An Overview on XML Semantic Disambiguation from Unstructured Text to Semi-Structured Data: Background, Applications, and Ongoing Challenges

Joe Tekli

Abstract— Since the last two decades, XML has gained momentum as the standard for Web information management and complex data representation. Also, collaboratively built semi-structured information resources, such as Wikipedia, have become prevalent on the Web and can be inherently encoded in XML. Yet most methods for processing XML and semi-structured information handle mainly the syntactic properties of the data, while ignoring the semantics involved. To devise more intelligent applications, one needs to augment syntactic features with machine-readable semantic meaning. This can be achieved through the computational identification of the meaning of data in context, also known as (a.k.a.) automated semantic analysis and disambiguation, which is nowadays one of the main challenges at the core of the Semantic Web. This survey paper provides a concise and comprehensive review of the methods related to XML-based semi-structured semantic analysis and disambiguation. It is made of four logical parts. First, we briefly cover traditional word sense disambiguation methods for processing flat textual data. Second, we describe and categorize disambiguation techniques developed and extended to handle semi-structured and XML data. Third, we describe current and potential application scenarios that can benefit from XML semantic analysis, including: data clustering and semantic-aware indexing, data integration and selective dissemination, semantic-aware and temporal querying, Web and Mobile Services matching and composition, blog and social semantic network analysis, and ontology learning. Fourth, we describe and discuss ongoing challenges and future directions, including: the quantification of semantic ambiguity, expanding XML disambiguation context, combining structure and content, using collaborative/social information sources, integrating explicit and implicit semantic analysis, emphasizing user involvement, and reducing computational complexity.

Index Terms—H.3.1 [Content Analysis and Indexing]; H.3.3 [Information Search and Retrieval]; I.7.1 [Document and Text Processing]: Document and Text Editing – *Document management*; I.7.2 [Document Preparation]: Document Preparation – *Markup languages*. I.2.4 [Knowledge Representation Formalisms and Methods]: Semantic Networks.

1. Introduction

With the increasing amount of information published on the Web, there is an ever-swelling demand for methods to effectively store, describe, access and retrieve Web data. After all, the value of information depends on how easy it is to search and manage [81]. In this context, a major breakthrough has been achieved in the past decade with the development and widespread adoption of XML (eXtensible Markup Language) as a standard semi-structured data representation model on the Web [20]. The ability to distil free-form information and reshape structured (e.g., relational, hierarchical, and/or graph-based) data into a unified semi-structured format has proven central in facilitating large-scale automatic data processing (with the proliferation of XML-based Web formats such as SOAP[1], RSS[2], SVG[3], MPEG-7[4], GML[5], etc.). Also, collaboratively built semi-structured information resources are becoming increasingly available [78] (such as Wikipedia[6], Wikitionary[7], Flickr[8], Twitter[9], and Yahoo Answers[10]) describing different kinds of textual and multimedia information which can be naturally represented and processed using standardized XML (and related) technology. Nonetheless, attaining a higher degree of human-machine cooperation requires yet another technological breakthrough: *extending the Web by giving information well defined semantic meaning*, i.e., the motto of the Semantic Web [13]. Thus, following the unprecedented Web exploitation and abundance of XML and semi-structured data, the identification of semantic meaning for XML-based information becomes a key challenge at the core of Semantic Web applications.

For the past decade, most existing research around XML data processing has focused on handling the syntactic and structural properties of XML documents [187, 191], while neglecting the semantics involved [183]. Yet, various studies have highlighted the impact of integrating semantic features in XML-based applications, ranging over semantic-aware query rewriting and expansion [36, 130] (expanding keyword queries by including semantically related terms from XML documents to obtain relevant results), document classification and clustering [182, 190] (grouping together XML documents based on their semantic similarities), schema matching and integration [46, 192] (considering the semantic meanings and relationships between XML schema elements and data-types), and more recently Web and mobile services' discovery, recommendation, and composition [94, 113, 220] (searching and mapping semantically similar WSDL/SOAP descriptions when processing Web Services, and XHTML/free-text descriptions when dealing with RESTful/mobile services), XML-based knowledge engineering (semantic annotation of XML data using Linked Data constructs) [70, 112], and semantic blog analysis and event detection in social networks [2, 12, 154], among other applications (cf. Section 6).

1.1 Problem Statement and Motivation

Here, a major challenge remains unresolved: *XML semantic disambiguation*, i.e., how to solve the semantic ambiguities and identify the meanings of terms in XML documents [89], which is central to improving the performance of XML-based applications. The problem is made even harder with the huge volume and diversity of XML and semi-structured data on the Web.

Joe Tekli is an Assistant Professor in the Electrical and Computer Engineering Department (ECE), Lebanese American University (LAU), 36 Byblos, Lebanon. Email: joe.tekli@lau.edu.lb

[1] http://www.w3.org/TR/soap/
[2] http://www.rssboard.org/rss-specification
[3] http://www.w3.org/Graphics/SVG/
[4] http://mpeg.chiariglione.org/standards/mpeg-7
[5] http://www.opengeospatial.org/standards/gml
[6] http://www.wikipedia.org [7] http://www.wikitionary.org
[8] http://www.flickr.com [9] http://twitter.com [10] http://answers.yahoo.com

On one hand, heterogeneous XML data sources often exhibit different and subjective ways to annotate similar (or identical) data. XML documents may have common structures across different topics, or may encompass common topics across different structures. A simple example is shown in Fig. 1, where two different XML documents describe the same *Hitchcock movie* even though they have different structures and tag names. On the other hand, a common limitation with most existing XML data processing methods is that they often represent information according to its syntactic and/or stylistic properties, neglecting its actual semantic meaning [183]. The center problem here is lexical ambiguity: a term (e.g., an XML element/attribute tag name or data value) may have multiple meanings (*polysemy*), a word can be implied by other related terms (*metonymy*), and/or several terms can have the same meaning (*synonymy*) [89]. For instance (according to a general purpose knowledge base such as WordNet [51]), the term *"Kelly"* in XML document 1 of Fig. 1 may refer to *Emmet Kelly: the circus clown*, *Grace Kelly: Princess of Monaco*, or *Gene Kelly: the dancer*. In addition, XML allows the definition of syntactic markup: customized (element/attribute) tags describing corresponding data values [20], which are equally prone to lexical ambiguity. For instance, the XML element tag name *"Director"* in document 1 (Fig. 1) can have several meanings, e.g., *"Manager of a company"*, *"Film director"*, *"Theater director"* or *"Music director"* (likewise for most terms/tag names in XML documents 1 and 2, e.g., *"Action"*, *"Plot"*, *"Cast"*, *"Star"*, etc., which can have more than 2 or 3 different semantic senses each, following WordNet). However, looking at their context in the document, a human user can tell that term *"Kelly"* here refers to *Grace Kelly*, and that element label *"Director"* refers to *"Film director"*.

The most obvious and accurate solution to the problem would be to manually annotate terms, mapping them with their intended meanings in a reference knowledge base. However, this remains practically infeasible due to the sheer amount of XML and semi-structured data on the Web. Thus, *word sense disambiguation* (WSD), i.e., the computational identification of the meaning of words in context [126], becomes central to automatically resolve the semantic ambiguities and identify the intended meanings of XML tag names and data values. Yet while WSD has been extensively studied for flat textual data [80, 126], nonetheless, the disambiguation of XML and semi-structured data remains in its early stages. Existing approaches have been straightforwardly extended from traditional flat text WSD, and thus show several limitations and challenges when handling (semi-)structured information (cf. Section 7).

1.2 Contribution and Organization of the Paper

In this paper, we provide a concise and comprehensive review of the methods related to XML semantic analysis and disambiguation. The objective of this study is to briefly describe, compare, and categorize the different techniques and methods related to the problem, while illustrating some of the main challenges and potential application scenarios that can benefit from XML semantic analysis. To our knowledge, this is the first review study dedicated to the XML semantic disambiguation domain, which we hope will foster and guide further research on the subject. Note that while mainly focused on XML (as the present W3C standard for semi-structured data representation on the Web), yet most concepts and methods covered in this paper can be easily adapted/extended to handle alternative semi-structured data models (e.g., JSON[1]). The remainder of the paper is organized as follows. Section 2 presents a glimpse on XML data and knowledge representations. Section 3 briefly reviews the background in traditional WSD. Section 4 reviews and categorizes XML semantic disambiguation techniques, followed by a description of experimental evaluation metrics and test data in Section 5. XML semantic-aware applications and potential uses are described in Section 6. Ongoing challenges and future directions are covered in Section 7, before concluding in Section 8.

[1] www.json.org

```
<?xml version= "1.0"?>
<Films>
  <Picture title= "Rear Window">
    <Director> Hitchcock </Director>
    <Year> 1954 </Year>
    <Genre> Thriller </Genre>
    <Cast>
      <Star> Stewart </Star>
      <Star> Kelly </Star>
    </Cast>
    <Plot>A wheelchair bound
        photographer spies on
        his neighbors …</Plot>
    …
  </Picture>
</Films>
```

a. XML document 1

```
<?xml version= "1.0"?>
<Movies>
  <Movie year= "1954">
    <Name> Rear Window </Name>
    <Directed_By>Alfred Hitchcock</Directed_By>
    <Actors>
      <Lead_Actor>
        <FirstName>James</FirstName>
        <LastName>Stewart</LastName>
      </Lead_Actor>
      <Actor>
        <FirstName>Grace</FirstName>
        <LastName>Kelly</LastName>
      </Actor>
    </Actors>
    …
  </Movie>
</Movies>
```

b. XML document 2

Fig. 1. Sample documents with different structures and tagging, yet describing the same information.

2. XML Data and Semantic Knowledge

2.1. XML Data Representation

XML documents represent hierarchically structured information and are generally modeled as Rooted Ordered Labeled Trees (ROLT, Fig. 2), based on the Document Object Model (DOM) [211]. A ROLT is a tree[2] with a single root node, in which the nodes are labeled and ordered. Given a ROLT T, we refer to $T[i]$ as the i^{th} node of tree T in pre-order (post-order or breadth-first) traversal, with $T[i].\ell$ its label. An XML document tree is typically represented as a ROLT where nodes represent XML elements/attributes, labeled using element/attribute tag names, and ordered following their order of appearance in the XML document[3] (which corresponds to pre-order traversal following the ROLT structure). Attribute nodes usually appear as children of their containing element nodes, sorted[4] by attribute name, and appearing before all sub-elements [134, 228]. Other types of nodes, such as entities, comments and notations, are commonly disregarded in most XML data processing approaches since they are not part of the core XML data [40, 182].

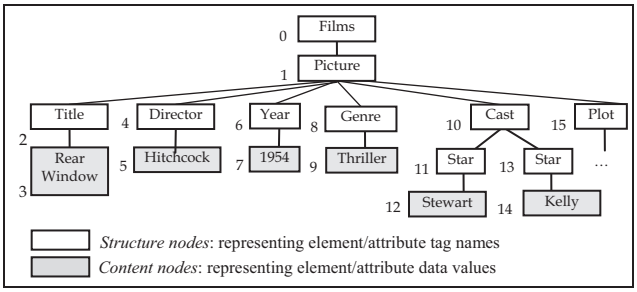


Fig. 2. XML document tree T representing XML document 1 in Fig. 1.

Also, one of the main characteristics that distinguish XML documents from plain semi-structured data is the notion of XML grammar. An XML grammar (i.e., DTD [20] or XSD [141]) is a set of

[2] *Tree* and *rooted ordered labeled tree* are used interchangeably hereafter.

[3] Element node ordering is disregarded in certain applications (such as in keyword-based information retrieval, cf. Section 6.3).

[4] While the order of attributes (unlike elements) is irrelevant in XML, yet most studies adopt an ordered tree model to simplify processing [134, 228].

definitions and declarations for modeling XML documents, defining the elements and attributes of the documents they describe, as well as element/attribute structural positions, data-types, and the rules they adhere to in the documents [147]. XML grammars can be viewed as schemas in traditional DBMS, necessary for the efficient indexing, storage, and retrieval of document instances.

As for XML element/attribute values, they can be disregarded (*structure-only*) or considered (*structure-and-content*) in XML data processing following the application scenario at hand (cf. Fig. 2).

In general, element/attribute values are disregarded when evaluating the structural properties of *heterogeneous* XML documents (originating from different data-sources and not conforming to the same grammar) [183], so as to perform XML structural classification/clustering [40, 182] or structural querying [16, 158] (i.e., querying the structure of documents, disregarding content). Yet, values are usually considered with methods dedicated to XML versioning [31, 37], data integration [62, 102], and XML *structure-and-content* querying applications [162, 163], where documents tend to have similar structures (probably conforming to the same grammar [98], cf. Section 6.2). With such methods, XML text sequences can be decomposed into word tokens, mapping each token to a leaf node labeled with the respective token, appearing as children of their container element/attribute node, and ordered following their order of appearance in the element/attribute text value [162, 163] (Fig. 2).

In the following, we will refer to the tree node representations of element/attribute tags as *structure nodes*, and to those of element/attribute values as *content nodes*.

2.2. Semantic/Knowledge Representation

The description of the semantic meaning of words/expressions and their relationships, also known as semantic/knowledge representation, has been a central topic in the fields of *Natural Language Processing* (NLP), *Information Retrieval* (IR), and *Artificial Intelligence* (AI) for the past two decades [126]. Here, *Description Logic* (DL) has been introduced as a family of formal knowledge representation languages, allowing to represent and describe semantic meaning [28], to be stored in so-called *Knowledge Bases* (KBs), i.e., repositories of machine-readable knowledge, available for automated processes aiming to achieve semantic-aware processing. Many languages for DL have been proposed [71, 197]: *Propositional Logic*, *First-Order Logic*, *Temporal Logic*, etc., with specific properties and applications mainly dedicated to semantic data analysis.

In this context, a typical KB structure is composed of a *Terminology-Box* (T-Box) and an *Assertion-Box* (A-Box) [28]. The T-Box underlines the set of concept definitions, while the A-Box consists of the collection of concept instances (also called individuals). With respect to (w.r.t.) a relational database, the T-Box is similar to the structure of the tables (database schema) whereas the A-Box is more like the data rows (tuples) inserted into the tables [14, 28]. As a result, various KB structures such as taxonomies, thesauri, and ontologies have been investigated and developed in NLP, IR, and AI to define, organize, and link semantic concepts in a KB [84].

A KB usually comes down to a *semantic network*[1] which is basically a graph consisting of nodes and arcs, organizing words/expressions in a semantic space [152] (cf. Fig. 3). Each node represents a *semantic concept* underlining a group of words/expressions, designating word *senses*. Arcs underline the semantic links connecting the concepts, representing semantic relationships (e.g., *synonymy*, *hyponymy* (*IsA*), *meronymy* (*PartOf*), etc. [123, 152]). Examples of lexical KBs are Roget's thesaurus [225], WordNet [123], and Yago [74]. In such structures, semantic information can be expressed as sets of triplets: $concept_1$-*relationship*-$concept_2$ (e.g., *Actor-IsA-Performer*, *Scene-PartOf-Movie* in Fig. 3), which are more commonly referred to as: *subject-predicate-object* triplets following the Semantic Web terminology [65] (covered in Section 6.6).

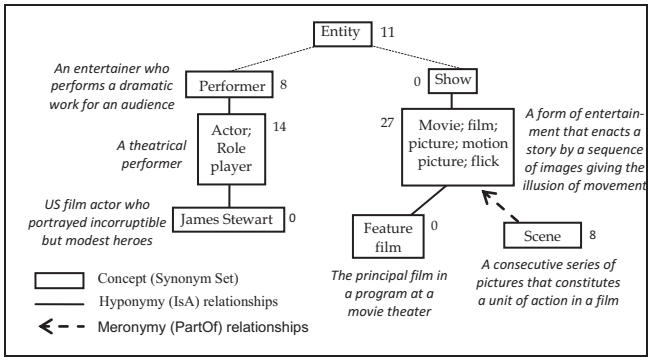


Fig. 3. Extract of the WordNet semantic network. Numbers next to concepts represent concept frequencies (based on the Brown corpus [53], cf. Section 3.5). Sentences next to concepts represent concept glosses.

3. Background in Word Sense Disambiguation

WSD underlines the process of computationally identifying the senses (i.e., semantic concepts designating the meanings) of words in context, to discover the author's intended meaning [80]. Different from traditional text mining techniques (e.g., lexical pattern discovery, syntactic dependency, co-occurrence, etc. [6, 69, 171]) which mainly capture the lexico-syntactic nature of text, and thus barely go beyond the surface appearance of words [126], WSD aims at identifying the underlying semantic meaning of information formulated with (possibly) different wordings and syntactic styles, in order to help identify the information that is most pertinent to the user's needs. The general WSD task consists of the following main elements: i) selecting words for disambiguation, iii) identifying and representing word contexts, ii) using reference knowledge sources, iv) associating senses with words, and v) evaluating semantic similarity between senses.

3.1. Selecting Words for Disambiguation

There are two possible methods to select target words for disambiguation: i) *all-words*, or ii) *lexical-sample*. In all-words WSD, e.g., [29, 144], the system is expected to disambiguate all words in a (flat) textual document. Although considered as a complete and exhaustive disambiguation approach, yet the *all-words* approach remains extremely time-consuming and labor intensive, where the high (time and processing) costs usually barely meet performance expectations [126]. In *lexical-sample* WSD, e.g., [64, 144], specific target words are selected for disambiguation (usually one word per sentence). These words are often the most ambiguous, and are usually chosen using supervised learning methods trained to recognize salient words in sentences [126]. Experimental results reported in [126] show high disambiguation accuracy using the *lexical-sample* approach, in comparison with the *all-words* approach.

Yet, a major difficulty in adopting the *lexical-sample* approach is in selecting ambiguous (target) words, due to the lack of formal methods to quantify *semantic ambiguity*, since current supervised learning approaches are time-consuming, including a training phase that requires training data which is not always available.

[1] In the remainder of the paper, terms *knowledge base* (KB) and *semantic network* are used interchangeably.

3.2. Identifying and Representing Context

Once words have been selected for disambiguation, their contexts have to be identified, to be utilized in the disambiguation process. In fact, WSD relies on the notion of context such that words that appear together in the same context usually have related meanings [100]. The *context* of a word in traditional flat textual data usually consists of the set of terms in the word's vicinity, i.e., terms occurring to the left and right of the considered word, within a certain predefined window size [100]. Other features can also be used to describe context, such as information resulting from linguistic preprocessing including part-of-speech tags (e.g., verb, subject, etc.), grammatical relationships (e.g., verb-subject, verb-object, etc.) [126]. Once the context has been identified, it has to be effectively represented to perform disambiguation computations. Here, the traditional *bag-of-words* paradigm is broadly adopted with flat textual data [80, 126], where the context is processed as a set of terms surrounding the word to disambiguate. A vector representation considering the number of occurrences of words in context can also be used [126], along with more structured context representations using co-occurrence graphs [1, 208]. Yet, the latter representations require substantial additional processing in comparison with the *bag-of-words* model.

3.3. Using Reference Knowledge Sources

In addition to the contexts of target words, external knowledge is essential to perform WSD, providing reference data which are needed to associate senses with words. Here, WSD methods can be distinguished as i) *corpus-based* or ii) *knowledge-based*, depending on the kind of external knowledge sources they rely on. The *corpus-based* approach is data-driven, e.g., [119, 138], as it involves information about words previously disambiguated, and requires supervised learning from sense-tagged corpora (e.g., SemCor [124] and OntoNotes [143]) where words/expressions have been associated with explicit semantic meaning, in order to enable predictions for new words. A *sense* here underlines a *labeled semantic category*[1] to be used in supervised learning, and is not formally defined (as opposed to having a formal *semantic concept* defined in a KB, cf. Section 2.2). A more restrictive view of *corpus-based* methods, known as *word sense induction*, e.g., [21, 87], seeks to automatically identify so-called *implicit senses*, i.e., *unlabelled semantic categories*[1] (designating the *uses*) of target words in a raw (unlabeled) corpus (e.g., the Brown corpus [53]). Methods in this category rely exclusively on corpus data where word relationships are derived from their contextualization and co-occurrence in the corpus itself. They cluster words with similar corpus statistical properties, producing an implicit sense inventory made of unlabelled word categories (clusters), where each category denotes an implicit sense [129][2].

On the other hand, *Knowledge-based* methods are knowledge-driven, e.g., [122, 128], as they handle a structured (and explicit) sense inventory and/or a repository of information about words that can be automatically exploited to distinguish their meanings in the text. Machine-readable KBs (dictionaries, thesauri, and/or lexical ontologies, e.g., Roget's thesaurus [225], WordNet [51], and Yago [74]) provide ready-made sources of information about word senses to be exploited in knowledge-based WSD. While *corpus-based* methods have been popular in recent years, e.g., [4, 7, 36], they are generally data hungry and require extensive training, huge text corpora, and/or a considerable amount of manual effort to produce a relevant sense-annotated corpus, which are not always available and/or feasible in practice. In addition, with *corpus-based* statistical approaches, the "true" understanding of words is hardly obtainable [84], since words are evaluated according to their statistical distribution in a corpus, often capturing syntactic or stylistic factors instead of semantic meaning [150]. Therefore, *knowledge-based* methods have been receiving more attention lately, e.g., [122, 126, 182], and include most XML disambiguation solutions (Section 4.2).

3.4. Associating Senses with Words

The culminating step in WSD is to associate senses with words, taking into account the target words' contexts as well as reference external knowledge about word senses. This is usually viewed as a word-sense classification task. In this regard, WSD approaches can be roughly categorized as *supervised* or *unsupervised*. On one hand, *supervised* methods, e.g., [119, 126, 203], involve the use of machine-learning techniques, using samples (a human expert manually annotates examples of words with the intended sense in context, where each sense underlines a *labeled semantic category*) provided as training data for a learning algorithm. The algorithm then induces rules to be used for assigning meanings to other occurrences of the words. External knowledge (mainly *corpus-based*) is used, and is combined with the human experts' own knowledge of word senses when manually tagging the training examples. While effective, yet *supervised* methods include a learning phase which is highly time-consuming, and requires a reliable training set with a wide coverage which is not always available.

On the other hand, *unsupervised* methods, e.g., [58, 137, 224], are usually fully automated and do not require any human intervention or training phase. Most recent (and XML-related) approaches, e.g., [115, 116, 137, 183], make use of a machine-readable KB (e.g., WordNet [51]) represented and processed as a *semantic network* (cf. Section 2.2). Given a target word to be disambiguated, WSD consists in identifying the semantic concept (word sense), in the reference semantic network, that best matches the semantic concepts (word senses) of terms appearing in the context of the target word. Semantic concept matching is usually performed using a measure of semantic similarity between concepts in the reference semantic network [22, 136] (cf. Section 3.5).

Note that a more restrictive view of *unsupervised* WSD applies to methods for *word sense induction*, which aim at clustering words which are (supposedly) semantically similar and can thus convey a specific meaning in a text corpus, without any external training data or predefined sense inventory (i.e., without predefined labeled categories or a KB) [19, 88]. Here, the induced senses have no external meaning, as they only match statistical patterns and syntactic divisions in the text corpus at hand [87] (refer to *word sense induction* in Section 3.3). In the remainder of this study, we constrain our presentation to the general definition of *unsupervised* WSD using a reference KB (i.e., *unsupervised* and *knowledge-based* WSD) [126].

3.5. Evaluating Semantic Similarity between Senses

Methods for evaluating semantic similarity between concepts (word senses) in a KB (semantic network), in order to perform *unsupervised* and *knowledge-based* sense matching, can be classified as [22]: i) *edge-based*, ii) *node-based*, and iii) *gloss-based* measures. *Edge-based* methods, e.g., [95, 145], are the most intuitive, estimating

[1] In machine learning and statistical classification, a *labeled category* is a class or group of entities/variables (e.g., words) sharing the same properties, and having an identifying (semantically meaningful) label. An *unlabelled category* however does not have an identifying label [129].

[2] In the remainder of this paper, the term *sense* means *explicit sense*, designating either a *labeled semantic category* (following the *corpus-based* approach) or a *semantic concept* in a KB (following the *knowledge-based* approach), unless stated otherwise.

similarity as the shortest path (in edges, or number of nodes) between the two concepts being compared, e.g., [219]:

$$\mathrm{Sim}_{\mathrm{Edge}}(c_1, c_2, \mathrm{KB}) = \frac{2 \times N_0}{N_1 + N_2 + 2N_0} \in [0,1] \quad (1)$$

where c_1 and c_2 designate two semantic concepts (word senses) in a reference KB, N_1 and N_2 are respectively the lengths of the paths separating c_1 and c_2 from their lowest common ancestor c_0 in KB, and N_0 is the length of the path separating c_0 from the root of KB.

Node-based approaches, e.g., [84, 150], incorporate an additional knowledge source: *corpus statistical analysis*, to augment the information already present in the reference KB. They estimate similarity as the maximum amount of *information content* (i.e., a function of concept occurrence probability, computed based on corpus statistics and KB structure [150]) that concepts share in common, e.g., [103]:

$$\mathrm{Sim}_{Node}(c_1, c_2, \mathrm{KB}, C) = \frac{2 \log p(c_0)}{\log p(c_1) + \log p(c_2)} \in [0,1] \quad \text{having} \quad p(c_i) = \frac{Freq(c_i)}{W} \quad (2)$$

where $p(c_i)$ is the occurrence probability of concept c_i designating the normalized frequency of occurrence of c_i in a reference corpus C (e.g., the Brown corpus [53]), W designates the size (total number of words) in the corpus (cf. concept frequencies in Fig. 3).

Gloss-based methods, e.g., [10, 100], evaluate semantic similarity as the word overlap between the glosses of concepts (word senses) being compared, a gloss underlining the textual definition describing a word sense (e.g., the gloss of the 1st sense of word *"Actor"* in WordNet is *"A theatrical performer"*, cf. Fig. 3), e.g., [10]:

$$\mathrm{Sim}_{Gloss}(c_1, c_2, \mathrm{KB}) = \left(\mathrm{gloss}(c_1) \cup \mathrm{gloss}(\mathrm{Rel}(c_1))\right) \cap \left(\mathrm{gloss}(c_2) \cup \mathrm{gloss}(\mathrm{Rel}(c_2))\right) \quad (3)$$

where $gloss(c_i)$ is the bag of words in the textual definition of concept (word sense) c_i, and $Rel(c_i)$ is the set of concepts related to c_i through a semantic relationship in KB.

It has been shown that *gloss-based* measures evaluate, not only *semantic similarity*, but also *semantic relatedness* [137], which is a more general notion: including similarity as well as any kind of functional relationship between terms (e.g., *"penguin"* and *"Antarctica"* are not similar, but they are semantically related due to their *natural habitat* connection), namely antonymy (e.g., *"hot"* and *"cold"* are dissimilar: having opposite meanings, yet they are semantically related), which makes *gloss-based* measures specifically effective in WSD [126]: matching not only similar concepts, but also semantic related ones[1].

3.6. Discussion

To sum up, WSD relies on the notion of context, such that words that appear together in the same textual context have related meanings. On one hand, *supervised* and *corpus-based* methods match words in context with senses represented as labeled categories, using machine learning techniques applied on text corpus statistics to categorize words w.r.t. senses. They usually require extensive training and large statistical corpora [126], and thus do not seem practical for the Web. In addition, they evaluate the meanings of words according to their statistical distribution in a corpus, often capturing syntactic factors instead of the "true" semantic meaning of words which is hard to obtain with this category of methods [84]. On the other hand, *unsupervised* and *knowledge-based* WSD have been largely investigated recently (including most methods targeting XML data, cf. Section 4.2), where a target word in context is matched with senses represented as concepts in a machine readable KB, using semantic similarity measures to compare and identify the best matches among target and context word senses in the KB. While usually more efficient than their *supervised* and *corpus-based* counterparts, yet the quality of *unsupervised* and *knowledge-based* approaches largely depends on the accuracy, coverage, and extensibility of the KB used as semantic reference [126], where KBs are not easy to handle and maintain (cf. Section 7.6). Fig. 4 depicts a simplified taxonomy summarizes the main characteristics of traditional WSD approaches developed in the literature. The interested reader can refer to [80, 87, 126] for extensive reviews on traditional WSD.

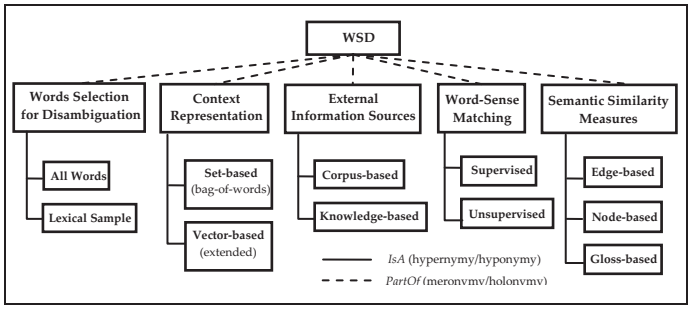


Fig. 4. Simplified taxonomy of the properties of WSD approaches.

4. XML Semantic Analysis and Disambiguation

4.1. XML Semantic-Aware Processing

Given the semi-structure (tree-like) nature of XML, most methods for processing XML data (including XML querying, classification, clustering, and integration techniques, cf. Section 6) have leveraged results from prominent research on combinatorial pattern/tree matching [166], namely tree edit distance and approximation methods, e.g., [40, 134]. Other works have focused on extending conventional information retrieval techniques [120], using vector-space models to represent and handle XML in structural feature spaces, e.g., [27, 202]. Extensive reviews of both families of XML-based methods can be found in [61, 187, 191]. Among those, the majority of existing approaches use only syntactic information in processing XML data, while ignoring the semantics involved. Yet, recent methods have attempted to integrate semantic and structural features in handling XML information. One of the early solutions to propose such a method is [198], where the authors make use of a textual similarity operator and utilize *Oracle*'s *InterMedia* text retrieval system to improve XML similarity search. In a more recent extension of their work [161], the authors define a generic ontological model, built on WordNet, to account for semantic similarity (instead of utilizing *Oracle InterMedia*). Another approach in [96] integrates XML label semantics (synonyms, compound words, and abbreviations, identified using WordNet) within an XML vector-space representation, so as to improve XML similarity evaluation in XML mining applications. More recent XML structure-based methods in [131, 132] have identified the need to support XML tag semantic similarity (including synonyms, hyponyms, etc., using WordNet) instead of only tag syntactic equality while performing XML clustering. In [109], the authors introduce a structure and content based method for comparing XML documents conforming to the same grammar (DTD/XSD), and consider semantic similarity evaluation between element/attribute values, using a variation of

[1] The reader can refer to [22, 136, 159] for comprehensive reviews and evaluations of semantic similarity measures.

the edge-based semantic similarity methods [145]. In [188], the authors introduce a hybrid XML similarity approach integrating a node-based semantic similarity measure [103] within a classic tree edit distance algorithm [31], to compare XML tag names. The approach is later extended to consider XML sub-tree structural and semantic similarities [193]. Similar techniques have been exploited in a wide array of XML similarity-based processing applications, which we further categorize and describe in Section 6.

While the aforementioned methods have endeavored to integrate a dose of semantic analysis in XML processing, yet, most of them completely neglect the problem of semantic ambiguity, or implicitly consider it as *already solved*. In other words, they consider XML labels to be inherently associated with disambiguated semantic concepts in the reference KB, which is unfortunately not the case in practice. Recent studies (described in the following section) have affirmed that the semantic analysis of XML documents involves, first and foremost, the identification of the possible senses of XML tag names/values (which are typically ambiguous, similarly to flat textual data). Hence, a word sense disambiguation task is required in order to assign each XML node label with the most appropriate sense, as a prerequisite to XML semantic-aware processing.

4.2. XML Semantic Disambiguation

The main challenges in XML semantic disambiguation reside in: i) how to define the notion of XML (structural) contextualization, ii) how to process XML context information for disambiguation, and iii) how to assign senses to XML node labels.

4.2.1. XML Context Identification

While the context of a word in traditional flat textual data consists of the set of terms in the word's vicinity [100], yet there is no unified definition of the context of a node in an XML document tree. Different approaches have been investigated, namely: i) *parent node context*, ii) *root path context*, iii) *sub-tree context*, and iv) *crossable edges context*, which we describe in the following sub-sections.

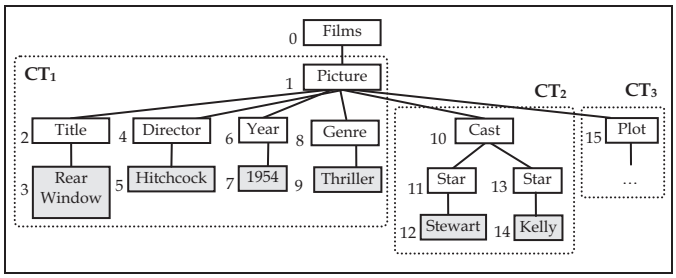


Fig. 5. *Canonical trees* identified in XML tree *T* from Fig. 2.

4.2.1.1. Parent Node Context (PNC)

The authors in [184, 185] consider that the context of an XML leaf element (i.e., an element containing a data value) or an attribute can be efficiently determined by its parent element, and thus process a parent node and its children leaf element/attribute nodes as one unified *canonical* entity. The approach is based on the observation/assumption that an XML leaf element constitutes by itself a semantically meaningless entity. As a result, the authors introduce the notion of *canonical tree* as a structure grouping together a leaf element node with its parent node, which is deemed as the simplest semantically meaningful structural entity. For instance, Fig. 5 depicts three canonical trees: CT_1, CT_2, and CT_3 identified in XML document tree *T* of Fig. 2. Here, one can realize that the context of leaf element nodes *T*[2] (*T*[2].ℓ = *"Title"*), *T*[4] (*"Director"*), *T*[6] (*"Year"*), and *T*[8] (*"Genre"*) is node *T*[1] (*"Picture"*). Likewise, the context of leaf element nodes *T*[11] and *T*[13] (*T*[11].ℓ = *T*[13]. ℓ = *"Star"*) is node *T*[10] (*"Cast"*).

4.2.1.2. Root Path Context (RPC)

In [182, 183], the authors extend the notion of XML node context to include the whole XML root path, i.e., the path consisting of the sequence of nodes connecting a given node with the root of the XML document (or document collection). For instance, Fig. 6 represents the contexts of each XML node in XML document tree *T* of Fig. 2. Note that the approach targets structure-only XML disambiguation and disregards data values.

Node	Context root path
Films	/Films
Picture	/Films/Picture
Title	/Films/Picture/Title
Director	/Films/Picture/Director
Year	/Films/Picture/Year

Node	Context root path
Genre	/Films/Picture/Genre
Cast	/Films/Picture/Cast
Star	/Films/Picture/Cast/Star
Plot	/Films/Picture/Plot

Fig. 6. *Root path* contexts identified in XML tree *T* from Fig. 2.

The authors consequently perform per-path sense disambiguation, comparing every node label in each path with all possible senses of node labels occurring in the same path. Each XML path is transformed into a weighted graph, with nodes underlining the senses of each path element, and edges connecting node senses following path direction and node sense semantic similarities (Fig. 7). The authors utilize an existing gloss-based WordNet similarity measure [10] (Formula 3) and introduce an edge-based measure (similar to Formula 2) to compare semantic senses in the weighted graph, where semantic similarity scores are assigned to corresponding graph edge weights. Then, selecting the appropriate sense for a given node label consists in identifying the set of node senses, in the weighted graph, where the sum of the weights over their edges is maximum (cf. highlighted nodes in Fig. 7).

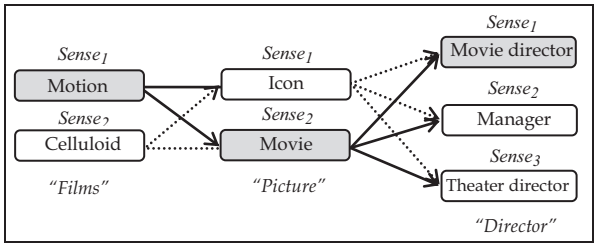


Fig. 7. Sample weighted graph for the root path *Films/Picture/Director*. We only show a limited number of senses for each node label, and omit edge weights (designating semantic similarities between concepts) for ease of presentation.

4.2.1.3. Sub-tree Context (STC)

Different from the notions of parent context and path context, the authors in [200] consider the set of XML nodes contained in the sub-tree rooted at a given element/attribute node, i.e., the set of labels corresponding to the node at hand and all its subordinates, to describe the node's XML context. For instance, Fig. 8 represents the sub-tree contexts for each XML element node from XML document tree *T* in Fig. 2, where sub-trees are numbered following the corresponding sub-tree root node (pre-) order. Note that XML data values in [200] are considered as part of the context information of an XML element/attribute node, yet are not processed separately as nodes targeted for disambiguation.

The authors apply a similar paradigm to identify the contexts of all possible node label senses in the WordNet KB (i.e., sub-tree

contexts, similar to the ones in Fig. 8, can be identified for each sense in the sample WordNet extract in Fig. 3). As a result, the target XML node to be disambiguated (along with its XML context sub-tree), and each of its potential node label senses in the reference KB (each with its own KB context sub-tree) are represented each as a set of lexical words/expressions: extracted from the corresponding sub-tree context node labels. For instance, the context set of node T[2] (*T*[2].ℓ = *"Title"*) in Fig. 3 is {*"Title"*, *"Rear Window"*}, whereas the context set of node T[10] (*"Cast"*) consists of terms {*"Cast"*, *"Star"*, *"Stewart"*, *"Kelly"*}. Then, the target XML node's label is processed for sense disambiguation by comparing the XML node context set with each of the candidate sense context sets. The authors in [200] utilize the *cosine* similarity measure to perform context set comparison, where sets are extended to vectors including *TF-IDF* [1] word frequencies. The target XML node is finally mapped to the semantic sense where their context sets (vectors) have the highest similarity.

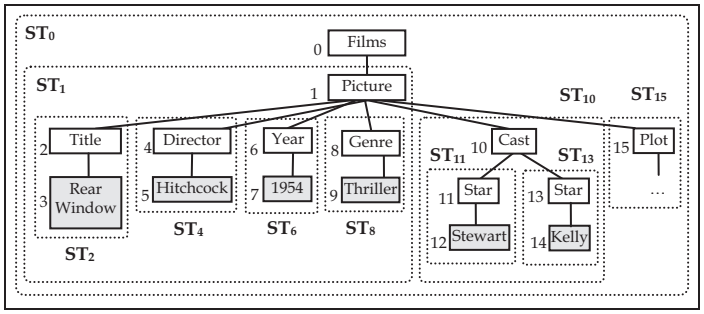


Fig. 8. *Sub-tree* contexts identified in XML tree *T* from Fig. 2.

4.2.1.4. CROSSABLE EDGES CONTEXT (CEC)

In [115, 116], the authors combine the notions of parent context and descendent (sub-tree) context in disambiguating generic structured data (e.g., XML, web directories, and ontologies). The authors consider that a node's context definition depends on the nature of the data and the application domain at hand. They propose various edge-weighting measures, namely a Gaussian decay function (cf. Formula 4) to identify *crossable edges*, such that nodes reachable from a given node through any *crossable edge* (following a user-specified direction, e.g., *direct*: ancestor/descendent, *opposite*: descendent/ancestor, or both) belong to the target node's context:

$$weight(n_c, n) = 2 \times \frac{e^{-\frac{d^2}{8}}}{\sqrt{2 \times \pi}} + 1 - \frac{2}{\sqrt{2 \times \pi}} \quad \textbf{(4)}$$

where n_c is a context node, n is the target node, and d is the distance (in number of edges) separating n_c form n in the XML tree. For instance, given XML document *T* tree in Fig. 2 (reported in Fig. 8), assume that the weight of the parent/child arcs is 1 in the *direct* direction and 0.5 in the *opposite* one, and that the maximum number of allowed crossings (to determine the context of a target node) is 2. As a result, the graph context of node *T*[13] (*"Star"*) would be made of the terms *"Star"* (*T*[13]), *"Kelly"* (*T*[14]), *"Cast"* (*T*[10]), *"Picture"* (*T*[1]), and *"Star"* (*T*[11]). In fact, the respective distances between node *T*[13] and nodes *T*[14], *T*[10], *T*[1], and *T*[11] are: 1 (i.e., 1 arc crossed in the *direct* direction with weight 1), 0.5 (i.e., 1 arc crossed in the *opposite* direction with weight 0.5), 1 (i.e., 2 arcs crossed in the *opposite* direction with weight 0.5), and 2 (i.e. 2 arcs crossed in the *opposite* direction with weight 0.5, and 1 arc crossed in the *direct* direction with weight 1). Then, following Formula 4, *weight*(*T*[14], *T*[14]) = 1, *weight*(*T*[14], *T*[13]) = 0.91, *weight*(*T*[10], *T*[13]) =0.95, *weight*(*T*[1], *T*[13]) = 0.91, and *weight*(*T*[11], *T*[13]) = 0.8.

Structure disambiguation is then undertaken by comparing the target node label with each candidate sense (semantic concept) corresponding to the labels in the target node's context, taking into account corresponding XML edge weights. The authors utilize an edge-based semantic similarity measure [93] (cf. Formula 1), exploiting the *hypernymy/hyponymy* relationships (and excluding remaining relationships such as *meronymy* and *holonymy*), to identify the semantic sense which best matches the target node label.

4.2.2. XML CONTEXT REPRESENTATION

Another major issue in XML semantic disambiguation is how to effectively represent and process the context of an XML node, once it has been identified, taking into account the structural positions of XML data in order to perform disambiguation.

In fact, most existing WSD methods - developed for flat textual data (Section 3) and/or XML-based semi-structured data [182-185] - adopt the *bag-of-words* paradigm where context is processed as a plain set of words surrounding the term/label (XML node) to be disambiguated. Hence, all context nodes are treated the same, despite their structural positions in the XML tree. One approach identified as the *relational information model* in [115, 116] (developed within the *CEC* approach, cf. Section 4.2.1.4) extends the traditional *bag-or-words* paradigm toward a vector-based representation with *confidence scores* combining: i) distance weights separating the context and target nodes, and ii) semantic weights highlighting the importance of each sense candidate. On one hand, the authors introduce a distance Gaussian decay function (cf. Formula 4) estimating edge weights such that the closer a node (following a user-specified direction), the more it influences the target node's disambiguation [115, 116]. The distance decay function is not only utilized in identifying the context of a target node (Section 4.2.1.4), but also produces weight scores which are assigned to each context node in the context vector representation, highlighting the context node's impact on the target node's disambiguation process (Fig. 9).

Context $_{BOW}$ (T[13]) = {T[13].ℓ, T[14].ℓ, T[10].ℓ, T[1].ℓ, T[11].ℓ } = {*"Star"*, *"Kelly"*, *"Cast"*, *"Picture"*} [2]

Context $_{RIM}$ (T[13]) =

"Star"	*"Kelly"*	*"Cast"*	*"Picture"*
1.8	0.91	0.95	0.91

Fig. 9. Sample bag-of-words (BOW) context representation versus relational information model (RIM) context representation for target node T[14] ("Kelly") in document tree *T* in Fig. 2, built with a maximum of 2 edge crossings (cf. example in Section 4.2.1.4).

On the other hand, the authors also include a semantic decay function, considering the frequency of usage of senses (i.e., designating how often a sense is used in common language), in order to assign a higher/lower weight (confidence) score highlighting the impact of each candidate sense on the disambiguation process. This is based on the assumption that senses with a higher usage frequency should be deemed more relevant in semantic evaluation. To do so, the authors exploit WordNet's ordered lists of senses ranked based on their frequencies of usage in the Brown text corpus:

[1] *Term Frequency – Inverse Document Frequency* is a term weighting score developped in information retrieval to highlight the relative importance of a term in describing a given document within a document collection [8].

[2] The second occurrence of node *"Star"* (*T*[11]) is removed from the bag-of-words context when adopting a *set-based* model, and can be sustained when utilizing a *multi-set* based model.

$$decay(s_i, t) = 1 - \rho \frac{pos(s_i) - 1}{|Senses(t, KB, C)|} \quad (5)$$

where t is a term (node label) being disambiguated, s_i is a candidate sense for t, $\rho \in [0, 1]$ is a parameter set at 0.8 (by the authors[1]), $Senses(t, KB, C)$ is the ordered list of candidate senses for t in KB based on their usage frequency in a corpus C (i.e. the first is the most common sense, and so forth), and $pos(s_i)$ is s_i's position in $Senses(t, KB, C)$. Given Formula 5, the weight (confidence) score in choosing a sense candidate s_i from $Senses(t, KB, C)$ is inversely proportional to its position, $pos(s_i)$, in $Senses(t, KB, C)$, which emulates human behavior in choosing the right meaning of a term [115].

4.2.3. Associating Senses with XML Nodes

Once the contexts of XML nodes have been determined, they can be handled in different ways to perform XML disambiguation. Two automated approaches, both *unsupervised* and *knowledge-based*, have been adopted in the literature, which we identify as: i) *concept-based* and ii) *context-based*. Also, semi-automated *feedback* techniques have been suggested to improve the results of disambiguation methods.

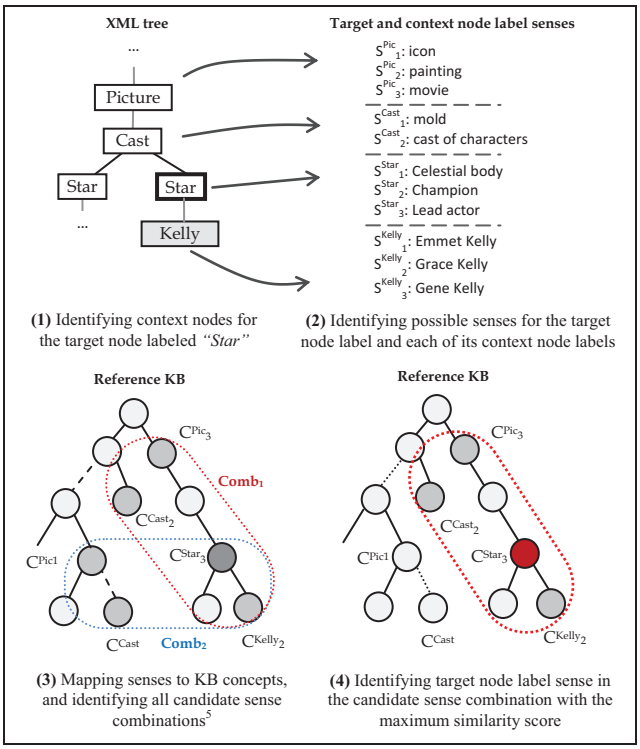


Fig. 10. Depiction of *concept-based* disambiguation approach.

4.2.3.1. Concept-based Approach

The *concept-based* approach adopted in [182, 183] was inspired by the *original Lesk algorithm* developed for disambiguating flat text [100], and consists in evaluating the semantic similarity between XML target node senses (concepts) and those of its context nodes, using measures of semantic similarity between concepts in a KB (cf. Section 3.5). The overall process is depicted in Fig. 10.

Given a target node to disambiguate, and after identifying context nodes (cf. Section 4.2.1) and performing XML context representation (cf. Section 4.2.2), the possible senses for each context node label as well as the candidate senses for the target node label are identified (Fig. 10, steps 1 and 2) and mapped to the reference KB. Then, each potential combination of context/target node senses is identified (Fig. 10, step 3), which comes down to $\prod_{i=1}^{N} |Senses(n_i.\ell)|$ *candidate combinations* where N represents all nodes in the disambiguation context, including the target node. Then, semantic similarity (cf. Section 3.5) is evaluated between pairwise senses in each candidate combination, averaged to produce a candidate combination score. The target node label is matched with the candidate sense corresponding to the candidate combination having the maximum score (Fig. 10, step 4).

4.2.3.2. Context-based Approach

The *context-based* approach introduced in [200] was inspired by the *simplified Lesk algorithm* developed for flat text [205], and consists in building a context set (the authors adopt the *bag-of-words* model, albeit a vector representation using the *relational information model* can be used, cf. Section 4.2.2) for each target node sense (concept) in the KB, as well as for the target node in the XML document tree, and then comparing context sets to select the target sense with maximum context set similarity. The overall process is depicted in Fig. 11. Given a target node to disambiguate, and after identifying context nodes, a context set representation is built for the target XML node in the XML tree (Fig. 11, steps 1 and 2) and a context set representation for each of the candidate senses in the KB (semantic network, Fig. 11, step 3). The XML context set is compared with each of the KB context sets, using a typical set comparison measure (e.g., *Jaccard* similarity). Then, the target node label is matched with the candidate sense having the KB context set with maximum score w.r.t. the XML target node context set (Fig. 11, step 4).

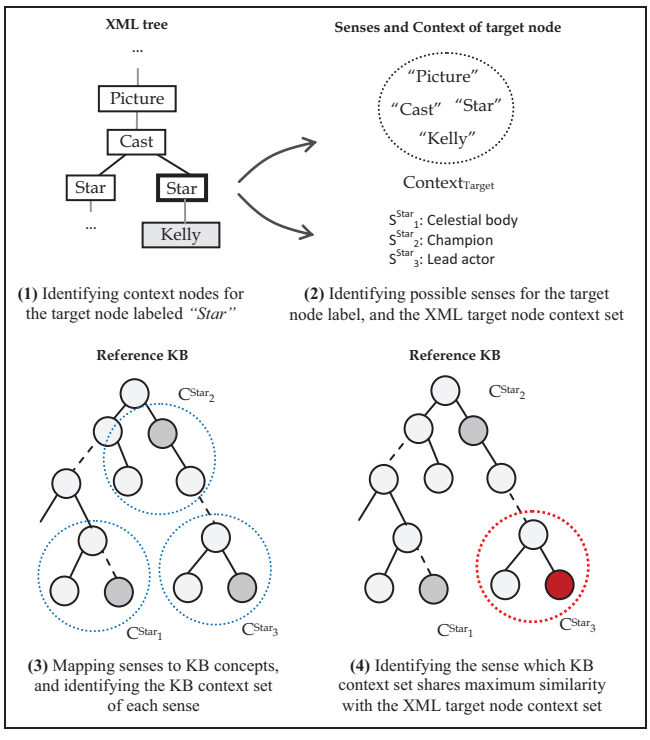


Fig. 11. Depiction of *context-based* disambiguation approach.

4.2.3.3. Feedback Techniques

While both automated disambiguation (sense assignment) approaches have been shown to produce useful results in a single run,

[1] Parameter ρ was set empirically by the authors in [115, 116] without any specific rationale.

yet recent studies in [115, 125] have argued the need for dedicated feedback techniques allowing the user (e.g., a human expert) to further refine the initial disambiguation result following her needs and individual understanding of the data. The main process consists in allowing the user to manually activate/deactivate the influence of a (set of) select candidate sense(s), after running the automate disambiguation process, and then repeating the same task by performing (as many) successive disambiguation runs (as needed) until reaching the desired result [125]. This can be achieved through two consecutive phases [115, 125]. First, producing a ranking of the plausible senses for each target node label, based on the similarity scores computed after each run of the automated disambiguation process. The ranking would highlight to the user the *confidence* of the automated process in choosing each sense as the right one for the target node label. Second, given the produced sense ranking, the user can retain (or disregard) the proper (noisy) sense(s) following her understanding of the meaning of the target node label. The process is repeated until the user identifies the right sense(s) for the target node label. In this context, the authors in [115] suggest multiple semi-automated strategies facilitating the user's task in providing feedback:

i) *Simple Feedback*: the top-N ranked senses are retained after each run, where N is a threshold value chosen by the user. When N=1, the system would require only one feedback iteration to select the topmost sense for the target node label.

ii) *Knockout feedback*: the sense with the lowest confidence (i.e., at the bottom of the ranked list of plausible senses) is deactivated (disregarded) at each run, until reaching the top-N senses. This approach requires more runs than *simple feedback*, but is usually more effective due to its greater gradualness [115].

iii) *Stabilizing knockout feedback*: a fix-point version of the *knockout feedback* method. It consists in computing an average confidence variation score for the target node label, w.r.t. all potential senses, between the current run and the previous run, then repeats the process until variation stabilizes (i.e., becomes lesser than a given threshold). Then, the top-N senses are retained. This method usually achieves the same effectiveness levels as *knockout feedback* while requiring less time [115].

While allowing user feedback through semi-automated processes seems certainly interesting, and promises to improve and adapt disambiguation results following the user's needs, nonetheless, feedback techniques inherently require substantial additional time (i.e., running successive iterations of the disambiguation process), and additional manual effort (i.e., fine-tuning thresholds, and manually validating senses). A possible compromise could be to limit feedback only to those most critical/ambiguous node labels (which we further discuss in Sections 7.1 and 7.8).

4.3. Comparative Complexity Analysis

Comparing the *effectiveness* (quality) and *efficiency* (time performance) levels of XML disambiguation approaches is not trivial theoretically, and requires a dedicated experimental study (cf. Section 5). Yet, one can theoretically analyze the computational complexity of existing methods, which would provide an informed hint on efficiency levels. Note that most existing studies do not provide complexity analyses.

On one hand, the complexity of *context identification* (cf. Section 4.2.1) and *context representation* (cf. Section 4.2.2) is almost linearly dependent on the size of the XML document at hand, and can be roughly performed in one single traversal over the XML document tree T, thus requiring $O(|T|)$ where $|T|$ is the number of structure and content nodes in T. On the other hand, when it comes to associating senses with XML nodes (i.e., the core task in the disambiguation process, cf. Section 4.2.3), one can realize that the *concept-based* approach is more computationally complex (less efficient) than the *context-based* approach.

First, the complexity of the *concept*-based approach in disambiguating one target node label comes down to $O(\prod_{i=1}^{N} |Senses(n_i.\ell)| \times N \times ((N\text{-}1)/2) \times Sim_{Semantic})$ time, considering respectively: the number of candidate combinations to process $\prod_{i=1}^{N} |Senses(n_i.\ell)|$, the number of pairs of senses to compare within each combination $N\times((N\text{-}1)/2)$, as well as the complexity of the semantic similarity measure utilized $Sim_{Semantic}$ (which can be either a node-base, edge-based, or gloss-based measure, or their combination, cf. Section 3.5), which simplifies to $O(|Senses(n.\ell)|^N \times N^2 \times Sim_{Semantic})$. Note that the complexity of a typical edge-based and/or node-based semantic similarity measure simplifies to $O(|KB| \times depth(KB))$ [103, 219] whereas the complexity of a typical gloss based similarity measure simplifies to $O(|gloss|^2)$ [10].

Second, the complexity of the *context-based* approach in disambiguating one target node label comes down to $O(|Senses(n.\ell)| \times Sim_{Set})$ time, considering respectively: the number of candidate (target node label) senses to process $|Senses(n.\ell)|$, which underlines the number of KB context sets built for each candidate sense in the KB, as well as the complexity of the set-based similarity measure used to compare each of the KB context sets with the XML target node context set, Sim_{Set}. For instance, $Sim_{Jaccard}$ simplifies to $O(N^2)$ time where N is the maximum number of elements per set.

A hybrid approach in [115, 116] (i.e., *CEC*, cf. Section 4.2.1.4) combines variants of the two preceding methods to disambiguate generic structured data.

4.4. Discussion

To sum up, XML disambiguation approaches have been extended from traditional flat text WSD, introducing adapted processes for identifying *disambiguation contexts*, producing *context representations*, and performing *word-sense matching* taking into account the semi-structured nature of XML. The main characteristics of existing solutions are summarized in TABLE 1.

Nonetheless, we argue that most XML disambiguation methods share several common limitations, namely: i) ignoring the problem of *semantic ambiguity* in identifying target nodes to process for disambiguation (i.e., processing all nodes for disambiguation, which might be needless), ii) neglecting XML structural features by using traditional flat text representation techniques (e.g., the *bag-of-words* paradigm) which are not designed for describing structured XML data [182, 185], iii) only partially considering the structural relationships of XML nodes (e.g., parent-node [185] or ancestor-descendent relationships [183]), and iv) using fixed contexts (e.g., parent node [185], root path [183]) or preselected semantic similarity measures (e.g., edge-based [93], or gloss-based [183]), thus minimizing the user's involvement in the disambiguation process. We further discuss each of above limitations along with other ongoing challenges and future directions in Section 7.

TABLE 1. COMPARING XML DISAMBIGUATION METHODS W.R.T. FIG. 4.

	Approaches	PNC [184, 185]	RPC [182, 183]	STC [200]	CEC [115, 116]
Target	**Node Selection** 1- All nodes 2- Sample nodes	All nodes	All nodes	All nodes	All nodes
	XML Data 1- Structure-only 2- Structure & Content	Structure and Content	Structure only	Structure only	Structure only
Context	**Context Model** 1- Parent node 2- Root path 3- Sub-tree 4- Crossable edges	Parent node	Root path	Sub-tree	Crossable edges
	Context Representation 1- Set (BOW) 2- Vector (Extended)	Set	Set	Set	Vector
	Context Size 1- Fixed 2- Flexible	Fixed	Fixed	Fixed	Flexible
Source	**External Information** 1- Corpus 2- KB	KB	KB	KB	KB
Matching	**Word-Sense Matching** 1- Supervised 2- Unsupervised	Unsupervised	Unsupervised	Unsupervised	Unsupervised
	Semantic Similarity 1- Edge based 2- Node based 3- Gloss based	Edge-based[1]	Edge-based & Gloss-based	NA	Edge-based
Characteristics	**User Involvment** 1. Limited 2. Partial	Limited	Limited	Limited	Partial
	User feedback	NA	NA	NA	Semi-automatic
	Complexity	NA	$O(\lvert Senses(n.\ell)\rvert^N \times N^2 \times Sim_{Semantic})$	$O(\lvert Senses(n.\ell)\rvert \times Sim_{Set})$	NA
	Evaluation method 1- Standalone 2- Embedded[2]	Embedded	Standalone	Embedded	Standalone
	Applications	XML Search	XML clustering	Document classification	Generic

5. EVALUATION METHODOLOGY

As for empirical evaluation, XML disambiguation methods can be assessed (similarity to WSD) in two ways [126]: i) *stand-alone*, i.e., as a separate application, where the tester evaluates the quality of the disambiguation process, or ii) *end-to-end*, i.e., as a component embedded within an application (such as data search, document clustering, or document classification, cf. Section 6), where the tester aims to demonstrate whether the disambiguation task improves (or not) the performance of the application as a whole. For clearness of presentation, we present *stand-alone* evaluation measures in this paper, and omit *end-to-end* evaluation measures since they are application dependent[3] (the interested reader can refer to Section 6 for a detailed discussion and references regarding XML semantic-aware applications and evaluation measures).

5.1. TEST MEASURES

The *effectiveness* (i.e., quality) of an automatic (XML) disambiguation approach can be evaluated based on the quality of the mapping between user identified senses and system generated senses for a select number of test data (e.g., XML element/attribute tag names and/or values targeted for disambiguation). In this context, most existing approaches suggest to i) first manually solve the disambiguation task, and then ii) use the results as a reference to evaluate the quality of the senses produced by the system [126].

Here, the *Precision* and *Recall* evaluation measures adopted from the field of information retrieval [157] are usually utilized to compare user and system generated senses [126]. *Precision* (*PR*) identifies the number of correctly identified senses, w.r.t. the total number of generated senses (correct and false) produced by the system. *Recall* (*R*) underlines the number of correctly identified senses, w.r.t. the total number of correct senses, including those not identified by the system. Having:

- *A* the number of correctly identified senses (true positives),
- *B* the number of wrongly identified senses (false positives),
- *C* the number of correct senses not identified by the system (false negatives). *Precision* and *recall* are computed as follows:

$$PR = \frac{A}{A+B} \in [0,1] \quad \text{and} \quad R = \frac{A}{A+C} \in [0,1] \qquad \textbf{(6)}$$

High *precision* denotes that the disambiguation process achieved high accuracy in identifying correct senses, whereas high *recall* means that few correct senses were missed by the system. In addition to evaluating *precision* and *recall* separately, it is a common practice to consider *F-value* as a combined measure, representing the harmonic mean of *precision* and *recall*. High *precision* and *recall*, and thus high *F-value* indicate high disambiguation quality:

$$F\text{-}Value = \frac{2 \times PR \times R}{PR + R} \in [0, 1] \qquad \textbf{(7)}$$

Disambiguation *coverage* is another interesting measure that can also be utilized to evaluate an aspect of disambiguation quality which is not captured by *precision* and *recall*: the case in which the system does not provide an answer (i.e., when the system is unable to match an XML node label with any semantic sense). *Coverage* (*CR*) measures the percentage of XML nodes in the test document (collection) for which the system provided a sense assignment, over the total number of assignments (answers) that should have been provided by the system:

$$CR = \frac{A + B}{A + C} \in [0, 1] \qquad \textbf{(8)}$$

Besides evaluating *effectiveness* (i.e., quality), note that evaluating the *efficiency* (i.e., time and/or space performance) of XML disambiguation methods is almost completely dismissed in existing approaches (along with complexity analyses), and needs to be addressed in upcoming studies.

5.2. TEST DATA

Recent studies in [182, 183] have published information regarding XML-based test data and manually annotated tag names, which can be used as a baseline for future evaluation/comparative studies.

TABLE 2. CHARACTERISTICS OF TEST DATASETS (FROM [182, 183]).

	# Leaf Nodes	Max Fan-out	Max/Avg Depth	Size	# Docs	# Elements	# Terms in Text Values
DBLP	13,209	20	3/3	822KB	3000	8231	1437
IEEE	228,869	43	8/5.15	150MB	4874	135,869	5224
PubMed	18,202	40	7/6.27	2608KB	1000	11,489	7838
Reuters	15,159	12	4/3.92	1911KB	572	7727	3235
Shakespeare	13,856	194	7/5.92	1446KB	7	7517	2049
Wikipedia	267,718	141	12/3.79	122MB	10,000	174,688	5989
Hybrid-Data	24,769	80	7/4.33	4712KB	2325	16,341	3213

The test data were collected from various application domains with different characteristics, which we briefly describe in TABLE 2 : i) *DBLP*: a subset of the DBLP XML data archive[4] containing data

[1] The authors in [184, 185] use multiple heuristic functions which are mostly comparable to edge-based semantic similarity measures.

[2] Standalone disambiguation solutions are evaluated independently, whereas embedded solutions are evaluated within specific applications (e.g., XML semantic search), which we discuss in Section 5.

[3] E.g., evaluating the quality of the XML disambiguation process embedded within a document clustering application comes down to measuring the quality of the produced clusters using cluster evaluation measures such as *inter-cluster* and *intra-cluster* indexes. The same goes for all applications.

[4] http://dblp.uni-trier.de/xml/

concerning scientific bibliography. The *DBLP* dataset exhibits high structural variety and short text values (e.g., author names, paper titles, conference names, etc.); ii) *IEEE*: a subset of the IEEE collection version 2.2, which has been used in the INEX document mining track 2008[1]. IEEE articles generally follow a complex structural schema, with lots of abbreviations in element names; iii) *PubMed*: a scientific dataset, containing biomedical articles obtained as results of the query *"protein"* submitted to the *PubMed* search engine[2]. *PubMed* data exhibits deeply nested elements with relatively long text values (e.g., journal abstracts); iv) *Reuters*: news headlines from the Reuters RSS channel[3]. Documents in this dataset have a regular structure with short text values (e.g., news titles, links, descriptions, etc.); v) *Shakespeare*: a selection of plays from the Shakespeare 2.0 collection[4]. This dataset has a rich structure and contains long text values (e.g., actors' speeches, where all the lines corresponding to the same speech were concatenated to form a unique element); vi) *Wikipedia*: a subset of the Wikipedia XML corpus used in the INEX 2007 document clustering track[5], including data representing encyclopedia articles extracted from Wikipedia. This collection contains big articles, with long textual values (e.g., paragraphs); vii) *Hybrid-Data*: a heterogeneous dataset combining documents from the above collections (i.e., *DBLP* 30%, *Wikipedia* 15%, *Reuters* 20%, *PubMed* 20%, and *Shakespeare* 15%).

Note that the datasets in TABLE 2 have been partly utilized for *stand-alone* XML disambiguation evaluation in [183], and for *end-to-end* evaluation within an XML document clustering application in [182]. Other experimental datasets have been partly described in alternative studies, e.g., [115, 116][6]. A major challenge in this context is to integrate experimental data in a unified benchmark to be used as a gold standard data repository for future XML (and semi-structured) disambiguation methods (cf. Section 7.10).

6. XML Semantic-Aware Applications

As for usage in practical scenarios, the diversity of XML and semi-structured data highlights a wide spectrum of applications which can benefit, in one way or another, from XML semantic analysis and disambiguation. Most applications in this context are built around methods for *XML structure* and *semantic similarity* evaluation, e.g., [3, 187, 191], i.e., comparing the structural positions of XML element/attribute nodes in the XML tree while taking into account the semantic similarities between node labels and/or values. In this context, developing semantic-aware applications usually requires three main steps:

a. *XML semantic disambiguation*: an initial pre-processing step to identify the intended meanings of node labels and/or values,
b. *XML similarity evaluation*: comparing XML trees w.r.t. the meanings of node labels/values identified in the initial step,
c. *Semantic-aware processing*: an application specific step, where semantic-aware processing is undertaken based on XML semantic similarity evaluation.

Accordingly, in this section, we present an overview of such applications which we gradually look at from different angles, starting from i) the layer of abstraction at which XML similarity is evaluated (Sections 6.1), and ii) the kind of XML information being assessed (Section 6.2), and then describing high-end application domains covering: iii) Information Retrieval (Section 6.3), iii) Image and Multimedia Retrieval (Section 6.4), iv) Web and Mobile Services (Section 6.5), and v) the (Social) Semantic Web (Section 6.6).

6.1. Following the Similarity Abstraction Layer

Following the XML similarity abstraction layer, three groups of semantic-aware approaches emerge, targeting: i) the data layer, i) the type layer, and ii) in-between the data and type layers [16].

Similarity at XML data layer, i.e., performing XML document/document comparison, is relevant in a variety of applications (cf. detailed reviews in) [3, 191], such as data *versioning*, *monitoring*, and *temporal querying*: a user may want to view or access a version of a particular document (e.g., an XHTML Web site, a Web Service SOAP description, an RSS feed, or an MPEG-7 video description, etc.) which was available during a certain period of time, or may want to view the results of a continuous query, or monitor the evolution of a certain document in time. All these tasks require sophisticated and semantic-aware *version control* capabilities, i.e., maintaining, describing, and learning about changes in the data, while taking into account the semantic meaning of the data captured in a pre-processing disambiguation step. Such tasks can be implemented using semantic-aware *tree edit distance* similarity measures (a.k.a. *differencing* measures) which produce, along with the similarity score, an *edit script* (a.k.a. *diff*) consisting of a set of *edit operations* describing semantic changes to the disambiguated data (e.g., inserting/deleting/updating semantically related nodes, to transform one XML document tree into another) [188, 189, 193]. Another major application is *document clustering*, i.e., grouping XML documents together, based on their structural and semantic similarities, which can improve *data storage indexing* [153, 165] and thus positively affect the *data retrieval* process [3, 182]. Afterall, semantically similar documents/elements would likely satisfy or not a given query, and thus would be easier to retrieve when stored together [101]. Also, clustering is essential is *information extraction*, *wrapping*, and *summarization*, allowing to automatically identify the sets of semantically similar XML elements to be extracted from documents in order to be reformulated (e.g., substituting disambiguated node labels with semantically related ones), restructured, or summarized to make it more processible in enterprise applications (e.g., adapting/simplifying the semantic content of a Web page, blog, RSS feed, or Web Service description for select users/non-experts) [78, 182].

Similarity at XML type layer, i.e., performing XML grammar/grammar comparison, is also useful for many tasks (cf. reviews in) [46, 168], namely *data integration*, which consists in: i) comparing (matching) grammars to identify semantically related elements [192], and then ii) merging the matched elements under a coherent grammar or semantic view [178]. Here, a disambiguation step is necessary to capture the semantic meaning of grammar elements prior to performing grammar matching. Data integration allows the user to efficiently access and acquire more complete information (e.g., accessing similar Websites, blogs, or RSS feeds simultaneously) [180, 181]. It is also essential in performing *data warehousing*[7], where XML information is transformed from different data-sources complying with different grammars into data conforming with grammars defined in the data warehouse [47]. Other applications include *message translation* in Business-to-Business (B2B) integration

[1] http://www.inex.otago.ac.nz/data/documentcollection.asp
[2] http://www.ncbi.nlm.nih.gov/entrez/
[3] http://www.reuters.com/tools/rss
[4] http://metalab.unc.edu/bosak/xml/eg/shaks200.zip
[5] http://www-connex.lip6.fr/~denoyer/wikipediaXML/
[6] The authors in [115, 116] did not provide the manual mappings used as reference in the evaluation process.
[7] A warehouse is a decision support database that is extracted from a set of data sources (e.g., different databases describing related data) [146].

[26]: reconciling the semantics of XML message grammars used by trading partners in order to translate in-coming and out-going messages accordingly, which is central in E-commerce and B2B applications [91, 92]; and XML *data maintenance* and *schema evolution*: detecting the structural and semantic differences/updates between different versions of a given grammar to consequently revalidate corresponding XML documents [16, 99].

Similarity between XML data and type layers, i.e., performing XML document/grammar comparison, can also benefit from XML disambiguation applied on the documents and grammars being compared, highlighting various applications (cf. review in) [187]. One such application is *XML document classification*, i.e., categorizing XML documents gathered from the Web against a set of reference grammars declared in an XML repository. In this context, evaluating semantic similarity between incoming disambiguated documents on one hand, and reference disambiguated grammars on the other hand (e.g., defined in a data warehouse), allows the identification of entities that are conceptually close, but not syntactically identical, which is common in handling heterogeneous XML repositories, particularly on the Web where users have different backgrounds and no precise definitions about the matter of discourse [16, 111]. Evaluating semantic similarity between documents and grammars can also be exploited in XML document retrieval via *structural queries* [59, 177]: a query being represented as an XML grammar with additional constraints on content; as well as in the *selective dissemination of information:* user profiles being expressed as grammars against which the incoming XML document stream is semantically matched [16, 176].

6.2. Following the Kind of XML Information

Considering the kind of XML information being evaluated, semantic-aware applications can be grouped in two main categories: i) *structure-only*, and ii) *structure-and-content* XML.

XML structure-only applications: Methods in this category compare the structure of XML documents and/or grammars, i.e., they compare the structural positions and ordering of XML element/attribute nodes identified by their labels, while disregarding their values. Processing the semantic meaning of XML tag names allows to improve the performance of structure-only XML applications, namely: *structural clustering* [40, 97, 134] and *classification* [16, 25] of heterogeneous XML documents from different sources (i.e., having different structures and not conforming to the same grammar); XML *structural querying* [16]: searching for documents/elements based on their structural and semantic properties; and XML *grammar integration*: semantic matching and merging of two grammars into one unified view [178, 186].

XML structure-and-content applications: Methods in this category compare the structure and content of XML documents and/or grammars[1], i.e., they compare element/attribute values taking into account their structural positions in the XML documents. Processing the semantic meaning of XML content is central with methods dedicated to XML *versioning* and *monitoring* [32, 37], *data integration* [62, 102], and XML structure-and-content *querying* (*retrieval*) applications [162, 163], where documents tend to have relatively similar structures and semantics (probably conforming to the same or similar grammars [98, 215]). The semantics of XML content could also be exploited in XML *grammar matching* [45], by processing the meanings of disambiguated element values in the document instances corresponding to the grammars being compared, to identify semantically matching elements (Section 6.1).

Note that methods for comparing *content-only* XML process element/attribute values only, while disregarding their structural positions (disregarding element/attribute tag names, and their containment relations). In other words, methods that target *content-only* XML handle XML documents as traditional flat textual files [108], and consequently exploit classic DB, IR, and semantic processing and disambiguation techniques (Section 3) in managing (e.g., searching, clustering, and classifying) the XML data [8, 120].

6.3. Information Retrieval

Information Retrieval (IR) is one of the foremost application domains requiring sophisticated semantic-aware and similarity-based processing, where systems aim at providing the most relevant (similar) documents w.r.t. a user information need expressed as a search query. In this context, a wide range of techniques extending traditional IR systems to handle XML IR have been designed (cf. extended reviews in) [149, 191]. In brief, XML IR systems accept as input: i) a user query: expressed as an XML document [142], an XML fragment [27], an XML structured query (e.g., XPath [199] or XQuery [18]), or as a set of keywords [221], and ii) an indexed XML document repository [108], and produce as output: a ranked list of XML elements (and their sub-trees)[2] selected form the repository, and ordered following their relevance (similarity) w.r.t. the user query [185]. Hence, the quality of an XML IR engine depends on two key issues: i) how documents and queries are represented (indexed), and ii) how these representations are compared (matched) to produce relevant results. In this context, most solutions in the literature have explored syntactic XML indexing paradigms (based on node positions, paths, or structural summaries) integrated in dedicated inverted indexing structures, e.g., [108, 212].

Nonetheless, as XML data on the Web became more prevalent and diverse, element/attribute labels and values became *noisier*, such that syntactic indexing techniques could not keep pace [48]. As a result, non-expert users have been increasingly faced with what is described as the *vocabulary problem* [54]: query keywords chosen by users are often different from those used by the authors of the relevant documents, lowering the systems' *precision* and *recall* rates. To tackle the problem, various approaches have suggested expanding the original query with terms synonymous to XML tag names and/or values, [107, 160]. Yet, query expansion methods remain of limited capabilities since the relationships between keywords chosen by users and those used by authors often extends beyond simple *synonymy* [48], thus highlighting the need for a more aggressive approach: *XML disambiguation*, taking into account all semantic relationships in the disambiguation process.

With XML disambiguation, both the XML query and documents can be processed and represented using semantic concepts, instead of (or in addition to) syntactic keywords and element names/values (e.g., typical XML indexing techniques can be used, except that element names/values would be replaced with semantic concepts). Consequently, query/document matching can be performed in the semantic concept space, instead of performing syntactic keyword/node label matching, thus extending XML IR toward *semantic XML IR*, or so-called *concept-based XML IR*. Prelimi-

[1] In the context of XML similarity evaluation, we underline by *XML grammar content*, the content of the document instances conforming to the grammar. In other words, the content of a given grammar element comes down to the contents of its corresponding instance document elements.

[2] Selecting a whole XML document as a potential answer comes down to selecting its root node (along with the corresponding sub-tree).

nary studies on *semantic XML IR* have shown that representing documents and queries using semantic concepts usually results in a retrieval model that is more effective and less dependent on the specific terms/node labels used [164]. Such a model could yield matches even when the same notion is described by different terms/node labels in the query and target documents, thus increasing the system's *recall*. Similarly, if the correct concepts are chosen for ambiguous terms appearing in the query and in the documents, then non-relevant documents that were retrieved with traditional (syntactic) XML IR could be eliminated from the results, thus increasing the system's *precision* [149].

Semantic XML IR, along with ontological (RDF/OWL[1]) IR, is currently a hot research topic [149, 164]. Aside from the promise of achieving improved quality (*effectiveness*), the time performance (*efficiency*) of: i) XML disambiguation during query/document preprocessing and representation, and then ii) semantic concept comparison during query/document matching, remains a major challenge, which might require dedicated optimization and/or XML parallelization techniques, e.g., [49, 68] (cf. Section 7.9).

6.4. Image and Multimedia Retrieval

A more specialized application area which extends XML IR is XML-based Image (and multimedia data) retrieval, i.e., *XML ImR*. In fact, for the last two decades, image datasets (and other kinds of multimedia data such as videos and audios[2]) have become increasingly available, especially on the Web considered as the largest multimedia database to date [76]. Thus, the need to efficiently index and retrieve images (and multimedia data) is becoming ever-more important. In this context, the large battery of existing ImR methods can be roughly categorized as: i) *text-based*, ii) *content-based*, and iii) *hybrid* methods, where a range of recent hybrid methods utilize *XML-based* IR techniques [42].

On one hand, most existing Web image search engines (such as Google Images[3]) and photo sharing sites (such as Flickr and Picasa[4]) mainly adopt the keyword (*text-based*) querying paradigm, where images are indexed based on their textual descriptions (e.g., tags, annotations, surrounding text, links, etc.), which are then mapped to keyword queries using adapted IR techniques. While *text-based* image search is time efficient, yet it shows various limitations, namely: poor result quality, since the automated engines are guessing image visual contents using indirect textual clues [213], and are thus usually unable to confirm whether the retrieved images actually contain the desired concepts expressed in the user queries [52]. In *content-based* ImR systems, e.g., [35, 106], images are indexed based on their visual content, e.g., color, texture, and shape descriptors, and are then processed via search engines devised to handle and compare low level feature descriptors (e.g., dominant color, color and edge histograms, etc.) [105, 106]. The main problems with this category of methods are: i) computational efficiency (low-level feature indexing and mapping is time consuming), and ii) the so-called *semantic gap*: low-level features are usually unable to capture the high-level semantic meaning in the image [105].

To address some the limitations mentioned above, various hybrid approaches have been developed, integrating both *text-based* and *content-based* image processing capabilities [42, 105]. Most methods in this category target Web images where both low-level and text-based image clues are available such as: i) the Web links of image files (e.g., URLs) which have a clear hierarchical structure including useful information such as image Web categories [105], as well as ii) the Web documents in which images are imbedded (e.g., HTML or XHTML) which encompass textual metadata (e.g., image label, Webpage title, ALT-tag, etc.) [24]. Hence, given the semi-structured nature of Web image annotations, XML-based solutions have been recently introduced, e.g., [82, 201, 204], organizing Web images into an XML document tree hierarchy, and then applying image search and retrieval operations on the obtained XML multimedia tree. The general process consist of three main steps [201]: i) placing images into a hierarchy made of link connectivity and Web document metadata, ii) defining multiple evidence scores based on image ascendants, brothers, and children, evaluated using an XML retrieval system, and then iii) retrieving multimedia fragments from relevant images. Recent methods have extended XML solutions toward MPEG-7 retrieval, e.g. [5, 11], providing a higher level of semantic expressiveness with the use of MPEG-7 constructs[4].

In this context, given that the meaning of an image is rarely self-evident using traditional *text-based* and/or *content-based* descriptions, the semantic analysis and disambiguation of XML-based data becomes central to enrich, with as little human intervention as possible, a collection of raw Web images (or multimedia data) into a searchable semantic-based structure that encodes semantically relevant image contents. This would provide the stepping stone toward full-fledged semantic image processing [156], which could be exploited to improve a range of applications namely: i) (semi-) *automatic image annotation* (using semantic clues and fact deduction to infer semantic annotations [50, 214]), ii) *semantic image clustering* and *classification* (grouping together similar images based on their semantic meaning [73, 213]), and iii) *semantic image retrieval* (finding, ranking, and re-organizing image search results according to their semantic similarity, with respect to an image and/or a keyword query) [106, 172]. In this context, dedicated image (multimedia) ontologies can be utilized, relating low-level features with high-level semantic concepts (e.g., color ontologies where colors are defined using color names – *red*, *blue*, etc. – linked with numerical representations [148, 175]), allowing to generate semantic templates to support high-level semantic ImR solutions integrating *text-based* and *content-based* features (e.g., the retrieval of named events, or of pictures with emotional significance such as "*find pictures of a joyful crowd*", e.g., [30, 227]). The main premise with this family of hybrid techniques is to simulate the visual concept space in terms of lexical concepts as perceived by humans, which remains a major ongoing challenge in ImR [42, 105].

6.5. Web and Mobile Services

Another interesting application area which requires XML semantic disambiguation is the matching, search, and composition of Web Services (WS). WS are software systems designed to support interoperable machine-to-machine interactions over a network (namely the Internet) [34]. An individual WS comes down to a self-contained, modular application that can be described, published, and invoked over the Internet, and executed on the remote system where it is hosted [155]. WS rely on two standard XML schemata: i) WSDL (Web Service Description Language) [34] allowing the definition of XML grammar structures to support the machine-readable

[1] Refer to Section 6.6.

[2] We focus on image retrieval here for clarity of presentation, and since it is a typical (and one of the most widespread) example(s) of multimedia IR [76].

[3] https://images.google.com/

[4] http://picasa.google.com/

[4] MPEG-7 provides standardized *Descriptors* (*D*): representing low-level features (date of creation, author, time, etc.) and high-level features (dominant color, edge histogram, etc.), and *Description Schemes* (*DS*): defining the structure and semantic relationships between descriptors and schemes [81].

description of a service's interface and the operations it supports, and ii) SOAP (Simple Object Access Protocol) [216] for XML-based communications and message exchange among WS end-points. RESTful services have been recently promoted as a simpler alternative to SOAP and WSDL-based WS: communicating over HTTP using HTTP request methods (e.g., *Get*, *Post*, *Put*, etc., instead of exchanging SOAP messages), and using XHTML or free test to describe the services (instead of WSDL) [151]. While WS (XML-based) descriptions are inherently more expressive than RESTful service descriptions (using XHTML or keywords), yet RESTful services can be specifically useful in developing mobile services with reduced processing and bandwidth requirements [207].

Hence, when searching for WS (or RESTful services) achieving specific functions, XML (or XHTML/keyword) based service requests can be issued, to which are then matched and ranked service WSDL (or XHTML/keyword) descriptions, thus identifying those services answering the desired requests. Here, matching and ranking service descriptions requires effective XML semantic analysis and disambiguation techniques, due to service author/user heterogeneity (same as the *vocabulary problem* in XML IR, cf. Section 6.3). The same applies for services discovery, recommendation, and composition: searching and mapping together semantically similar WSDL/SOAP descriptions when processing WS, and performing semantic-aware mapping of XHTML/keyword descriptions when dealing with RESTful and/or mobile services [94, 113, 220].

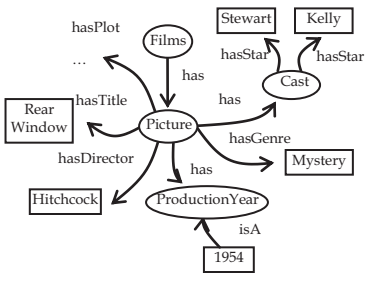
XML similarity and differential encoding can also be used to boost SOAP performance [194, 195]: comparing new SOAP messages with predefined message patterns or WSDL grammars (at the sender/receiver side), processing only those parts of the messages which are different, thus reducing processing cost in SOAP *parsing*, (*de*)*serialization* and *communications* (cf. review in) [196] .

6.6. Semantic Web and Social Semantic Web

Above all, the Semantic Web vision [13] benefits from most of the above-mentioned applications, as it naturally requires XML disambiguation to deal with the semantics of Web documents (encoded in XML-based format), in order to enable and improve interoperability between systems, ontologies, and users. Typically, XML disambiguation can be utilized in *ontology learning*, to build domain taxonomies [88, 206] and enrich/update large-scale semantic networks [139], based on user input data (e.g., Web pages, blogs, image annotations, etc.) encoded in XML. Here, technologies such as RDF (Resource Description Framework) [117, 173]), and OWL (Web Ontology Language) [121] can be used to construct such ontologies.

Subject	Predicate	Object
Films	has	Picture
Picture	hasTitle	"Rear Window"
Picture	hasDirector	"Hitchock"
Picture	has	productionYear
"1954"	isA	productionYear
Picture	hasGenre	"Mystery"
Picture	has	Cast
Cast	hasStar	"Stewart"
Cast	hasStar	"Kelly"
Picture	hasPlot	...

a. Sample RDF triples describing XML document 1 in Fig. 1.



b. RDF graph (Linked Data) representation.

Fig. 12. Sample RDF triples describing XML document 1 from Fig. 1, with the corresponding graph representation.

RDF enables the definition of statements specifying relationships between instances of data in the form of < *subject*, *predicate*, *object* > triples, which designate that a resource (i.e., the subject) has a property (i.e., the predicate) whose value is a resource or a literal (i.e., the object). OWL is built on top of RDF and adds additional expressiveness which depends on the type of *Description Logic* (DL) language applied (OWL allows different levels of semantic expressiveness, ranging from OWL-Lite, to OWL-DL and OWL-Full [57, 121]). Fig. 12.a shows sample RDF triples generated based on XML document 1 in Fig. 1. RDF (OWL) triples can also be represented as a labeled directed graph (cf. Fig. 12.b), and can thus be considered as a kind of semantic network where the concepts (nodes) are related to one another using meaningful relationships (edges).

As a result, RDF and OWL highlight the concept of *Linked Data*: the seamless connection of pieces of information and knowledge on the Semantic Web [70], where a given resource (i.e., subject) can be associated with new properties (i.e., objects) via new relationships (i.e., predicates), and where additional statements (i.e., triples) can be easily added to describe resources and properties [57]. Here, XML disambiguation is central: allowing to extract the semantic information form XML data so that it can be utilized or integrated with semantic annotations from: i) reference ontologies, ii) previously annotated (disambiguated) XML documents, or iii) user generated annotations (e.g., social tagging). Practical examples include integrating hotel and airline reservations, order processing, and insurance renewal with social networking information [112]. Also, augmenting Web data (in XML) with semantic annotations (i.e., triples) provides a way of blending traditional information with Linked Data and Semantic Web constructs. A real-world example is the US retailer *Best Buy* who annotates its XHTML pages that describe products (e.g., business record), with RDFa (i.e., RDF annotations) that can be processed programmatically by search engines and classifiers [110].

An emerging trend in this context is the integration of user information (e.g., user annotations, hash-tags, search queries, and selected search results), i.e., so-called *social semantics* [164], to semantically augment Web (XML-based) data. This highlights the concept of the *Social Semantic Web* [78, 164], a Web in which social interactions lead to the creation of collective and collaborative knowledge representations such as Wikipedia, Wikitionary, Yahoo Answers, and Flikr (cf. Section 7.6), providing semantic information based on human contributions and paving the way for various new applications ranging over: i) *blog classification*, e.g., introducing simple and effective methods to semantically classify blogs, determining their main topics, and identifying their semantic connections [167, 170], ii) *social semantic network analysis*, e.g., disambiguating entities in social networks, and identifying semantic relationships between users based on their published materials [12, 154], and iii) *socio-semantic information retrieval*, e.g., taking into account user information to improve/adapt Web data indexing, query formulation, search, result ranking, and result presentation techniques [140, 164] (cf. reviews on Semantic Web and Social Semantic Web applications in [39, 164]).

7. Discussion and Ongoing Challenges

To wrap up, we discuss in this section some of the major challenges facing existing XML disambiguation methods, which were partially covered throughout the paper, and outline some ongoing and future directions. Note that we mainly emphasize XML-based disambiguation challenges here and do not address general WSD challenges [87, 126] (which could also affect XML disambiguation, such as: bootstrapping and active learning [133], knowledge enrichment and integration [17], and domain-oriented WSD [23]) since the latter are out of the scope of this paper.

7.1. Evaluating Semantic Ambiguity

Most existing XML disambiguation methods completely neglect the problem of evaluating the *semantic ambiguity* of an XML node within the tree structure. In other words, to our knowledge, existing methods do not address the issue of *selecting words for disambiguation*, and rather perform semantic disambiguation (and/or semi-automatic feedback) on all XML document nodes. This is inherently time consuming, and might even be needless. For instance, there is no need to disambiguate node labels *"Movies"*, "*Thriller*", and *"Hitchcock"* in the XML documents of Fig. 1 since they have one prevalent meaning each (based on WordNet[1]). Here, it would be more efficient to select the most ambiguous nodes in the XML document tree as target nodes to be processed for disambiguation, before running the disambiguation process on the document as a whole. Hence, a formal mathematical approach for *semantic ambiguity evaluation* is required to quantitatively assess ambiguity, taking into account different XML features such as: i) the number of node label senses: the more senses a node label has, the more ambiguous it is, ii) the node's structural position and depth w.r.t. the root of the document tree: nodes closer to the root of the document tend to be more descriptive of the whole document, i.e., having a broader and more ambiguous meaning, than information further down the XML hierarchy which tends to be more specific [15, 228], and iii) the number of children nodes having distinct labels: a greater number of children nodes provide more hints on the actual meaning of the parent node, thus making it less ambiguous. For instance, in Fig. 13, one can clearly identify the meaning of root node label "*Picture*" (i.e., *"Motion picture"*) in Fig. 13.a, by simply looking at the node's distinct children labels. Yet the meaning of "*Picture*" remains ambiguous in the XML tree of Fig. 13.b (having several children nodes but with identical labels).

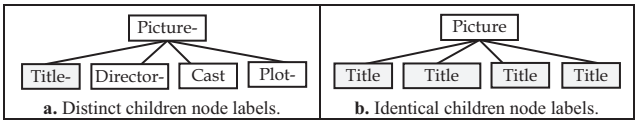


Fig. 13. Sample XML document trees.

7.2. Expanding Structural Context

Most disambiguation methods limit XML context to specific structural features, e.g., parent nodes (*PNC*) [185], root node paths (*RPC*) [182, 183], node sub-trees (*STC*) [200], or nodes reachable through certain crossable edges (*CEC*) [115, 116]. Consequently, most approaches are static in that the size/span of the XML context is predefined (e.g., the parent node, the root path, or the node sub-tree), which makes the context relatively poor for the disambiguation process. For instance, in the document tree *T* of Fig. 2, given XML node *"Cast"* as the target for disambiguation: considering (exclusively) the parent node label (i.e., *"Picture"*), the root node path labels (i.e., *"Films"* and *"Picture"*), or the node sub-tree labels (i.e., *"Star"*) remains insufficient for effective disambiguation. Note that in contrast with the above methods, the *CEC* approach in [115, 116] allows the user to adapt context size by tuning XML edge weights (using dedicated decay functions, cf. Formulas 4 and 5) to identify *crossable edges*. Yet, tunig the proposed weight functions might prove unintuitive: chiefly for non-experts.

Also, an XML document may encompass elements defining hyper-links to other documents, and/or referencing other elements in the same document (e.g., elements of type XLink, or elements which are associated special attributes ID, IDREF and/or IDREFS). Including such links in the XML data model would give rise to a graph rather than a tree. While such links might not be important as far as the structure of the document at hand [134] (i.e., they are usually disregarded in most XML *structure-only* comparison methods and applications such as document clustering and classification, e.g., [40, 72]), yet hyper-links can be important in the use of XML data content, i.e., in *structure-and-content* applications, namely in XML data search and integration [60, 63] (cf. Section 6.2).

In addition, XML documents can represent different kinds of data: both rigorourisly structured (e.g., normalized relational) data, as well as loosely structured and graph-based data (e.g., Web directories, ontological structures). As a result, the notion of structural context can be expanded/fine-tuned w.r.t. the kind of data at hand. For instance, it would be interesting to consider the primary-key/foreign-key (PK/FK) relationships (joins) in defining the context of a node representing a tuple in a relational table, including in its structural context: nodes representing tuples in other tables linked to the latter through the PK/FK join [115]. Also, we might need to disregard certain ontological links (e.g., *IsA*, *PartOf*, *RelatedTo*, etc.) in defining the context of a target node representing an ontological concept, disregarding nodes which might not be useful (or might be noisy) for disambiguation [115].

Hence, expanding/adapting the XML context of a target node to consider nodes connected to the latter via different kinds of relatiohships, e.g., hyper-links, PK/FK joins, or ontological links, in addition to those connected via structural containment relations, would provide more adapted context information, and thus improve disambiguation quality.

7.3. Combining Structure and Content

Most XML disambiguation methods target XML *structure-only* disambiguation, e.g., [116, 182, 183, 200], taking into account XML element/attribute tag names and disregarding data values. Yet, many applications rely on XML *structure-and-content* processing, ranging over document versioning, monitoring, integration, and retrieval (cf. Section 6.2). However, those disambiguation approaches which do handle both XML structure and content, e.g., [184, 185], process content (data values) similarly to structure (element/attribute tag names), which might not always be effective.

On one hand, we believe integrating XML structure and content is beneficiary in resolving ambiguities in both tag names and data values. For instance, it would be beneficial to consider data values *"Stewart"* and *"Kelly"* in disambiguating XML node *"Cast"* in Fig. 2. Likewise the other way around: considering tag name *"Cast"* would help disambiguate data values *"Stewart"* and *"Kelly"*. On the other hand, while XML element/attribute tag names represent *string* values and can be processed using traditional lexico-syntactic disambiguation techniques (cf. Section 4.2), yet corresponding data values could be of different types, e.g., *decimal*, *Boolean*, *date*, *year*, etc. Thus for each data-type, a different method should be utilized to identify semantic meaning. In such a framework, an XML grammar/schema might have to be utilized in the disambiguation process, an XML schema defining element/attribute data types which can be used in disambiguating corresponding element/attribute data values[2]. For instance,

[1] Following WordNet, term "*Movie*" has one single meaning: "*A motion picture*". Similarly, *"Thriller"* has one sense: "*A suspenseful adventure story, play, or movie*", and *"Hitchcock"* has one sense: *"Sir Alfred Hitchcock: English film director noted for his skill in creating suspense"*.

[2] XML Schemas (XSDs) [140] enable a thorough management of data-types (19 different data-types are supported, the user being able to derive new data-types), which is otherwise very restricted in DTDs [20].

knowing that data values *"Stewart"* and *"Kelly"* correspond to data-type *Person_Name*, we can use a dedicated knowledge base such as FOAF [2] to identify corresponding semantic relations, instead of running them through a general purpose knowledge base such as WordNet where they might not link to any semantic concepts.

Handling XML data values also relates to the problems of *Named Entity Recognition* [135] (i.e., NER, identifying named entities in text) and *Named Entity Disambiguation* [66] (i.e., NED, which can be viewed as domain-specific WSD which associates NEs with appropriate references in a dedicated KB, e.g., FOAF [2] or Wikipedia [217]). NER and NED methods could be integrated within XML disambiguation, identifying NEs in data values (e.g., *"Hitchcock"*, *"Stewart"* and *"Kelly"* in document tree *T* of Fig. 2), and then linking them with proper references (e.g., Wikipedia pages describing *Alfred Hitchcock*, *James Stewart*, and *Grace Kelly* respectively). Using Wikis to bridge (the different yet complementary tasks of) WSD, NER, and NED, has been recently identified as the task of *Wikification* [78], and has been receiving increasing attention lately in the domains of collaborative semantic analysis and processing [56, 164] (cf. Section 7.6).

7.4. Extending Syntactic Processing

Most XML disambiguation methods follow the *bag-of-words* paradigm, e.g., [182, 183, 185], such that the XML context is processed as a homogeneous set of words surrounding the term/label (node) to be disambiguated, regardless of XML node structural relationships/proximity. Nonetheless, based on the structured nature of XML, nodes closer together in the XML hierarchy are usually more related than separated nodes, and thus should be considered differently in the disambiguation process. For instance, considering target node label *"Star"* in document tree *T* (Fig. 2), one can realize that XML nodes *"Stewart"* and *"Cast"*, which are closer to the target node, can better influence the latter's disambiguation, whereas nodes farther away such as: *"Picture"*, *"Year"*, *"Genre"*, etc., would have a lesser impact on the target's disambiguation. To our knowledge, only one approach, i.e., *CEC* [115, 116], considers XML node proximity in the disambiguation process, introducing the so-called *relational information model* where the semantic contribution of each context node is weighted following its relative distance from the target node (computed as the sum of the weights of *crossable edges*, evaluated using a dedicated weight function, cf. Formula 4). Yet, the authors do not compare their solution with existing XML disambiguation methods.

Also, recent studies in [9, 79] have highlighted the usefulness of going beyond the simple *bag-of-words* model in describing semi-structured data, including in the vector representation of documents additional features that explicitly model structural information gathered: i) from within the document itself (similar to the *relation information model* in [115, 116]), and ii) from external structured sources (such as mapping document terms with labels from the Wikipedia category hierarchy [9, 79]). Comparing the *bag-of-words* model with expanded structural models in [9, 79] showed significant improvement in document clustering quality and semantic concept matching [127] (cf. Sections 6.1 and 6.2), which highlights the importance of extending syntactic processing when handling XML and semi-structured data.

7.5. Handling Compound XML Tag Names

Most XML disambiguation methods process XML element/attribute tag names as independent textual tokens (terms) similarly to tokens in flat text, while neglecting to handle compound tags. In fact, following the XML data model (cf. Section 2.1), we distinguish three kinds of textual input: i) element/attribute tag names (i.e., structure node labels) consisting of individual terms, ii) element/attribute tag names consisting of compound words, usually made of two individual terms (t_1 and t_2)[1] separated by special delimiters (namely the underscore character, e.g., *"Directed_By"*, or using upper/lower case to distinguish the individual terms, e.g., *"FirstName"*), and iii) element/attribute text values (e.g., content node labels) consisting of sequences of terms separated by the space character. To process these textual input, the WSD task is typically preceded by a linguistic pre-processing phase which performs: i) tokenization[2], ii) stop word removal[3], and iii) stemming[4], applied on the input document's tag names and text values, to produce corresponding XML tree structure and content node labels. While most existing approaches process tag names and values similarly (cf. Section 4.2), we argue that special care should be taken in handling compound tags.

Considering the first case (i.e., tag names made of a single term), no significant pre-processing is required, except for stemming (e.g., tag names *"Films"* and *"Movies"* in the XML documents of Fig. 1 can be respectively stemmed into *"Film"* and *"Movie"*). As for the second case (i.e., tag names made of compound terms t_1 and t_2), most existing methods consider t_1 and t_2 as a sequence made of two separate terms, which are then processed for stop word removal and stemming (similarly to processing element/attribute values). The resulting stemmed terms are represented in separate structure nodes (x_i) labeled with the corresponding terms ($x_i.\ell = t_i$), and are then processed separately for sense disambiguation, each structure node label being associated with the most relevant sense [116, 200].

Yet, we argue that processing compound tag terms as separate entities in the XML tree might not be the best strategy for disambiguation. On one hand, if t_1 and t_2 match a single concept in the reference KB (e.g., synset *first name* in WordNet), they need not be tokenized and can be considered as a single token, kept in a single structure node label, and assigned to a single KB sense. On the other hand, if t_1 and t_2 do not match a single semantic concept (e.g., terms of compound label *LeadActor* in document 2 of Fig. 1.b, which do not match any sense in WordNet), it seems more relevant to keep both terms within a single XML structure node label (ℓ) in order to be treated together for disambiguation, i.e., one sense will be finally associated to ℓ, which can be formed by the best combination of t_1 and t_2's senses. This taps into a related research area which could also benefit from XML disambiguation: *lexicography*, i.e., the creation of new senses, dictionaries, and KBs [126], which is central in performing *ontology learning* on the Semantic Web (e.g., building/enriching domain specific taxonomies to describe Web pages, blogs, image annotations, cf. Section 6.6). While traditional flat text WSD helps provide empirical/statistical sense groupings to generate new senses or adapt the definitions of existing senses, e.g., [86, 87], XML compound tag names could allow a similar functionality with i) less processing (e.g., accessing compound tags in the XML tree is less costly then performing statistical analysis) and with ii) higher expressiveness (taking into account the XML hierarchical relationships and hyper-links connecting nodes to identify new semantic relationships between concepts).

[1] Having more than two terms per XML tag name is unlikely in practice [200].
[2] Decomposing a sentence/sequence of terms, into a set of individual tokens.
[3] Removing tokens which useless for semantic analysis (e.g., "the", "that").
[4] Identifying the word's lexical root or origin (e.g., *"Acting"* becomes *"Act"*).

7.6. Using Collaborative/Social Knowledge Sources

Another central issue which could promote the development of more sophisticated XML disambiguation methods is to go beyond the usage of conventional KBs as reference semantic information sources. In fact, conventional KBs like Roget's thesaurus [225] and WordNet [51] (cf. Section 2.2) are extremely powerful tools and have helped achieve quality WSD and XML disambiguation methods [126], mainly since they are i) *manually-built* (word senses and relationships have been vetted by experts), and ii) *fully-structured* (which is easier to process by machines than unstructured corpora). Yet, conventional KBs also show many limitations [78]: i) the need for significant *human effort* to create and maintain, ii) the need for *wide coverage* which is difficult to achieve by means of manual input from experts, and iii) the need to *update information* in the KB in a timely manner which cannot always be achieved by experts.

To overcome these limitations, there is an increasing need to consider non-conventional sources such as Wikipedia [41], Google [87], Yahoo [179], and Flikr [169], which are *collaboratively built*, where content can be efficiently created and updated by non-experts from different domains, and then the produced content is simply vetted by experts. This requires significantly less effort than manually creating the information from scratch, and allows for a wider coverage and faster update [78]. While, such sources are *semi-structured* in nature, nonetheless information is not fully formalized or disambiguated. For instance, Wikipedia's text corpus is partially structured in pages (articles) and info-boxes[1], with various relationships among pages, namely: *redirection pages*, *internal hyperlinks*, *inter-language links*, as well as *category pages*. Yet, Wikipedia pages also contain vast amounts of unstructured text with no linkage to any pages or other terms (i.e., with no semantic meaning). Thus, dedicated additional processing is required, such as mapping ambiguous/related terms/pages together [41, 83] and identifying missing relationships between terms/pages [75, 217], in order to perform WSD. Also, expanding/integrating semi-structured sources such as Wikipedia, with structured KBs such as WordNet, to fill-in the semantic knowledge gap, has been a promising research direction with successful projects such as Yago [74] and DBPedia [17].

Also, another form of collaboratively built knowledge sources has emerged in the context of the *Social Semantic Web*, known as *folksonomies* [140, 164], as a user-driven approach to annotate Web resources, and which could be particularly useful in semi-structured WSD. A folksonomy is basically a 3-dimensional data structure (usually represented as a 3-uniform graph) whose dimensions are represented by *users*, *tags*, and *resources*, where users assign tags to Web resources, such that tags are freely chosen by the users without a reference KB [140]. Then a post-processing, organization, and mining of the *tags* and their relationships with *users* and *resources* would identify *user vocabularies* describing the resources [12, 170]. Such vocabularies are commonly referred to as *emergent semantics* [140], and represent a (bottom-up) complement to the more formalized (top-down) knowledge organization in conventional KBs [164]. Here, producing high quality semantics requires dedicated additional processing and disambiguation techniques to map the semi-structured and ambiguous tags in the folksonomy with concepts in the KB, allowing to extend the vocabulary and/or the KB with new concepts/relationships highlighting the users' perception of the semantic meaning of information. Then, either the user vocabulary or the extended KB can be used (with corpus-based and/or knowledge-based analysis) to achieve quality WSD.

[1] Tables summarizing important attributes and contents in a Wikipedia page.

Here, note that since XML inherently describes semi-structured data, processes originally developed for XML disambiguation can be utilized to handle collaboratively built semi-structured sources, and vice versa: techniques devised to disambiguate concepts and identify relationships in semi-structured knowledge sources like Wikipedia or folksonomies could inspire the development of new XML disambiguation techniques.

7.7. Using Explicit and Implicit Semantic Analysis

7.7.1. Explicit Semantic Analysis

An original method for using the semi-structured representation of non-conventional KBs (such as Wikipedia's links and category pages) was motivated in [55, 56]. The authors introduce a novel technique titled: *Explicit Semantic Analysis* (ESA) which relies on statistical and distributional analysis of term occurrences in Wikipedia's corpus. The ESA approach attempts to evaluate the semantic relationships between terms in a (flat text) document against a high-dimensional space of concepts, automatically derived from Wikipedia. Here, the semantics of a given term are described by a vector storing the term's association strengths with Wikipedia-derived concepts. A concept is generated from a single Wikipedia article, labeled with the article's title, and is represented as a vector of terms that occur in the article, weighted by their *TF-IDF* scores. Once these concept vectors are generated, an inverted index is created to map back from each term to the concepts it is associated with. Each term appearing in the Wikipedia corpus can be seen as triggering each of the concepts it points to in the inverted index, with the attached weight representing the degree of association between that term and the concept.

For instance, some of the main Wikipedia concepts triggered by the term *"actor"* are (articles titled) *Actor*, *Movie star*, *Acting*, *Film*, and *Academy award*. Even without reading the Wikipedia articles associated with these concepts, it is clear to most readers that these concepts are relevant to the input term. One can also realize that the concepts' labels exhibit a degree of semantic relatedness with the input term that extends simple *synonymy*. As a result, performing semantic analysis and computing word relatedness between words based on their Wikipedia-ESA representation has been shown highly effective in comparison with traditional KB approaches [55].

In this context, we believe that ESA can be adapted/extended toward XML semantic analysis and disambiguation, such that an input XML document is represented as a vector (*term-element* matrix) whose weights measure: the strength of association between terms in XML element tag names/text values on one hand, and ESA concepts extracted from Wikipedia's corpus on the other hand. Note that applying ESA on flat text documents has been shown effective in a wide range of applications, namely computing the *semantic relatedness between text fragments* (i.e., terms or documents) [55], *text categorization* [56], and *semantic information retrieval* [48], and would most likely benefit XML-based applications.

7.7.2. Implicit Semantic Analysis

Another direction to improve XML semantic analysis and disambiguation is to incorporate *implicit semantics* (a.k.a. *latent semantics*) inferred from the statistical analysis of XML element tag names/text values, following the basic idea that: documents which have many node labels in common are semantically closer than ones with fewer node labels in common [164]. *Implicit concepts* are *synthetic* concepts generated by extracting latent relationships between terms in a document (or a document collection), or by calculating probabilities of encountering terms, such that the generated concepts do

not necessarily align with any human-interpretable concept [43, 226]. This is different from conventional concept-based semantic analysis, which utilizes *explicit concepts* representing *real-life* entities/notions defined following human perception (e.g., concepts defined within a conventional/non-conventional KB such as WordNet or Wikipedia) [210]. In this context, the few existing methods toward XML implicit semantics fall into two main categories: i) *Kernel Matrix Learning* (KML), and *Latent Semantic Indexing* (LSI).

On one hand, the KML approach [222, 223] is based on the notion that XML elements might have different contributions to the meaning of the XML document, where an element's contribution depends on its relationship with other elements (rather than only the relative position of the element) in the XML document. To quantify element contribution, the authors utilize supervised and/or unsupervised learning [104] to produce a so-called *kernel matrix*: i.e., an *m*-by-*m* matrix representing the implicit semantic relationships between each pair of *m* elements in the documents being processed. The kernel matrix is then used as reference in comparing and evaluating the similarity between XML documents.

On the other hand, the LSI approach [38, 202] projects an XML document from the original *term-element* document feature space onto a *concept-element* (or so-called *latent semantic*) space, made of *implicit concepts* identified via *Singular Value Decomposition* (SVD). LSI can be viewed as a feature transformation method, taking as input a set of (XML syntactic) features, and producing as output a set of new (*latent semantic*) features. The document is then represented and processed using its latent semantic *concept-element* features. An approach in [222] combines both KML and LSI such that the *kernel matrix* is used to compare document representations in the *concept-element* (latent semantic) space.

While performing XML-based implicit semantic analysis is promising, yet existing KML and LSI methods suffer from various limitations, namely: i) implicit concepts are difficult to understand and evaluate by human users, ii) the number of generated implicit concepts depends on statistical/algebraic analysis rather than the actual meaning of the data, and iii) existing methods are relatively complex and do not scale well [164]. Some of these limitations have been recently addressed, introducing innovative techniques such as distributed LSI [209], probabilistic LSI [44], paralleled probabilistic LSI [85], and optimized LSI [174] to deal with flat documents. These (and other optimization techniques) need to be further explored toward handling XML and semi-structured data.

7.8. Emphasizing User Involvement

Furthermore, existing XML disambiguation methods are mostly static in adopting a fixed context size (e.g., parent node [185], root path [183] or sub-tree [200]) or using preselected semantic similarity measures (e.g., edge-based measure [116], or gloss-based measure [183]), such that user involvement/system adaptability is minimal. Here, a more dynamic approach, allowing the user i) to choose context size, ii) to choose the semantic similarity measure to be used, and iii) to fine-tune the impact/weight of context nodes and semantic similarity measures on the disambiguation process, can help the user optimize the disambiguation process following her needs, taking into account the nature and properties of the XML data being disambiguated. For instance, increasing context size with highly ambiguous or structure-rich XML (i.e., nodes having many siblings/descendents) could increase the chances of including noise (e.g., unrelated/heterogeneous XML nodes) in the disambiguation context and thus disrupt the process. Yet, increasing context size with less ambiguous/poorly structured XML could actually help in creating a large-enough and/or rich-enough context to perform effective disambiguation.

Having a dynamic approach, disambiguation parameters can be fine-tuned: i) manually by human experts, or ii) using automatic or semi-automatic optimization techniques where parameters are chosen to maximize disambiguation quality through some cost function (such as *precision* or *f-measure* [87]). This requires the investigation of optimization techniques that apply linear programming and/or machine learning in order to identify the best weights for a given problem class, e.g., [77, 118]. Providing such a capability, in addition to manual tuning, and the use of semi-automatic feedback techniques (cf. Section 4.2.3), would enable the user to start from a sensible choice of values (e.g., identical weight parameters to consider all disambiguation features equally) and then optimize and adapt the disambiguation process following the scenario and optimization (cost) function at hand.

7.9. Reducing Computational Complexity

Also, one of the main concerns in WSD in general, and in XML disambiguation in particular, remains: high computational complexity (cf. Section 4.2.3.3). WSD has been described as an *AI-complete* problem [114] in comparison with *NP-completeness* in complexity theory, i.e., a problem whose difficulty is equivalent to solving centrals problems in AI (e.g., the Turing Test) [126]. Various adaptations and simplifications of traditional WSD techniques have been proposed in the literature (e.g., the adapted and simplified versions of the Lesk algorithm [100, 205]), some of which have been extended toward XML disambiguation (cf. Section 4.2.3). Yet, despite the various efforts to simplify the algorithmic nature of the process, it has been widely accepted that WSD's main complexity resides in the so-called *knowledge bottleneck* [164], i.e., its heavy reliance on external knowledge which requires substantial time to acquire and process. In fact, without external knowledge, it would be virtually impossible for both humans and machines to identify the meaning of words and expressions [126], since external knowledge serves as a common reference of semantic meaning.

In this context, performing efficient semantic analysis and disambiguation requires more sophisticated knowledge (and semi-structured data) indexing capabilities, e.g., [33, 90], along with more powerful (parallel) processing architectures, e.g., [49, 196].

Note that in light of the increasing growth and evolving nature of collaborative knowledge sources, the *knowledge bottleneck* can be viewed as a variant of the *Big Data* problem, highflying similar issues of *Volume* and *Velocity* (as well as *Variety* and *Veracity*) which are among the hottest in DB, IR, and AI research [67, 218].

7.10. Creating an Experimental Benchmark

Last but not least, a major challenge for future XML and semi-structured disambiguation studies is to develop a comprehensive experimental benchmark: i) implementing existing disambiguation methods to be used in comparative *stand-alone* evaluation, and which can also be embedded and evaluated *end-to-end* within various application scenarios (cf. Section 6), enabling the user to evaluate the effectiveness and efficiency of various algorithms in each application domain, and then choose the one that is most adapted to her needs, ii) implementing dedicated test measures (e.g., *precision*, *recall*, and *coverage*) for evaluating the effectiveness of different methods, iii) providing readily available test data with manual annotations serving as a baseline (gold standard) for testing, and iv) allowing testers to easily append their own algorithms, test measures, and test data in order to dynamically extend the bench-

mark for future empirical evaluations. Providing an experimental benchmark would facilitate future empirical studies and thus foster further research in the area.

8. CONCLUSION

In this survey paper, we have given an overview of current research related to XML-based semi-structured semantic analysis and disambiguation. We have described how different techniques were extending from traditional flat text WSD toward disambiguating semi-structured XML data. We have identified and utilized different criteria to categorize and compare existing XML disambiguation approaches, ranging over: *target node selection* (all nodes, or sample nodes), *context identification* (parent node, root path, subtree, or crossable edges), *context representation* (set-based, or vector-based), *context size* (fixed, or flexible), the *kind of XML data* targeted for disambiguation (structure-only, or structure-and-content), the type of *external information* used (corpus-based, knowledge-based, or collaborative), the *word-sense matching* approach adopted (concept-based, or context-based), the *semantic similarity measure* used (edge-based, node-based, or gloss-based), and the *evaluation method* used for empirical testing (standalone, or embedded within an holistic application). We have also presented and discussed a wide variety of applications requiring XML semantic-aware processing, and concluded by identifying and discussing some of the main challenges and future directions in the field.

Recall that we focus on XML as the present W3C standard for semi-structured data representation on the Web, yet most concepts and methods covered in this paper can be easily adapted or extended to handle alternative or future semi-structured data models (e.g., JSON). We hope that the unified presentation of XML-based semantic disambiguation in this paper will contribute to strengthen further research on the subject.

ACKNOWLEGMENTS

This study is partly funded by the National Council for Scientific Research (CNRS), Lebanon, project: NCSR_00695_01/09/15, and by LAU research fund: SOERC1516R003.

REFERENCES

[1] Agirre, E. *et al.*, *Two Graph-based Algorithms for State of the art Word Sense Disambiguation*. Conference on Empirical Methods in Natural Language Processing (EMNLP), 2006. pp. 585–593.
[2] Aleman-Meza B. *et al.*, *Scalable Semantic Analytics on Social Networks for Addressing the Problem of Conflict of Interest Detection.* ACM Transaction on the Web (TWeb), 2008. 2(1):7.
[3] Algergawy A. *et al.*, *XML Data Clustering: An Overview.* ACM Computing Survey 2011. 43(4):25.
[4] Amitay E. *et al.*, *Multi-Resolution Disambiguation of Term Occurrences.* CIKM, 2003. pp. 255-262.
[5] Angelopoulou A. *et al.*, *Mapping MPEG-7 to CIDOC/CRM.* TPDL Conf., 2011. pp. 40-51.
[6] Arimura H. *et al.*, *Efficient Data Mining from Large Text Databases.* Progress in Discovery Science 2002. pp. 123-139.
[7] Artiles J. *et al.*, *Word Sense Disambiguation based on Term to Term Similarity in a Context Space.* In Senseval-3: Third International Workshop on the Evaluation of Systems, 2004. pp. 58-63.
[8] Baeza-Yates R. and Ribeiro-Neto B., *Modern Information Retrieval.* 1999. Addison-Wesley, MA.
[9] Banerjee S. *et al.*, *Clustering short texts using Wikipedia.* ACM SIGIR Conf., 2007. pp. 787–788.
[10] Banerjee S. and Pedersen T., *Extended Gloss Overlaps as a Measure of Semantic Relatedness.* International Joint Conference on Artificial Intelligence (IJCAI'03), 2003. p. 805-810.
[11] Benitez A.B. *et al.*, *Enabling MPEG-7 structural and semantic descriptions in retrieval applications.* JASIST journal, 2007. 58(9):1377-1380.
[12] Berendt B. *et al.*, *Bridging the Gap - Data Mining and Social Network Analysis for Integrating Semantic Web and Web 2.0.* Elsevier JWS, 2010. 8(2-3): 95-96.
[13] Berners-Lee T. *et al.*, *The Semantic Web.* Scientific American, 2001. 284(5):1:19.
[14] Bertails A. *et al.*, *The Semantic Web: The Internet & Tomorow's Web.* Industrial Realities, 2010.
[15] Bertino E. *et al.*, *Measuring the structural similarity among XML documents and DTDs.* Journal of Intelligent Information Systems (JIIS), 2008. 30(1):55-92.
[16] Bertino E. *et al.*, *A Matching Algorithm for Measuring the Structural Similarity between an XML Documents and a DTD and its Applications.* Elsevier Information Systems, 2004. (29):23-46.
[17] Bizer C. *et al.*, *DBpedia – a crystallization point for the web of data.* Elsevier JWS, 2009. 7:154–165.
[18] Boncz P. A. *et al.*, *MonetDB/XQuery: a fast XQuery processor powered by a relational engine.* International ACM SIGMOD Conference, 2006. pp. 479-490.
[19] Bordag S., *Word Sense Induction: Triplet-based Clustering and Automatic Evaluation.* In Proc. of EACL Conf., 2006. pp. 137–144.
[20] Bray T. *et al.*, *Extensible Markup Language (XML) 1.0 - 5th Edition. W3C recommendation, 26 Novembre 2008.* http://www.w3.org/TR/REC-xml/ [cited July 2015].
[21] Brody S. and Lapata M., *Bayesian word sense induction.* In Proc. of EACL Conf., 2009. pp. 103–111.
[22] Budanitsky A. and Hirst G., *Evaluating WordNet-based Measures of Lexical Semantic Relatedness.* Computational Linguistics, 2006. 32(1): 13-47.
[23] Buitelaar P. *et al.*, *Domain-specific WSD.* WSD: Algs. & Apps., 2006. pp. 275–298, Springer.
[24] Cai D. *et al.*, *Hierarchical Clustering of WWW Image Search Results using Visual, Textual and Link Information.* Proc. of the Inter. ACM Multimedia Conference, 2004. pp. 952-959.
[25] Candillier L. *et al.*, *Transforming XML Trees for Efficient Classification and Clustering.* Proc. of the Workshop of the Initiative for the Evaluation of XML Retrieval (INEX), 2005. pp. 469-480.
[26] Cardoso J. and Bussler C., *Mapping between heterogeneous XML and OWL transaction representations in B2B integration.* Journal of Data & Knowledge Engineering (JDKE), 2011. 70(12): 1046-1069
[27] Carmel D. *et al.*, *An Extension of the Vector Space Model for Querying XML Documents via XML Fragments.* Proc. of the ACM SIGIR Workshop on XML and Information Retrieval, 2002. pp. 14-25.
[28] Champavère J., *From Knowledge Representation to the Semantic Web: An Overview.* pp.14, 2010.
[29] Chan Y. S. *et al.*, *NUS-PT: Exploiting parallel texts for word sense disambiguation in the English all-words tasks.* In Proc. of SemEval, 2007. pp. 253-256.
[30] Chang S. *et al.*, *Semantic visual templates: linking visual features to semantics,* ICIP, 1998, 531–534.
[31] Chawathe S., *Comparing Hierarchical Data in External Memory.* VLDB Conf., 1999. pp. 90-101.
[32] Chawathe S. *et al.*, *Change Detection in Hierarchically Structured Information.* SIGMOD, 1996, 26-37.
[33] Chbeir R. *et al.*, *SemIndex: Semantic-Aware Inverted Index.* ADBIS Conf., 2014. pp. 290-307.
[34] Chinnici R. *et al.*, *Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language, W3C Recomm. 26 June 2007*, [cited June 2015]; Available from: http://www.w3.org/TR/wsdl20/.
[35] Chua T.S. *et al.*, TREC 2003 Video Retrieval and Story Segmentation Task at NUS PRIS, 2003.
[36] Cimiano P. *et al.*, *Towards the Self-Annotating Web.* WWW Conf., 2004. pp. 462-471.
[37] Cobéna G. *et al.*, *Detecting Changes in XML Documents.* ICDE Conf., 2002. pp. 41-52.
[38] Costa G. and Ortale R., *A Latent Semantic Approach to XML Clustering by Content and Structure Based on Non-negative Matrix Factorization.* ICMLA Conf., 2013. pp. 179-184.
[39] d'Aquin M. *et al.*, *Toward a New Generation of Semantic Web Applications.* IEEE Intelligent Systems, 2008. 23(3):20–28.
[40] Dalamagas T. *et al.*, *A Methodology for Clustering XML Documents by Structure*. Information Systems, 2006. 31(3):187-228.
[41] Dandala B. *et al.*, *Sense Clustering using Wikipedia.* RANLP Conf., 2013. pp. 164-171.
[42] Datta R. *et al.*, *Image Retrieval: Ideas, Influences and Trends of the New Age.* ACM Computer Surveys, 2008. 40(2):1-60.
[43] Deerwester S. *et al.*, *Indexing by Latent Semantic Analysis.* JASIST journal, 1990. 41(6):391–407.
[44] Ding C. *et al.*, *On the equivalence between Non-negative Matrix Factorization and Probabilistic Latent Semantic Indexing.* Computational Statistics & Data Analysis, 2008. 52(8): 3913-3927.
[45] Do H. and Rahm E., *COMA: A System for Flexible Combination of Schema Matching Approaches.* VLDB Conf., 2002. pp. 610-621.
[46] Do H. and Rahm E., *Matching Large Schemas: Approaches and Evaluation.* Information Systems, 2007. 32(6): 857-885.
[47] Doan A. *et al.*, *Learning to Match the Schemas of Data Sources: A Multistrategy Approach.* Machine Learning, 2003. 50(3):279-301.
[48] Egozi O. *et al.*, *Concept-Based Information Retrieval Using Explicit Semantic Analysis.* ACM Transactions on Information Systems (TOIS) 2011. 29(2):8.
[49] Fadika Z. *et al.*, *Parallel and Distributed Approach for Processing Large-Scale XML Datasets.* IEEE/ACM GRID Conf., 2009. pp. 105-112.
[50] Fadzli S. A. and Setchi R., *Concept-based indexing of annotated images using semantic DNA.* 25(8):1644-1655, Journal of Engineering Applications of Artificial Intelligence, 2012.
[51] Fellbaum C., *Wordnet: An Electronic Lexical Database.* MIT Press, Cambridge, MA, 1998. 422 p.
[52] Feng H. *et al.*, *A Bootstrapping Framework for Annotating and Retrieving WWW Images.* ACM Multimedia Conf., 2004. pp. 960-967.
[53] Francis W. N. and Kucera H., *Frequency Analysis of English Usage*. Houghton Mifflin, Boston, 1982.
[54] Furnas G. *et al.*, *The vocabulary problem in human-system communication.* Communications of the ACM, 1987. 30(11):964–971.
[55] Gabrilovich E. and Markovitch S., *Computing semantic relatedness using Wikipedia-based explicit semantic analysis.* IJCAI Joint Conf., 2007. pp. 1606–1611.
[56] Gabrilovich E. and Markovitch S., *Wikipedia-based semantic interpretation for Natural Language Processing.* Journal of Artificial Intelligence Research (JAIR) 2009. 34:443–498.
[57] Garcia-Castro R. and Gomez-Perez A., *Interoperability Results for Semantic Web Technologies using OWL as the Interchange Language.* Elsevier JWS, 2010. 8(4):278-291.
[58] Gliozzo A. *et al.*, *Unsupervised domain relevance estimation for word sense disambiguation.* In EMNLP Conf., 2004. pp. 380-387.
[59] Grahne G. & Thomo A., *Approximate Reasoning in Semi-structured Databases.* KRDB; 2001, Vol. 45.
[60] Graupmann J. *et al.*, *The SphereSearch Engine for Unified Ranked Retrieval of Heterogeneous XML and Web Documents.* VLDB Conf., 2005. pp. 529-540.
[61] Guerrini G. *et al.*, *An overview of similarity measures for clustering XML documents*. Web Data Management Practices: Emerging Techniques and Technologies. IDEA Group, 2006.
[62] Guha S. *et al.*, *Approximate XML Joins.* SIGMOD Conf., 2002. pp. 287-298.
[63] Guo L. *et al.*, *XRANK: ranked keyword search over XML documents.* SIGMOD Conf., 2003. pp. 16-27.
[64] Guo Y. *et al.*, *HIT-IR-WSD: A WSD System for English Lexical Sample Task.* SemEval 2007, 165-168.
[65] Guo Y. *et al.*, *A Requirements Driven Framework for Benchmarking Semantic Web Knowledge Base Systems.* IEEE Trans. Knowl. Data Eng. (TKDE), 2007. 19(2): 297-309.
[66] Hachey B. *et al.*, *Evaluating Entity Linking with Wikipedia.* Artificial Intelligence, 2013. 194:130-150.
[67] Hashem I. *et al.*, *The rise of "big data" on cloud computing: Review and open research issues*. Information Systems, 2015. 47:98–115.
[68] Head M.R. and Govindaraju M., *Performance Enhancement with Speculative Execution Based Parallelism for Processing Large-scale XML-based Application Data.* HPDC Symposium, 2009, 21-30.
[69] Hearst M. A., *Untangling Text Data Mining.* ACL Meeting, 1999. Maryland, USA.
[70] Heath T. and Bizer C., *Linked Data: Evolving the Web into a Global Data Space.* Synthesis Lectures on the Semantic Web: Theory and Technology, Morgan & Claypool, 2011.
[71] Heflin J., *Knowledge Representation on the Internet: Achieving Interoperability in a Dynamic, Distributed Environment.* PhD Thesis, University of Maryland, USA., 2000. AAAI/IAAI 2000:1074
[72] Helmer S., *Measuring the Structural Similarity of Semistructured Documents Using Entropy.* VLDB Conf., 2007. pp. 1022-1032.
[73] Hirota M. *et al.*, *A Robust Clustering Method for Missing Metadata in Image Search Results.* Journal of Information Processing (JIP), 2012. 20(3):537-547.
[74] Hoffart J. *et al.*, *YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia.* Artificial Intelligence, 2013. 194: 28-61.
[75] Hoffmann R. *et al.*, *Learning 5000 relational extractors.* ACL'10, 2010. pp. 286–295.
[76] Hong R. *et al.*, *Multimedia encyclopedia construction by mining web knowledge.* Signal Processing 2013. 93(8):2361-2368.
[77] Hopfield J. and Tank D., *Neural Computation of Decisions in Optimization Problems.* Biological Cybernetics, 1985. 52(3):52–141.
[78] Hovy E.H. *et al.*, *Collaboratively built semi-structured content and Artificial Intelligence: The story so far.* Artificial Intelligence, 2013. pp. 194: 2-27.
[79] Hu X. *et al.*, *Exploiting Wikipedia as External Knowledge for Doc. Clustering.* SIGKDD 2009 389-396.
[80] Ide N. and Veronis J., *Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art.* Computational Linguistics, 1998. 24(1):1-40.
[81] International Organization for Standardisation (ISO), *MPEG-7 Overview.* ISO/IEC JTC1/SC29/WG11, Coding for Moving Pictures and Audio, 2004. Martinez J.M., N6828.
[82] Iskandar D. *et al.*, *Social media retrieval using image features and struct. text.* INEX, 2007, 358-372.
[83] Ito M. *et al.*, *Association thesaurus construction methods based on link co-occurrence analysis for Wikipedia.* CIKM Conf., pp. 817–826.
[84] Jiang J. and Conrath D., *Semantic Similarity based on Corpus Statistics and Lexical Taxonomy.* Proc. of CICLing Conf., 1997. pp 19-33.
[85] Jin Y. *et al.*, *P2LSA and P2LSA+: Two Paralleled Probabilistic Latent Semantic Analysis Algorithms Based on the MapReduce Model.* IDEAL Conf. 2011. pp. 385-393.
[86] Kilgarriff A., *Word Senses.* Word Sense Disambiguation: Algorithms and Applications, 2006, 29-46.
[87] Klapaftis I. and Manandhar S., *Evaluating Word Sense Induction and Disamiguation Methods.* Language Resourses and Evaluation, 2013. 47(3):579-605.
[88] Klapaftis I.P. and M. S., *Taxonomy Learning Using Word Sense Induction.* ACL Conf. 2010, 82–90.
[89] Krovetz R. & Croft W., *Lexical Ambiguity and Information Retrieval.* ACM TOIS, 1992. 10(2):115-141.
[90] Kumar S. *et al., Ontology based Semantic Indexing Approach for Information Retrieval System.* International Journal of Computer Applications (IJCA), 2012. Volume 49– No.12.
[91] Lampathaki F. *et al.*, *Business to business interoperability: A current review of XML data integration standards.* Computer Standards & Interfaces, 2009. 31(6):1045-1055.
[92] Lau R., *Towards a Web Services and Intelligent Agents-based Negotiation System for B2B ECommerce.* Electronic Commerce Research and Applications archive, 2007. 6(3):260-273
[93] Leacock C. and Chodorow M., *Combining Local Context and WordNet Similarity for Word Sense Identification.* The MIT Press, Cambridge, 1998. pp. 265-283.
[94] Lecue F. and Mehandjiev N., *Seeking Quality of Web Service Composition in a Semantic Dimension.* IEEE Trans. on Knowledge and Data Engineering (TKDE), 2011. pp. 942-959.
[95] Lee J.; Kim M.; and Lee Y., *Information Retrieval Based on Conceptual Distance in IS-A Hierarchies.* Journal of Documentation, 1993. 49(2):188-207.
[96] Lee J.W.; Lee K. and Kim W., *Preparations for Semantic-based XML Mining.* ICDM, 2001, 345-352.

[97] Lee M. *et al., XClust: Clustering XML Schemas for Effective Integration.* CIKM, 2002. pp. 292-299.
[98] Leitao L. *et al., Structure-Based Inference of XML Similarity for Fuzzy Duplicate Detection.* CIKM Conf., 2007. pp. 293-302.
[99] Leonardi E. *et al., DTD-Diff: A Change Detection Algorithm for DTDs.* DASFAA, 2006. pp. 384-402.
[100] Lesk M., *Automatic Sense Disambiguation using Machine Readable Dictionaries: How to tell a Pine Cone from an Ice Cream Cone.* SIGDOC Conf., 1986.
[101] Lian W. *et al., An Efficient and Scalable Algorithm for Clustering XML Documents by Structure.* IEEE Transactions on Knowledge and Data Engineering (TKDE), 2004. 16(1):82-96.
[102] Liang W. and Yokota H., *LAX: An Efficient Approximate XML Join Based on Clustered Leaf Nodes for XML Data Integration.* BNCOD Conf., 2005. pp. 82-97.
[103] Lin D., *An Information-Theoretic Definition of Similarity.* ICML Conf., 1998. pp. 296-304.
[104] Liu N. *et al., Learning Similarity Measures in the Non-orthogonal Space.* CIKM, 2004. pp. 334-341.
[105] Liu Y. *et al., A Survey of Content-Based Image Retrieval with High-Level Semantics*, Pattern Recognition, 2006. 40(1):262-282.
[106] Long F. *et al., Fundamentals of Content-based Image Retrieval.* Multimedia IR & Mgmt., 2003.
[107] Lu J., *An Introduction to XML Query Processing and Keyword Search.* Springer-Verlag, 2013. pp. 292.
[108] Luk R. *et al., A Survey in Indexing and Searching XML Documents.* JASIST, 2002. 53(6):415-437.
[109] Ma Y. and Chbeir R., *Content and Structure Based Approach for XML Similarity.* CIT, 2005, 136-140.
[110] MacManus R., *How Best Buy is using The Semantic Web.* The New York Times, 2010. July 1st.
[111] Maguitman A. *et al., Algorithmic Detection of Semantic Similarity.* WWW Conf., 2005. pp. 107-116.
[112] Malaika S. and Nicola M., *Data normalization reconsidered: An examination of record keeping in computer systems.* Developer Works, IBM Corporation 2010. pp. 32.
[113] Malki A. *et al., Composing Data Services with Uncertain Semantics.* IEEE TKDE, 2015. pp. 936-949.
[114] Mallery J. C., *Thinking about Foreign Policy: Finding an Appropriate Role for Artificial Intelligence Computers.* Ph.D. dissertation. MIT Political Science Department, Cambridge, MA, 1988.
[115] Mandreoli F. and Martoglia R., *Knowledge-based sense disambiguation (almost) for all structures.* Information Systems, 2011. 36(2): 406-430.
[116] Mandreoli F. *et al., Versatile Structural Disambiguation for Semantic-Aware Applications.* CIKM Conf., 2005. pp. 209-216.
[117] Manola F. and Miller E., *Resource Description Framework (RDF) Primer : Model and Syntax Specification.* W3C Recommendation, 2004. http://www.w3.org/TR/rdf-primer/.
[118] Marie A. and Gal A., *Boosting Schema Matchers.* OTM Conf., 2008. pp. 283 – 300.
[119] Marquez L. *et al., Supervised corpus-based methods for WSD. In Word Sense Disambiguation: Algorithms and Applications.* E. Agirre and P. Edmonds, Eds. Springer, NY, 2006. pp. 167-216.
[120] McGill M., *Introduction to Modern Information Retrieval.* 1983. McGraw-Hill, New York.
[121] McGuinness D.L. and Van Harmelen F., *OWL 2 Web - Ontology Language Document Overview.* W3C Proposed Edited Recommendation, 2012. http://www.w3.org/TR/owl2-overview/.
[122] Mihalcea R., *Knowledge-based Methods for WSD.* WSD: Algs. & Apps., 2006. pp. 107–131, Springer.
[123] Miller G., *WordNet: An On-Line Lexical Database.* International Journal of Lexicography, 1990. 3(4).
[124] Miller G.A. *et al., A Semantic Concordance.* ARPA HLT Workshop, 1993. pp. 303–308.
[125] Navigli R., *A Structural Approach to the Automatic Adjudication of Word Sense Disagreements.* Natural Language Engineering, 2008. 14(4):547–573.
[126] Navigli R., *Word Sense Disambiguation: a Survey.* ACM Computing Surveys, 2009. 41(2):1–69.
[127] Navigli R. and Ponzetto S.P., *BabelNet: the Automatic Construction, Evaluation and Application of a Wide-Coverage Multilingual Semantic Network.* Artificial Intelligence, 2012. 193 (2012) 217–250.
[128] Navigli R. and Velardi P., *Structural Semantic Interconnections: A Knowledge-Based Approach to Word Sense Disambiguation.* IEEE Trans. Pattern Anal. Mach. Intell., 2005. 27(7):1075-1086.
[129] Navigli R. and Crisafulli G., *Inducing Word Senses to Improve Web Search Result Clustering.* EMNLP Conf., 2010. pp. 116–126, MIT, USA.
[130] Navigli R. and Velardi P., *An Analysis of Ontology-based Query Expansion Strategies.* IJCAI Joint Conf., 2003. pp. 42-49.
[131] Nayak R., *Fast and effective clustering of XML data using structural information.* Knowledge and Information Systems (KIS), 2008. 14 (2): 197-215.
[132] Nayak R. and Tran T., *A Progressive Clustering Algorithm to Group The XML Data By Structural and Semantic Similarity.* IJPRAI, 2007. 21(4):723–743.
[133] Ng V. and Cardie C., *Weakly supervised natural language learning without redundant views.* HLT/NAACL Conf., Edmonton, Alta., Canada, 2003. pp. 173-180.
[134] Nierman A. & Jagadish H. V., *Evaluating structural similarity in XML documents.* WebDB, 2002, 61-66.
[135] Nothman J. *et al., Learning Multilingual Named Entity Recognition from Wikipedia,* Artificial Intelligence, 2013. 194:151-175.
[136] Patwardhan S. *et al., Using Measures of Semantic Relatedness for Word Sense Disambiguation.* CICLing Conf., 2003. pp. 241-257.
[137] Patwardhan S. *et al., SenseRelate:TargetWord – A Generalized Framework forWord Sense Disambiguation.* AAAI Conf., Anthony Cohn (Ed.), Vol. 4. AAAI Press 1692-1693. .
[138] Pedersen T., *Unsupervised corpus-based methods for WSD.* In Word Sense Disambiguation: Algorithms and Applications, E. Agirre and P. Edmonds, 2006. pp.133-166, Springer, New York, NY,.
[139] Pennacchiotti M. and Pantel P., *Ontologizing semantic relations.* ACL-COLING, 2006. pp. 793–800.
[140] Peters, I., *Folksonomies. Indexing and Retrieval in Web 2.0.* De Gruyter, 2009. pp. 443.
[141] Peterson D. *et al. W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes.* http://www.w3.org/TR/xmlschema11-2/ [cited July 2015].
[142] Pokorny J. and Rejlek V., *A Matrix model for XML Data.* Chap. DB&IS, 2005. IOS Press, pp. 53-64.
[143] Pradhan S. *et al., Modeling Unrestricted Coreference in OntoNotes.* CoNLL Shared Task, 2011, 1-27.
[144] Pradhan S. *et al., Semeval-2007 task-17: English lexical sample, SRL and all words.* SemEval, 2007. pp. 87-92.
[145] Rada R. *et al., Development and Application of a Metric on Semantic Nets.* IEEE Transactions on Systems, Man, and Cybernetics (SMC), 1989. 19(1):17-30.
[146] Rahm E. and Bernstein P., *A Survey of Approaches to Automatic Schema Matching.* The VLDB Journal, 2001. (10):334-350.
[147] Ray E.T., *Introduction à XML*, ed. O'Reilly. 2001, Paris. p. 327.
[148] Ren J. *et al., A Novel Image Retrieval Based on Representative Colors.* ICIVC Conf., pp. 102-107.
[149] Renteria-Agualimpia W. *et al., Exploring the advances in semantic search engines.* DCAI Symposium, 2010. vol. 79, pp. 613–620.
[150] Resnik P., *Using Information Content to Evaluate Semantic Similarity in a Taxonomy.* IJCAI Joint Conf., 1995. Vol 1, pp. 448-453.
[151] Richardson L. and Ruby S., *RESTful Web Services.* O'Reilly Media, Inc., 2008. pp. 454.
[152] Richardson R. and Smeaton A., *Using WordNet in a Knowledge-based approach to information retrieval.* Proceedings of the BCS-IRSG Colloquium on Information Retrieval, 1995.
[153] Rusu L.I. *et al., Storage Techniques for Multi-versioned XML Documents.* DASFAA, 2008, 538-545.
[154] Saari P. and Eerola T., *Semantic Computing of Moods Based on Tags in Social Media of Music.* . IEEE Trans. on Knowledge and Data Engineering (TKDE), 2014. pp. 2548-2560.
[155] Sahai A. and Machiraju V., *Enabling fo the Ubiquitous e-services Vision on the Internet* Hewlett-Packard Laboratories, HPL-2001-5, 2001.
[156] Salameh K. *et al., SVG-to-RDF Image Semantization.* SISAP Conf., 2014. pp. 214-228.
[157] Salton G. and Mcgill M.J., *Introduction to Modern Information Retrieval,*. 1983. McGraw-Hill, Tokio.
[158] Sanz I. *et al., Approximate Subtree Identification in Heterogeneous XML Documents Collections.* XML Symposium, 2005. pp. 192-206.
[159] Saruladha K. *et al., A Survey of Semantic Similarity Methods for Ontology Based Information Retrieval.* ICMLC Conf., 2010. pp. 297 - 301
[160] Schenkel R. and Theobald M., *Structural Feedback for Keyword-Based XML Retrieval.* ECIR Conf., 2006. pp. 326-337.
[161] Schenkel R. *et al., Semantic Similarity Search on Semistructured Data with the XXL Search Engine* Information Retrieval 2005. (8):521-545.
[162] Schlieder T., *Similarity Search in XML Data Using Cost-based Query Transformations.* WebDB workshop, 2001. pp. 19-24.
[163] Schlieder T. and Meuss H., *Querying and Ranking XML Documents.* JASIST, 2002. 53(6):489-503.
[164] Schoefeggera K. *et al., A survey on socio-semantic information retrieval.* CS Review, 2013. 8:25–46.
[165] Schöning H., *Tamoni – A DBMS Designed for XML.* ICDE Conf., 2001. pp. 149-154.
[166] Shasha D. and Zhang K., *Approximate Tree Pattern Matching.* Pattern Matching in Strings, Trees and Arrays, Oxford University Press, 1995. chapter 14.
[167] Shi C. *et al., Chinese SNS blog classification using semantic similarity.* CASoN, 2013. pp. 1-6.
[168] Shvaiko P. and Euzenat J., *A Survey of Schema-Based Matching Approaches.* JoDS, 2005, 146-171.
[169] Sigurbjorsnsson B. and van Zwol R., *Flickr Tag Recommendation based on Collective Konwledge.* WWW Conf., 2008. pp. 327-336.
[170] Singh A. K. and Joshi R.C., *Semantic Tagging and Classification of Blogs.* ICCCT, 2010. pp. 455-459.
[171] Smadja F. A., *Retrieving Collocations from Text: Xtract.* Comput. Linguistics, 1993. 19(1): 143-177.
[172] Smeulders A. W. M. *et al., Content-based Image Retrieval at the End of the Early Years.* IEEE TPAMI, 2000. 22(12):1349–1380.
[173] Snow R. *et al., Semantic taxonomy induction from heterogenous evidence.* COLING-ACL, 2006.
[174] Song W. and Park S.C., *Genetic algorithm for text clustering based on latent semantic indexing.* Journal of Computers & Mathematics with Applications, 2009. 57(11):1901-1907.
[175] Stanchev P.L. *et al., High Level Color Similarity Retrieval.* IJITA journal, 2003. 10(3):363-369.
[176] Stanford Center for Biomedical Informatics Research. *Protégé Ontology Editor.* [cited June 2015].
[177] Stanoi I. *et al., A framework for the selective dissemination of XML documents based on inferred user profiles.* ICDE Conf., 2003. pp. 531 – 542.
[178] Staworko S. and Chomicky J., *Validity-Sensitive Querying of XML Databases.* EDBT, 2006, 164-17.
[179] Su H. *et al., Identification of Syntactically Similar DTD Elements for Schema Matching.* WAIM Conf., 2001. pp. 145-159.
[180] Surdeanu M. *et al., Learning to rank answers to non-factoid questions from web collections.* Computational Linguistics, 2011. 3:351–383.
[181] Taddesse F.G. *et al., Semantic-based Merging of RSS Items.* WWW Journal, 2010. Vol. 12, Springer.
[182] Taddesse F.G. *et al., Relating RSS News/Items.* ICWE Conf., LNCS, 2009. pp. 44-452.
[183] Tagarelli A. and Greco S., *Semantic Clustering of XML Documents.* ACM TOIS, 2010. 28(1):3.
[184] Tagarelli A. *et al., Word Sense Disambiguation for XML Structure Feature Generation.* ESWC Conf., 2009. LNCS 5554, pp. 143–157.
[185] Taha K. and Elmasri R., XCDSearch: An XML Context-Driven Search Engine. IEEE Trans. Knowl. Data Eng. 22(12): 1781-1796 (2010)
[186] Taha K. and Elmasri R., OOXKSearch: A Search Engine for Answering XML Keyword and Loosely Structured Queries Using OO Techniques. J. Database Manag. 20(3): 18-50 (2009)
[187] Tansalarak N. and Claypool K. T., *QMatch - Using paths to match XML schemas.* Data Knowledge Engineering (DKE), 2007. 60(2):260-282.
[188] Tekli J. *et al., XML Document-Grammar Comparison: Related Problems and Applications.* Central European Journal of Computer Science (CEJCS), 2011. Inaugural Issue, 1(1):117-136.
[189] Tekli J. *et al., Semantic and Structure Based XML Similarity: An Integrated Approach.* COMAD Conf., 2006. pp. 32-43.
[190] Tekli J. *et al., Efficient XML Structural Similarity Detection using Sub-tree Commonalities.* SBBD Conf. and SIGMOD DiSC, 2007. pp. 116-130.
[191] Tekli J. *et al., A Fine-Grained XML Structural Comparison Approac*h. ER Conf., 2007, pp. 582-598
[192] Tekli J. *et al., An Overview of XML Similarity: Background, Current Trends and Future Directions.* Elsevier Computer Science Review, 2009. 3(3):151-173.
[193] Tekli J. *et al., Minimizing User Effort in XML Grammar Matching.* Elsevier Info. Sci. J., 2012. 210:1-40.
[194] Tekli J. *et al., A Novel XML Document Structure Comparison Framework based-on Sub-tree Commonalities and Label Semantics.* Elsevier Journal of Web Semantics (JWS), 2012. 11:14-40.
[195] Tekli J. *et al., Differential SOAP Multicasting.* IEEE Inter. Conf. on Web Services (ICWS) 2011. pp. 1-8.
[196] Tekli J. *et al., SOAP Processing Performance and Enhancement.* IEEE TSC, 2012. 5(3): 387-403.
[197] Terzi E. *et al., Knowledge Representation, Ontologies, & the Semantic Web.* APWeb, 2003, 382-387.
[198] Theobald A. and Weikum G., *Adding Relevance to XML.* WebDB, 2000. pp. 105-124. Dallas, USA.
[199] Theobald M. *et al., TopX: Efficient and Versatile Top-k Query Processing for Semistructured Data.* The VLDB Journal, 2008. 17:81–115.
[200] Theobald M. *et al., Exploiting Structure, Annotation, and Ontological Knowledge for Automatic Classification of XML Data.* ACM WebDB, 2003. pp. 1-6, San Diego, California.
[201] Torjmen M. *et al., XML Multimedia Retrieval: From Relevant Textual Information to Relevant Multimedia Fragments.* INEX: Initiative for the Evaluation of XML Retrieval, 2009.
[202] Tran T. *et al., Utilizing the Structure and Content Information for XML Document Clustering.* INitiative for the Evaluation of XML Retrieval - INEX 2008, 2008. pp. 460-468.
[203] Tratz S. *et al., PNNL: A supervised maximum entropy approach to word sense disambiguation.* SemEval, Prague, Czech Republic, 2007. pp. 264-267.
[204] Tsikrika T. *et al., Structured Document Retrieval, Multimedia Retrieval, and Entity Ranking Using PF/Tijah.* INEX: Initiative for the Evaluation of XML Retrieval, 2008. pp. 273–286.
[205] Vasilescu F. *et al., Evaluating Variants of the Lesk Approach for Disambiguating Words.* Language Resources and Evaluation (LREC'04), 2004. pp. 633-636.
[206] Velardi P. *et al., OntoLearn Reloaded: A Graph-Based Algorithm for Taxonomy Induction.* Computational Linguistics, 2013. 39(3): 665-707.
[207] Verma R. and Srivastava A., *A Novel Web Service Directory Framework for Mobile Environments.* IEEE ICWS Conf., 2014. pp. 614-621.
[208] Veronis J., *Hyperlex: Lexical cartography for information retrieval.* Comput. Speech Lang., 2004. 18(3):223–252.
[209] Vigna S., *Distributed, large-scale latent semantic analysis by index interpolation.* InfoScale, 2008. 18.
[210] Voorhees E., *Using Wordnet to Disambiguate Word Senses for Text Retrieval.* SIGIR, 1993, 171-180.
[211] W3 Consortium. *The Document Object Model*. 2005, Available from: http://www.w3.org/DOM.
[212] Wang H. & Meng X., *On the sequencing of tree structures for XML indexing*, ICDE, 2005. pp. 372-383.
[213] Wang S. *et al., iGroup: Presenting Web Image Search Results in Sem. Clusters.* CHI, 2007, 587-596.
[214] Weinberger K. *et al., Resolving Tag Ambiguity.* MM'08 Conf., 2008. pp. 111-120, Vancouver, Canada.
[215] Weis M. and Naumann F., *Dogmatix Tracks down duplicates in XML.* SIGMOD, 2005. pp. 431-442.
[216] W3C, *SOAP Version 1.2.* W3C Recomm. 2007, http://www.w3.org/TR/soap/ [cited July 2015].
[217] Wu F. and Weld D., *Open information extraction using Wikipedia,* ACL'10, 2010. pp. 118–127.
[218] Wu X. *et al., Data Mining with Big Data.* IEEE TKDE, 2014. 26: 97-107.
[219] Wu Z. and Palmer M., *Verb Semantics and Lexical Selection.* ACL, 1994. pp. 133-138.
[220] Xia X. *et al., Evolving Recommender System for Mobile Apps: A Diversity Measurement Approach.* Smart Computing Review, 2013. 3(3):139-154.
[221] Xiang Y. *et al., A New Indexing Strategy for XML Keyword Search.* FSKD Conf., 2010.
[222] Yang J. *et al., Integrating Element and Term Semantics for Similarity-Based XML Document Clustering.* Web Intelligence, 2005. pp. 222-228.
[223] Yang J. *et al., Learning the Kernel Matrix for XML Document Clustering.* EEE, 2005. pp. 353-358.
[224] Yarowsky D., *Unsupervised word sense disambiguation rivaling supervised methods.* ACL Meeting, 1995. pp. 189-196.
[225] Yaworsky D., *Word-Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora.* COLING, 1992. Vol 2, pp. 454-460. Nantes.
[226] Yi X. and Allan J., *A comparative study of utilizing topic models for information retrieval.* ECIR Conf., 2009. pp. 29–41.
[227] Yuan J. *et al., Mining Compositional Features From GPS and Visual Cues for Event Recognition in Photo Collections.* IEEE Transactions on Multimedia, 2010. 12(7):705-716.
[228] Zhang Z. *et al., Similarity Metric in XML Documents.* KMEM Workshop, 2003.

Joe Tekli is an Assistant Professor in the ECE Department, Lebanese American University (LAU). He holds a PhD in Computer Science from the Univ. of Bourgogne (UB), LE2I CNRS, France (2009), awarded with Highest Honors. He has completed three post-docs: in the Univ. of Milan, Italy (Fall 2009), in the Univ. of Shizuoka, Japan (Spring 2010), and in ICMC, Univ. of Sao Paulo (USP), Brazil (2010-2011). He has also conducted multiple visiting research missions in the University of Pau and Adour Countries (UPPA), France between Fall 2012 and Fall 2015. He was awarded various prestigious fellowships: French Ministry of Education (France), Fondazione Cariplo (Italy), JSPS (Japan), FAPESP (Brazil), and AUF (France). His research covers XML; MM data semantics, data-mining, and information retrieval. He has coordinated and participated in various international and national research projects, including CEDRE (2012-13), STICAmSud (2013-14), and CNRS Lebanon (2016-17). He is a member of IEEE and ACM SIGAPP, and an organizing member of various international conferences such as IEEE ICWS, AIAI, SITIS, ACM MEDES, etc. He has more than 30 publications in various prestigious journals and conferences including: IEEE TSC, WWW J., Elsevier JWS, Elsevier Info. Sciences, Elsevier CS Review, IJWSR, IEEE ICWS, EDBT, ER, WISE, ADBIS, SBBD, COMAD, ICWE, SOFSEM, etc.