FLUX-CiM: Flexible Unsupervised Extraction of Citation Metadata

Eli Cortez[1] Altigran S. da Silva[1] Marcos André Gonçalves [2]

Filipe Mesquita[1] Edleno S. de Moura[1]

[1]Universidade Federal do Amazonas
Departamento de Ciência da Computação
Manaus, AM, Brazil
{eccv,alti,fsm,edleno}@dcc.ufam.edu.br

[2]Universidade Federal de Minas Gerais
Departamento de Ciência da Computação
Belo Horizonte - MG - Brazil
mgoncalv@dcc.ufmg.br

ABSTRACT

In this paper we propose a knowledge-base approach to help extracting the correct components of citations in any given format. Differently from related approaches that rely on manually built knowledge-bases (KBs) for recognizing the components of a citation, in our case, such a KB is automatically constructed from an existing set of sample metadata records from a given area (e.g., computer science or health sciences). Our approach does not rely on patterns encoding specific delimitators of a particular citation style. It is also unsupervised, in the sense that it does not rely on a learning method that requires a training phase. These features assign to our technique a high degree of automation and flexibility. To demonstrate the effectiveness and applicability of our proposed approach we have run experiments in which we applied it to extract information from citations in papers of two different domains. Results of these experiments indicate precision and recall levels above 94% and perfect extraction for the large majority of citations tested.

Categories and Subject Descriptors

H.3.7 [**Information Storage and Retrieval**]: Digital Libraries—*standards, systems issues*

General Terms

Algorithms, Standardization

Keywords

Citation Management, Metadata Extraction

1. INTRODUCTION

Citation management is a central aspect of modern digital libraries. Citations serve, for example, as a fundamental evidence of the impact or significance of particular scientific articles, and therefore of the research they report. Evaluation of individual's performances for promotions and grants may use citations as evidence to evaluate competence and the impact of a researcher's work. Citations have also been used as an auxiliary evidence in Information Retrieval tasks such as automatic document classification [3, 4], indexing and ranking [16], and quality assessment [9]. Bibliographic measures that rely on citations have served as inspiration for modern Web link analysis algorithms such as PageRank [2]. Citations in a broader sense [1] are the basis of important projects such as the Digital Bibliography & Library Project (DBLP) [2] and the Computer Science Bibliography [3].

Citation management in a digital library involves aspects such as: (i) data cleaning to correct mistakes, such as assignment of improper authorship or splitting of a researcher's production due to the use of multiple names in publications; and (ii) removal of duplicates, mainly after data integration or data input tasks. Most of the techniques to perform these tasks rely on the assumption that we can correctly identify main components within a citation, such as authors' names, title, publication venue, year, pages, etc. This, although is not an easy task due to a variety of reasons such as [17]: data entry errors, various citation formats, lack of (the enforcement of) a standard, imperfect citation gathering software, common author names, abbreviations of publication venues and large-scale citation data.

In this paper we propose a knowledge-base approach to help extracting the correct components of citations in any given format. Differently from related approaches such as [7, 6] that rely on manually built knowledge-bases (KBs) for recognizing the components of a citation, in our case, such a KB is automatically constructed from an existing set of sample metadata records from a given area (e.g., computer science or health sciences). Such sample metadata records are very easy to obtain nowadays, for instance, collected directly from the web or harvested from open archives. The extraction process in our technique is based on: (1) estimating the probability of given term found on a citation

JCDL'07, June 18–23, 2007, Vancouver, British Columbia, Canada.

[1]Here interpreted as a set of bibliographic information such as author name, title, publication venue, or year that are pertinent to a particular article.
[2]`http://www.informatik.uni-trier.de/~ley/db`
[3]`http://liinwww.ira.uka.de/bibliography`

to occur as a value of a given citation field according to the information encoded in KB, and (2) the use of generic structural properties of bibliographic citations. This means that our approach does not rely on patterns encoding specific delimitators of a particular citation style. This assigns to our technique a high degree of automation and flexibility, as demonstrated by experiments we have performed and reported here.

It is also worth noting that our proposed technique can be considered unsupervised, as it does not rely on a learning method that requires a, sometimes very expensive, training phase. It can be applied in any bibliographic citation field as long as a knowledge base can be constructed, which, as mentioned before, is easily done with relatively little effort as we shall see.

To demonstrate the effectiveness and applicability of our proposed approach we have run experiments in which we applied it to extract information from citations in papers of two different domains. In the Computer Science area we used data from the ACM Digital Library and in the Health Sciences area data from several journal papers sponsored by the U.S. National Institutes of Health (NIH). To build the knowledge base we used, in the CS case, data from several distinct BibTex files collected from the Internet with no particular selection method. In the Health Sciences case, we used PubMed Central (PMC) a free digital archive of biomedical and life sciences journal literature. Results of these experiments indicate that our method was able to correctly extracted, in average, over 94% of the fields values present in the citations. In addition, for more than 82% of the citations the extraction was perfect, with all fields correctly extracted.

This paper is organized as follows. Section 2 covers related work. Section 3 gives background on the concepts used in our approach and presents in details the proposed method. Section 4 describes our experiments and discusses results. Section 5 concludes the paper giving directions for future work.

2. RELATED WORK

In past years, several tools have been proposed to address the issue of data extraction from textual documents, with a focus on documents available on the Web. A brief survey on such tools is presented in [15]. In general, these approaches are based on several distinct techniques such as HTML structure analysis [5, 1, 22, 18], natural language processing [8, 20, 23], machine learning [11, 13], data modeling [14] and ontologies [7].

A research initiative closely related to what we present in this paper is the one carried out by the Embley et. al on ontology-based data extraction [7]. This approach uses a semantic data model to provide an ontology that describes the data of interest, including relationships, lexical appearances and context keywords. By parsing this ontology, a relational database schema and a constant/keyword recognizer are automatically generated, which are then used to extract the data that will populate the database. While most approaches rely on the textual context surrounding the data of interest, the ontology-based approach relies mainly on the expected contents of the pages, according to what was anticipated by a pre-specified ontology built by a specialist. If the ontology is representative enough, the extraction process is fully automated. In this case, the extraction process is inherently resilient (i.e., it works properly even if the formatting features of the source documents change) and adaptable (i.e., it works for documents from many distinct sources belonging to a same application domain).

Specifically in the DL realm, a fast growing related area of research which has been gaining much attention recently is the area of automatic metadata extraction. Han et. al. describe a Support Vector Machine classification-based method for metadata extraction from the header part of research papers and show that it outperforms other machine learning methods on the same task [10]. MetaExtract is a system to automatically assign Dublin Core + GEM metadata using extraction through natural language processing techniques applied to educational documents [24]. In [12], the authors focus on title extraction from general documents (e.g., presentations, book chapters, technical papers, brochures, reports, and letters). Paynter [21] focus on the evaluation of automatic metadata assignment tools and discuss its advantages and limitations. In [6] an approach is proposed for metadata extraction based on an ontological knowledge representation framework called INFOMAP. This approach, similarly to [7], requires an ontology to be built, in this case with the help of the Compass editing tool. The authors report good extraction results considering 6 different (although fixed) citation patterns for journal papers only.

3. THE FLUX-CIM METHOD

In this section, we present the details of our citation metadata extraction method: FLUX-CiM. We begin by providing some concepts and definitions used throughout the discussion. Next, we discuss each step that composes our method. First, we discuss the *blocking* step, in which a citation string containing the metadata to be extracted is split in syntactic units called *blocks*. Next, we discuss the *matching* step, which attempts to associate a citation metadata field to each block based on the information available on the knowledge base. After this, we discuss the *binding* step, in which blocks left unassociated in the previous step are further analyzed for associations based on their relative position on the citation string. Finally, we discuss the *joining* step in which blocks are joined to form the values of fields that compose a metadata record.

3.1 Basic Concepts

Knowledge Base

A knowledge base is a set of pairs $KB = \{\langle m_1, O_1\rangle, \ldots, \langle m_n, O_n\rangle\}$ in which each m_i is a distinct bibliographic metadata field, and O_i is a set of strings $\{o_{i,1}, \ldots, o_{i,n_i}\}$ called *occurrences*. Intuitively, O_i is set of typical values for field m_i.

The process of building a knowledge-base is trivial. Given a set of bibliographic metadata records for a given area, we simply process each record, and, for each field we found, we extract the values as occurrences. We notice that this process requires no human effort for selecting some form of "gold standard" records. Indeed, the process is most likely to be carried out automatically by using format conversion. Thus, since there is no human driven training involved, we regard our method as completely unsupervised. Regarding implementation, in the prototype we used for our experiments, the knowledge base is represented as an inverted index composed by the terms found in the occurrences.

$$KB = \{ \quad \langle Author, O_{Author} \rangle, \langle Title, O_{Title} \rangle \}$$
$$O_{Author} = \{ \quad \text{"J. K. Rowling","Galadriel Waters", "Beatrix Potter"} \}$$

$$O_{Title} = \{ \quad \text{"Harry Potter and the Half-Blood Prince",}$$
$$\text{"A Guide to Harry Potter", "Petter Rabbit's Halloween"} \}$$

Figure 1: A sample knowledge base.

In Figure 1 we present a very simple example of a knowledge base, which includes only two metadata fields: *Author* and *Title*.

Citation String

A *citation string* is a text portion encompassing a complete citation from the list of citations in a file. In our method, citation strings are obtained using simple format converters that extract text from files in PDF and other popular formats. In Figure 2(a) we present an example of a citation string.

p-delimiters

a *p-delimiter*, or *potential delimiter character* is any character other than `A`,...,`Z`, `a`, ...,`z`, `0`,...,`9`. We notice that we do not assume p-delimiters as field delimiters intrinsically. Instead, as explained below, we keep track of them to verify if they indeed are used as delimiters in the citation string being processed.

3.2 Method Steps

3.2.1 Blocking

The first step in our extraction method consists of splitting a citation string into substrings we call *blocks*. Let p_l and p_r be p-delimiters and C be a citation string. A block b is an string containing no p-delimiters that occurs in a sequence $p_l b p_r$, or $b p_r$ where b is a prefix of C, or $p_l b$ where b is a suffix of C.

In our method, we consider blocks as sets of terms that will compose a value of a certain field. In a same citation string, there could be more than one block that will be associated to a same field. In Figure 2(b) the blocks identified for our example citation string are marked with rectangles. The rationale behind the idea of identifying blocks is the observation that, in general, in a citation string, every field value is bounded by a p-delimiter, but not all p-delimiters bound a field.

3.2.2 Matching

The matching step consists of associating each block with a bibliographic metadata field. To accomplish this, we match each block against the occurrences composing the knowledge base and evaluate to which field the block is more likely to belong to. For certain terms this is very easy to accomplish. For instance, the term "procedure" is clearly unrelated to all fields but *Title*. In other cases, we have ambiguous terms and we need to use the occurrences to estimate the degree of ambiguity of terms with respect to the fields on the knowledge based. For instance, consider the simple knowledge base in Figure 1. In these occurrences, the term Pottter is considered ambiguous, since it is found in both occurrences *Author* and *Title*. On the other hand, the term Halloween is typical of the *Title* occurrences, and thus unambiguous.

To account for this, we use for the matching a function we call *FF* (Field Frequency), which is an adaptation of the AF function proposed in [19]. The FF function is defined below.

$$FF(b, m_i) = \frac{\sum_{t \in T(m_i) \cap T(b)} fitness(t, m_i)}{|T(b)|} \quad (1)$$

where $T(m_i)$ is the set of all terms found on the occurrences of metadata field m_i, and $T(b)$ is the set of terms found in block b.

The FF function estimates the probability of b being a part of an occurrence of m_i, by evaluating how typical the terms in b are in the occurrences of this field according to the knowledge base. For this, a *fitness measure* is defined as follows.

Given an ambiguous term, the *fitness* function attempts to measure how typical this term is in each field where it occurs. For instance, in the occurrences of Figure 1, the ambiguous term `Potter` is more typical in field *Title* than in field *Author*.

The fitness measure is computed by the following formula:

$$fitness(t, m_i) = \frac{f(t, m_i)}{N(t)} \times \frac{f(t, m_i)}{f_{max}(m_i)} \quad (2)$$

where $f(t, m_i)$ is the number of occurrences $o_{i,k} \in O_i$ associated with field m_i in the knowledge base which contain the term t, $f_{max}(m_i)$ is the highest frequency of any term among the occurrences $o_{i,k} \in O_i$, and $N(t)$ is the total number of occurrences of term t in the knowledge base.

The first fraction in Equation 2 expresses the probability of term t be part of an occurrence of m_i in the knowledge base. Such probability would be suitable for our purposes with all m_i had the same number of occurrences in the knowledge based. As this not true in general, fields with more occurrences would tend to have higher probability values. Therefore, we add the second fraction, as a normalization factor to avoid this problem. This fraction gives the frequency of t in occurrences of m_i normalized by maximum frequency of a term in occurrences of m_i. Thus, it varies from 0, which means completely infrequent, to 1, which means this is the most frequent. This normalization is also useful for making the frequency values comparable among all fields.

Thus, for each block b in the citation string, we calculate $FF(m_i, b)$, for every field m_i in the knowledge base. Finally, b is associated to the field which gives the maximum FF value.

After the matching step, most of the blocks are associated to one of the fields in the knowledge base. We refer to these blocks as *matched*. However, there may also occur *unmatched* blocks, that is, some blocks may remain unassociated to any field after the matching phase. This situation occurs with blocks composed by terms not present on the occurrences in the knowledge base.

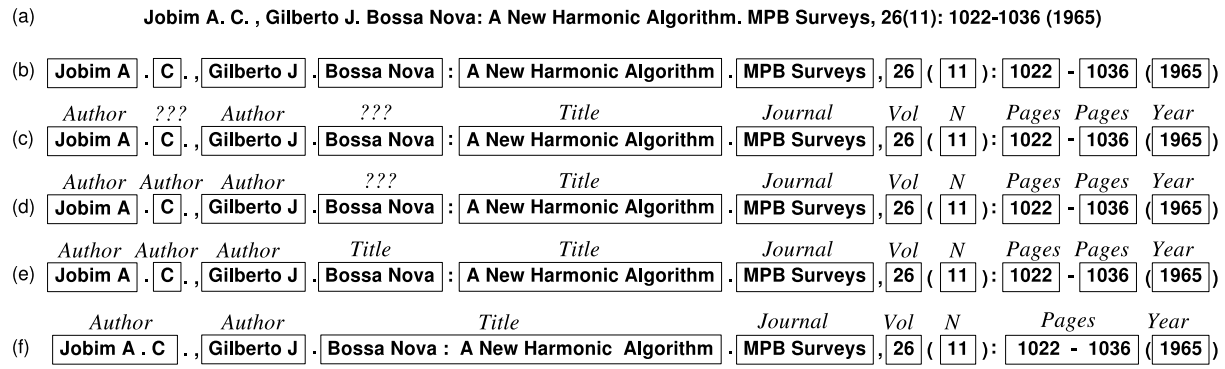

Figure 2: An sample citation string (a) and the extraction steps: blocking (b), matching (c), binding (d and e), and joining (f).

In Figure 2(c) we exemplify the output of the matching step. In this figure, unmatched blocks are labeled with *???* and matched blocks are labeled with the names of their corresponding fields. Cases such as these must be addressed, and this is the task carried out by the `binding` step explained in what follows.

3.2.3 Binding

In the matching step, several blocks were associated to a field from the knowledge base. Based on this information, the binding step will associate remaining unmatched blocks with fields. In Figure 2(c), we illustrate two cases of single unmatched blocks (marked with *???*). However, in general, there could be a sequence of unmatched blocks that need to be associated to some field. The way we solve this problem depends on the neighborhood of the sequence of unmatched blocks on the citation strings. There are three distinct cases we consider: *homogeneous neighborhood*, *partial neighborhood* and *heterogeneous neighborhood*. For each of these cases, we detail below the specific binding strategy adopted.

Homogeneous Neighborhood

Let l and r be matched blocks associated to a same field m. Suppose these blocks occur in a sequence $l, p_0, u_1, p_1, \ldots, u_n, p_n, r$, in which each u_i is a unmatched block and each p_i is p-delimiters. In this case, all u_i will be associated to m. An example of homogeneous neighborhood is illustrated in Figure 2(c), where the block containing the term "C" is associated to *Author* in Figure 2(d) since both of its neighbors are associated to this field.

Partial Neighborhood

Let b be a matched block associated to field m. Suppose this block occur in a sequence $I = u_1, p_1, \ldots, u_n, p_n, b$ or in a sequence $F = b, p_0, u_1, p_1, \ldots, u_n$, in which each u_i is a unmatched block and each p_i is a p-delimiter. In this case, all u_i will be associated to m. Notice that in I, blocks u_i begin the citation string, while in F, blocks u_i end the citation string.

Heterogeneous Neighborhood

Consider the example in Figure 2(c), where we must decide whether the block containing "Bossa Nova" should be associated to *Author*, as the block on the left, or to *Title* as the block on the right.

In such situations, our method resorts to the available p-delimiters surrounding the unmatched blocks, and verifies if (1) they are typically found between contiguous blocks of distinct fields; or (2) they are typically found between contiguous blocks of a same field. In the first case, we regard the p-delimiter as being indeed a field delimiter, and, thus, the two blocks it separates cannot be associated to the same field. In the second case we regard the p-delimiter as being simply a character that appears in values of a field, and, thus the two blocks it separates are likely to be associated to a same field. This verification is carried out based on the results of the matching step for a set of citations, where several blocks are labelled with their corresponding field. Then, we can analyse how common a p-delimiter is for each field and how they tipically behave, i.e., which of the cases (1) or (2), described above, apply.

For instance, in Figure 2, because "." is likely to be a delimiter between *Author* and *Title* and ":" is likely to be a character occurring in values of *Title*, we would choose to associate "Bossa Nova" to *Title* rather than to *Author*. These ideas are elaborated in the following.

Consider the sequence $l, p_0, u_1, p_1, \ldots, u_n, p_n, r$, where l and r are matched blocks associated to distinct fields m_l and m_r, respectively, u_i are unmatched blocks and p_i are p-delimiters. Our problem is to determine, for each u_i, whether it will be associated to m_l or to m_r.

First of all, we consider that only one of the p-delimiters p_i is indeed a field delimiter. Based on this, once we find that some p_i is a field delimiter, then we associate all unmatched blocks u_j $(0 < j \leq i)$ to m_l, i.e., same field as the block on the left, and we associate all u_k $(i > k \geq n)$ to m_r, i.e., same field as the block on the right.

Now, consider the following expressions:

$$T(p_k, m_l, m_r) = \frac{f(p_k, m_l, m_r)}{\sum_{p_j \in P} f(p_j, m_l, m_r)} \quad (3)$$

where $f(p, m_l, m_r)$ is the frequency of p-delimiter p between

contiguous blocks associated to fields m_l and m_r by the matching step, and P is the set of all p-delimiters.

$$C(p_k, m) = \frac{f(p_k, m)}{\sum_{p_j \in P} f(p_j, m)} \quad (4)$$

where $f(p, m)$ is the frequency of a p-delimiter p between contiguous blocks associated to a same field m by the matching step, and P is the set of all p-delimiters.

Intuitively, Equation 3 estimates the probability of a given p-delimiter p_i be a delimiter between fields m_l and m_r, while Equation 4 estimates the probability of p_i be a character occurring as part of the values of a field m. It is worth noticing that the frequencies used in these equations are obtained after analyzing each p-delimiter in all citations to be extracted. This is done to ensure that meaningful statistics on the role and position of the p-delimiter are produced.

In our method, these factors are considered for deciding which p-delimiter p_i is the field delimiter in the sequence. For this, we use Equation 5, defined as follows.

$$\begin{aligned} D(p_k, m_l, m_r) = \quad & 1 - [(1 - T(p_k, m_l, m_r) \quad \times \\ & \prod_{0 \le j < k} 1 - C(p_j, m_l) \quad \times \\ & \prod_{k > j \ge n} 1 - C(p_j, m_r)] \end{aligned} \quad (5)$$

where p_k is a p-delimiter.

Given a delimiter p_k, Equation 5 takes into account: (1) the probability of p_k be a typical field delimiter between values of m_l and m_r; (2) the probability of the p-delimiters on the left of p_k be part of the values of field m_l; and (3) the probability of the p-delimiters on the right of p_k be part of the values of field m_r.

Thus, the problem of binding the sequence of unmatched blocks within a heterogeneous neighborhood is solved by calculating $D(p_k, m_l, m_r)$ for each p-delimiter p_k in the sequence. The field delimiter is selected as the one for which this equation gives the largest value.

In Figure 2(e), for instance, the block containing the term "Bossa Nova" is associated to *Title*, since $D(\text{“ : ”}, Title, Author) < D(\text{“.”}, Title, Author)$.

3.2.4 Joining

When the binding step is over, each block in the citation string is associated to a metadata field. Then, the last step in our extraction method consists in joining together blocks associated to a same field to form the values of that field. For most of the cases, this step is straightforward to accomplish, since it simply requires joining contiguous blocks associated to a same field. However, joining blocks associated to the *Author* field requires a more careful procedure, since there may be several *Author* values on a citation string. Thus, in this section we describe how we handled joining blocks to form values for the *Author* field. For instance, the *Author* blocks in Figure 2(e), must be joined to form *Author* values illustrated in Figure 2(f).

The solution for this problem relies on the information available on the knowledge base. Let η be the average number of terms in the occurrences of the *Author* field in the knowledge base. We assume that the number of terms found in the values of *Author* in any citation string is approximately equal to η.

Now, consider that there is some set s of strings used as implicit delimiters for separating the values of *Author* in a citation string. For instance, in a given citation, the string "," may be used as a delimiter for all values of *Authors*, except for the last value which is separated by the string "and". In this case, s ={",","and"}. In our method, as mentioned, we rely on the observation that, the number of terms bounded by the strings in s should be approximately equal to η.

Consider a sequence of blocks that must be joined to compose the values of *Author*. Given a set of delimiter strings s, two or more contiguous blocks are to be joined if the p-delimiter p between them is not a delimiter string, i.e, $p \notin s$. Hence, we must determine which p-delimiters compose s.

The solution we adopt is to take candidate sets of delimiters and for each candidate set, evaluate if this set is the one that results in values of *Author* with a number of terms closest to η. For this we define a metric we call *delimiting error* that is based on the difference between the lengths of the values (in number of terms) and the average length found in the knowledge base (η).

$$de(s, a, \eta) = \prod_{x \ \in \ split(s,a)} dif(len(x), \eta) \quad (6)$$

where s is a set of delimiters, a is the portion of the citation string composed by *Author* blocks, and the following auxiliary functions are used:

- $split(s, a)$ returns all substrings of a that are bounded by some delimiter $p \in s$.
- $len(x)$ returns the number of terms in the string x.
- $dif(l_1, l_2) = |l_1 - l_2|$ iff $l_1 \neq l_2$, and $dif(l_1, l_2) = \epsilon_0$ otherwise, where ϵ_0 is a small constant.

Intuitively, given a citation string with a set of *Author* blocks to be joined, Equation 6 calculates a score based on the distance between η and the number of terms of each *Author* value obtained when using s as the set of delimiters.

Thus, let P be the set of p-delimiters between *Author* blocks. We evaluate the delimiting error for each subset of p-delimiters $s \subseteq P$ using Equation 6. The set of delimiters used for *Author* values will be the one with smallest delimiting error.

As an example, consider the citation in Figure 2(e), in which the set of delimiters between *Author* blocks is {".",","}. Also, assume $\eta = 2.7$ [4]. When the delimiter "," is used as a separator of *Author* values, the delimiting error is about 0.21, while using the delimiter "." or the delimiter set {".",","}, the delimiting error is about 0.83 in both cases. Thus, the delimiter "," is the best choice. In Figure 2(f) we show the *Author* values obtained with this delimiter.

4. EXPERIMENTS

In this section, we present the experiments we have performed to evaluate our approach on the task of extracting metadata from citation strings. We apply our method in the domains of *health sciences* (HS) and *computer science* (CS), conducting similar experiments for both. In general, we use a citation metadata collection of a specific domain

[4]This is the actual value we have found in one of the citation collections used in our experiments.

Domain	KB size	# Fields	# Citations
HS	5000	6	2000
CS	1950	10	300

Table 1: Features of the collections used in the experiments.

to generate the knowledge base. Then, we execute our extraction method for a set of citations strings from the same domain. Table 1 presents some features of the collections we have used in our experiments with these domains.

4.1 Setup

For the experiments in the HS domain we have used a controlled and well organized collection of citations from the PubMed Central (PMC)[5]. Each citation record in the PMC collection presents the citation string as well as the metadata record, where the fields of the citation are explicitly identified. We have collected a subset of the PMC collection and separate five thousand citations to build the knowledge base and two thousand citations to compose the set of citations strings for the extraction process. By doing so, we guarantee that there is no overlap between the data on the knowledge base and on the citations set. Therefore, by carrying out this experiment over a controlled collection, we can automatically verify the extraction results for a large number of citation strings.

For the CS field, we have gathered a heterogeneous collection composed by assorted references from several conferences and journals in this area. By using such a collection to build the knowledge based, we aimed at evaluating our approach with a less controlled collection. For the extraction process we have used a set of 300 citation strings randomly selected from the ACM Digital Library. Therefore, we manually verify the experimental results for the CS domain.

As mentioned earlier, to the best our knowledge, there is no other unsupervised, automatic, knowledge-based method in the literature that addresses the problem of citation metadata extraction. We believe that a comparison with machine learning or ontology-based approaches cited in the related work would be interesting. However, this may be unfair, since neither the training data nor the ontologies are available. Therefore, we have made available our experimental data set at `http://www.dcc.ufam.edu.br/~eccv/flux-cim` for the sake of future comparison.

In the experiments we evaluated the extraction results obtained after the matching, binding and joining steps discussed in Section 3. We aim at verifying how each step contributes to the overall effectiveness of our approach. In the evaluation we used the well known precision, recall, and F-measure metrics.

Let B_i be a reference set and S_i be a test set to be compared with B_i. We define precision (P_i), recall (R_i) and F-measure (F_i) as:

$$P_i = \frac{|B_i \cap S_i|}{|S_i|} \quad R_i = \frac{|B_i \cap S_i|}{|B_i|} \quad F_i = \frac{2(R_i.P_i)}{(R_i + P_i)} \qquad (7)$$

[5] `http://www.pubmedcentral.nih.gov/`

4.2 Results

Verifying the Blocking Hypothesis

The first result we report aims at verifying in practice the hypotheses we have formulated regarding blocking, i.e., that, in general, in a citation string, every field value is bounded by a p-delimiter, but not all p-delimiters bound a value. To verify this, we look into the citation in the collections we have used for the experiments and count the field values that are bounded by some p-delimiter. As expected, the HS and CS collections present, respectively, 100% and 99.80% of the field values bounded by a p-delimiter. The value lower than 100% for the CS collection was due to a few cases in which field values are not separated by any p-delimiter.

Block-level Results

We now present results that show how correctly the blocks were associated to their respective fields in our method.

Consider the set of citation strings we used for evaluating the extracting process in a given domain. Let B_i be the set of all blocks in the strings in this set which compose the values of a metadata field m_i. These blocks were used as references to our block-level verification. For the case of the HS experiments, for each m_i, B_i was automatically obtained, since the correct metadata values were available from PMC. In the case of the CS experiments, we had to manually build the B_i sets.

Now, let S_i be the set of blocks associated to m_i after a given step of our method, e.g., matching or binding. The precision and recall obtained with Equation 7 for these experiments are presented in Tables 2(a) for the HS domain and (b) for the CS domain. To compare the outcome of the first two steps of our method, we separately present the results obtained after the matching step and after the binding step, which are cumulative. We also present the number of blocks which were left unmatched after the matching step.

In Section 3 we argue that the matching step is the main step of our approach. To verify this, we notice that, on average, less than 5% of the blocks are left unmatched for both domains. This occurs because blocks that present at least one of its terms occurring on the knowledge base are matched. However, this fact alone would be not enough to guarantee the high precision and recall results obtained, which are due to the suitability of the FF function (Equation 1) we propose for the matching.

The results in Tables 2(a) and (b) also show that the binding step plays an important role in our method, since it was able to significantly improve the results of recall by keeping precision levels very similar to the ones in the matching step.

We notice that field *Author* in CS was an exception with respect to the number of matched blocks, since more than 20% of blocks for this field were left unmatched. This is explained by the fact that in CS domain the way author names are presented is much less uniform than in the HS domain. For instance, in the CS collection, there were many distinct ways of using initials of authors' names. An example of cases found on this collection is the real citation we present in Figure 3. In this citation, it is hard even for humans to correctly separate the author names correctly. We searched for other citation of the same paper and find out that the correct author values are "Clayton Lewis", "D. Charles Hair" and "Victor Schoenberg". Notice that the first value was represented distinctly from the other two.

Field	Matching			Unmatched	Binding		
	P	R	F	Blocks	P	R	F
Author	99.04%	94.33%	0.9662	4.96%	98.89%	99.26%	0.9907
Title	93.71%	90.54%	0.9209	6.17%	92.90%	95.96%	0.9440
Journal	97.51%	89.22%	0.9319	2.22%	97.15%	89.32%	0.9307
Date	99.85%	99.50%	0.9967	0.35%	99.85%	99.50%	0.9967
Pages	99.90%	99.45%	0.9967	0.35%	99.70%	99.45%	0.9957
Volume	98.53%	99.51%	0.9902	0.20%	97.96%	99.56%	0.9875
Average	98.09%	95.42%	0.9671	2.38%	97.74%	97.17%	0.9742

(a) HS Domain

Field	Matching			Unmatched	Binding		
	P	R	F	Blocks	P	R	F
Author	99,78%	79,29%	0,8836	20,63%	99,82%	98,96%	0,9939
Title	98,11%	90,43%	0,9412	7,83%	97,19%	97,61%	0,9740
Journal	95,80%	97,86%	0,9682	1,43%	95,80%	97,86%	0,9682
Date	99,70%	97,38%	0,9853	2,04%	97,98%	99,13%	0,9855
Pages	97,87%	98,71%	0,9829	1,29%	97,06%	99,14%	0,9809
Conference	100,00%	96,00%	0,9796	0,40%	99,18%	96,40%	0,9777
Place	98,88%	89,85%	0,9415	9,64%	98,48%	98,48%	0,9848
Publisher	100,00%	100,00%	1,0000	0,00%	100,00%	100,00%	1,0000
Number	97,87%	97,87%	0,9787	2,13%	97,87%	97,87%	0,9787
Volume	100,00%	98,25%	0,9912	0,00%	100,00%	98,25%	0,9912
Average	98,80%	94,56%	0,9652	4,54%	98,34%	98,37%	0,9835

(b) CS Domain

Table 2: Block-level precision and recall for each field after the matching and after the binding steps for (a) the HS domain and for (b) the CS domain. The percentage of unmatched blocks after the matching step is also presented.

```
Lewis, Clayton, D. Charles Hair, and Victor
Schoenberg (1989). Generalization Consistency
Control. In Proceedings of ACM CHI'89
Conference on Human Factors in Computing
Systems. pages 1-5.
```

Figure 3: Example of a real citation found on the CS collection.

Field-level Results

Now, to demonstrate the effectiveness of the whole extraction process with our method, we evaluate the extraction quality after the joining step, where blocks are joined to compose the values of fields. Here, instead of blocks, we analyze for each field occurring in the citations, if the values assigned by our method to this field are correct. This is important specially for the *Author* field to check if the blocks associated to this field were correctly joined, i.e., if terms from the same author names were joined in the same field value.

In this case, we redefine Equation 7, by considering B_i the set of complete values of m_i and S_i the set of complete values associated to m_i by our method. Again, the B_i sets were automatically obtained in the case of the HS domain, and manually built in the case of the CS domain. The results are show in Table 3 for the HS (a) and CS (b) domains. Notice that precision and recall are defined here for complete field values. Thus, if at least one block of the m_i value was not associated to m_i, we consider that all the m_i value was incorrectly extracted.

From Table 3 (a) and (b) we notice that the high accuracy levels reached after the matching and binding steps remain after the joining steps. Indeed, all but one of the F-measure results were higher or equal to 0.93. The exception was the F-measure value for the field *Title* from the HS, which was around 0.85. A closer look in the values of this field revealed a large overlap with the terms in the values of field *Journal* in this domain. Because of this, some *Journal* blocks were wrongly associated to *Title* in the matching step. This can be observed by looking at the recall value for *Journal* (89.32%) and the precision value for *Title* (93.7%) after the matching step in Table 2(a), which are relatively low. This situation was propagated through the binding step until the joining step.

Citation-level Results

The final aspect we have analyzed in our experiments is how well each citation record was extracted by our method, that is, we want to verify whether the fields composing each record were correctly extracted or not. Notice that while the field-level results presented above involve all values from a given field, regardless of the citations in which they occur, in this section we examine the extraction results on a per-citation basis, averaging the results.

To present these results, consider each reference set B_i as the set of field values in a given citation record C_i. Now, let S_i be the set of field values extracted for C_i by our method. Then, precision and recall are calculated using Equation 7. In Table 4 we presented the average of precision and recall obtained in the experiments with HS (a) and CS (b) domains.

The values in Table 4 were obtained by taking into consideration all the field values occurring in each citation, which may vary for each individual citation. For instance, while some CS citations have fields related to a journal publication only (e.g., *Volume*), others will have fields related to a conference publication only (e.g.,*Place*). These results demon-

Field	Precision	Recall	F-measure
Author	98.57%	99.04%	0.9881
Title	84.88%	85.14%	0.8501
Journal	97.23%	89.35%	0.9312
Date	99.85%	99.50%	0.9967
Pages	99.70%	99.20%	0.9945
Volume	98.20%	98.75%	0.9847
Average	96.41%	95.16%	0.9578

(a) HS Domain

Field	Precision	Recall	F-measure
Author	93.59%	95.58%	0.9457
Title	93.00%	93.00%	0.9300
Journal	95.71%	97.81%	0.9675
Date	97.75%	97.44%	0.9759
Pages	97.00%	97.84%	0.9741
Conf	97.47%	95.45%	0.9645
Place	96.83%	97.60%	0.9721
Pub	100.00%	100.00%	1.0000
Number	97.87%	97.87%	0.9787
Volume	100.00%	98.25%	0.9912
Average	96.92%	97.08%	0.9700

(b) CS Domain

Table 3: Field-level precision and recall for each field after the joining step for (a) the HS domain and for (b) the CS domain.

Domain	Precision	Recall	F-measure
HS	94.82%	95.10%	0.9496
CS	95.85%	96.22%	0.9604

Table 4: Average citation-level precision and recall for citations after the joining step for (a) the HS domain and for (b) the CS domain.

strate that our method is able to deal with a variety of citation types, without having to rely on a pre-defined set of patterns.

Figure 4, we show a different perspective on the recall results of Table 4. In this Figure, we plot, for each recall level r%, the number of citations for which a recall of at least r% was achieved. Notice, that for more than 82% of the citations, all fields were correctly extracted, resulting in a 100% recall. From the remaining citations, more that 14% achieved a recall greater than 60%. This result indicates that our method can be deployed in practical situations for helping to manage citations in a digital library.

In our final experiment, we verify how the performance of our method behaves when the size of the knowledge base varies. The result of this experiment is presented in Figure 5, in which for the HS domain, we used an increasing number of sample citation metadata records, from 50 to 10000, and calculated the citation-level F-measure resulting from running the extraction process over the HS citation collection. As this curve shows, the F-measure quickly stabilizes, reaching 0.95 with 3000 sample citation records, and this value remains the same until 10000 sample citation records. This shows that our method does not require a large knowledge base to reach a good extraction quality in the HS collection we use.

Although the experimental results we have presented here demonstrate the high effectiveness of our proposed method, the problem of citation extraction is still a challenge, mainly

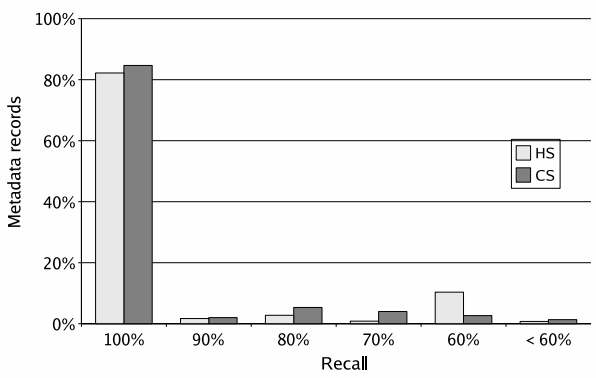

Figure 4: Recall values achieved in citation extraction.

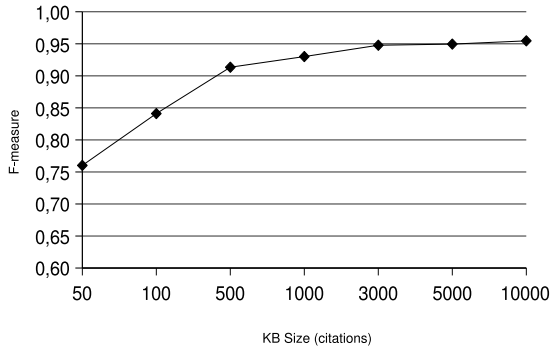

Figure 5: Behavior of the citation extraction performance with the increasing of the knowledge base.

due to some pathological cases that would prevent any method from achieving a perfect result.

In Figure 3 we have shown one of such cases, and Figure 6, another case of a real citation in the HS collection we use is presented. In this citation, one of the values of the field *Author* is "Pathology Review Committee", that can be easily misidentified, for instance, as a value of field *Title*, since the term "Pathology" is typical of this field.

5. CONCLUSIONS AND FUTURE WORK

In this paper we propose a novel approach for extracting components of citations in any given format, which relies on a knowledge base automatically constructed from an existing set of sample metadata records of a given area. The

```
Nagtegaal ID, Klein Kranenbarg E, Hermans J, van
de Velde CJH, van Krieken JHJM, Pathology Review
Committee .  Pathology data in the central
database of multicenter randomized trials need
to be based on pathology reports and controlled
by trained quality managers.  J Clin Oncol.
2000;18:1771-1779.
```

Figure 6: Example of a real citation found on the HS collection.

extraction process in our technique is based on: (1) estimating the probability of given term found on a citation to occur as a value of a given citation field according to the information encoded in the knowledge base, and (2) the use of generic structural properties of bibliographic citations. The effectiveness and applicability of our proposed approach were demonstrated by experiments for extracting information from citations in papers of two different domains: Health Sciences (HS) and Computer Science (CS).In these experiments, we obtained precision and recall over 95% for the fields present in the set of citations and average recall of over 94% for the fields present in each citation. Also important is the fact that for more than 82% of the citations the extraction was perfect, with all fields correctly extracted.

Our approach differs from related knowledge based approaches that rely on manually built knowledge bases for recognizing the components of a citation. Also, our approach does not rely on patterns encoding specific delimitators of a particular citation style, or in some learning method that requires a, sometimes very expensive, training phase. This assigns to our technique a high degree of automation and flexibility, as demonstrated by experiments we have performed and reported here. This occurs even with citations such as the ones found in the CS collection we used in our tests, in which journal and proceedings citations occur with a variety of distinct fields.

Among the paths we intent to explore as future work we may cite to investigate the use of feedback techniques to automatically expand the knowledge base with metadata extracted. This poses an additional challenge since only high quality data can be added to the knowledge base. Otherwise, there is a risk of ruin all future extraction process. We are also considering to investigate the applicability of our method for extracting citations form sources other than citation lists from papers. For instance, it seems interesting to have a mechanism to automatically populate a digital library with metadata directly from web sites of recent conferences.

Acknowledgments

This work is partially supported by projects GERINDO (CNPq/CT-INFO 552.087/02-5), SIRIAA (CNPq/CT-Amazônia 55.3126/2005-9), 5S-VQ (CNPq/CT-INFO 55.1013/2005-2), Tamanduá (FINEP/CT-INFO-Grade-01/2004), FAPEAM/PAPPE and by individual grants from CNPq to Altigran S. da Silva, Edleno S. de Moura, Marcos André Gonçalves and by CAPES to Filipe Mesquita.

6. REFERENCES

[1] A. Arasu and H. Garcia-Molina. Extracting structured data from web pages. In *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 337–348, New York, NY, USA, 2003. ACM Press.

[2] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30(1-7):107–117, 1998.

[3] P. Calado, M. Cristo, M. A. Gonçalves, E. S. de Moura, B. Ribeiro-Neto, and N. Ziviani. Link-based similarity measures for the classification of web documents. *J. Am. Soc. Inf. Sci. Technol.*, 57(2):208–221, 2006.

[4] T. Couto, M. Cristo, M. A. Gonçalves, P. Calado, N. Ziviani, E. Moura, and B. Ribeiro-Neto. A comparative study of citations and links in document classification. In *JCDL '06: Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, pages 75–84, New York, NY, USA, 2006. ACM Press.

[5] V. Crescenzi, G. Mecca, and P. Merialdo. Roadrunner: Towards automatic data extraction from large web sites. In *VLDB '01: Proceedings of the 27th International Conference on Very Large Data Bases*, pages 109–118, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[6] M.-Y. Day, T.-H. Tsai, C.-L. Sung, C.-W. Lee, S.-H. Wu, C.-S. Ong, and W.-L. Hsu. A knowledge-based approach to citation extraction. In *IRI '05: Proceedings of the 2005 IEEE International Conference on Information Reuse and Integration*, pages 50–55, New York, NY, USA, 2005. IEEE Systems, Man, and Cybernetics Society.

[7] D. W. Embley, D. M. Campbell, Y. S. Jiang, S. W. Liddle, D. W. Lonsdale, Y.-K. Ng, and R. D. Smith. Conceptual-model-based data extraction from multiple-record web pages. *Data Knowl. Eng.*, 31(3):227–251, 1999.

[8] D. Freitag and A. McCallum. Information extraction with hmm structures learned by stochastic optimization. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 584–589. AAAI Press / The MIT Press, 2000.

[9] M. A. Gonçalves, B. L. Moreira, E. A. Fox, and L. T. Watson. What is a good digital library? - defining a quality model for digital libraries. To appear in Information Processing and Management, 2007.

[10] H. Han, C. L. Giles, E. Manavoglu, H. Zha, Z. Zhang, and E. A. Fox. Automatic document metadata extraction using support vector machines. In *ACM/IEEE Joint Conference on Digital Libraries, JCDL 2003*, pages 37–48. IEEE Computer Society, 2003.

[11] C.-N. Hsu and M.-T. Dung. Generating finite-state transducers for semi-structured data extraction from the web. *Inf. Syst.*, 23(9):521–538, 1998.

[12] Y. Hu, H. Li, Y. Cao, D. Meyerzon, and Q. Zheng. Automatic extraction of titles from general documents using machine learning. In *JCDL'05: Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries*, Tools & techniques: supporting classification, pages 145–154, 2005.

[13] N. Kushmerick. Wrapper induction: efficiency and expressiveness. *Artif. Intell.*, 118(1-2):15–68, 2000.

[14] A. H. F. Laender, B. A. Ribeiro-Neto, and A. S. da Silva. Debye - data extraction by example. *Data Knowl. Eng.*, 40(2):121–154, 2002.

[15] A. H. F. Laender, B. A. Ribeiro-Neto, A. S. da Silva, and J. S. Teixeira. A brief survey of web data extraction tools. *SIGMOD Record*, 31(2):84–93, 2002.

[16] S. Lawrence, C. L. Giles, and K. Bollacker. Digital libraries and autonomous citation indexing. *Computer*, 32(6):67–71, 1999.

[17] D. Lee, J. Kang, P. Mitra, C. L. Giles, and B.-W. On.

Are your citations clean? new scenarios and challenges in maintaining digital libraries. To appear in Communications of the ACM, 2007.

[18] B. Liu, R. Grossman, and Y. Zhai. Mining data records in web pages. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601–606, New York, NY, USA, 2003. ACM Press.

[19] F. Mesquita, A. S. da Silva, E. S. de Moura, P. Calado, and A. H. F. Laender. Labrador: Efficiently publishing relational databases on the web by using keyword-based query interfaces. *Information Processing & Management*, 2007. Article in Press, Corrected Proof.

[20] I. Muslea, S. Minton, and C. A. Knoblock. Hierarchical wrapper induction for semistructured information sources. *Autonomous Agents and Multi-Agent Systems*, 4(1-2):93–114, 2001.

[21] G. W. Paynter. Developing practical automatic metadata assignment and evaluation tools for internet resources. In M. Marlino, T. Sumner, and F. M. S. III, editors, *ACM/IEEE Joint Conference on Digital Libraries, JCDL 2005, Denver, CA, USA, June 7-11, 2005, Proceedings*, pages 291–300. ACM, 2005.

[22] D. C. Reis, P. B. Golgher, A. S. Silva, and A. F. Laender. Automatic web news extraction using tree edit distance. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 502–511, New York, NY, USA, 2004. ACM Press.

[23] S. Soderland. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34(1-3):233–272, 1999.

[24] O. Yilmazel, Finneran, C. M., Liddy, and E. D. Metaextract: an NLP system to automatically assign metadata. In *JCDL'04: Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries*, Collaboration and group work, pages 241–242, 2004.