches, mainly Hidden Markov Model (HMM) [9] and Conditional Random Fields (CRF) [15] since the processes satisfy the Markov property. Due to the aperiodicity of some states for both extraction and segmentation tasks, CRF has been widely and could achieve satisfactory accuracy. More specifically, references are assumed to appear in one section, while each reference string has a unique title, a unique page range and a unique source. Also, Neural Networks are also employed to segment reference strings [16], by replacing the words with their numerical representation as an input of the CRF model. Considering the correlation between the two steps, the result of segmentation highly depends on the result of the extraction. In this regard, any deficiency in the extraction's performance will certainly influence the entire process in a negative way.

Although the citation style is relatively standard in terms of reference content, the citation practice differs from a community to another. Some disciplines, like German social sciences or humanities, commonly use word processors without tool support to represent references, leading to a large variety of reference characteristics within and between publications. Variations include the locations of references in footnotes, endnotes or in specific sections as well as the manner in which reference components are delineated from each other. Basically, this variation has different facets, making the automatic recognition of references more challenging. Regarding reference extraction, the main variation is the section in which references are located. Contrary to the standard practice, where there is a single section containing references, it is very common in some communities that the references are placed in separated parts of the publication such as footnotes and endnotes. Here, it is difficult to adapt it to the problem of estimating a sequence of hidden variables. Another facet of this variety is the way of listing the references, which can be either in numerical order, by the full first author's name and the year or only by the first few letters of the name.

Furthermore, there are other varieties within the references being cited in one publication, where references are not composed of the same number of components. For example, social scientists often cite grey literature publications, which do not have author names or page range. Consequently, the order of components changes from

a reference to another depending on the existence of these components. This leads to making the number of possible transitions higher with fewer examples and hence an imprecise estimation for small datasets.

In order to overcome the above-mentioned issues, this paper proposes a robust approach to extract and segment difficult references, with a case study of German social science publications. For this, each line in the publication is classified, using Random Forest, into a reference or not. Each classified line is associated with classification scores to all classes (non-, first, intermediate or last reference line). Next, CRF is applied on potentially reference lines, with the support of a probabilistic approach, inspired from Metropolis-Hasting (MH), to form complete segmented reference strings. A feedback mechanism is adopted to reduce the accumulation of error among steps, where the approach does not independently rely on each model. Instead, it incorporates the trained models under a probabilistic approach to enhance the results of all models. Moreover, we provide an online tool that is implemented based on the developed approach[1].

The remainder of this paper is structured as follows: Section 2 discusses the related and prior work. Section 3 introduces the proposed approach starting with feature extraction, the core of the method and filtering process. Section 4 describes the performed experiments and reports the study results of different evaluations. Finally, Section 5 draws conclusions and communicate findings of the benefit of the proposed method.

2 RELATED WORK

A prior literature review demonstrates that much more effort has been devoted to developing techniques for reference segmentation than for reference extraction. This section discusses the novelties of our method by elaborating on prior work about both reference extraction and reference segmentation.

2.1 Reference Extraction

Reference extraction refers to the task of recognizing a section containing reference strings and identifying them afterward. In many disciplines like computer science or mathematics, the reference section is most often clearly delineated from the main text and labeled with a title like 'References' or 'Literature'. Based on this assumption, most prior work recognizes the beginning and end of such a reference section using rule-based or machine learning-based techniques.

Zou *et al.* [22] employ Support Vector Machine (SVM) to locate the reference section in HTML medical articles. They extracted geometric and text features from paper zones and combine them to distinguish between sections. Due to its efficiency, Tkaczyk *et al.* [19] use SVM to predefine a number of frequently occurring document segments, including 'abstract', 'body', 'references' and 'appendix'. Then, they extract the references from only the reference segment. Instead of SVM, the approaches proposed by Patrice Lopez [13] and Körner *it al.* [11] use CRF to extract reference strings in view of its capability to model decision boundaries among different classes. A popular reference extracting tool, called *ParsCit* [5], uses a set of heuristics to identify the references by scanning the entire document for section headers such as "Reference", "Bibliography", "Notes" or any possible variations.

Considering the difficulties of identifying the sections in PDF documents, Bergmark [1] first converted the document into a well-formed XML format and then parsed it to find the section labeled with "References". Another way to identify the section containing references is a rule-based approach [2], which tends to achieve better results as the rules are customized for a specific domain. However, taking into account the differences among reference styles and articles in general, the rule-based techniques do not perform well on articles for which rules were not priorly defined.

2.2 Reference Segmentation

In the literature, many methods for reference string segmentation (also known as reference string parsing) exist, differing in their techniques [18], assumptions and target datasets. We broadly classify these methods into two categories, namely Classifier-based and Template-based.

Classifier-based Techniques: Machine Learning (ML) algorithms are popular among researchers of information extraction as they are capable to learn from different data and achieve higher performance. Supervised algorithms such as CRF [15], Hidden Markov Models (HMM) [9] and SVM [21] proved their efficiency to achieve a satisfactory result in reference segmentation. Another direction to segment references adopts unsupervised learning techniques such as Hierarchical and Agglomerative clustering [8, 14], where little or no data is required to train the classifier.

SVM is one of the most frequently used methods for classifying tokens of a given reference string. Zhang *et al.* [21] propose a structural SVM to segment references of biomedical literature. Due to the strong regularity of the reference structure, the task is considered as a sequence learning problem, where the achieved accuracy is about 98%. Zou *et al.* [22] used and compared Support Vector Machine and Conditional Random Fields for reference segmentation, focusing on articles in the medical domain. The comparison concluded that both approaches achieve nearly the same accuracy (97%).

Since segmenting references corresponds to the problem of finding the most likely sequence of hidden states, HMM is a suitable tool for estimating a sequence of hidden variables given the sequence of observed events. In [9], a simple first-order HMM is applied, where the data is smoothed using naive smoothing to handle the absence of some state transitions and emissions. The model was trained on handcrafted citation training data and achieved an accuracy of more than 90% when tested on health science datasets consisting of homogeneous citations. Furthermore, Yin *et al.* [20] used Bigram HMM to solve this problem, claiming that their method yields better results than Unigram HMM. The difference between Unigram HMM and Bigram HMM is that the latter uses a modified model for computing the emission probability, without changing the structure of HMM itself.

Conditional Random Fields are statistical models used to compute the probability of hidden states given the sequence of observations [12]. Peng and McCallum [15] conducted an empirical study on Gaussian variations, Exponential and Hyperbolic-L1 prior and several class of features. They claimed that their method attained the state-of-the-art performance for extracting standard fields in

[1] http://excite.west.uni-koblenz.de/excite

the citations. Romanello *et al.* [17] used CRF for extracting and segmenting canonical references found in the field of classical studies. The popular citation extraction tool *ParsCit* developed by Councill *et al.* [5] relies on CRF model to identify the labels such as *Author*, *Title* and *Year* for the tokens being observed in the references.

Since the task of building a gold standard dataset is cumbersome and resource intensive, the problem of segmenting references was addressed using unsupervised learning approaches. For example, Grenager *et al.* [8] adopted this approach with small amounts of prior knowledge to segment fields, considering two different datasets, namely bibliographic citations and classified advertisements. They used Unconstrained HMM and Hierarchical Mixture Emission model for evaluating it against a supervised first-order HMM model in terms of average accuracy. Although the accuracy obtained by the unsupervised model was not better than that of the supervised model, the authors assume that the accuracy can further be increased by rectifying the structure of the model.

Semi-supervised learning approach was also explored in [8], where an increase of the model accuracy was noticed by incrementally introducing annotated citation data into the dataset being used to learn the unsupervised model. Chambers and Jurafsky [3] also leveraged unsupervised learning methods for template mining without knowing the templates in advance.

Basically, the performance of a machine learning algorithm depends on the features that represent the data. In the context of extracting metadata from references, many techniques achieved compelling performance using different features, which were manually designed. It is also important to note that these features are engineered for a specific domain and are not necessarily well generalized when applied to another domain or reference style. This deficiency can be overcome with the help of Deep Neural Networks which are considered to be more effective in obtaining an accurate and generalized representation of the data. Prasad *et al.* [16] exploited this approach by employing a Long Short Term Memory (LSTM) neural network model to represent tokens. Afterward, CRF model was trained on the extracted features, where this method showed strong performance over manually defined features.

Template-based Techniques: To extract relevant information, templates are formed based on prior domain knowledge. These templates are directly applied to the text data. Relevant information is extracted when certain conditions included in the template are matched [7]. Yang *at al.* [4] enhanced their previous work on sequence alignment techniques by replacing the reliance on background knowledge (e.g., author name database) with the punctuation symbols to identify the reference format. The authors demonstrate that the improved "BibPro" citation parser significantly performed better than INFOMAP and ParsCit metadata extraction tools if it used with a dataset containing six different citation styles. Also, a Hierarchical template-based approach *INFOMAP* was proposed by Day *et al.* [6]. Considering the disadvantages of traditional rule-based approaches such as lack of customization of the rules, the template-based hierarchical approach overcomes them by representing the information about the reference components (e.g., Author, Title, Volume, and Issue) in a tree structure. This structure represents patterns for the reference components found in different reference styles. The performance of INFOMAP was evaluated on 10,000 reference strings randomly selected from six different reference styles (e.g., IEEE, APA, ACM, etc.), demonstrating an average accuracy of 92.39%.

The template-based techniques require a thorough domain knowledge for designing the templates. Accordingly, Chambers and Jurafsky [3] proposed a method to learn the template structures instead of designing them. For this, the authors relied on two unsupervised learning algorithms: Latent Dirichlet Allocation (LDA) and Agglomerative clustering. The obtained result on the MUC-4 corpus leads to the conclusion that the learned templates perform better than the templates created by domain experts.

3 APPROACH

The proposed method of extracting references from PDF documents operates in two correlated phases: reference line classification and reference segmentation & identification. Before applying the method on query documents, an off-line process has to be carried out on the training set in order to train the necessary models. Subsequently, the online process employs the trained models to extract and segment reference strings from different PDF documents. Fig. 1 illustrates the general overview of the proposed approach consisting of an offline and online processes.

In order to train the models, layout information in text format are automatically extracted from the training PDF documents using Cermine [19]. From the output of each PDF document, the references were manually recognized and annotated, where each reference was in return manually segmented into basic components (e.g., author, title, source, etc.). These references, consolidated with the remaining text of their corresponding PDFs, represent the gold standard of our dataset. Detailed information about the dataset is given in Section 4.

Using the training set of the gold standard, a Random Forest model is trained for line classification considering four classes: non-, first, intermediate and last reference line. The reason of choosing Random Forest over other classifiers such as SVM is its capability to handle multiple class and to support features on various scales. Beside, the manually segmented reference strings in the gold standard are used to train CRF model in order to observe the relationships among components. Additionally, the completeness of a reference is estimated as probability density functions of the set of segmented reference strings. Each function is estimated with Multivariate Kernel Density Estimator given one of the forming properties (i.e., existing components, number of token per each component, etc.) of the training dataset. Below the extraction of features, core of the approach and filtering process are discussed:

3.1 Features

Both models, responsible for classifying lines (i.e Random Forest) and segmenting reference string candidates (i.e. Conditional Random Fields), use a set of discriminative features extracted from each line and each token, respectively. List. 1 presents a brief description of the features used by our method, where a detailed explanation can be found under this repository[2].

[2]Git repository: https://github.com/exciteproject/Exparser

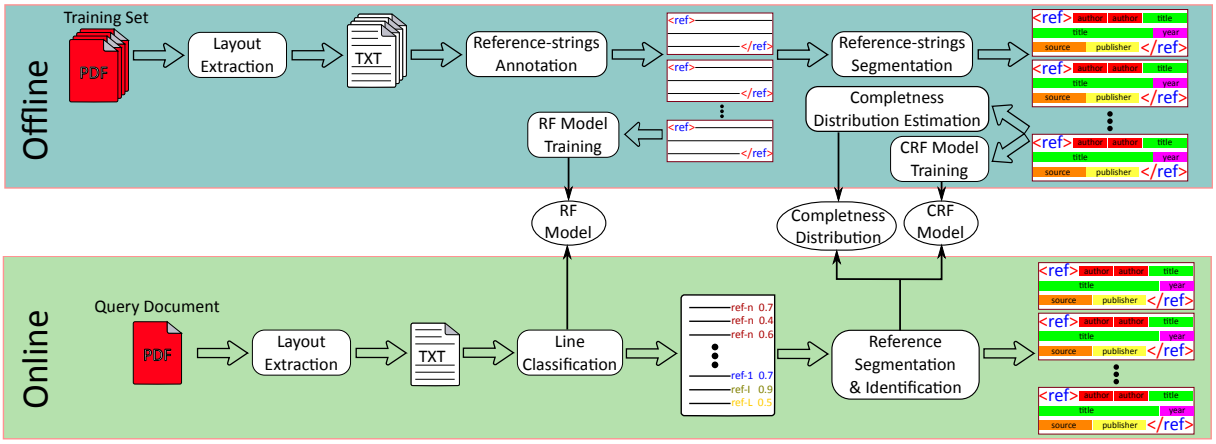

Figure 1: A general overview of the reference extraction approach. The offline process is dedicated to train the classifiers: reference line (Random Forest 'RF') and reference segmentation (Condition Random Fields 'CRF') and estimate the completeness distribution using the manually extracted and segmented references. The online process uses the trained models to extract and segment the references of a query PDF document.

List 1: Overview of the considered features in line classification and reference segmentation.

a Format-based
- Existence of year format, e.g., *1999*
- Existence of page format, e.g., *25–32*
- Existence of hyperlink format, e.g., *http://www.xyz.com*
- etc.

b Lexical-based
- Existence of the keyword *Vol.*
- Existence of the keyword *Eds.*
- Existence of the keyword *pp.*
- etc.

c Semantic-based
- Existence of a first or last name, e.g., *Alexander*
- Existence of a city name, e.g., *Paris*
- Existence of a source name, e.g., *International Conference on...*
- etc.

d Shape-based
- Ratio of digits in a Line/Token
- Ratio of capital letters in a Line/Token
- Histogram of words length in a Line
- etc.

3.2 Reference Extraction & Segmentation

Given a query document, the layout information is similarly extracted using Cermine [19]. Subsequently, the pre-trained Random Forest model is used to classify each line into either: non-, first, intermediate or last reference line (i.e., ref-0, ref-1, ref-I and ref-L, respectively). Here, each classified line is associated with the probabilities of its belonging to all classes. The classified lines and the associated probabilities are used afterwards to compose and segment consistent references. A consistent reference denotes a combination of lines that potentially fulfill the conditions of a complete and coherent reference string. In this regards, the method starts with the line $\ell_{\hat{i}}$ having the highest reference probability among all lines (Λ) being classified as a *reference line* (i.e. ref-1, ref-I or ref-L) , where $\hat{i}$ is obtained as follows:

$$\hat{i} = \underset{\substack{0<i\leq N \\ i\in\Lambda}}{\operatorname{argmax}} P_e\left(\omega^e(\ell_i) \neq \text{ref-0} \mid \boldsymbol{c}_i^e\right) , \tag{1}$$

N is the total number of lines in Λ and ℓ_i denotes the ith line. $\omega^e(.)$ is the line class and $\boldsymbol{c}_i^e$ is the corresponding extracted feature vector. $P_e(. \mid .)$ represents the conditional probability of the line class, given the corresponding feature vector, which is computed by the pre-trained Random Forest model.

The unique selected line is considered as an initial reference-string candidate $\psi_{t=0}$. Then, a series of candidates are sequentially generated in a random process, starting from $\psi_{t=0}$, where the best candidate is assumed to be approached with the progress of this process. Considering that the number of lines composing a reference string is unpredictable, the number of candidates (α) in our experiments is set to 30 to ensure that the best candidate is reached. Let ψ_t and ψ_{t-1} be two consecutive reference-string candidates, the superiority between them is assessed by the acceptance ratio a, which measures the quality of reference-string candidates in terms of line combination (Δ_e), segmentation (Δ_s) and completeness (Δ_c). Here, each candidate is compared to its predecessor, where it is accepted if it better, otherwise, it is rejected and substituted with the predecessor candidate. In other terms, since the comparison is achieved by computing the acceptance ratio a, the new candidate is accepted only if $a > 1$. The acceptance ratio between the current candidate ψ_t and its predecessor ψ_{t-1} ($a_{t,t-1}$) is computed as follows:

$$a_{t,t-1} = \frac{\Delta_e(\psi_t)}{\Delta_e(\psi_{t-1})} \times \frac{\Delta_s(\psi_t)}{\Delta_s(\psi_{t-1})} \times \frac{\Delta_c(\psi_t)}{\Delta_c(\psi_{t-1})} . \tag{2}$$

Eq.2 approximates the optimal candidate of line combination for each reference by assessing the disparity among the qualities of sampled candidates. It is important to note that the number of samples (α) should be sufficient to obtain a stabilized reference-string.

Considering a candidate ψ, the quality measure of line-combination validates the determination of ψ by first-reference line from top and last reference line from bottom. In this context, an adequate reference candidate is characterized with top and bottom lines having high probabilities of first and last reference lines, respectively. The line-combination measure of a given candidate $\Delta_e(\psi)$ is obtained as follows:

$$\begin{aligned}\Delta_e(\psi) =& P_e\left(\omega^e(\psi^1) = \text{ref-1} \mid \boldsymbol{c}^e(\psi^1)\right) \\ &\times\ P_e\left(\omega^e(\psi^L) = \text{ref-L} \mid \boldsymbol{c}^e(\psi^L)\right)\ ,\end{aligned} \tag{3}$$

where ψ^1 and ψ^L correspond to the first and last line of ψ, respectively.

The segmentation measure (Δ_s) is a precision evaluation of the segmentation that is applied on ψ, and it is computed as follows:

$$\Delta_s(\psi) = \prod_{j}^{M} \max_{0<k\leq K} P_s\left(\omega_k^s \mid \boldsymbol{c}_j^s\right)\ , \tag{4}$$

where M is the number of token in ψ, ω^s is the class of reference component (i.e., Author, Title, Year, etc.) and K denotes the number of all possible reference components. $\boldsymbol{c}_j^s$ represents the extracted feature vector from the jth token in the corresponding reference candidate.

As its name indicates, the completeness measure evaluates the completeness of the reference candidate given the references in the training dataset.

Let f_a, f_b and f_c be the estimated functions that represent the distributions of $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$, respectively, obtained using Multivariate Kernel Density Estimation. $\mathbf{x} = (x_1, x_2, \cdots, x_R)$ represents the existence of key-components (i.e. Author name, Year, Page, Source and Editor) in the training dataset of length (R). Similarly, $\mathbf{y} = (y_1, y_2, \cdots, y_R)$ and $\mathbf{z} = (z_1, z_2, \cdots, z_R)$ represent the arrangements of the key-components in terms of their first and last appearance, respectively. Furthermore, other heuristic completeness factors (f_d) are used, for example, the length of lines and punctuation marks at the end of lines. A detailed explanation about the completeness factors is found under the corresponding repository link[2].

For measuring the completeness of a reference string candidate ψ, giving its set of tokens and their corresponding components, $\Delta_c(\psi_t)$ is computed as: the product of $f_a(\psi)$, $f_b(\psi)$, and the other completeness factors.

$$\Delta_c(\psi) = f_a(\psi) \times f_b(\psi) \times f_c(\psi) \times f_d(\psi)\ . \tag{5}$$

For the sake of simplicity, Fig. 2 shows an illustrative example of the computation of quality measures given a reference-string candidate. In the top-left, the line combination measure is obtained as the product of two scores: 1) the probability that the first line of the reference-string candidate corresponds to ref-1 and 2) the probability that the last line of the reference-string candidate corresponds to ref-L. In the top-right, the products of the completeness densities implies the completeness measure. In the bottom side, the segmentation measure is represented by the product of components' scores.

After determining the first reference string based on the above process, the method iteratively searches for the remaining reference strings until no reference line remains. It starts with the line having the highest probability among the remaining lines and follows the same process to form and segment a new coherent reference string. Algorithm. 1 summarizes the above discussed steps.

Algorithm 1 Algorithm of extracting coherent reference strings.

```
Θ = ∅                                                    ▷ Set of references
while Λ ≠ ∅ do
    Initialise with ℓ_î (among Λ_{0:N}) using Eq. 1
    Determine the initial candidate: ψ_{t=0} = ℓ_î
    Compute the measures of ψ_{t=0} following: Eq. 3, Eq. 4 and Eq. 5
    for t ← 1 to α do                                    ▷ α = 30
        Generate a new candidate ψ_t = ℓ_{x(t):y(t)} from ψ_{t−1}, ensuring ℓ_{x(t):y(t)} ⊂ Λ.
        Compute the measures of ψ_t following: Eq. 3, Eq. 4 and Eq. 5 ⇒ obtain a
        if a < 1 then
            ψ_t = ψ_{t−1}
        end if
    end for
    Θ = Θ ∪ ψ_α
    Λ = Λ \ ℓ_{x(α):y(α)}
end while
```

A detailed explanation of the method is given in an online documentation[3].

3.3 Filtering

In the online process, the lines of the query document are independently classified, without considering the classes of subsequent and former lines. Despite that this is considered an advantage to find references in different parts of the document (e.g., footnote), it is highly likely that the classification output is noisy with wrong classifications. To overcome this problem, a filtering process is necessary to smooth the output. The reference regions are first searched by employing a capacity distribution that expresses each line with the amount of neighbor reference lines. Let ε be the stride parameter, the density d_i of each line ℓ_i is computed as:

$$d_i = \sum_{\iota=i-\varepsilon}^{i+\varepsilon} h(\ell_\iota),\ \text{ where }\ h(\ell_\iota) = \begin{cases} 0 & \text{if } \omega^e(\ell_\iota) = \text{ref-0} \\ 1 & \text{Otherwise} \end{cases}\ . \tag{6}$$

Based on the above equation, the wrongly detected reference lines can be discarded. Also, the wrongly missed reference lines can be reconsidered. Note that this rectification does not affect the scores of classes given by Random Forest, where they remain the same for the rectified lines.

[3] https://exparser.readthedocs.io

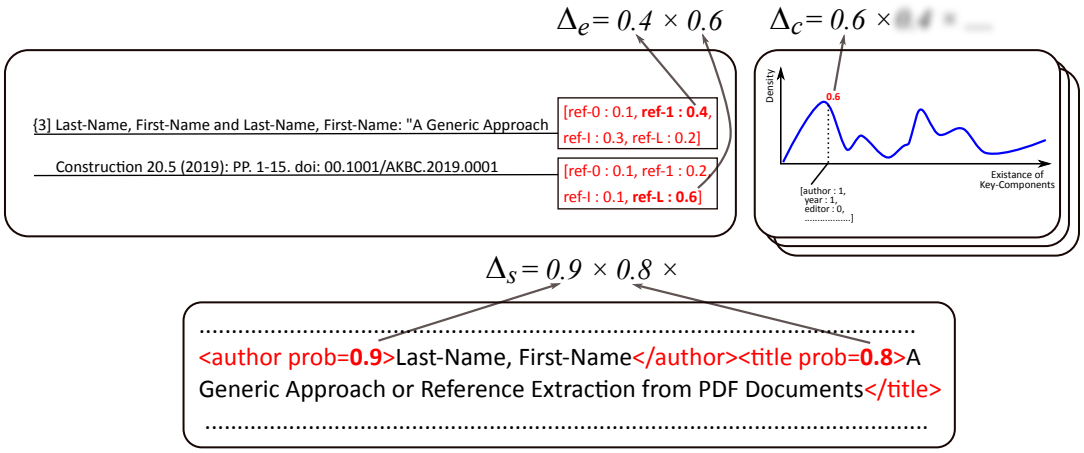

Figure 2: An illustration of the quality measure given a reference candidate.

4 EXPERIMENTS

To validate the effectiveness of the proposed method in terms of both reference extraction and segmentation, several experimental evaluations have been carried out on different datasets due to the variety of their properties and characteristics. Another reason to use different datasets is the fact that each dataset is derived from a well-known state-of-the-art method. Therefore, for precise comparison, the proposed approach is compared to *Cermine* [19], *Grobid* [13] and *ParsCit* [5] on their corresponding datasets. Additionally, this paper proposes a new challenging dataset, on which the proposed method, *Grobid* and *Cermine* are applied and compared. The efficiency of the methods is assessed in terms of precision, recall and F-score, where at the end the complexity of each method is discussed. In the following, the considered datasets are described:

Proposed Dataset in German language (PGS):. This dataset consists of 125 annotated articles in German language collected from SSOAR[4]. All articles are on social science and can be divided into two categories: 1) 100 articles have references in a specific section and 2) the references in the remaining 25 articles are sparsely located in the document and not only in the reference section. Fig 3 illustrates two examples of references and notes in German social science publications. As shown, the similarity between references in footnotes and notes is very high. This includes; 1) similar location in the paper, same font style and size, and 3) numerated in the same order. On the other hand, a reference in the reference section appears differently in terms of format and content.
The references in each article are manually identified, segmented and consolidated with the remaining text in the layout file. By considering the reference strings in all articles, 2652 reference strings are extracted and assigned to 6711 text lines. Here, the lines are extracted using a tool from *Cermine* that extracts each sequence of words in the PDF with respecting line break and multi-column PDF. It has to be noted that the references in this dataset do not correspond to only academic literature but grey literature as well. Therefore, different unusual references are more likely to appear in this dataset. This include; references without authors (e.g. organizational reports), references with only title and URL (e.g. datasets). This variety is considered challenging in view of 1) the existence and absence of essential components (e.g. author), 2) the non-unified arrangement of reference components (both inter- and intra-articles), 3) the variety of component number.
In the remaining of this section, the parts of extraction and segmentation of *PGS* are referred by PGS_e and PGS_s, respectively.

Proposed Dataset in English language (PES):. Similarly to *PGS*, *PES* is a set of 100 PDF articles collected from SSOAR, where all articles are in English language. However, the references in this collection are from different languages. After annotating all the references in *PES*, 2838 are counted.

Cermine Dataset (CDS):. It is a part of GROTOAP2 dataset[5], which consists of 13210 articles collected from medicine domain. Due to the considerable time consuming to train the reference segmentation model of *Cermine*, only 6858 reference strings from GROTOAP2 are considered, each of which is segmented into components.

Grobid Dataset (GDS):. From 1943 articles available in PMC_sample[6], 100 articles are randomly selected to constitute this dataset, where reference strings in each document are annotated and segmented.

For detailed evaluations, reference identification and reference segmentation are independently discussed below:

4.1 Reference Identification

Evaluating the quality of reference identification is subjected to several criteria. First, the retrieval accuracy, which is characterized by 1) the number of relevant reference lines among the retrieved ones and 2) the number of retrieved reference lines among the total amount of existing reference lines in the document. As a reference consists of multiple lines, it might be retrieved, either precisely,

[4]https://www.gesis.org/ssoar/home/

[5]https://repod.pon.edu.pl/dataset/grotoap2
[6]https://grobid.readthedocs.io

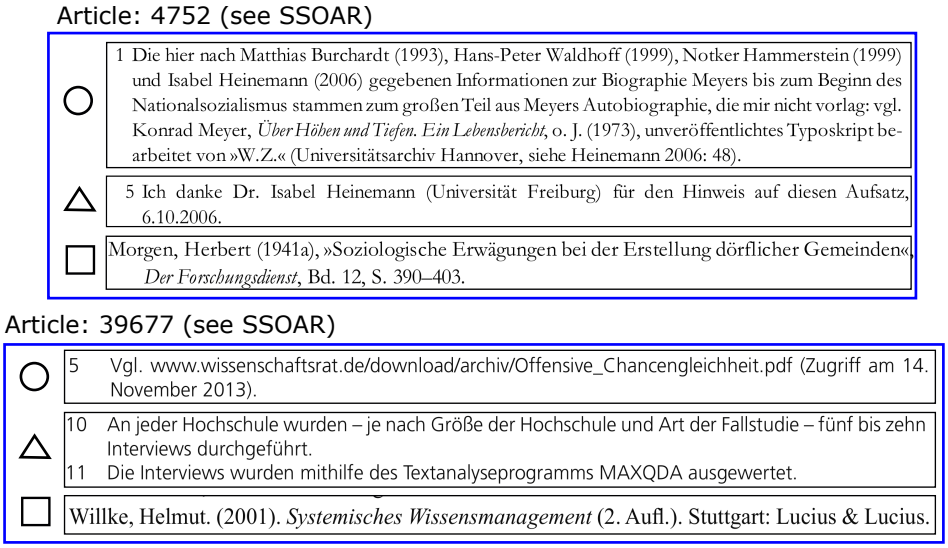

Figure 3: Examples from two different articles showing the difference and similarity between references and footnotes. ◯: reference in footnote, △: footnote (non-reference) and □: a reference in a reference section.

missing relevant line(s) or including irrelevant line(s). Therefore, the second criterion is the inner precision within the reference itself. Accordingly, a reference is considered to be precisely identified if all its lines are gathered without including outlier lines.

The qualities of reference identification of the proposed method (*Proposed*), *Grobid* and *Cermine* are assessed by applying each method to PGS_e using 10-fold cross-validation. To ensure equitable learning of all the models, we trained *Grobid* on the level of reference-line segmentation and section segmentation. Here, footnotes containing references were re-annotated as reference sections for both training and testing sets. Considering the identification of each reference line independently, Table 1 presents the results of the three datasets in terms of three evaluation macro averaged metrics; precision, recall and F1-score. Here, each line in the document can belong to either reference line or non-reference line. As demonstrated, *Cermine* achieves the highest precision among the three models, where about 77% of the retrieved references are relevant. However, most of the reference lines are not retrieves. On the contrary, the proposed method retrieved almost 90% of the reference lines, which is higher with 20% from the successor approach.

Table 1: Result of independent reference line extraction on PGS_e using *Proposed*, *Grobid* and *Cermine*.

	Precision	Recall	F1-Score
Proposed	0.69	**0.89**	**0.78**
Grobid	0.66	0.69	0.67
Cermine	**0.77**	0.26	0.39

In addition to precisely and accurately identifying reference lines, it is also necessary to consolidate them together to form consistent reference strings. Considering that a reference string is characterized by a first line (ref-1), intermediate line(s) (ref-I) and last line (ref-L), Table 2 demonstrates the results of the three approaches, where that obtained by *Proposed* is very similar to the result in Table 1. This means that most of the retrieved reference lines are precisely identified (ref-1, ref-I and ref-L) and thus references are well composed.

4.2 Reference Segmentation

The quality of reference segmentation is assessed on the basis of several experimental evaluations. In order to avoid the influence of former phases on the evaluation of this phase, the references are assumed to be properly extracted for all methods. Hence, the input in this evaluation is a set of references, each of which is segmented into components such as '*author*', '*title*', '*pages*', etc. It is important to note that the compared methods don't consider similar components, for example, '*URL*' is not recognized by *Cermine* and '*Identifier*' is considered only by the proposed approach. Moreover, the evaluation is carried out after removing non-alpha-numeric characters from the output of all models as well as the ground truth. Due to the variety of reference styles and for objective evaluation, five datasets are used to compare the *Proposed* to the state-of-the-art methods.

To ensure that all models benefit from all available information, the segmented references in PGS_s and PES_s are prepared according to the corresponding format of each model without changing any of their properties (i.e. spacing, punctuation, capitalisation, etc.). Furthermore, to validate the effectiveness of the proposed approach in segmenting regular reference strings, Table 3 presents the results of our model and three other baseline models: *Grobid*, *Cermine* and *ParsCit* on PES_s. The result of each model was obtained by applying 10-fold cross-validation on the 2838 reference strings, where the data is split based on the 100 blocks of reference strings (i.e. articles). Specifically, for each fold, the reference strings in 90 articles are used to train the models and the reference of the remaining articles

Table 2: Result of reference extraction on PGS_e using *Proposed*, *Grobid* and *Cermine*, where M-Aver. denotes Micro-Average.

	Precision				Recall				F1-Score			
	ref-1	*ref-I*	*ref-L*	*M-Aver.*	*ref-1*	*ref-I*	*ref-L*	*M-Aver.*	*ref-1*	*ref-I.*	*ref-L*	*M-Aver.*
Proposed	0.73	**0.51**	0.73	**0.67**	**0.84**	**0.84**	**0.86**	**0.84**	**0.78**	**0.64**	**0.79**	**0.74**
Grobid	**0.74**	0.42	**0.74**	0.65	0.56	0.73	0.6	0.62	0.64	0.53	0.66	0.61
Cermine	0.63	0.09	0.12	0.31	0.47	0.01	0.01	0.19	0.54	0.01	0.01	0.22

Table 3: Result of reference segmentation on *PES* using: P: *Proposed*, G: *Grobid*, C: *Cermine* and R: *ParsCit*.

	Precision				Recall				F1-Score			
	P	*G*	*C*	*R*	*P*	*G*	*C*	*R*	*P*	*G*	*C*	*R*
Publisher^{1,2,3}	0.959	**0.964**	0.773	0.921	0.845	**0.877**	0.58	0.528	0.897	**0.917**	0.611	0.665
First Page^{1,2,3}	**0.997**	0.988	0.982	0.908	**0.98**	0.963	0.972	0.014	**0.989**	0.976	0.977	0.027
Last Page^{2}	0.994	0.917	**0.996**	N/A	**0.984**	0.906	0.975	N/A	**0.989**	0.911	0.985	N/A
Title^{1,2,3}	0.932	**0.952**	0.829	0.787	**0.973**	0.958	0.958	0.908	0.951	**0.955**	0.888	0.843
URL^{3}	**0.965**	0.944	N/A	0.865	0.764	**0.849**	N/A	0.445	0.809	**0.868**	N/A	0.564
Author^{1,3}	**0.971**	0.891	0.946	0.963	**0.91**	0.899	0.9894	0.884	**0.938**	0.894	0.918	0.921
Author Surname^{2}	**0.952**	0.891	0.889	N/A	0.884	**0.909**	0.873	N/A	**0.915**	0.899	0.88	N/A
Author Given-name^{2}	**0.941**	0.79	0.938	N/A	**0.912**	0.855	0.849	N/A	**0.925**	0.821	0.887	N/A
Volume^{1,2,3}	0.956	**0.992**	0.957	0.971	**0.937**	0.925	0.928	0.242	0.925	**0.957**	0.942	0.374
Source^{1,2,3}	**0.943**	0.941	0.903	0.698	**0.835**	0.832	0.641	0.469	**0.884**	0.883	0.745	0.558
Editor^{3}	0.898	**0.906**	N/A	0.498	**0.778**	0.494	N/A	0.12	**0.832**	0.638	N/A	0.19
Identifier	**0.96**	N/A	N/A	N/A	**0.701**	N/A	N/A	N/A	**0.733**	N/A	N/A	N/A
Year^{1,2,3}	0.944	**0.991**	0.972	0.96	0.933	**0.95**	0.946	0.766	0.939	**0.97**	0.958	0.85
Issue^{2}	0.958	**0.981**	0.978	N/A	**0.889**	0.781	0.846	N/A	**0.922**	0.867	0.906	N/A
Other^{3}	**0.846**	0.783	N/A	0.485	0.722	**0.78**	N/A	0.775	0.777	**0.778**	N/A	0.595
macro Average^{1}	0.957	**0.96**	0.91	0.887	**0.916**	0.915	0.859	0.544	0.932	**0.936**	0.868	0.605
macro Average^{2}	**0.958**	0.941	0.922	N/A	**0.917**	0.896	0.857	N/A	**0.934**	0.916	0.878	N/A
macro Average^{3}	**0.941**	0.935	N/A	0.806	**0.868**	0.858	N/A	0.515	**0.894**	0.884	N/A	0.559

are used for testing. The reason is to avoid having relatively similar reference strings (in terms of style, arrangement and components) in training and testing sets, assuming that the reference strings in the same article follow the same style. Since each of the applied method admits a different set of components, three average results are computed for different set of components. E.g., $macro\,Average^{1}$ is the average of all components associated with the attribute (^{1}).

As the table demonstrates, the average results of *Proposed*, *Grobid* and *Cermine* are considering the seven common components are similar. However, considering the result per each component, *Proposed* achieves a satisfactory result for all components, where the minimum precision, recall and F1-score, without considering '*Other*', are (0.898, 0.764, 0.733), respectively. Also, most essential components (i.e. author, title, source, etc.), that can be used to identify articles, are retrieved by *Proposed*.

Moreover, we evaluated the approaches on the segmentation set of PGS_S, where the results of *Proposed*, *Grobid* and *Cermine* are also very similar with higher precision for *Grobid* and higher recall for *Proposed* as can be seen in Table 4.

Since the variety of reference style is large and the importance of components differ from a community to another, as we observed in the annotation of other datasets, we evaluate *Proposed* considering *CDS* dataset using 10-fold cross-validation. In this evaluation, we split the reference strings to training and testing parts without taking into account the articles citing these references. In addition to *Proposed*, *Cermine* is also applied to this dataset and their results are demonstrated in Table. 5. In contrast to the previous evaluation and although its high precision, the recall of the proposed method is relatively lower. This decrease in performance is explained by the annotation of this dataset, in which a lot of content is not annotated. More precisely, there exist references, in which URLs and information about the editor are present but not annotated. In our approach, any content in the reference which is not annotated is considered as either '*Other*' or '*Empty*'. '*Other*' is assigned to content that might be useful such as note and place of publication and its goal is to help the prediction of neighbouring components by exploring the transitions among states. '*Empty*' is any content that is not useful as information but it helps the understanding of the edges between components by CRF. Examples of '*Empty*' include: punctuation, key-words (e.g. '*In:*'), parentheses.

As the previous evaluation showed the negative impact of non-annotated content in the prediction of some components, we aim in this evaluation to examine the influence of '*Other*' and '*Empty*' on guiding the prediction. For this, we used *GDS* dataset, in which the contents of the annotated reference strings are reordered compared to the raw data. In addition, these annotated reference strings

Table 4: Result of reference segmentation on PGS_s using: R: *Proposed*, G: *Grobid*, C: *Cermine* and R: *ParsCit*.

	Precision				Recall				F1-Score			
	P	*G*	*C*	*R*	*P*	*G*	*C*	*R*	*P*	*G*	*C*	*R*
Publisher[1,2,3]	0.964	**0.97**	.966	0.747	0.811	**0.814**	0.705	0.384	0.875	**0.878**	0.765	0.496
First Page[1,2,3]	0.979	**0.989**	0.957	0.934	**0.938**	0.846	0.887	0.04	**0.958**	0.91	0.919	0.079
Last Page[2]	0.991	0.988	**0.995**	N/A	**0.962**	0.946	0.95	N/A	**0.976**	0.966	0.972	N/A
Title[1,2,3]	0.894	**0.921**	0.817	0.653	**0.961**	0.937	0.921	0.866	0.925	**0.929**	0.865	0.743
URL[3]	**0.996**	0.977	N/A	0.985	0.8	**0.961**	N/A	0.687	0.881	**0.969**	N/A	0.783
Author[1,3]	0.926	0.809	0.882	**0.945**	0.793	**0.884**	0.857	0.732	0.854	0.844	**0.867**	0.824
Author Surname[2]	**0.91**	0.843	0.782	N/A	0.787	**0.881**	0.821	N/A	0.843	0.**861**	0.799	N/A
Author Given-name[2]	**0.89**	0.778	0.887	N/A	0.823	**0.856**	0.791	N/A	**0.855**	0.813	0.828	N/A
Volume[1,2,3]	0.932	**0.988**	0.872	0.926	**0.78**	0.748	0.757	0.144	**0.848**	**0.848**	0.808	0.245
Source[1,2,3]	0.89	**0.898**	0.784	0.488	**0.749**	0.746	0.542	0.487	0.81	**0.814**	0.636	0.484
Editor[3]	0.878	**0.898**	N/A	0.596	**0.751**	0.489	N/A	0.029	**0.808**	0.631	N/A	0.048
Identifier	**0.902**	N/A	N/A	N/A	**0.706**	N/A	N/A	N/A	**0.754**	N/A	N/A	N/A
Year[1,2,3]	0.904	0.977	0.933	**0.98**	0.901	0.907	**0.92**	0.529	0.903	**0.941**	0.926	0.684
Issue[2]	0.964	0.979	**0.99**	N/A	**0.703**	0.574	0.521	N/A	**0.799**	0.715	0.658	N/A
Other[3]	**0.848**	0.695	N/A	0.438	0.735	**0.78**	N/A	0.721	**0.785**	0.73	N/A	0.54
macro Average[1]	0.927	**0.936**	0.887	0.81	**0.848**	0.84	0.798	0.455	**0.882**	0.881	0.827	0.508
macro Average[2]	0.927	**0.936**	0.898	N/A	**0.841**	0.825	0.781	N/A	**0.879**	0.867	0.818	N/A
macro Average[3]	**0.921**	0.912	N/A	0.769	**0.822**	0.812	N/A	0.462	**0.865**	0.849	N/A	0.493

Table 5: Result of reference segmentation on *CDS* using: P: *Proposed* and C: *Cermine*.

	Precision		Recall		F1-Score	
	P	*C*	*P*	*C*	*P*	*C*
Publisher	**0.94**	0.93	0.35	**0.61**	0.43	**0.74**
First Page	0.98	**0.99**	0.96	**0.98**	0.97	**0.98**
Last Page	0.997	**0.99**	0.93	**0.99**	0.96	**0.99**
Title	**0.91**	0.83	0.96	**0.97**	**0.93**	0.89
A.Surname	**0.88**	0.8	0.24	**0.92**	0.38	**0.86**
A.Given-name	**0.98**	0.89	0.16	**0.9**	0.27	**0.892**
Volume	**0.99**	**0.99**	0.91	**0.97**	0.95	**0.98**
Source	0.92	**0.93**	**0.96**	0.61	**0.94**	0.74
Year	0.95	**0.99**	0.91	**0.97**	0.93	**0.98**
Issue	0.95	**0.98**	0.8	**0.86**	0.87	**0.92**
macro Average	**0.95**	0.93	0.72	**0.88**	0.76	**0.89**

Table 6: Result of reference segmentation on *GDS* using: P: *Proposed* and G: *Grobid*.

	Precision		Recall		F1-Score	
	P	*G*	*P*	*G*	*P*	*G*
Publisher	**0.93**	0.90	**0.77**	0.72	**0.83**	0.79
First Page	**0.99**	0.98	**0.98**	0.93	**0.9**	0.95
Last Page	**0.98**	0.92	**0.97**	0.92	**0.98**	0.92
Title	**0.99**	0.98	**0.99**	0.98	**0.99**	0.98
URL	**0.99**	0.98	0.38	**0.92**	0.39	**0.95**
A.Surname	**0.99**	0.8	**0.99**	0.95	**0.99**	0.87
A.Given-names	**0.99**	0.33	**0.99**	0.49	**0.99**	0.394
Volume	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**
Source	**0.96**	0.95	0.91	**0.93**	**0.94**	**0.94**
Editor	**0.99**	0.96	0.02	**0.62**	0.04	**0.71**
Year	**0.99**	**0.94**	0.98	**0.98**	0.96	**0.99**
Issue	**0.99**	0.98	**0.94**	0.92	**0.96**	0.95
Other	0.89	**0.98**	0.7	**0.93**	**0.8**	0.95
*Place**	N/A	0.95	N/A	0.07	N/A	0.13
macro Average	**0.97**	0.9	0.82	**0.87**	0.83	**0.87**

(ground truth) are filtered from additional content, including punctuation and key-words. Since both models *Proposed* and *Grobid* rely on the sequential property of reference strings, we preferred to test both methods on the ground truth after removing the annotation so that we ensure that the training and testing phases are consistent. The result presented in Table 6 shows a high average precision of the proposed method without relying on additional content. It shows also that the relatively low recall is due to two components: '*Editor*' and '*URL*'. For '*Editor*', the reason is the lack of instances assigned to this component and the remarkable presence of *Other* which is employed in this case to identify 'place of publication'.

The above evaluations demonstrates the capability of our method to extract and segment references from a challenging dataset accurately. It indicates also the necessity of training the model on a well annotated dataset. The details of all evaluations presented in this papers, including datasets, source codes and evaluation metrics, can be found in our public repository[2]. The developed method is employed in the toolchain of the EXCITE project, which is dedicated to publish open literature references [10]. In addition, an online demo is made available to the public and can be easily used[1].

5 CONCLUSION

A novel approach for extracting and segmenting references is proposed in this paper. The benefit of combining the different steps in a coherent mechanism is demonstrated and validated with the obtained result. The presented approach is non-parameterized, where it takes the PDF document as input and outputs a list of segmented reference strings. As a result, the approach achieved a satisfactory result on different datasets overcoming state-of-the-art methods. This effectiveness is validated in terms of reference extraction and reference segmentation. Moreover, we introduced a new challenging dataset dedicated to both tasks.

For future work, on the one side, we will improve our method by combining classical features and word embedding to obtain a better representation of tokens and thus better extraction and segmentation result. On the other side, we will apply the method on a collection of articles, e.g. on German social science, and match them against the records of existing bibliographic databases. The reason is to enrich the citation network with citations, which are unintentionally neglected because of their old publication dates, the linking inability of their publishers, etc.

Acknowledgment: This work has been funded by Deutsche Forschungsgemeinschaft (DFG) as part of the project "Extraction of Citations from PDF Documents (EXCITE)" under grant numbers MA 3964/8-1 and STA 572/14-1.

REFERENCES

[1] Donna Bergmark. Automatic extraction of reference linking information from online documents. Technical report, Cornell University, 2000.

[2] Kurt D Bollacker, Steve Lawrence, and C Lee Giles. Citeseer: An autonomous web agent for automatic retrieval and identification of interesting publications. In *Proceedings of the second international conference on Autonomous agents*, pages 116–123. ACM, 1998.

[3] Nathanael Chambers and Dan Jurafsky. Template-based information extraction without the templates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 976–986. Association for Computational Linguistics, 2011.

[4] Chien-Chih Chen, Kai-Hsiang Yang, Hung-Yu Kao, and Jan-Ming Ho. Bibpro: A citation parser based on sequence alignment techniques. In *Proceedings of the 22nd International Conference on Advanced Information Networking and Applications*, pages 1175–1180. IEEE, 2008.

[5] Isaac G. Councill, C. Lee Giles, and Min-yen Kan. Parscit: An open-source crf reference string parsing package. In *Proceedings of the 6th edition of the Language Resources and Evaluation Conference*. European Language Resources Association, 2008.

[6] Min-Yuh Day, Richard Tzong-Han Tsai, Cheng-Lung Sung, Chiu-Chen Hsieh, Cheng-Wei Lee, Shih-Hung Wu, Kun-Pin Wu, Chorng-Shyong Ong, and Wen-Lian Hsu. Reference metadata extraction using a hierarchical knowledge representation framework. *Decision Support Systems*, 43(1):152–167, 2007.

[7] Ying Ding, Gobinda Chowdhury, Schubert Foo, et al. Template mining for the extraction of citation from digital documents. In *Proceedings of the Second Asian Digital Library Conference*, pages 47–62, 1999.

[8] Trond Grenager, Dan Klein, and Christopher D. Manning. Unsupervised learning of field segmentation models for information extraction. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 371–378. Association for Computational Linguistics, 2005.

[9] Erik Hetzner. A simple method for citation metadata extraction using hidden markov models. In *Proceedings of the 8th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 280–284. ACM, 2008.

[10] Azam Hosseini, Behnam Ghavimi, Dagmar Kern, and Philipp Mayr. EXCITE - A toolchain to extract, match and publish open literature references. In *Proceedings of the 19th ACM/IEEE on Joint Conference on Digital Libraries*. ACM, 2019.

[11] Martin Körner, Behnam Ghavimi, Philipp Mayr, Heinrich Hartmann, and Steffen Staab. Evaluating reference string extraction using line-based conditional random fields: A case study with german language publications. In *Proceedings of the first Workshop on Data-Driven Approaches for Analyzing and Managing Scholarly Data*, pages 137–145. Springer, 2017.

[12] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. pages 282–289, 2001.

[13] Patrice Lopez. Grobid: Combining automatic bibliographic data recognition and term extraction for scholarship publications. In *Proceedings of the 13th European Conference on Digital Libraries*, pages 473–474. Springer Berlin Heidelberg, 2009.

[14] Hanna Pasula, Bhaskara Marthi, Brian Milch, Stuart Russell, and Ilya Shpitser. Identity uncertainty and citation matching. In *Proceedings of the 15th Advances in Neural Information Processing Systems Conference*. Neural information processing systems foundation, 1 2003.

[15] Fuchun Peng and Andrew McCallum. Information extraction from research papers using conditional random fields. *Information Processing and Management*, 42(4):963–979, Jul 2006.

[16] Animesh Prasad, Manpreet Kaur, and Min-Yen Kan. Neural parscit: a deep learning-based reference string parser. *International Journal on Digital Libraries*, pages 1–15, 2018.

[17] Matteo Romanello, Federico Boschetti, and Gregory Crane. Citations in the digital library of classics: extracting canonical references by using conditional random fields. In *Proceedings of the 2009 Workshop on Text and Citation Analysis for Scholarly Digital Libraries*, pages 80–87. Association for Computational Linguistics, 2009.

[18] Dominika Tkaczyk, Andrew Collins, Paraic Sheridan, and Joeran Beel. Machine learning vs. rules and out-of-the-box vs. retrained: An evaluation of open-source bibliographic reference and citation parsers. In *Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries*, pages 99–108. ACM, 2018.

[19] Dominika Tkaczyk, Paweł Szostek, Mateusz Fedoryszak, Piotr Jan Dendek, and Łukasz Bolikowski. Cermine: automatic extraction of structured metadata from scientific literature. *International Journal on Document Analysis and Recognition (IJDAR)*, 18(4):317–335, Dec 2015.

[20] Ping Yin, Ming Zhang, ZhiHong Deng, and DongQing Yang. Metadata extraction from bibliographies using bigram hmm. In *Proceedings of the 7th International Conference on Asian Digital Libraries*, pages 310–319. Springer, 2004.

[21] Xiaoli Zhang, Jie Zou, Daniel X Le, and George R Thoma. A structural svm approach for reference parsing. In *Proceedings of the 9th International Conference on Machine Learning and Applications*, pages 479–484, Dec 2010.

[22] Jie Zou, Daniel Le, and George R. Thoma. Locating and parsing bibliographic references in html medical articles. *International Journal on Document Analysis and Recognition (IJDAR)*, 13(2):107–119, Jun 2010.