Pre-print paper. Accepted for publication in JASIST, June 2016.

Learning to Cite Framework: How to Automatically Construct Citations for Hierarchical Data

Gianmaria Silvello
Department of Information Engineering, University of Padua, Via Gradenigo 6/b, Padua, Italy
silvello@dei.unipd.it
tel. +39 049 827 7500

Abstract

The practice of citation is foundational for the propagation of knowledge along with scientific development and it is one of the core aspects on which scholarship and scientific publishing rely.

Within the broad context of data citation, we focus on the *automatic construction of citations problem* for hierarchically structured data. We present the "learning to cite" framework which enables the automatic construction of human- and machine-readable citations with different level of coarseness. The main goal is to reduce the human intervention on data to a minimum and to provide a citation system general enough to work on heterogeneous and complex XML datasets.

We describe how this framework can be realized by a system for creating citations to single nodes within an XML dataset and, as a use case, show how it can be applied in the context of digital archives.

We conduct an extensive evaluation of the proposed citation system by analyzing its effectiveness from the correctness and completeness viewpoints, showing that it represents a suitable solution that can be easily employed in real-world environments and that reduces human intervention on data to a minimum.

Introduction

"*If I have seen further, it is by standing on the shoulders of giants*". This notorious maxim attributed to Sir Isaac Newton evokes the importance of building on prior results by referring and citing related works in the quest for scientific advancement. As a matter of fact, the practice of citation is foundational for the propagation of knowledge along with scientific development and it is one of the core aspects on which scholarship and scientific publishing rely [Cronin, 1984]. In the traditional context of printed books and journals, citation procedures improved and adapted over recent centuries [Borgman, 2015] and they are now well-understood and established; they have also been successfully resettled to work with digital publications and on-line journals, which resemble traditional journals although they adopt different formats and supports.

Nonetheless, traditional citation procedures cannot be straightforwardly applied to data citation which calls for new methodologies and solutions [Buneman et al. 2014]. Data citation is of upmost importance for giving credit to data curators and for connecting scholarly publications to data with the purpose of sustaining and validating scientific claims and results. In particular, data citation has a fundamental role in the call for better transparency and reproducibility in science [Baggerly, 2010] which has been embraced by several fields such as Astronomy [Kurtz, 2012], Information Retrieval [Arguello et al., 2015], Database Systems [Freire et al., 2012], Biomedical research [AMS, 2015], and Public Health Research [Carr and Littler, 2015], just to name a few.

Data citation has been predominantly analyzed from the scholar publishing and the infrastructural viewpoint. The former has been investigating policies and meanings of data sharing and citation as a support for reproducibility and validation in science [Borgman, 2012a]; the necessity to connect (cite) scientific publications with the data used for supporting the reported results [Lawrence et al., 2011; Callaghan et al., 2012] as in the case of enhanced publications

[Vernooy-Gerritsen, 2009; Bardi and Manghi, 2015]; the role of data journals [Candela et al., 2015]; and, how to give credit to data creators and curators [Borgman, 2012b]. From the infrastructural viewpoint, research has been focusing on the information and publishing infrastructures required to handle dynamic data changing through time [Auer et al., 2012, Pröll and Rauber, 2013], to use of persistent identifiers for the identification and access to data [Simons, 2012], and to realize data repositories to store, preserve and provide access to data [Burton et al., 2015].

Within the infrastructural viewpoint, data citation has started to be considered specifically from the computational perspective [Buneman et al. 2016] further strengthening the necessity to design tools and systems able to automatically construct both machine- and human-readable data citations (i.e., references or citation snippets), to cite data at different level of coarseness, to cite evolving datasets, and to group and structure sets of citations.

In this work, by focusing on XML structured datasets, we tackle the the *automatic construction of citations problem*, which is composed of two key challenges: (i) modeling the referent of a citation and (ii) the automatic generation of citations.

The first challenge requires us to define a general framework for specifying what a citation-to-data should look like and what the elements that compose a citation are. In a traditional setting, citations are structured around well-accepted concepts, for example the elements composing a citation to a journal article may be title, authors, pages, year; data citations by contrast do not fit this framework – the elements structuring a citation may vary from dataset to dataset and may need to be decided on-the-fly by considering the specific characteristics of the dataset being cited. This challenge also comprises the need to cite data at different levels of coarseness, i.e. to produce deep citations [Buneman, 2006]. For instance, if we consider an XML file, then every attribute or data element at any level (the root, an internal node or a leaf) of the XML hierarchy is a viable citable unit[1]. When XML is considered, all relevant information required to construct a citation may be directly available in the citable unit or, more likely, it can be distributed in coarser data elements related to the citable unit.

The second challenge, i.e., the automatic generation of citations, requires defining a methodology to automatically produce data citations because we cannot assume that the people citing the data understand the complexity of the dataset, know how data should be cited in a specific context, and select relevant information to form a complete and correct citation.

To the best of our knowledge, only one solution for addressing the problem of the automatic construction of citations has been defined [Buneman 2006; Buneman and Silvello, 2010], and it is based on a rule-based system to build citations for XML files. This approach exploits the hierarchical nature of XML files to cite data at different levels of coarseness, create human- and machine-readable citations and associate description metadata with the cited data. This approach is computationally efficient and effective for XML, but has some limitations when it comes to being adopted by practitioners: (i) citation rules have to be embedded in the XML files and thus a not negligible amount of work is required to prepare the data in order to make it citable; (ii) the definition of the rules requires both the knowledge of the data domain and XML technology; (iii) heterogeneity of the XML files (e.g. differences in the use of tags, tag nesting and/or the intended tag semantics) directly reflects on the rules that need to be customized to adapt to it, thus general rules may not apply for all the XML files in a given collection.

We propose the "learning to cite" framework, which enables the automatic construction of human- and machine-readable citations to XML data with different level of coarseness, with the final goal of reducing human intervention on data to a minimum and to providing a citation system general enough to work on different data collections. The basic

[1] In this work, any element in a dataset that can be cited is considered a citable unit.

idea is to learn a citation model directly from a given data collection by using a sample set of human-readable citations for training purposes and then exploit such a model to build citations on-the-fly for any citable unit within that collection; we remove the need to set up rules or to prepare the data in order to make it citable. Basically, with the learning to cite framework we are proposing a citation mechanism based on a machine learning approach where knowledge – i.e. what and how to cite – is learned from data, rather than on a knowledge engineering approach where *"knowledge is programmed by human experts* [into systems]*"* [Domingos, 2015] and customized from case to case, when necessary. Learning how to cite data from the data itself allows us to define citation methods which adapt to the diversified citation practices and better fit to a context where "*citation methods tend to be learned by example rather than taught*" [Borgman, 2015].

We instantiate the learning to cite framework by means of a citation system for XML data; this system exploits the hierarchical nature of XML data and the logic behind the XML rule-based system discussed above to automatically learn how to cite any element in a XML file in a given collection.

We conduct an extensive evaluation of this citation system by employing the *Library of Congress* (LoC) collection of archival finding aids[2] encoded in XML (i.e. *Encoded Archival Description* (EAD) files) as test-bed. This collection is well-suited for the evaluation purposes because it is made up of thousands of XML files with different number of nodes, breadths and depths, makes a heterogeneous use of XML elements and attributes, and describes archives with different purposes and containing heterogeneous material. Within this use case the "data" we are considering are in the form of archival descriptions encoded in XML, i.e. EAD files. So, in this context an archival collection of EAD files is a data collection where each single XML element within a file is seen as a datum that may require an individual citation. The archival files are suitable for testing the proposed framework because of their heterogeneity even within the same collection; this heterogeneity is useful to verify the flexibility of the framework because we can test its ability to adapt to structural variations from file to file.

It is worth mentioning that a citation system based on the learning to cite framework produces citations which are not formally exact, but as close as possible to what is considered a "correct citation"; these can be seen as "best match citations" as opposed to the "exact match citations" produced by a knowledge system such as the rule-based one. In order to evaluate best match citations produced by the citation system, we compare them against a ground-truth made up of manually constructed citations and we define evaluation measures to assess the correctness and the completeness of the automatically generated citations.

The rest of the paper is organized as follows: the "Background" section reports on the related works on data citation and some basics concepts about the XML model as well as XML processing and accessing; the "Digital archives: A use case" section presents the use-case we employ in this work; the "Learning to cite framework" section gives an intuitive view of the framework which is then described in detail in the "Training phase" and "Validation phase" subsections; the "Implementation of the framework" subsection details how the system has been implemented from the technological viewpoint; the "Evaluation" section reports on the XML collection employed, how the ground-truth has been created and the experimental results; and finally, the "Conclusions and Future Work" section draws some final remarks and discusses future work.

Background

Note on terminology

[2] http://findingaids.loc.gov/

In this work we adopt the terminology defined in [CODATA-ICSTI, 2013] where the term *citation* is used to refer to the full reference information regarding an object; in traditional print, citations are usually composed of an in-text citation pointer and a full bibliographic reference, which in the digital realm are both referred to with the term citation.
The elements composing a citation are often referred as citation metadata; these metadata could be collected either from the actual data being cited as we do in this work or from some external sources. The actual data being cited can be identified by an organization of elements superimposed over the data or by a query identifying the data.
We consider two types of citation: machine-readable citation and human-readable citation. The former type refers to a citation which is machine actionable (e.g., it can be used to retrieve and access an element of the citation in the cited dataset) and automatically interpretable such as a set of XPath [W3C, 2007] if we are citing an XML file; the latter refers to a text-based citation readable by a human that can be seen as the digital counterpart of a traditional print reference. We assume that from a machine-readable citation it is always possible to create a human readable citation, for instance if we consider a machine-readable citation composed of a sequence of XPaths, its human-readable equivalent will be composed of the text elements identified by each single XPath.

Principles and methods for data citation

Data citation is a complex problem that can be tackled from many perspectives and involves different areas of information and computer science. Several international initiatives have focused on the definition of the core principles for data citation which can also be seen as a set of conditions that any data citation solution should meet. The work on these principles has been carried out by several groups [Brase et al., 2014]. The most relevant initiatives include the International Council for Science: Committee on Data for Science and Technology[3] which in 2013 published a major report [CODATA-ICSTI, 2013] on data citation principles; and FORCE 11 (The Future of Research Communications and e-Scholarship)[4] which in 2014 published a list of principles as the synthesis of the work of a number of working groups (which also included some CODATA representatives) [FORCE11, 2014]. These principles can be classified into two main groups: the former states the importance of data citation in scholarly and research activities and the latter defines the main guidelines a data citation methodology should respect. The former group includes three important principles: the importance of data as it is a first-class product of research and it must be cited and citable as other research objects; the need to give credit and attribution to data creators and curators as it is granted to authors of traditional publications; and the importance of connecting a scientific claim with a citation to the data on which such a claim is based. The latter set of principles states that a citation must guarantee four criteria: the *identification* and access to the cited data, in particular the citation should be machine-actionable and provide access also to the metadata or documentation that are required both by humans and machines to use the data; the *persistence* of data identifiers as well as related metadata; the *completeness* of the reference, meaning that a data citation should contain all the necessary information to interpret and understand the data even beyond the lifespan of the data they describe; and the *interoperability* of citations that should be usable both by humans and machines coming from different communities with different practices.
These principles highlight the importance of providing access to the cited data as well as of defining a complete and persistent reference that can be understood by both humans and machines [Starr et al., 2015]. In particular, references should be self-contained and sufficient to sustain a claim based on the cited data as well as to understand the data given that they may outlive the data itself.

[3] http://www.codata.org/task-groups/data-citation-standards-and-practices
[4] http://www.force11.org/

Several studies [Klump et al., 2015; Simons, 2012] focus on the use of persistent identifiers such as Digital Object Identifiers (DOI), Persistent Uniform Resource Locator (PURL) and the Archival Resource Key (ARK). The main goal of these works is to target the identification problem of cited data by providing a unique and persistent means to identify and retrieve the cited data. The use of persistent identifiers provides us with a pointer to the data to be cited and is an important component of any data citation solution. On the other hand, it addresses just one facet of the problem leaving several other open, such as how to handle citations with variable granularity a.k.a. *deep citations* [Buneman, 2006] where we may need to cite a whole dataset, subset of data or a single datum; in this case providing a persistent identifier for each datum in a dataset may be unfeasible. For this reason, the use of persistent identifiers, their study and evaluation is mainly related to the publishing of research data [Klump et al., 2015; Mooney and Newton, 2012] in order to provide a handle for subsequent citation purposes rather than a data citation solution itself.

The learning to cite framework makes use of persistent identifiers to retrieve the dataset to be cited – i.e. an XML file in this particular instantiation of the framework – and then exploit different means to retrieve the specific cited datum within the dataset; on the other hand, the whole automatic methodology defined for generating both machine- and human-readable citations is agnostic to the use of persistent identifiers.

Data citation systems

Many of the existing approaches to data citation allow us to reference datasets as a single unit having textual data serving as metadata source. As pointed out by [Proll and Rauber, 2013] most data citations "*can often not be generated automatically and they are often not machine interpretable*"; furthermore, most data citation approaches do not provide ways to cite datasets with variable granularity.

The problem of how to cite a dataset at different levels of coarseness has been tackled by Pröll and Rauber [Pröll and Rauber, 2013] who proposed an approach relying on persistent and timestamped queries to cite relational databases and implemented to work also with Comma-Separated Values (CSV) files [Pröll and Rauber, 2015b], by Silvello [Silvello, 2015] who proposed a methodology based on named meta-graphs to cite RDF sub-graphs, and by Buneman and Silvello [Buneman and Silvello, 2010] who proposed a rule-based citation system for XML. The work by Pröll and Rauber is focused on defining a scalable system to cite data with variable granularity by handling their dynamicity, and they do not target the problem of producing human-readable and machine-actionable citations by considering the completeness requirement. Silvello's solution for RDF graphs targets the variable granularity problem and proposes an approach to create human-readable and machine-actionable data citations even though the actual elements composing a citation are not automatically selected.

In [Buneman, 2006; Buneman and Silvello, 2010] a citation system to create machine-actionable citations to XML data is described; this system creates citations by using only the information present in the data. Given an XML file, this rule-based system requires identifying the nodes corresponding to citable units and tagging them with a rule that is then used to generate a citation; the form of the rule is $C \leftarrow P$ where C provides a concrete syntax of a human or machine-readable citation and P is an XPath augmented with decorated variables. The purpose of P is to bind the decorated variables in order to use them in C. Once the given XML file has been prepared to be cited (i.e. the rules are in place), the citation of a citable unit within this file is generated by a conjunction of the rules (i.e. XPaths) retrieved from the node corresponding to the citable unit up to the root of the XML file. Basically, the system gathers all the rules in the path from the citable unit to the root and each rule contains a specification of the elements to be comprised in the citation that has to be generated. This system allows the automatic generation of both human- and machine-readable citations and these citations are *exact* because they contain all and only the required information which were specified

by the expert who defined the rules.

The main drawback of this approach is that the rules have to be defined by hand and they require the active involvement of an expert(s) (data creators and data curators) of the dataset who also need to know XML syntax. A set of rules has to be defined and/or customized (potentially) for several XML files within a collection thus requiring a high amount of resources that may impair the employment of such system in a real-world environment. The learning to cite framework we propose builds on this approach by exploiting its model and its efficient implementation, but overcomes its main drawback by easing the creation of rules by lowering the barriers (resource- and knowledge-wise) to adopt and use such a citation system in a real-world application.

XML and XPath

The *eXtensible Markup Language* (XML) is widely used to mark-up documents with a meaningful semantics and it is the *de-facto* standard for data exchange on the Web. An XML document is seen as a tree structure of nodes of three main types: elements, attributes and text nodes. Element nodes have a name (*label*) and may carry text; they are internal nodes and thus may have child nodes. Attributes have a name and carry text, whereas text nodes carry text, but do not have a name; both are external nodes. Elements are associated with an index determined by the order of the sub-elements in the documents; as an example, if the element e has three children, say e_a, e_b, e_c, e_a has index 1, e_b has index 2 and e_c has index 3. Attributes are not associated with an integer index and they can be identified because their names are unique within an element.

In Figure 1 we can see a sample XML file taken from the Library of Congress finding aid collection[5] and its tree representation. We can see that the XML document is composed of seven element nodes, one attribute and three text elements. The element nodes are all internal and the attributes as well as the text nodes are external and do not have sub-nodes. The XML structure allows for uniquely identifying a given element as the sequence of node labels (with indexes) from the root of the tree [Buneman et al., 2002]; we call these sequences *node paths*; note that an attribute can appear only at the end of a node path.

XPath is a language for addressing parts of an XML document; it provides basic facilities for manipulation of several data types (e.g. strings, numbers and Booleans) and adopts a path notation for navigating through the hierarchical structure of an XML document [W3C, 2007].

XPath exploits the tree structure of XML documents and its primary construct is the expression which is evaluated by an "XPath engine" to yield an object that can be a node-set, a Boolean, a number or a string. One of the main kinds of expressions is the so-called *location path* which selects a set of nodes relative to a given node (i.e. context node); the output of evaluating such an expression is the node-set containing the nodes selected by the location path.

Each part of an XPath expression (i.e. *location step*) can be composed of three parts: (i) an axis, which specifies the tree relationship between the nodes selected by the location step and the context node; (ii) a node test, which specifies the node type and expanded-name of the nodes selected by the location step; and (iii) zero or more predicates that can further refine the set of nodes selected by the location step.

[5] `http://findingaids.loc.gov/`

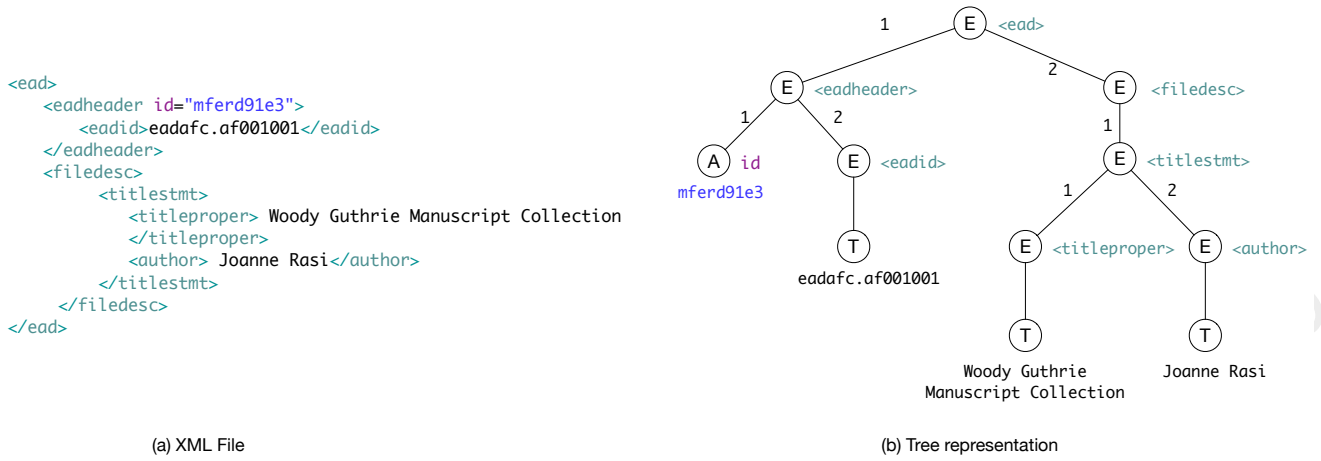


Figure 1. *(a) A sample XML file (*Encoded Archival Description *(EAD) format) inspired by the Library of Congress finding aids collection. (b) The tree representation of the XML file where each node reports the type (E = elements, A = attributes, and T = text), the name or label and the content, if any.*

In this work we specify a node path as an XPath and thus, without loss of generality, we assume that every node in an XML tree can be uniquely identified by an XPath.

From the data citation viewpoint, we can say that every node (element, attribute and text) in an XML document is citable and that there is a unique XPath for each citable node within a given XML document. For instance, the following XPath uniquely identifies the text element with content `Joanne Rasi` in the XML reported in Figure 1(a): `/ead/filedesc/titlesmtm/author`

Digital archives: A use case

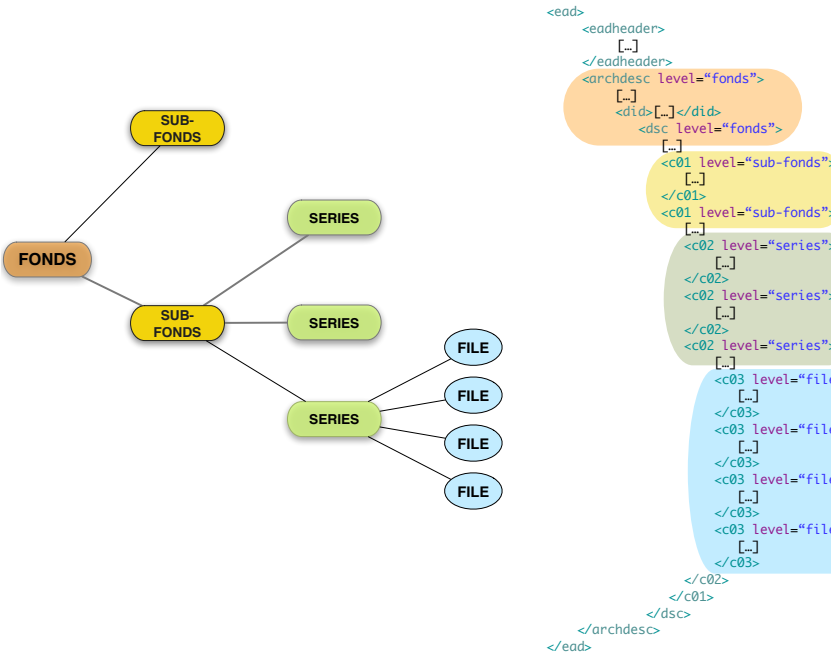
Archives are composed by "*unique records of corporate bodies and the papers of individuals and families*" [Pitti and Duff, 2001]. The original order – i.e. the principle of provenance – of the documents within an archive is preserved because the context and the order in which the documents are held are as valuable as their content [Duranti, 1998].

Archival documents are strongly interlinked and their relationships have to be retained to preserve their informative content and provide understandable and useful information over time. Therefore, archives explicitly model and preserve the provenance of their records by means of a hierarchical method, which maintains the context in which they have been created and their relationships. According to the *International Standard for Archival Description (General)* (ISAD(G)) [ISAD, 1999], archival description (i.e. the finding aids) proceeds from general to specific as a consequence of the provenance principle and has to show, for every unit of description, its relationships and links with other units and to the general fonds, taking the form of a tree as shown in Figure 2 on the left.

The digital encoding of ISAD(G) is the Encoded Archival Description (EAD) [Pitti, 1999], shown in Figure 2 on the right, which is an XML description of a whole archive, reflects the archival structure, holds relations between entities and retains context. We can say that the EAD files closely resemble the description of the archival material and provide a means to represent the internal logic of organization of information in an archive.

EAD is composed of three high-level components: `<eadheader>`, `<frontmatter>` and `<archdesc>`. The `<eadheader>` contains metadata about the archive descriptions and includes information about them such as title, author, and date of creation. The `<frontmatter>` supplies publishing information and is an optional element, while the `<archdesc>` contains the archival description itself and constitutes the core of EAD. The `<archdesc>` may include many high-level sub-elements, most of which are repeatable. The most important element is the `<did>` or descriptive identification which describes the collection as a whole. Finally, the `<archdesc>` contains an element that

facilitates a detailed analysis of the components of a fonds, the <dsc> or description subordinate components. The <dsc> contains a repeatable recursive element, called <c> or component. Components not only are nested under the <archdesc> element, they are also usually nested inside one another. Components usually are indicated with <cN> tag ($N \in \{01,02,\dots,12\}$).



(a) Archival Tree (b) EAD representation

Figure 2. *A sample archival tree and its EAD representation*.

EAD fully enables the expression of multiple description levels central to most archival descriptions and reflects hierarchy levels present in the resources being described, and because of its flexible structure and broad applicability it has been embraced by many repositories [Kiesling, 2001].

On the other hand, EAD allows for several degrees of freedom in tagging practice, which may turn out to be problematic in the automatic processing of EAD files, since it is difficult to know in advance how an institution will use the hierarchical elements. Indeed, [Wisser and Dean, 2013] conducted a thorough study on many EAD repositories and found substantial variability in element and attribute usage both within and across repositories, with variability more prevalent within the <dsc> section of the finding aid which contains the actual descriptions of the archival objects. [Francisco-Revilla et al., 2014] conducted a similar study determining that in half of the studied EAD files the variability in the use of tags and "localized modifications of EAD" had the potential to generate machine handling problems of the files.

The EAD files represent a good test-bed for the citation framework we propose, because: (i) they are deep files not easy to navigate and understand for the users that may need to reconstruct the context of a node to create a suitable and complete citation; (ii) there is a wide variability in the use of tags that makes it difficult to re-use citation rules across files in a collection and even more across collections; (iii) every node (data node or attribute) in an EAD file is a potential citable unit, thus for every EAD file thousands of citations can be generated.

As an example, let us consider the EAD file from the Library of Congress describing the collection of "*Huntington Cairns papers, 1780-1984*"[6] and let us suppose we need to create a citation for the node corresponding to the file

[6] http://hdl.loc.gov/loc.mss/eadmss.ms001024

```
<ead>
     <eadheader id="ms001024">
          <eadid>http://hdl.loc.gov/loc.mss/eadmss.ms001024</eadid>
         <filedesc>
          <publicationstmt>
             <publisher> Manuscript Division, Library of Congress
             </publisher>
          </publicationstmt>
        </filedesc>
   </eadheader>
   <archdesc>
        <did>
            <unittitle>Huntington Cairns Papers</unittitle>
                <dsc>
                     <c01>
                          <did>                                    /ead/archdesc/dsc/c01/did/unittitle
                               <unittitle>Part II: Writings<unitdate>1905-1984</unitdate></unittitle>
                               <container type="box">129-152</container>
                          </did>
                          <c02>
                               <did>
                                    <unittitle>By Cairns</unittitle>
                                    <container type="box">129</container>
                               </did>
                               <c03>
                                    <did>
                                         <unittitle>Books</unittitle>
                                         <container type="box">135</container>
                                    </did>
                                    <c04>
                                         <did>
                                              <unittitle>"The Elements of Legal Theory" (unpublished)</unittitle>
                                         </did>
                                         <c05>
                                              <did>
                                                   <unittitle>Correspondence, 1951-1956</unittitle>
                                              </did>
                                         </c05>
                                    </c04>
                               </c03>
                          </c02>
                     </c01>
                </dsc>
        </did>
   </archdesc>
</ead>
```

Figure 3. *An extract of an EAD file where all and only the nodes required to cite the "Correspondence, 1951-1956" data element are reported.*

containing the "*Correspondence, 1951-1956*". This EAD file is composed of more than 8 thousand nodes, thus in Figure 3 we show a simplified version of it, where we report only the nodes related to the citation and the data information we use to generate it. A complete and correct human-readable citation for "*Correspondence, 1951-1956*" can be the following:

```
Correspondence, 1951-1956, "The Elements of Legal Theory" (unpublished). Book, box 135;
By Cairns, box 129. Part II: Writings (1905-1984), box 129-152. Huntington Cairns Papers,
Manuscript Division, Library of Congress. http://hdl.loc.gov/loc.mss/eadmss.ms001024
```

We can see that the element "*Correspondence, 1951-1956*" is cited within its context; indeed, the citation contains information from the upper levels of the archival tree up to the archival fonds "*Huntington Cairns Papers*"; the citation contains the information required to understand the meaning of the cited element as well as to find in the archive the physical object it describes. Furthermore, the citation also comprises administrative information taken from the `<eadheader>` section of the EAD file such as the unique identifier of the file and the publisher; the citation may also contain the author of the finding aids or similar data if they are available and considered useful for the citation. The actual data composing a citation to an EAD file could be decided *ad-hoc* by the archive responsible or by following

standard guidelines such as those provided by the Purdue University[7] to which we refer in this work; however, the learning to cite framework and the citation system we propose are independent of the choice of the elements composing the citations.

In Figure 4 on the left we show the textual tokens composing the human-readable citation and on the right the corresponding XPath retrieving the required token from the EAD file containing the element to be cited.

Human-Readable Citation	Machine-Readable Citation
http://hdl.loc.gov/loc.mss/eadmss.ms001024	/ead/eadheader/eadid
Manuscript Division, Library of Congress	/ead/eadheader/filedesc/publicationstmt/publisher
Huntington Cairns Papers	/ead/archdesc/did/unittitle
Part II: Writings	/ead/archdesc/dsc/c01[10]/did/unittitle
1905-1984	/ead/archdesc/dsc/c01[10]/did/unittitle/unitdate
box	/ead/archdesc/dsc/c01[10]/did/container/@type
129-152	/ead/archdesc/dsc/c01[10]/did/container
By Cairns	/ead/archdesc/dsc/c01[10]/c02[1]/did/unittitle
box	/ead/archdesc/dsc/c01[10]/c02[1]/did/container/@type
129	/ead/archdesc/dsc/c01[10]/c02[1]/did/container/
Books	/ead/archdesc/dsc/c01[10]/c02[1]/c03[4]/did/unittitle
box	/ead/archdesc/dsc/c01[10]/c02[1]/c03[4]/did/container/@type
135	/ead/archdesc/dsc/c01[10]/c02[1]/c03[4]/did/container
"The Elements of Legal Theory" (unpublished)	/ead/archdesc/dsc/c01[10]/c02[1]/c03[4]/c04[2]/did/unittitle
Correspondence, 1951-1956	/ead/archdesc/dsc/c01[10]/c02[1]/c03[4]/c04[2]/c05[1]/did/unittitle

Figure 4. *The correspondence between the elements of a human-readable and a machine-readable citation. Each XPath is used to retrieve the required textual token in the XML file containing the element to be cited.*

We can see that the machine-readable citation is composed of a conjunction of XPaths uniquely identifying text elements and attributes in the XML file which contains the chosen citable unit; by resolving the XPaths we can straightforwardly generate the corresponding human-readable reference. As a consequence, a human-readable reference contains as many data elements as the number of XPaths in the corresponding machine-readable reference. All the information required to generate the citation is gathered from the element to be cited and the surrounding elements (i.e. siblings, ancestors and descendants). We can see that in order to create a complete citation we need to visit all the nodes of the tree from the citable unit up to the root, which in this case requires us to climb ten levels. The additional difficulty for a user who may need to manually build this citation is to filter out all the non-relevant information these nodes and relative nodes contain.

Learning to cite framework

Overview

The goal of the learning to cite framework is to address the problem of the automatic generation of citations for XML data without requiring any effort from the curators and without any modification to the data that has to be cited. This framework is realized by a system that, given a collection of XML files: (i) takes as input a number of sample human-readable citations created from the XML collection; (ii) learns a model from these citations; and (iii) uses this model to create both human- and machine-readable citations for whatever citable unit within the considered XML collection.

Therefore, the main idea on which the learning to cite framework relies is that we can generate a model from a training set data and then use this model to automatically produce references of potentially unknown elements. The six main conceptual blocks realizing this framework are shown in Figure 5.

[7] http://guides.lib.purdue.edu/c.php?g=352889&p=2378064

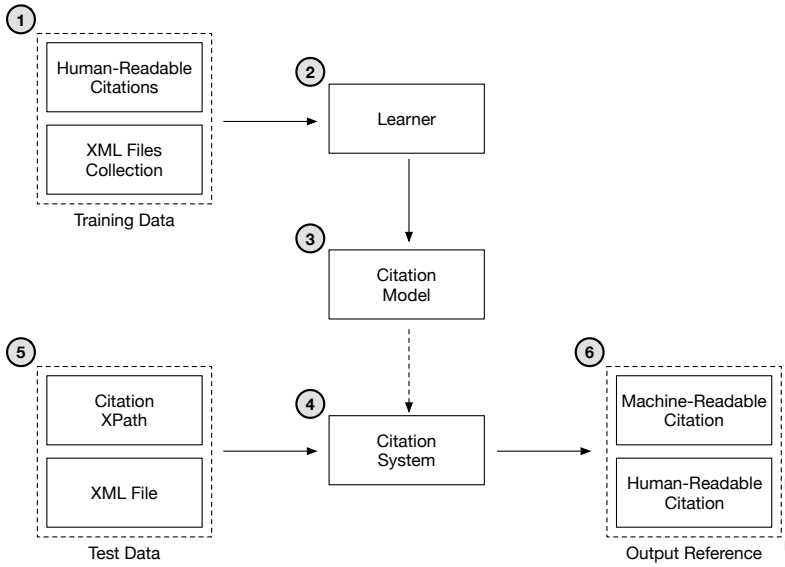


Figure 5. *The building blocks of the "Learning to Cite" framework.*

Training data plays a key role in this context and it is the first block constituting the framework; it is composed of a set of pairs where each pair contains an XML file and a human-readable citation associated to it. The citation is composed of a group of textual tokens, which are the basis for the learner (block 2) to build the model. Each textual token is sought in the associated XML file by employing a retrieval algorithm and the location paths identifying the retrieved elements (there may be none, one or many retrieved elements for each token) are used to build the citation model (block 3). This process is repeated for each textual token composing each citation present in the training set.

Once the whole training set has been processed, we obtain a citation model which is an XML tree made up of the union of the location paths identifying the retrieved elements. This tree is used by the citation system (block 4) to generate the citations of the test data (block 5). The test data is composed of a set of pairs, where each pair contains an XML file and the XPath identifying a citable unit in the file. The citation system parses the given test XPath and produces a set of progressively shorter paths, one for each location step of the test XPath: for each path, the citation system uses the citation model to predict which elements of the test XML file have to be used to generate the final citation. The output reference (block 6) contains a machine- and a human-readable citation; the machine-readable citation is composed of a conjunction of the XPaths identifying the elements in the test XML selected by the citation system and the human-readable citation is composed of the textual tokens retrieved by the XPaths of the machine-readable citation.

Overall, the presented learning to cite framework follows two main phases: the training phase, where the citation model is learned, and the validation phase, where the parameters of the model are optimized.

Training phase

In Figure 6 we can see the main components of the learner which takes the training data as input and produces the citation model as output. The training data is composed of a set of human-readable citations $\mathcal{H} = \{\mathrm{H}_1, \mathrm{H}_2, \dots, \mathrm{H}_\mathrm{m}\}$ and a set of XML trees $T = \{t_1, t_2, \dots, t_n\}$; each citation $\mathrm{H}_i \in \mathcal{H}$ is associated to one and only one tree $t_j \in T$ and each tree has at least one associated citation.

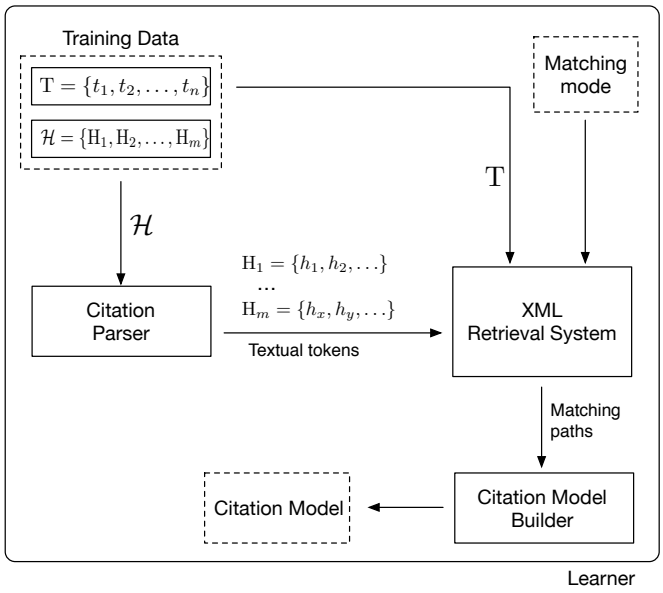


Figure 6. *The blocks composing the* Learner *component of the learning to cite framework*.

The "Citation Parser" parses each citation in $H_i \in \mathcal{H}$ by obtaining a set of textual tokens $h_k \in H_i$ such that $H_i = \{h_1, h_2, \ldots\}$. Given a textual token, say $h_k \in H_i$, and the corresponding tree, say $t_j \in T$, the learner seeks $h_k$in t_j and returns the XPath p_k identifying the token in the tree. The retrieval of a textual token in an XML tree is performed by using one of the following *matching modes* implementing different retrieval algorithms:

1. *Exact match mode*: retrieves those elements containing all and only the words in the given token;
2. *Shallow match mode*: retrieves those elements containing all but not only the words in the given token;
3. *Mixed match mode*: uses the exact match mode first and if no result is returned it uses the shallow mode.

All these three matching modes may return none, one or many elements and the learning system assigns a score to every retrieved element indicating its similarity with the sought textual token; all the elements returned by the exact match algorithm score 1, whereas the elements returned by the shallow algorithm score in the range [0, 1]. Thus, at the end of the retrieval process, for each textual token we end up with a ranked list of elements ordered by similarity score where only those elements whose score is above a given threshold are considered to be relevant. The threshold value and the retrieval mode are set by the learner in the validation phase.

If the path p_k is already present in the citation tree, the score of the element identified by the path is updated by summing the old score with the new one and a new attribute called *frequency* is added to the element.

As an example, let us consider the simple XML reported in Figure 3. In this case, if we are looking for the "129" textual element and we are using the exact match mode, then a single element would be returned, i.e. the one identified by the XPath `/ead/archdesc/dsc/c01/c02/did/container` (i.e., "129"); whereas, by employing the shallow match mode both the elements identified by `/ead/archdesc/dsc/c01/did/container` (i.e., "129-152" which only partially matches the sought textual token) and `/ead/archdesc/dsc/c01/c02/did/container` (i.e., "129") would be returned. In this case the mixed match mode would return the same element returned by the exact match mode. In the shallow mode case, two elements are returned and in this case they would be both considered relevant and used to populate the XML tree.

```
<cite-tree>
    <ead score="0" frequency="0" type="element">
        <eadheader score="0" frequency="0" type="element">
            <eadid score="1.0" frequency="1" type="element"/>
            <filedesc score="0" frequency="0" type="element">
                <publicationstmt score="0" frequency="0" type="element">
                    <publisher score="0.9413807890996732" frequency="1.0" type="element">
                        <extptr score="0.9413807890996732" frequency="1" type="element"/>
                    </publisher>
                </publicationstmt>
                <titlestmt score="0" frequency="0" type="element">
                    <titleproper score="0.9311609000928631" frequency="1" type="element"/>
                </titlestmt>
            </filedesc>
        </eadheader>
        <archdesc score="0" frequency="0" type="element">
            <did score="0" frequency="0" type="element">
                <unittitle score="0.9740960467331898" frequency="1" type="element">
                    <unitdate score="0.9740960467331898" frequency="1" type="element"/>
                </unittitle>
            </did>
            <dsc score="0" frequency="0" type="element">
                <c01 score="0" frequency="0" type="element">
                    <did score="0" frequency="0" type="element">
                        <unittitle score="0.9197168697545395" frequency="1" type="element">
                            <unitdate score="0.9369796895969322" frequency="2.0" type="element"/>
                        </unittitle>
                        <container score="0.9357849740192015" frequency="1.0" type="element">
                            <type score="1.0" frequency="3.0" type="attribute"/>
                        </container>
                    </did>
                    <c02 score="0" frequency="0" type="element">
                        <did score="0" frequency="0" type="element">
                            <container score="1.0" frequency="1.0" type="element">
                                <type score="1.0" frequency="3.0" type="attribute"/>
                            </container>
                            <unittitle score="0.9542425094393249" frequency="1" type="element"/>
                        </did>
                        […]
                    </c02>
                </c01>
            </dsc>
        </archdesc>
    </ead>
</cite-tree>
```

Figure 7. *A part of the XML tree obtained by processing the human-readable reference reported in the Use Case Section and* by using a mixed match mode.

Indeed, the *frequency* attribute indicates how many times an element has been considered relevant within the training set, whereas the *score* attribute quantifies its relevance; note that the score attribute is normalized to be within the [0,1] range.

In Figure 7 we can see an extract of the citation tree created from the human-readable reference shown in Figure 4 above by using a mixed match mode. We can see that all but not only the XPaths reported in Figure 4 are present in the citation tree. For instance, the citation model contains the path identifying the element `container` within the `c01` element which is also present in the machine-readable citation reported in Figure 4. On the other hand, the element `extptr` within the `publisher` element is present in the citation model even though it is not part of the citation; this shows that the citation model comprehends more elements and paths than those exactly matching the elements composing the citation used for building the model.

Given that this citation model is created from just one citation, the score values are all close to one as well as the frequencies; it is clear that one citation is not enough to build a solid citation model and in the Experiment Section we analyze this aspect in order to define the minimum number of citations required to build an effective model.

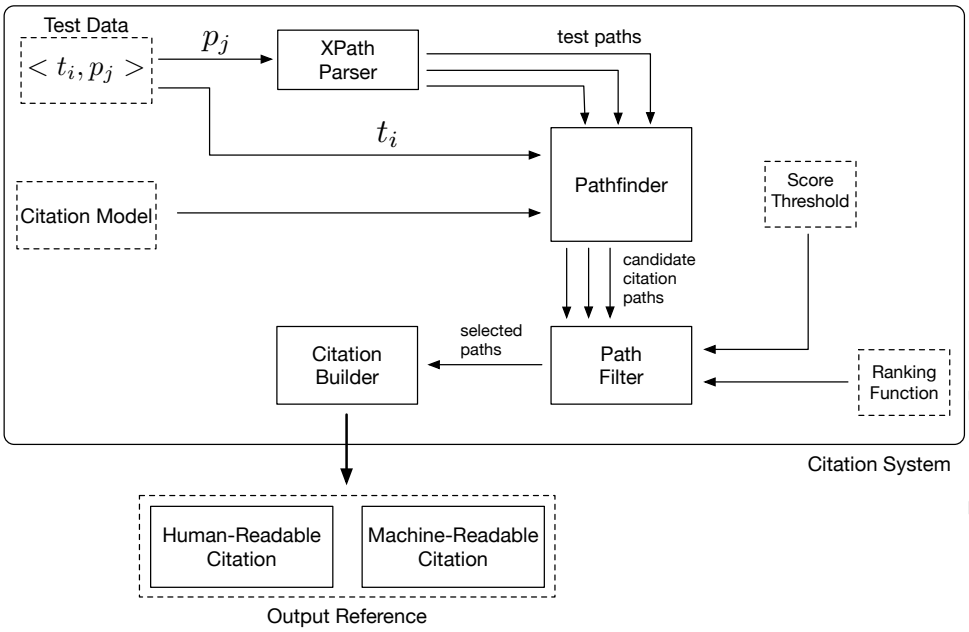


Figure 8. *The blocks composing the* Citation System *of the learning to cite framework and the output reference composed of the human- and the machine-readable reference*.

The citation system, the main components of which are shown in Figure 8, takes the citation model and the test data as inputs. The test data is a pair $\langle p_t, t_t \rangle$ where p_t is the XPath of the citable unit within the XML tree t_t; as an example, referring to the use case presented above, t_t could be the XML tree shown in Figure 3 and p_t could be the XPath

`/ead/archdesc/dsc/c01[10]/did/unittitle`

which identifies the "*Part II: Writings*" element (enclosed in a box in Figure 3) in the test tree.

The first step taken by the citation system is to parse the test path into progressively shorter paths by considering only the labels while ignoring the indexes of the path; each one of these parsed paths may identify an XML element containing relevant data to build the citation. Referring to our example, the "XPath Parser" outputs six XPaths: (1) `/ead/archdesc/dsc/c01/did/unittitle`; (2) `/ead/archdesc/dsc/c01/did`; (3) `/ead/archdesc/dsc/c01`; (4) `/ead/archdesc/dsc`; (5) `/ead/archdesc`; and, (6) `/ead`.

Each path is matched with the citation model by the "Pathfinder" component and if there is an exact match, meaning that the test path is present as it is in the citation model, the XPath of matched element along with the XPaths of its descendants with score and frequency bigger than zero are inserted in a *candidate citation set*; the citation system builds a candidate set for each processed XPath. In our example the XPath (1) will produce a candidate set containing two XPaths: itself and the XPath to the `unitdate` element; whereas the XPath (6) will produce a candidate set containing thirteen XPaths, i.e. all the elements in the citation model shown in Figure 7 with frequency and score greater than zero. In this example, all the tested XPaths have an exact match in the citation model given that this sample model has been built from a single citation of an element in the same tree as the tested one; however, in a real setting an exact match is not always found, so let us think about all those elements which are seen for the first time in the test phase and thus cannot be present in the citation model built in the training phase.

In these cases, a *best match* between the XPath being processed and the citation model is sought by the Pathfinder. Given an XPath, say XPath (1) in the example, the Pathfinder seeks the element identified by the deepest location step (`unittitle` in the example); if there is a match, then the Pathfinder seeks the longest location path within XPath (1) with a match in the citation model; if there is more than one match, then only the longest path is kept. Once a best match is found, then a candidate citation set is created for each XPath parsed from the test path as described for the

exact match case.

At the end of this process, the candidate citation sets become the inputs of the "Path Filter" component which selects the most promising paths from each set which will be the constituent of the final citation. Each path in a candidate set comes with a frequency, a score and a relative depth (*relDepth*) which indicates the distance from the element identified by the candidate path and the element identified by the test path; for instance, the distance between the elements identified by the XPath (6) `(unittitle)` and the candidate path `/ead/archdesc/dsc/c01[10]/did/unittitle/unitdate` is one.

Frequency, score and relative depth are used by the Path Filter to rank the candidate paths and select the ones that will be the constituents of the citation; the rationale is that the paths with higher score, higher frequency and lower relative depth are the most relevant. The Path Filter employs four *ranking functions*:

- *Frequency Score Depth Normalization (FSDN)*: $\frac{score * frequency}{relDepth}$
- *Score Depth Normalization (SDN)*: $\frac{score}{relDepth}$
- *Frequency Depth Normalization (FDN)*: $\frac{frequency}{relDepth}$
- *Frequency Score (FS)*: $score * frequency$

The FSDN rank function takes into account both the frequency and the score of an element and it ranks at higher positions the elements with a high score – i.e., elements that have been exactly matched are preferred to those only partially matched – and high frequency – i.e., elements encountered many times in training set. These values are normalized by the relative depth, penalizing the elements at lower levels in the tree; the rationale behind this choice is that higher elements usually contain data which are more likely to convey useful and general information that the elements at lower levels. SDN and FDN follow the same rationale, but the former one considers only the score whereas the latter considers only the frequency. With SDN we prefer an exact match found just a few times over a shallow match found many times; with FDN this preference is reversed. Lastly, FS does not normalize the score and frequency value by using the relative depth and thus it does not discriminate between matching elements at different depths.

The Path Finder orders the paths in each one of the candidate sets created by the Pathfinder by employing one of these ranking functions and selects only those paths with a value above a given *score threshold*. Note that for selection purposes the ranking values are further normalized in the [0,1] interval.

The ranking function and the score threshold value may be defined *a priori* if we assume some knowledge about the test data or, more likely, may be estimated by the system in the validation phase which optimizes these parameters according to an *optimization measure* of choice. Lastly, the Citation Builder component gathers the selected paths and builds the machine-readable citation as a conjunction of the selected XPaths and the human-readable citation by retrieving the textual tokens from the test tree by using these XPaths.

Validation phase

The validation phase is required when the method of building the citation tree (exact match mode, shallow match mode, mixed match mode), the ranking function selecting the candidate paths (FSDN, SDN, SDN, FS) and the threshold values are not fixed *a priori*, but are set to maximize an optimization measure of choice.

To this end we define three measures to evaluate the performances of the citation system from an effectiveness viewpoint: *precision*, *recall* and *fscore*. These measures assess the quality of a citation generated by the citation system by comparing it with an ideal one which represents the perfect citation for a given element; the ideal citation is also

called *ground-truth citation*.

Let $MC_k = \{p_1, p_2, \dots, p_n\}$ be a machine-readable citation generated by the citation system for the element e_k where $\{p_1, p_2, \dots, p_n\}$ are the paths composing the citation; and let $GTC_k = \{p'_1, p'_2, \dots, p'_m\}$ be the ground-truth machine-readable citation for the same element e_k, where $|MC_k| = n$ and $|GTC_k| = m$.

Then, $Precision = \frac{|MC \cap GTC|}{|MC|}$, $Recall = \frac{|MC \cap GTC|}{|GTC|}$ and $fscore = 2 * \frac{precision * recall}{precision + recall}$.

Precision is the ratio between the number of correct paths that are present in the citation generated by the system and the total number of paths in the citation; thus it evaluates the *correctness* of the automatically generated citation. Precision is in the range [0,1] and can also be indicated as a percentage; for instance, if we achieve 100% precision, it means that all the paths in the system generated citation are present in the ground-truth citation and thus are correct.

Recall is the ration between the number of correct paths in the system generated citation and the total number of correct paths; thus, it evaluates the *completeness* of the generated citation. Like precision, recall is in the [0,1] interval and can be expressed as a percentage. A 100% recall indicates that all (but not necessarily only) the correct paths are present in the system generated citation. fscore is a synthesis measure that balances between precision and recall and weights both the correctness and completeness of a citation; fscore is in the [0,1] interval and can be expressed as a percentage as well. If we achieve a 100% fscore it means that the system generated citation has both 100% precision and 100% recall and thus it contains all and only the correct paths specified by the ground-truth citation.

The learning to cite framework adopts a *k*-fold validation strategy where the training data is randomly partitioned into *k* equal sized subsets. Of the *k* subsets, a single subset is retained as the validation data for testing the model, and the remaining *k*−1 subsets are used as training data. The cross-validation process is then repeated *k* times, with each of the *k* subsets used once as the validation data. The *k* results from the folds are then averaged to produce a single estimation according to a chosen evaluation measure.

Implementation of the framework

The presented learning to cite framework has been implemented in a working system for citing XML data. Basically, this system realizes block 2 (Learner) and block 5 (Citation System) of the conceptual design shown in Figure 5. All the components are implemented in Java 8.0 and the code is open source and publicly available as are all the libraries used for implementing the specific functionalities of the system.

In particular, we adopted the open-source Java-based XML DBMS BaseX 8.3.1[8] for realizing the "Pathfinder" (Figure 6) and the "XML Retrieval System" (Figure 7) components. BaseX is a state-of-the-art Java-based native XML database, which offers both in-memory and secondary-memory storage. BaseX uses compact memory structures and performs compression based on dynamic recognition of data types which, for instance, allows it to determine if a text node is a string or an integer to enable compact storage of the element. Moreover, BaseX provides effective full-text search capabilities which are exploited to perform exact and best match retrieval.

We chose to implement the system in Java in order to make it portable and platform independent; moreover, we employed Apache Maven[9] to simplify the build process and to provide a uniform build system.

The code of the system along with its documentation are publicly available at the following URL: `http://www.dei.unipd.it/~silvello/datacitation`.

[8] http://basex.org/
[9] http://maven.apache.org/

Experimental evaluation

The experimental evaluation we conducted has the aim to:

1. investigate the effect of parameter (matching mode, ranking function and score threshold) tuning and the choice of the optimization measure on system effectiveness;
2. evaluate the citation system from the correctness and completeness points of view;
3. analyze how the training set size and composition impact system effectiveness.

Experimental Collection

We created an experimental collection by building a training and validation set composed of XML tree and human-readable citation pairs and a test set composed of XML tree and machine-readable pairs. Both for the training and validation set and the test set we manually created the ground-truth machine-readable citations used to assess the quality of the citations automatically generated by the system.

We built the experimental collection by using the Library of Congress digital finding aids collection encoded in the EAD format which is publicly available at the following URL: `http://findingaids.loc.gov/`.

The full EAD collection is composed of 2,083 files with a maximum file size of 15 MB and average file size of 200 kB; the whole collection contains about 11 million citable units where on average each file contains 5,368 citable units and the biggest one contains 384,957 citable units. On average the EAD files have depth 10 with maximum depth 18 and minimum 7.

To build the training and validation set, we randomly selected 25 EAD files and for each of these files we randomly extracted 4 citable units; we obtained a set of 100 XPaths identifying an equal number of different citable units. For each citable unit (i.e. XML element), we manually created a human-readable citation to be used to train the citation system and a machine-readable citation to build the ground-truth to be used for validation purposes. We did not impose any constraints on the training and validation sets as for example on the granularity of the selected citable units.

The test set was built by following a similar procedure: from the whole EAD collection minus the 25 files selected for the training and validation set, we randomly selected 50 EAD files and for each one a single citable unit was selected at random. Then, we manually created a ground-truth machine-readable citation for each one of these randomly sampled citable units. Both the training set and the test set contains citable units with granularity ranging from depth 4 to depth 11 and thus the level of coarseness of the citations in the training set and those in the test test are comparable. In general, the training set should contain citations at different level of granularity in order to provide a good coverage of the citations that have to be produced in the test phase.

We created the ground-truth citations by following the guidelines provided by the archives of the Purdue University which follows the Modern Language Association (MLA)[10] citation style. For reproducibility purposes, the training and validation set, the test set and the ground-truth files as well as the code developed to conduct the experiments are openly available at the following URL: `http://www.dei.unipd.it/~silvello/datacitation`.

Results

First-of-all, on the full training and validation set we tested how the citation system behaves with different parameter tunings. In Table I we report the best configurations of parameters based on precision, recall and fscore calculated on the complete training set. We employed a 5-fold cross validation by using 4 folds for training and 1 fold for validation; each measure is calculated as the average over the five times the validation process was repeated. The optimization

[10] https://www.mla.org/

measure used for validating the citation model is the same used for evaluating it.

Table I. *The best configurations of parameters based on three validation measures (precision, recall and fscore) calculated on the complete training set. Each measure value is calculated as the average over the five validation folds; we report the standard deviation (std dev) for each measure.*

Tree Type	Ranking Function	Score Threshold	Avg Precision	Std dev Precision	Avg Recall	Std dev Recall	Avg Fscore	Std dev Fscore
exact	FDN	0.1	0.3789	0.06	0.8975	0.04	0.5231	0.04
exact	FDN	0.5	0.7356	0.01	0.7448	0.03	**0.7316**	0.01
exact	FDN	1.0	**0.7908**	0.04	0.4552	0.05	0.5702	0.04
exact	FS	0.1	0.3813	0.07	0.8962	0.03	0.5196	0.04
exact	FS	0.5	0.6042	0.01	0.6919	0.03	0.6372	0.01
exact	FS	1.0	0.7211	0.02	0.2949	0.05	0.4087	0.03
exact	FSDN	0.1	0.3769	0.06	0.8975	0.04	0.5208	0.04
exact	FSDN	0.5	0.7293	0.01	0.7440	0.03	0.7278	0.01
exact	FSDN	1.0	**0.7908**	0.04	0.4542	0.08	0.5694	0.05
exact	SDN	0.1	0.1845	0.04	**0.9052**	0.04	0.3014	0.04
exact	SDN	0.5	0.2607	0.00	0.7684	0.04	0.3857	0.01
exact	SDN	1.0	0.3564	0.01	0.3411	0.04	0.3411	0.02
mixed	FDN	0.1	0.3186	0.05	0.8942	0.04	0.4631	0.04
mixed	FDN	0.5	0.5957	0.02	0.7111	0.05	0.6403	0.03
mixed	FDN	1.0	0.6115	0.04	0.3636	0.04	0.4477	0.03
mixed	FS	0.1	0.3339	0.08	0.8901	0.05	0.4734	0.06
mixed	FS	0.5	0.6127	0.03	0.6473	0.04	0.6220	0.03
mixed	FS	1.0	0.7028	0.04	0.2990	0.10	0.4095	0.06
mixed	FSDN	0.1	0.3276	0.05	0.8942	0.04	0.4718	0.04
mixed	FSDN	0.5	0.6514	0.02	0.7252	0.05	0.6789	0.03
mixed	FSDN	1.0	0.7746	0.03	0.4472	0.05	0.5581	0.04
mixed	SDN	0.1	0.1469	0.05	**0.9045**	0.04	0.2493	0.05
mixed	SDN	0.5	0.2690	0.01	0.7676	0.05	0.3948	0.01
mixed	SDN	1.0	0.4234	0.01	0.3643	0.05	0.3822	0.02
shallow	FDN	0.1	0.1630	0.04	0.8679	0.04	0.2719	0.04
shallow	FDN	0.5	0.3645	0.02	0.2670	0.04	0.2973	0.03
shallow	FDN	1.0	0.4393	0.04	0.1817	0.03	0.2484	0.03
shallow	FS	0.1	0.1451	0.07	0.8647	0.04	0.2455	0.05
shallow	FS	0.5	0.2080	0.02	0.4693	0.04	0.2814	0.03
shallow	FS	1.0	0.4437	0.05	0.1731	0.06	0.2432	0.05
shallow	FSDN	0.1	0.1496	0.06	0.8673	0.04	0.2527	0.04
shallow	FSDN	0.5	0.4537	0.02	0.5782	0.04	0.4993	0.03
shallow	FSDN	1.0	0.4393	0.05	0.1817	0.04	0.2484	0.03
shallow	SDN	0.1	0.1057	0.08	0.8796	0.04	0.1866	0.04
shallow	SDN	0.5	0.1686	0.01	0.6982	0.05	0.2687	0.01
shallow	SDN	1.0	0.5177	0.01	0.3267	0.06	0.3957	0.02

We can see that the best results for precision are achieved when the citation model is built by employing the exact match mode. The exact match mode creates a narrower citation model where the correctness of the citation is the first priority. On the other hand, when completeness is the priority we need to optimize using the recall measure; in this case we see that the shallow match mode is a very good option. The shallow mode creates bigger citation models than the exact match one, thus fostering the completeness of the generated citations over their correctness. As we can see by adopting an exact match mode we obtain the highest precision values along with good recall performances as witnessed by the high fscore values; by contrast, with the shallow mode the values of recall are good, but precision and consequently also fscore are quite low showing that the generated citations are complete but also include noisy elements (i.e. useless or wrong paths). The mixed match mode is an interesting compromise, but it still tends to favor recall over precision, showing that the shallow mode somehow overshadows the exact match mode when both the modes are used.
The best results for all three measures are obtained when the ranking function adopts some forms of relative depth normalization (i.e. FSDN, FDN and SDN) given that the FS function never achieves the best performances. FDN and FSDN foster precision, whereas SDN fosters recall.

Lastly, low score thresholds work better with recall because they enable the inclusion of a higher number of paths in the citation fostering completeness at the price of a lower correctness; whereas, as expected, higher thresholds foster correctness over completeness and thus we obtain higher precision values than with lower score thresholds.

The results reported in Table I are useful for understanding how the different optimization parameters impact the performances of the citation system, but they are not indicative of how the system actually behaves in a real environment with real data.

To this end in Figure 9 we report the performances of the citation system evaluated with the three measures over the 50 test data samples. We conducted this test by varying the size of the training set from 100 training data samples (i.e. the whole set) to 5 data samples. The smaller training sets are obtained by randomly sampling the whole training set with repetitions; with this procedure, we can obtain as many different training sets as we need for the training sets with size smaller than 100. We performed ten random samplings for each training set size, thus we repeated each test ten times and then reported the average values.

In Figure 9, we see the box plots showing the distribution of average measures over the 50 test data samples; in this way, we can see how the system performances vary from test sample to test sample. The optimization measure used for the validation phase, in this case, is always fscore and we used the same 5-fold cross-validation strategy.

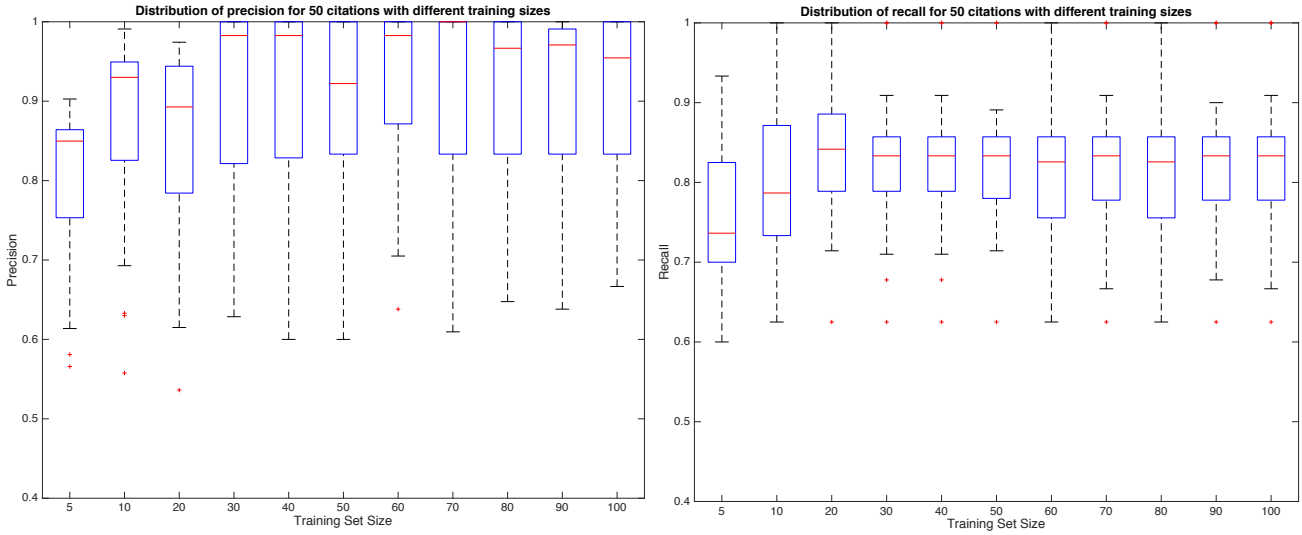


Figure 9. *Distribution of the measures for 50 test citations optimized with fscore as the training set varies.*

We can see that by optimizing with fscore, we get very good performances for precision which is on average above 90% for all the training sets bigger than 20 samples and fscore which is, on average, higher than 80% for training sets bigger than 10 samples. Both precision and recall have some low outliers with performances around 60%, whereas fscore is consistently above 70% when the training set size is bigger than 10 samples; this shows an inversely proportional relationship between precision and recall, such that when a citation has a low recall usually it is compensated by a high precision or vice versa.

It is interesting to note that the training set size has a small impact on the performances of the system when it is bigger than 10 samples. This is quite important for the citation system, given that one of its main goals is to produce citations requiring a low effort to data creators and curators; therefore, given that in this context building the training set is the only effort required of them, the smaller the required training set size the better.

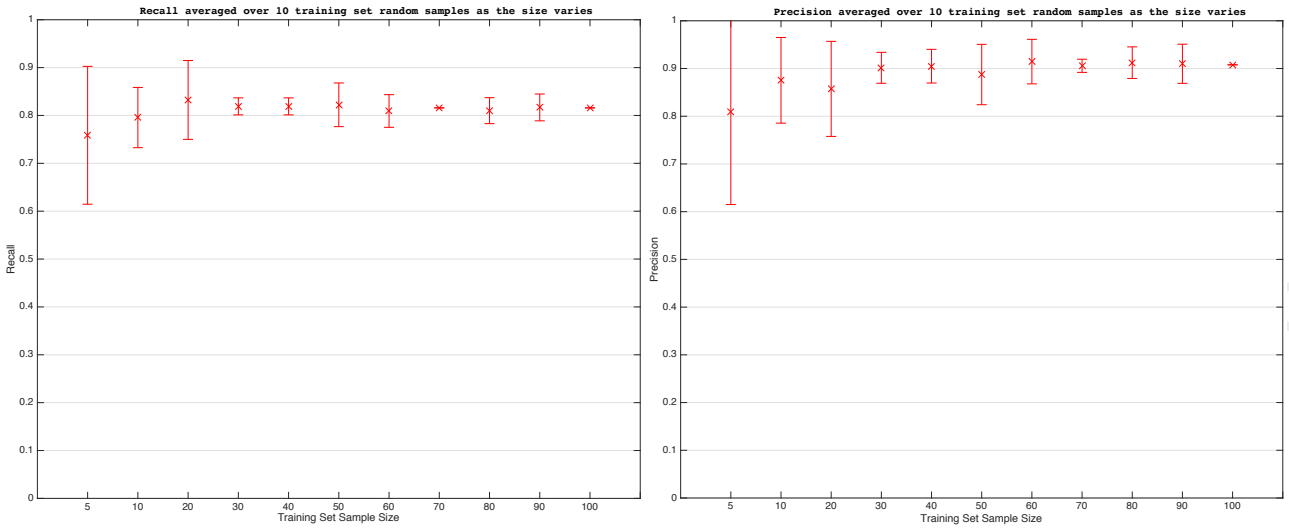


Figure 10. *Measures averaged over ten training sets randomly sampled as the size varies. The bars report the standard error. The optimization measure is fscore.*

In Figure 10 we report the performances averaged on the 50 test samples, for all the evaluation measures with the citation system optimized with fscore. Also in this case we repeated the test by randomly sampling the full training set and creating ten alternative training sets for each considered size; the bars report the standard error showing the variability of performances amongst training set samples.

As expected, the average performances are the same as those discussed for Figure 9, but the standard error shows us that the variability of performances decreases as the training set size increases; this is consistent across the different evaluation measures. Having stable performances with different samples of a training set with fixed size is important because we need a citation system whose performances do not depend on the specific citations used for creating the citation model and for optimizing its parameters. As we can see in Figure 10, the standard error for training sets with size bigger than 20 are quite small, whereas with size 5, 10 and 20 the error is greater, meaning that with very small training set sizes the specific composition of the training set has a direct impact on the performances of the citation system.

To further investigate whether the achieved performances are influenced by the specific composition of the training set, we conducted a 1-way ANOVA statistical test. We created ten training test sets for each tested size (from 5 to 90 with step 5) by randomly sampling with repetitions the full training set and we run the system on the test data for each training set by adopting 5-fold cross validation. In this case, we repeated this test by varying the optimization measure while keeping the evaluation measure fixed.For each system configuration and for each optimization measure we obtained ten different performance values for each test sample (500 values for each system configuration) and we performed the 1-way ANOVA statistical test on this data. The 1-way ANOVA tests if a null hypothesis is rejected or not; when the p-value is less than 0.01, then the null hypothesis is rejected with 99% probability, otherwise it cannot be rejected. In this test the null hypothesis is that system performances are the same as the training set composition varies, i.e. they do not depend on the composition of the training set; if the p-value returned by the ANOVA test is high (usually >0.01or >0.05) it means that the null hypothesis cannot be rejected and thus we can consider the performances of the system independent of the specific composition of the training set. Table II reports the 1-way ANOVA p-values for the different optimization measures as the training set sizes vary. We can see that by using fscore as the optimization measure we need a training set composed of at least 30 human-readable citations to obtain a citation model independent of the specific composition of the training set. The same value is achieved by using recall as the optimization measure,

whereas precision requires a training set with at least size 70.

Table II. *ANOVA 1-way p-values for different training set size as the optimization measure is varied; fscore is used as test measure. * denotes p-values of less than 5%. ** denotes p-values of less than 1%.*

Training Set Size	Optimization Measure	p-value	Training Set Size	Optimization Measure	p-value	Training Set Size	Optimization Measure	p-value
5	fscore	6.58e-87**	5	precision	4.24e-94**	5	recall	4.76e-93**
10	fscore	0.0051**	10	precision	2.43e-77**	10	recall	0.1000*
20	fscore	0.0158*	20	precision	8.61e-61**	20	recall	1.17e-33**
30	fscore	0.9881	30	precision	6.06e-53**	30	recall	0.1691
40	fscore	0.9895	40	precision	4.64e-23**	40	recall	0.7012
50	fscore	0.9228	50	precision	3.60e-23**	50	recall	1.0000
60	fscore	0.9689	60	precision	1.02e-37**	60	recall	1.0000
70	fscore	1.0000	70	precision	1.0000	70	recall	1.0000
80	fscore	0.9964	80	precision	1.0000	80	recall	0.9988
90	fscore	0.9570	90	precision	1.0000	90	recall	0.9972

In Figure 11 we report the plots of the Tukey HSD test [Tukey, 1949] where the same experiment configuration adopted for the ANOVA test is here used by maintaining the optimization measure fixed (fscore) and evaluating the performances by using precision, recall and fscore. Tukey HSD plots show the performances of the citation system grouped by training set size where the groups colored in blue do not statistically differ from another.

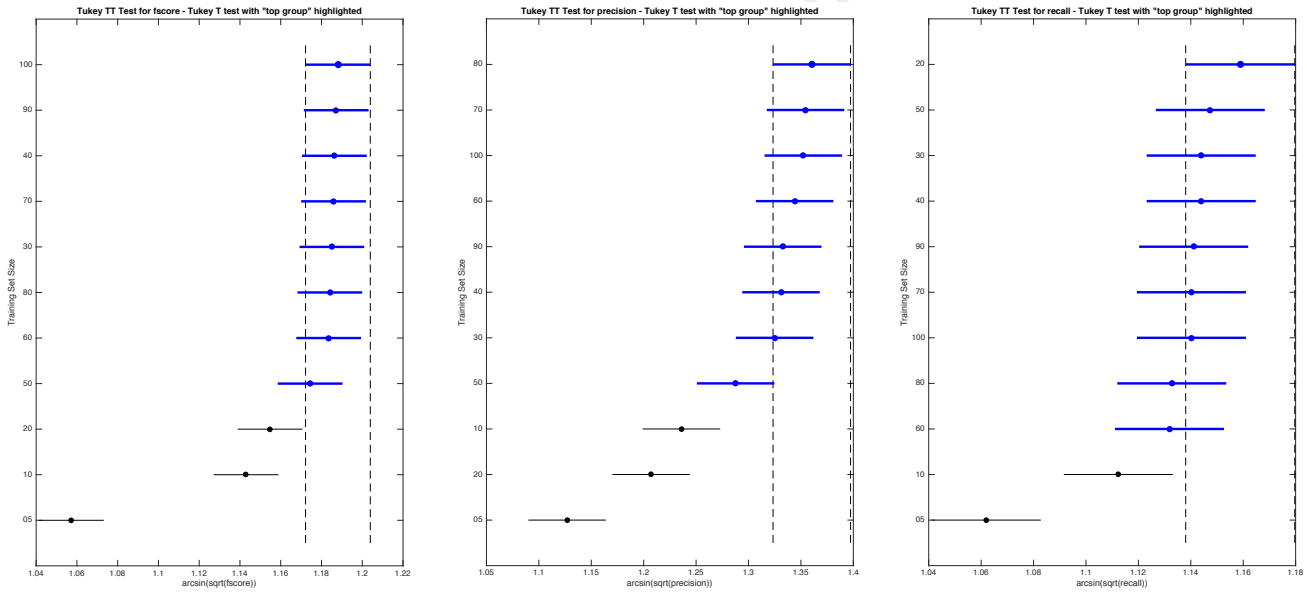


Figure 11. *Tukey HSD Test plots for different test measures (fscore, precision and recall) as the training set size varies by using fscore as optimization measure.*

We can see that by using fscore as the evaluation measure there is no significant difference between the performances obtained with training set sizes ranging from 100 to 30; the only significant differences are achieved with training set size 5, 10 and 20. This result further confirms what we have seen in Table II for the 1-way ANOVA test. The very same result is achieved by using precision as evaluation measure, and by using recall we have an even stronger result since the performances achieved with the full training set are significantly different only from those obtained with training set size 10. From this evaluation we can conclude that the learning to cite framework implemented for the XML data achieved good performances overall with fscore above 80%, precision above 90% and recall above 80% on average. We have seen that fscore is the most robust optimization measure and that with a training set containing as much as 30 human-readable citations we build a citation model solid enough to create citations for a collection of more than 2

thousand XML files.

Error types

As we discussed above, the learning to cite framework does not provide formally correct citation and by setting the optimization parameters of the framework it is possible to drive the citations produced towards a higher precision or a higher recall. In the former case, the produced citation may contain fewer selected elements possibly leaving some relevant elements out, whereas in the latter case the citation may contain many elements some of which could be not relevant to the citation.

The errors that could be present in the automatically produced citations can be classified into four main types.

The first type of error is the repetition of an element. This error is more likely to happen if the framework uses a shallow match mode to build the citation tree since it maximizes the number or matches than the exact approach. Moreover, by using recall as optimization measure the number of repetitions can increase since the higher recall values are obtained with shallow and mixed approaches as shown in Table I. Repetitions may make citations cumbersome even though they do not compromise their correctness and usefulness. Detecting a repetition is not a hard task and this error can be manually corrected; indeed, once the citation has been produced also a not-expert user can detect and delete repetitions from a citation in order to make it more readable. It is also possible to think about a (semi-)automatic method to detect and delete repetitions from the produced citations.

The second type of error is the absence of one or more elements from the citation. This error is more likely to happen when an exact match mode is adopted for building the citation tree and it is connected to the use of precision as optimization measure. This error has a bigger impact than repetitions since it is hard to automatically detect and correct the absence of an element; this error is hard to address also manually given that both expert and not-expert users may not realize that an element is missing from the produced citation.

The third type of error is the presence of a wrong element in the citation. This error requires a semantic analysis of the citation to be detected and it could be hard to solve even manually. The fourth type of error is the presence of a collateral element which is not part of the formally correct citation even though it is not wrong per se.

As an example, let us consider the formally correct citation reported in the "Digital Archives: A Use Case" Section and shown in Figure 4, which is composed of 15 distinct elements. A possible citation produced by the learning to cite framework optimized with recall is the following one where wrong elements are in bold:

Correspondence, 1951-1956, "The Elements of Legal Theory" (unpublished). Book, box 135; By Cairns, box 129. Part II: Writings, 1905-1984, box **1780-1984.** Huntington Cairns Papers 1780-1984. **MSS14746.** http://hdl.loc.gov/loc.mss/eadmss.ms001024; **Huntington Cairns Papers, A Finding Aid to the Collection in the Library of Congress**

In this case there is a repetition, a wrong element and the presence of collateral elements. As we can see the element "Huntington Cairns Papers" is repeated two times and its second occurrence can be deleted. The element "1780-1984" is wrong because after the element box there should be the element "129-152"; this means that a date has been inserted in the citation in place of the box number. We can see that this error can be detected with a semantic analysis of the citation since the correspondence is correctly places in "box 135" and "By Cairns" data are placed in box 129, so the expert user would see that it is not possible that the "Part II: Writings" documents are in "box 1780-1984". The element "MSS14746", which is an internal identifier, is a collateral element since its presence is not an error per se, but it simply provides additional (not strictly required) information. Two elements are missing: "Manuscript division" and "Library of Congress"; the first one cannot be derived from any other information in the citation, whereas the second one is

substituted by the valid collateral element "A Finding Aid to the Collection in the Library of Congress". The evaluation we conducted does not distinguish between error types, thus also the last collateral element which is not, strictly specking, an error, is considered such and contributes negatively to precision and fscore.

Let us see how the citation to the same element considered above could look like by optimising the framework with precision:

```
Correspondence, 1951-1956, Book, By Cairns, 1905-1984.  Part II: Writings, 1905-
1984. Huntington Cairns Papers, http://hdl.loc.gov/loc.mss/eadmss.ms001024
```

In this case, we can see that all the information present in the citation are formally correct, there are no repetitions, wrong and collateral elements. On the other hand, there are several missing elements which are box numbers, the title of the book "The Elements of Legal Theory (unpublished)" and the elements identifying the preserver of the archive which is the Library of Congress.

The first citation is more complete and can be manually fixed quite easily by expert and not-expert users, whereas the second one is more difficult to complete by hand even though it provides most of the fundamental information required and does not convey any wrong message.

Conclusions and Future Works

The practice of data citation is unanimously considered fundamental for scientific progress, but as of now it is not commonly adopted and encouraged in all scientific fields. The research community has been taking action to ease the process of citing data and to make it a core aspect of scholarship and scientific publishing. As a consequence, in recent years data citation received a great deal of attention, which has led to the definition of the basic principles for data citation, to the creation of data journals and data citation indexes, to the idea of "actionable papers" linking scientific claims to the data sustaining them and to the development of infrastructures to manage and access scientific publications along with the related data.

Nevertheless, in order to make data citation an everyday practice and to overcome the cultural and technical barriers still impairing its wide adoption, there is the pressing need to develop effective and easy to use citation tools. This requires studying data citation from a computational perspective as recently suggested by [Buneman et al., 2016].

It is important to consider that data citation is a compound and complex problem and it is accepted that a "one size fits all" system to address this problem does not exist. Indeed, flat data, relational databases, XML and RDF datasets are intrinsically different one from the other, present heterogeneous structures and functions and, as a consequence, require specific solutions for addressing data citation problems.

In this work, we stem from this consideration by focusing on how to generate citations to single nodes within a hierarchical dataset serialized as XML. To this end, we defined the so-called learning to cite framework which enables the automatic construction of human- and machine-readable citations to hierarchical data with variable granularity, with the final goal of reducing the human intervention on data to a minimum and to provide a citation system general enough to work on different hierarchical data collections.

The learning to cite framework represents a change of paradigm for the automatic creation of data citations because it shifts the focus from an approach where the citation process is decided and modelled *a priori* by human experts to one where this process is not taught but learned by example and can dynamically adapt to new data and contexts.

We described a concrete implementation of the learning to cite framework by developing an open-source citation system for XML datasets, which learns how to cite directly from the data and can be used in different settings and with different datasets.

We conducted a thorough evaluation of the developed citation system by employing a use case drawn from the digital archives domain, where there is the need to cite large, deep and heterogeneous XML files – i.e. finding aids encoded in the EAD format. In order to make our evaluation reproducible, we created an *ad-hoc* shared test collection based on the EAD files of the Library of Congress collection; the experimental collection includes a training set composed of one hundred human-readable citations to XML citable units and a test set composed of fifty citable units. Both for the training set and the test set we created a ground-truth composed of 150 manually crafted machine-readable citations to be used for validation purposes and to assess the quality of automatically generated citations. This experimental collection is made openly available to enable further experiments on the automatic creation of citations by the research community at large and represents the first concrete effort to provide a common and shared point of comparison for data citation methods and systems. Furthermore, we defined from scratch three evaluation measures to assess the correctness and completeness of automatically generated data citations: precision, recall and fscore.

The experiments we carried out investigated three main aspects: (i) the effect of parameters tuning and the choice of the optimization measure on the citation model and consequently on system effectiveness; (ii) the correctness and completeness of the citations generated by the system; and (iii) the impact of the training set size and composition on system performances.

We concluded that the parameters tuning has a considerable impact on system performances given that the choice of one matching mode and one ranking function over others fosters correctness over completeness or vice versa, and that fscore is a good choice for an optimization measure because it enables the system to achieve better performances by using smaller training sets than by using precision and recall as optimization measures. The citation system proved to be effective and on average achieves precision values above 90% and recall and fscore values consistently above 80% by using a minimum of 30 human-readable citations as the training set. We have seen that there are no significant performance differences by using training sets greater than 30 citations and that the specific composition of the training set does not have a significant impact on system effectiveness. This means that the only effort required of data creators and curators to employ our citation system in a real environment is to produce a few dozen human-readable citations as training and validation set from a randomly selected subset of citable units. It should be noticed that, even though the precision and recall are quite high, the citations produced by the proposed system are not perfect and formally correct, so we may take into account some manual work to refine the citations by removing redundant or evidently wrong elements or by adding missing elements.

There are some extensions to the framework that can be planned as future works. One extension concerns the production of citations with different structures and styles. Indeed, the training set determines the structure of the produced citations; so, if we need to produce citations with two different structures (or styles) we need to train two different citation models by using two different training sets. In some cases, a viable alternative would be to set up a set of rules to manipulate the produced citations in order to add or remove a predefined set of elements. Another possibility is to define a set of rules to format the produced citations. Indeed, every element returned by the learning to cite framework is related to a specific XML element, so we can define a formatting rule related to this element. For instance, in the context of EAD, in one set of rules we may decide that the “unittitle” element has to be printed in italic; in an alternative set of rules we may impose that the “unittitle” element has not to be inserted in the citation because it may be considered as superfluous for some reasons. In any case, these rules are related to the citation policies that are defined for the collection of files at hand and, if required, they have to be set by the collection owner/responsible.

As further future works we plan to extend the developed citation system in order to pass from the current single datum citation –i.e., single XML node citation – approach to a multiple data citation one, where the same citation can refer to

multiple elements or element aggregations or where we need to create citations for hundreds of independent citable units. We also aim to tackle the problem of citation identity by defining an efficient method to establish when two citations cite the same data and can be considered to be the same.

From the evaluation viewpoint, we should note that the collection we employed contains heterogeneous files created by different people in different dates and with different purposes, so the presented system has been already tested by considering heterogeneous hierarchies. It is also true that all the files come from the Library of Congress collection and so they may be more consistent one with the other than a bunch of files taken from, say, ten radically different organizations. In this last case, the training set has to comprehend a sufficient number of examples from each different considered collection. The citation model has been defined to be flexible enough to accommodate for these variations, but the required size of the training and validation sets in this case may need to be larger than the one determined for the Library of Congress collection. As future works, we want to test the system on the UK Archival Hub[11] which puts together files coming from many different archives intrinsically different one from the other. On the one hand, we want to apply the model learnt on the Library of Congress data to the UK Archival Hub. This last aspect is commonly called *transfer learning* from a dataset to another; the idea is to understand whether is it possible and to what extent, to define a method to generate citations for a collection of files by using the citation model created by employing a training set obtained from a different collection. Such a method would have a sizeable impact because it will further lower the barriers and the effort required to create data citations; indeed, we could train a citation system by using a well-suited available training set and then use such system in a context where there is no training set or the resources to create one.

Lastly, we will study how to implement the learning to cite framework for data models and formats other than XML, such as relational databases or RDF datasets. With these data models the logic behind the presented citation system has to be re-thought because it does not work as is for data that is flat, i.e. where there is no hierarchical structure to reveal the organization or the hierarchy is modelled as part of the data.

Acknowledgements

I would like to express gratitude to the people at the University of Padua, Italy and in particular to Maristella Agosti and Nicola Ferro for their support on my research activities throughout the years. Very special thanks go to Peter Buneman – who introduced me to data citation – and Susan Davidson for their insights and the continuous useful discussions about this topic. Last, but not least, I would like to thank the anonymous reviewers who kindly contributed to improve this paper.

References

[Altman and Crosas, 2013] Altman, C. and Crosas, M. (2013). The Evolution of Data Citation: From Principles to Implementation. *IASSIST Quarterly* 37:Spring, pages 62-70.

[AMS, 2015] The Academy of Medical Science (2015). Reproducibility and Reliability of Biomedical Research: Improving Research Practice. Technical report, October 2015. http://apo.org.au/node/58335

[Arguello et al., 2015] Arguello, J., Crane, M., Diaz, F., Lin, J., and Trotman, A. (2015). Report on the SIGIR 2015 Workshop on Reproducibility, Inexplicability, and Generalizability of Results (RIGOR). SIGIR Forum, 49(2).

[Auer et al., 2012] Auer, S., Dalamagas, T., Parkinson, H. E., Bancilhon, F., Flouris, G., Sacharidis, D., Buneman, P., Kotzinos, D., Stavrakas, Y., Christophides, V., Papastefanatos, G., and Thiveos, K. (2012). Diachronic Linked Data: Towards Long-Term Preservation of Structured Interrelated Information.

[11] http://archiveshub.ac.uk/

pages 31–39.

[Baggerly, 2010] Baggerly, K. (2010). Disclose All Data in Publications. *Nature*, (467):401.

[Bardi and Manghi, 2015] Bardi, A. and Manghi, P. (2015). A Framework Supporting the Shift from Traditional Digital Publications to Enhanced Publications. *D-Lib Magazine*, 21(1/2).

[Borgman, 2012a] Borgman, C. L. (2012). The Conundrum of Sharing Research Data. *Journal of the Association for Information Science and Technology (JASIST)*, 63(6):1059–1078.

[Borgman, 2012b] Borgman, C. L. (2012b). Why are the Attribution and Citation of Scientific Data Important? *National Academy of Sciences' Board on Research Data and Information*, Report from Developing Data Attribution and Citation Practices and Standards: An International Symposium and Workshop. National Academies Press: Washington DC.

[Borgman, 2015] Borgman, C. L. (2015). Big Data, Little Data, No Data. MIT Press.

[Brase et al., 2014] Brase, J., Socha, Y., Callaghan, S., Borgman, C. L., Uhlir, P. F., and Carroll, B. (2014). *Research Data Management: Practical Strategies for Information Professionals*, chapter Data Citation: Principles and Practice, pages 167–186. Purdue University Press.

[Buneman and Silvello, 2010] Buneman, P. and Silvello, G. (2010). A Rule- Based Citation System for Structured and Evolving Datasets. *IEEE Data Eng. Bull.*, 33(3):33–41.

[Buneman et al., 2002] Buneman, P., Davidson, S. B., Fan, W., Hara, C. S., and Tan, W. C. (2002). Keys for XML. *Computer Networks*, 39(5):473–487.

[Buneman et al., 2014] Buneman, P., Cohen-Boulakia, S., Davidson, S. B., Frew, J., and Tannen, V. (2014). Computational challenges in data citation. Technical report, University of Pennsylvania.

[Buneman et al., 2016] Buneman, P., Davidson, S., Frew, J. (2016). Why Data Citation is a Computational Problem. *Communications of the ACM*, Forthcoming

[Buneman, 2006] Buneman, P. (2006). How to Cite Curated Databases and How to Make Them Citable. In *Proc. of the 18th International Conference on Scientific and Statistical Database Management*, SSDBM 2006, pages 195–203.

[Burton et al., 2015] Burton, A., Koers, H., Manghi, P., Bruzzo, S. L., Aryani, A., Diepenbroek, M., and Schindler, U. (2015). On Bridging Data Centers and Publishers: The Data-Literature Interlinking Service. volume 544 of *Communications in Computer and Information Science*, pages 324–335. Springer.

[Candela et al., 2015] Candela, L., Castelli, D., Manghi, P., and Tani, A. (2015). Data Journals: A Survey. *Journal of the Association for Information Science and Technology (JASIST)*, 66(9):1747–1762.

[Carr and Littler, 2015] Carr, D., and Littler, K. (2015). Sharing Research Data to Improve Public Health: A Funder Perspective. *Journal of Empirical Research on Human Research Ethics*, 10(3), 314–316.

[CODATA-ICSTI, 2013] CODATA-ICSTI Task Group on Data Citation Standards and Practices (2013). Out of Cite, Out of Mind: The Current State of Practice, Policy, and Technology for the Citation of Data. *Data Science Journal*, 12:1–67.

[Costas et al., 2013] Costas, R., Meijer, I., Zahedi, Z., and Wouters, P. (2013). The Value of Research Data - Metrics for Datasets from a Cultural and Technical Point of View. A Knowledge Exchange Report. Technical Report, Danish Agency for Culture.

[Cronin, 1984] Cronin, B. (1984). The Citation Process. The Role and Significance of Citations in Scientific Communication. London: Taylor Graham.

[Domingos, 2015] Domingos, P. (2015). The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World. Basic Books.

[Duke and Ball, 2012] Duke, M. and Ball, A. (2012). How to Cite a Dataset and Link to Publications: A Report of the Digital Curation Centre. In *Proc. of the 23rd International CODATA Conference*.

[Duranti, 1998] Duranti, L. (1998). Diplomatics: New Uses for an Old Science. Society of American Archivists and Association of Canadian Archivists in Association with Scarecrow Press, Lanham, Maryland, USA.

[FORCE11, 2014] Data Citation Synthesis Group: Joint Declaration of Data Citation Principles. Martone M. (ed.) San Diego CA: FORCE11; 2014.

[Francisco-Revilla et al., 2014] Francisco-Revilla, L., Trace, C. B., Li, H., and Buchanan, S. A. (2014). Encoded Archival Description: Data Quality and Analysis. *Proceedings of the American Society for Information Science and Technology*, 51(1):1–10.

[Freire et al., 2012] Freire, J., Bonnet, P., and Shasha, D. (2012). Computational reproducibility: state-of-the-art, challenges, and database research opportunities. In Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012, pages 593–596.

[Goodman et al., 2014] Goodman A., Pepe A., Blocker A. W., Borgman C. L., Cranmer K., Crosas M., et al. (2014) Ten Simple Rules for the Care and Feeding of Scientific Data. *PLoS Comput Biol* 10(4): e1003542. doi:10.1371/journal.pcbi.1003542.

[Gottlob et al., 2003] Gottlob, G., Koch, C., and Pichler, R. (2003). XPath Processing in a Nutshell. *SIGMOD Rec.*, 32(1):12–19.

[Howison and Bullard, 2015] Howison, J. and Bullard, J. (2015). Software in the Scientific Literature: Problems with Seeing, Finding, and Using Software Mentioned in the Biology Literature. *Journal of the American Society for Information Science and Technology (JASIST)*. Forthcoming.

[Huang et al., 2015] Huang, Y.-H., Rose, P. W., and Hsu, C.-N. (2015). Citing a Data Repository: A Case Study of the Protein Data Bank. *PLoS ONE*, 10(8).

[ISAD, 1999] International Council on Archives (1999). ISAD(G): General International Standard Archival Description, 2nd edition. Ottawa: International Council on Archives.

[Kiesling, 2001] Kiesling, K. (2001). Metadata, Metadata, Everywhere - But Where Is the Hook? OCLC Systems & Services, 17(2):84–88.

[Klump et al., 2015] Klump, J., Huber, R., and Diepenbroek, M. (2015). DOI for Geoscience Data – How Early Practices Shape Present Perceptions. Earth Science Informatics, pages 1–14.

[Kurtz, 2012] Kurtz, M. J. (2012). Linking, Finding, and Citing Data in Astronomy. In National Research Council. In *For Attribution - Developing Data Attribution and Citation Practices and Standards: Summary of an International Workshop*. Washington, DC: The National Academies Press, 2012. doi:10.17226/13564.

[Lawrence et al., 2011] Lawrence, B., Jones, C., Matthews, B., Pepler, S., and Callaghan, S. (2011). Citation and Peer Review of Data: Moving Towards Formal Data Publication. *International Journal of Digital Curation*, 6(2):4–37.

[Maxwell and Delaney, 2004] Maxwell S. and Delaney H. D. (2004). *Designing Experiments and Analyzing Data. A Model Comparison Perspective*. Lawrence Erlbaum Associates, 2nd ed, 2004.

[Mooney and Newton, 2012] Mooney, H. and Newton, M. P. (2012). The Anatomy of a Data Citation: Discovery, Reuse, and Credit. *Journal of Librarianship and Scholarly Communication*, 1(1).

[Pitti and Duff, 2001] Pitti, D. V. and Duff, W. M. (2001). Introduction. In Pitti, D. and Duff, W. M., editors, Encoded Archival Description on the Internet, pages 1–6. The Haworth Press, Inc.

[Pitti, 1999] Pitti, D. V. (1999). Encoded Archival Description. An Introduction and Overview. D-Lib Magazine, 5(11).

[Pröll and Rauber, 2013] Pröll, S. and Rauber, A. (2013). Scalable Data Citation in Dynamic, Large Databases: Model and Reference Implementation. In Hu, X., Young, T. L., Raghavan, V., Wah, B. W., Baeza-Yates, R., Fox, G., Shahabi, C., Smith, M., Yang, Q., Ghani, R., Fan, W., Lempel, R., and Nambiar, R., editors, Proc. of the 2013 *IEEE International Conference on Big Data*, pages 307–312. IEEE Computer Society.

[Pröll and Rauber, 2015] Pröll, S. and Rauber, A. (2015). Asking the Right Questions - Query-Based Data Citation to Precisely Identify Subsets of Data. *ERCIM News*, (100).

[Pröll and Rauber, 2015b] Pröll, S. and Rauber, A. (2015). Scalable Dynamic Data Citation, Approaches, Reference Architectures and Applications. *RDA-WG-DC Position Paper*, https://rd-alliance.org/groups/data-citation-wg/wiki/scalable-dynamic-data-citation-rda-wg-dc-position-paper.html.

[Rauber et al., 2015] Rauber, A., Asmi, A., van Uytvanck, D. and Pröll, S. (2015). Data Citation of Evolving Data, Recommendations of the Working Group on Data Citation (WGDC)

[Robinson-Garcia et al., 2015] Robinson-Garcia, N., Jiménez-Contreras, E., and Torres-Salinas, D. (2015). Analyzing data citation practices according to the Data Citation Index. *Journal of the American Society for Information Science and Technology (JASIST)*. Forthcoming.

[Ruth, 2001] Ruth, J. E. (2001). The Development and Structure of the Encoded Archival Description Document Type Definition. In Pitti, D. and Duff, W. M., editors, Encoded Archival Description on the Internet, pages 27–59. The Haworth Press, Inc.

[Silvello, 2015] Silvello, G. (2015). A Methodology for Citing Linked Open Data Subsets. *D-Lib Magazine*,

21(1/2).

[Simons, 2012] Simons, N. (2012). Implementing DOIs for Research Data. *D- Lib Magazine*, 18(5/6).

[Starr et al., 2015] Starr J, Castro E, Crosas M, Dumontier M, Downs RR, Duerr R, Haak L, Haendel M, Herman I, Hodson S, Hourclé J, Kratz JE, Lin J, Nielsen LH, Nurnberger A, Pröll S, Rauber A, Sacchi S, Smith AP, Taylor M, Clark T. (2015). Achieving human and machine accessibility of cited data in scholarly publications. *PeerJ PrePrints* http://doi.org/10.7287/peerj.preprints.697v4

[Thanos and Rauber, 2014] Thanos, C. and Rauber, A. (2014). Scientific Data Sharing and Re-use: Introduction to the Special Theme. *ERCIM News*, 2014, pp.13.

[Tsikrika et al., 2011] Tsikrika, T., Garcia Seco de Herrera, A., and Muller, H. (2011). Assessing the Scholarly Impact of ImageCLEF. In Forner, P., Gonzalo, J., Kekalainen, J., Lalmas, M., and de Rijke, M., editors, Multilingual and Multimodal Information Access Evaluation. *Proceedings of the Second International Conference of the Cross-Language Evaluation Forum (CLEF 2011)*, pages 95–106. Lecture Notes in Computer Science (LNCS) 6941, Springer, Heidelberg, Germany.

[Tukey, 1949] Tukey, J. (1949). Comparing Individual Means in the Analysis of Variance. *Biometrics* 5(2): 99-114.

[UCL, 2010] University College London (2010). Advancing Research and Practice in Digital Curation and Publishing: Summary Report and Recommendations of the Workshop on Next Steps in Research, Education and Practice. Technical Report, University College London.

[Vernooy-Gerritsen, 2009] Vernooy-Gerritsen, M. (2009). Enhanced Publications: Linking Publications and Research Data in Digital Repositories. Amsterdam University Press.

[W3C, 2004] W3C (2004). Document Object Model (DOM) Level 3 Core Specification, Version 1.0 – W3C Recommendation 07 April 2004. http: //www.w3.org/TR/DOM-Level-3-Core/.

[W3C, 2006] W3C (2006). Extensible Markup Language (XML) 1.1 (Second Edition) – W3C Recommendation 16 August 2006, edited in place 29 September 2006. http://www.w3.org/TR/xml11/.

[W3C, 2007] W3C (2007). XML Path Language (XPath) 2.0 – W3C Recommendation 23 January 2007. http://www.w3.org/TR/xpath20/.

[Wisser and Dean, 2013] Wisser, K. and Dean, J. (2013). EAD Tag Usage: Community Analysis of the Use of Encoded Archival Description Elements. *American Archivist*, 76(2):542–566.