

Resolving XML Semantic Ambiguity

Nathalie Charbel, Joe Tekli, Richard Chbeir, Gilbert Tekli

▶ **To cite this version:**

Nathalie Charbel, Joe Tekli, Richard Chbeir, Gilbert Tekli. Resolving XML Semantic Ambiguity. 18th International Conference on Extending Database Technology, EDBT 2015, Mar 2015, Brussels, Belgium. pp.277-288, 10.5441/002/edbt.2015.25 . hal-01909107

HAL Id: hal-01909107

https://univ-pau.hal.science/hal-01909107

Submitted on 3 Apr 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Resolving XML Semantic Ambiguity

Nathalie Charbel
LE2I Lab. UMR-CNRS, University of Bourgogne (UB), 21078 Dijon, France
nathaliecharbel@gmail.com

Joe Tekli
ECE Dept., Lebanese American University (LAU), 36 Byblos, Lebanon
joe.tekli@lau.edu.lb

Richard Chbeir
LIUPPA Lab., University of Pau & Adour Countries (UPPA), 64600 Anglet, France
richard.chbeir@univ-pau.fr

Gilbert Tekli
NOBATEK,
67 Rue de Mirambeau
64600 Anglet, France
gtekli@gmail.com

ABSTRACT
XML semantic-aware processing has become a motivating and important challenge in Web data management, data processing, and information retrieval. While XML data is semi-structured, yet it remains prone to lexical ambiguity, and thus requires dedicated semantic analysis and sense disambiguation processes to assign well-defined meaning to XML elements and attributes. This becomes crucial in an array of applications ranging over semantic-aware query rewriting, semantic document clustering and classification, schema matching, as well as blog analysis and event detection in social networks and tweets. Most existing approaches in this context: i) ignore the problem of identifying ambiguous XML nodes, ii) only partially consider their structural relations/context, iii) use syntactic information in processing XML data regardless of the semantics involved, and iv) are static in adopting fixed disambiguation constraints thus limiting user involvement. In this paper, we provide a new XML Semantic Disambiguation Framework titled *XSDF* designed to address each of the above motivations, taking as input: an XML document and a general purpose semantic network, and then producing as output a semantically augmented XML tree made of unambiguous semantic concepts. Experiments demonstrate the effectiveness of our approach in comparison with alternative methods.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval - Content Analysis and Indexing; I.7.1 [**Document and Text Processing**]: Document and Text Editing – *Document management*; I.7.2 [**Document Preparation**]: Document Preparation – *Markup languages*.

General Terms
Algorithms, Measurement, Performance, Design, Experimentation.

Keywords
XML semantic-aware processing, ambiguity degree, sphere neighborhood, XML context vector, semantic network, semantic disambiguation.

1. INTRODUCTION
In the past decade, there has been extensive research around XML data processing taking advantage of the semi-structured nature of XML documents to improve the quality of Web-based information retrieval and data management applications [28]. The majority of existing approaches use syntactic information in processing XML data, while ignoring the semantics involved [48]. Yet, various studies have highlighted the impact of integrating semantic features in XML-based applications, ranging over semantic-aware query rewriting and expansion [11, 40] (expanding keyword queries by including semantically related terms from XML documents to obtain relevant results), XML document classification and clustering [49, 53] (grouping together documents based on their semantic similarities, rather than performing syntactic-only processing), XML schema matching and integration [13, 55] (considering the semantic meanings and relations between schema elements and data-types), and more recently XML-based semantic blog analysis and event detection in social networks and tweets [2, 7]. Here, a major challenge remains unresolved: *XML semantic disambiguation*, i.e., how to resolve the semantic ambiguities and identify the meanings of terms in XML documents [23], which is central to improving the performance of XML-based applications. The problem is made harder with the volume and diversity of XML data on the Web.

Usually, heterogeneous XML data sources exhibit different ways to annotate similar (or identical) data, where the same real world entity could be described in XML using different structures and/or tagging, depending on the data source at hand (as shown in Figure 1, where two different XML documents describe the same *Hitchcock movie*). The core problem here is lexical ambiguity: a term (e.g., an XML element/attribute tag name or data value) may have multiple meanings (homonymy), it may be implied by other related terms (metonymy), and/or several terms can have the same meaning (synonymy) [23]. For instance (according to a general purpose knowledge base such as WordNet [14]) the term *"Kelly"* in XML document 1 of Figure 1 may refer to *Emmet Kelly: the circus clown*, *Grace Kelly: Princess of Monaco*, or *Gene Kelly: the dancer*. However, looking at its context in the document, a human user can tell that *"Kelly"* here refers to *Grace Kelly*. Yet while seemingly obvious for humans, such semantic ambiguities remain extremely complex to resolve with automated processes.

```
<?xml version= "1.0"?>
<films>
  <picture title= "Rear Window">
    <director> Hitchcock </director>
    <year> 1954 </year>
    <genre> mystery </genre>
    <cast>
      <star> Stewart </star>
      <star> Kelly </star>
    </cast>
    <plot>A wheelchair bound
      photographer spies on his
      neighbors …</plot>
    …
  </picture>
</films>
```

a. Doc 1

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

b. Doc 2

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

In this context, *word sense disambiguation* (WSD), i.e., the computational identification of the meaning of words in *context* [39], could be central to automatically resolve the semantic ambiguities and identify the meanings of XML element/attribute

© 2015, Copyright is with the authors. Published in Proc. 18th International Conference on Extending Database Technology (EDBT), March 23-27, 2015, Brussels, Belgium: ISBN 978-3-89318-067-7, on OpenProceedings.org. Distribution of this paper is permitted under the terms of the Creative Commons license CC-by-nc-nd 4.0

10.5441/002/edbt.2015.25

tag names and data values, in order to effectively process XML documents. While WSD has been widely studied for flat textual data [20, 39], yet, the disambiguation of structured XML data remains largely untouched. The few existing approaches to XML semantic-aware analysis (cf. Section 2) have been directly extended from traditional flat text WSD, and thus show several limitations, motivating this work:

- **Motivation 1:** Completely ignoring the problem of *semantic ambiguity*: most existing approaches perform semantic disambiguation on all XML document nodes (which is time consuming and sometimes needless) rather than only processing those nodes which are most ambiguous,
- **Motivation 2:** Only partially considering the structural relations/context of XML nodes (e.g., solely focusing on parent-node relations [52], or ancestor-descendent relations [50]). For instance, in Figure 1, processing XML node *"cast"* for disambiguation: considering (exclusively) its parent node label (i.e., *"picture"*), its root node path labels (i.e., *"films"* and *"picture"*), or its node sub-tree labels (i.e., *"star"*), remains insufficient for effective disambiguation.
- **Motivation 3:** Making use of syntactic processing techniques such as the *bag-of-words* paradigm [49, 52] (commonly used with flat text) in representing XML data as a plain set of words/nodes, thus neglecting XML structural and/or semantic features as well as compound node labels,
- **Motivation 4:** Existing methods are mostly static in adopting a fixed context size (e.g., parent node [52], or root path [50]) or using preselected semantic similarity measures (e.g., edge-based measure [29], or gloss-based measure [50]), such that user involvement/system adaptability is minimal.

The main goal of our study is to provide an effective method to XML semantic analysis and disambiguation, overcoming the limitations mentioned above. We aim to transform traditional *syntactic XML trees* into *semantic XML trees* (or graphs, when hyperlinks come to play), i.e., XML trees made of concept nodes with explicit semantic meanings. Each concept will represent a unique lexical sense, assigned to one or more XML element/attribute labels and/or data values in the XML document, following the latter's structural context. To do so, we introduce a novel XML Semantic Disambiguation Framework titled *XSDF*, a fully automated solution to semantically augment XML documents using a machine-readable semantic network (e.g., WordNet [14], Roget's thesaurus [60], FOAF [2], etc.), identifying the semantic definitions and relationships among concepts in the underlying XML structure. Different from existing approaches, *XSDF* consists of four main modules for: i) linguistic pre-processing of XML node labels and values to handle compound words (neglected in most existing solutions), ii) selecting ambiguous XML nodes as primary targets for disambiguation using a dedicated *ambiguity degree* measure (unaddressed in existing solutions), iii) representing target nodes as special *sphere neighborhood* vectors considering a comprehensive XML structure context including all XML structural relations within a (user-chosen) range (in contrast with partial context representations using the *bag-of-words* paradigm), and iv) running *sphere neighborhood* vectors through a hybrid disambiguation process, combining two approaches: *concept-based* and *context-based* disambiguation, allowing the user to tune disambiguation parameters following her needs (in contrast with static methods). We have implemented *XSDF* to test and evaluate our approach. Experimental results reflect our approach's effectiveness in comparison with existing solutions.

The remainder of this paper is organized as follows. Section 2 reviews the background and related works. Section 3 develops our XML disambiguation framework. Section 4 presents experimental results. Section 5 concludes the paper with future works.

2. BACKGROUND & RELATED WORKS

2.1 Word Sense Disambiguation

WSD underlines the process of computationally identifying the senses (meanings) of words in context, to discover the author's intended meaning [20]. The general WSD task consists of the following main elements: i) selecting words for disambiguation, ii) identifying and representing word contexts, iii) using reference knowledge sources, and iv) associating senses with words.

Selecting words for disambiguation: two possible methods exist: i) *all-words*, or ii) *lexical-sample*. In all-words WSD, e.g., [10, 44], the system is expected to disambiguate all words in a (flat) textual document. In lexical-sample WSD, e.g., [18, 44], specific target words are selected for disambiguation, which are usually the most ambiguous words, chosen using supervised learning methods trained to identify salient words in phrases [39].

Identifying and representing context: 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 [26]. Thus, the traditional *bag-of-words* paradigm to represent context terms is broadly adopted with flat textual data [20, 39].

Using reference knowledge sources: distinguished as *corpus-based* or *knowledge-based*. The corpus-based approach, e.g., [3, 4, 11], considers previously disambiguated words, and requires supervised learning from sense-tagged corpora (e.g., SemCor [36]) to enable predictions for new words. Knowledge-based methods, e.g., [33, 39, 49], use machine-readable knowledge bases (i.e., ontologies, thesauri and/or taxonomies, e.g., WordNet [14], Roget's thesaurus [60], ODP [28], etc.) providing ready-made sense inventories to be exploited in WSD.

Associating senses with words: categorized as *supervised* or *unsupervised*. Supervised methods, e.g., [31, 39, 57], use machine-learning techniques with *corpus-based* training data provided to a learning algorithm that induces rules to be used for assigning meanings to words. Unsupervised methods, e.g., [29, 43, 48], are usually *knowledge-based* where reference knowledge bases (e.g., WordNet) are processed as *semantic networks* made of concepts representing word senses, and links connecting concepts, representing semantic relations (hyponymy, meronymy, etc., [14, 46], cf. Figure 2). Here, WSD consists in identifying the semantic concept (word sense) in the semantic network that best matches the semantic concepts of terms appearing in the context of the target word, using a measure of *semantic similarity* [9, 42].

Semantic similarity measures in a semantic network: can be classified as *edge-based*, *node-based*, and *gloss-based* [9]. Edge-based methods [25, 59] estimate similarity as the shortest path (in edges, weights, or number of nodes) between concepts being compared. Node-based approaches [27, 45] estimate similarity as the maximum amount of information content concepts share in common, based on the statistical distribution of concept (term) occurrences in a text corpus (e.g., the Brown corpus [15]). Gloss-based methods [5, 6] evaluate word overlap between the glosses of concepts being compared, a gloss underlining the textual

definition describing a concept (e.g., the gloss of the 1st sense of word *Actor* in WordNet is *"A theatrical performer"*, cf. Figure 2).

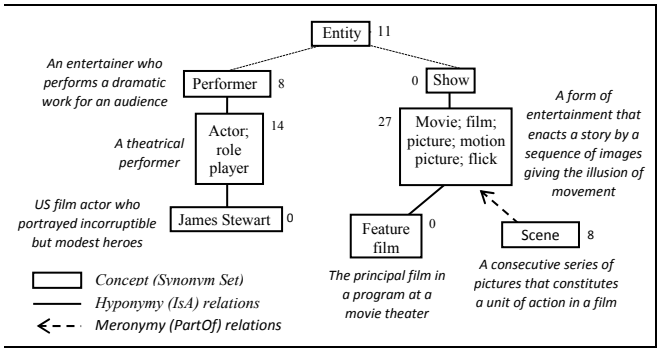


Figure 2. Extract of the (weighted) WordNet semantic network. Numbers next to concepts represent concept frequencies (based on the Brown text corpus [15]). Sentences next to concepts represent concept glosses.

Note that *unsupervised/knowledge-based* WSD has been largely investigated recently (including most methods targeting XML data), in comparison with *supervised* and *corpus-based* methods, which usually require extensive training and large test corpora [39], and thus do not seem practical for the Web. The reader can refer to [20, 39] for reviews on traditional WSD.

2.2 XML Semantic Disambiguation

Few approaches have been developed for semantic disambiguation of XML and semi-structured data. The main challenges reside in the notion of XML (structural) contextualization and how it is processed, as described below.

2.2.1 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 [26], yet there is no unified definition of the context of a node in an XML document tree. Different approaches have been investigated: i) *parent node*, ii) *root path*, iii) *sub-tree*, and iv) *versatile structural context*.

Parent node context: The authors in [51, 52] consider the *parent node* to be the context of an XML data element, and process a parent node and its children as one (canonical) entity, deemed as the simplest semantically meaningful structural entity. The authors utilize context-driven search techniques (structure pruning, identifying relatives, etc.) to determine the relations between canonical trees. These are used to assign semantic node labels using a reference ontology [47], generalizing/specializing node concepts following their labels and positions in the XML tree.

Root path context: In [49, 50], 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). They perform per-path sense disambiguation, comparing every node label in each path with all possible senses of node labels occurring in the same path (using gloss-based and edge-based semantic similarity measures from [6, 59] applied on WordNet) to select the appropriate sense for the node label being processed.

Sub-tree context: The authors in [56] consider the set of XML node labels contained in the sub-tree rooted at a given element node to describe the node's XML context. The authors apply a similar paradigm to identify the contexts of all possible node label senses in WordNet. Consequently, they compare the XML label context to all candidate sense contexts in WordNet, identifying the sense (concept) with the highest context similarity.

Versatile structural context: In [29], 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). They propose various edge-weighting measures (namely a Gaussian decay function) to identify *crossable* edges, such that nodes reachable from a target node through any *crossable* edge belong to the target node's context. Then, they compare the target node label with each candidate sense (concept) corresponding to the labels in the target node's context (using edge-based semantic similarity [24] applied on WordNet) in order to identify the highest matching concept.

2.2.2 XML Context Representation and Processing

Another concern in XML-based WSD is how to effectively process the context of an XML node (once it has been identified) considering the structural positions of XML data. Most existing WSD methods - developed for flat textual data (Section 2.1) and/or XML-based data [49-52] - adopt the *bag-of-words* model where context is processed as a set of words surrounding the term/label (node) to be disambiguated. Hence, all context nodes are treated the same, despite their structural positions in the XML tree. One approach in [29] extends the traditional *bag-or-words* paradigm with additional information considering distance weights separating the context and target nodes: identified as *relational information model* [29]. The authors employ a specially tailored Gaussian decay function estimating edge weights such as the closer a node (following a user-specified direction), the more it influences the target node's disambiguation [29].

2.2.3 Associating Senses with XML Nodes

Once the contexts of XML nodes have been determined, they can be handled in different ways. Two interesting approaches, both *unsupervised* and *knowledge-based*, have been adopted in this context, which we identify as: *concept-based* and *context-based*. On one hand, the concept-based approach adopted in [49, 50] consists in evaluating the semantic similarity between target node senses (concepts) and those of its context nodes, using measures of semantic similarity between concepts in a semantic network, selecting the target sense with maximum similarity. On the other hand, the context-based approach introduced in [56] consists in building context vectors for each target node sense (concept) in the semantic network, and building a context vector for the target node in the XML document tree, and then comparing context vectors to select the target sense with maximum vector similarity.

A hybrid approach in [29] combines variants of the two preceding approaches to disambiguate generic structured data (including XML). Yet while producing quality results, the authors do not compare their solution with XML disambiguation methods.

Wrapping up: we identify four major limitations motivating our work (which have been highlighted in Section 1): most existing methods i) completely ignore the problem of *semantic ambiguity*, ii) only partially consider the structural relations/context of XML nodes (e.g., parent-node [52] or ancestor-descendent relations [50]), ii) neglect XML structural/semantic features by using syntactic processing techniques such as the *bag-of-words* paradigm [49, 52], and iv) are static in choosing a fixed context (e.g., parent node [52], or root path [50]) or preselected semantic similarity measures, thus minimizing user involvement.

3. XML DISAMBIGUATION FRAMEWORK

In order to address all motivations above and provide a more complete and dynamic XML disambiguation approach, we introduce *XSDF* as an unsupervised and knowledge-based solution to resolve semantic ambiguities in XML documents. *XSDF*'s overall architecture is depicted in Figure 3. It is made of four modules: i) linguistic pre-processing, ii) nodes selection for disambiguation, iii) context definition and representation, and iv) XML semantic disambiguation. The system receives as input: i) an XML document tree, ii) a semantic network (noted *SN*), and iii) user parameters (to tune the disambiguation process following her needs), and produces as output a semantic XML tree.

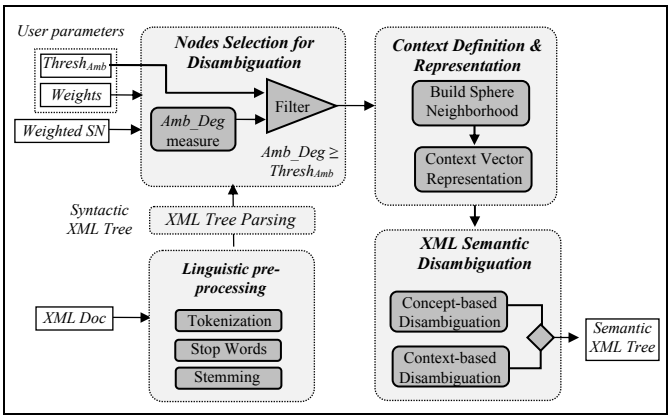


Figure 3. Overall *XSDF* architecture.

We develop *XSDF*'s main modules in the following, starting with the XML and semantic data models adopted in our study.

3.1 XML and Semantic Data Models

XML documents represent hierarchically structured information and can be modeled as *rooted ordered labeled trees* (Figure 4.a and b), based on the Document Object Model (DOM) [58].

Definition 1 - Rooted Ordered Labeled Tree: It is a rooted tree in which the nodes are labeled and ordered. We denote by $T[i]$ the i^{th} node of T in preorder traversal, $T[i].\ell$ its label, $T[i].d$ its depth (in number of edges), and $T[i].f$ its out-degree (i.e., the node's fan-out). $R(T)=[0]$ designates the root node of tree T[1] ●

An XML document can be represented as a *rooted ordered labeled tree* where nodes represent XML elements/attributes, labeled using element/attribute tag names. Element nodes are ordered following their order of appearance in the XML document. Attribute nodes appear as children of their containing element nodes, sorted[2] by attribute name, and appearing before all sub-elements [41, 61]. Element/attribute text values are stemmed and decomposed into tokens (using our linguistic pre-processing component), mapping each token to a leaf node labeled with the respective token, appearing as a child of its container element/attribute node, and ordered following their order of appearance in the element/attribute text value (Figure 4.a).

Note that element/attribute values can be disregarded (*structure-only*) or considered (*structure-and-content*) in the XML document tree following the application scenario at hand. Here, we believe integrating XML structure and content is beneficiary in resolving ambiguities in both element/attribute tag names (structure) and data values (content). For instance, in the document of Figure 1.a, considering data values *"Kelly"* and *"Stewart"* would be beneficial to disambiguate tag label *"cast"*. The same applies the other way: *"cast"* can help disambiguate *"Kelly"* and *"Stewart"*. Also, we provide the formal definition of a semantic network, as the semantic (knowledge base) data model adopted in our study[3].

Definition 2 – Semantic network: It is made of concepts representing groups of words/expressions designating word senses, and links connecting the concepts designating semantic relations, and can be represented as $SN=(C, L, G, E, R, f, g)$:

- C: set of nodes representing concepts in SN (*synsets* as in WordNet [14]),
- L: set of words describing concept labels,
- G: set of glosses describing concept definitions,
- E: set of edges connecting concept nodes, $E \subseteq C \times C$,
- R: set of semantic relations, $R = \{$*Is-A*, *Has-A*, *Part-Of*, *Has-Part*…$\}$, the synonymous words/expressions being integrated in the concepts themselves,
- f: function designating the labels, sets of synonyms, and glosses of concept nodes, f: $C \rightarrow L, L^n, G$ where n designates the number of synonyms per concept,
- g: function designating the labels of edges, g: $E \rightarrow R$.

Note that $c \in SN$ designates a semantic concept with $c.\ell$ its label, $c.syn$ its set of synonymous words, and $c.gloss$ its gloss. We also designate by $\overline{SN}$ a weighted semantic network: augmented with concept frequencies (cf. Figure 2) statistically quantified from a given text corpus (e.g., the Brown corpus [15]) ●

In our current study and tests, we adopt WordNet [14] as a reference semantic network, being a commonly used lexical reference for the English language. Yet, any other knowledge base can be used based on the application scenario, e.g., ODP [28] for describing semantic relations between Web pages, or FOAF [2] to identify relations between persons in social networks.

Note that after disambiguation, target nodes in the XML document tree would consist of semantic concept identifiers extracted from the reference semantic network, where non-target XML nodes remain untouched (cf. Figure 4.b).

3.2 Linguistic Pre-Processing

Linguistic pre-processing consists of three main phases: i) tokenization, ii) stop word removal, and iii) stemming, applied on each of the input XML document's element/attribute tag names and text values, to produce corresponding XML tree node labels. Here, we consider three possible inputs:

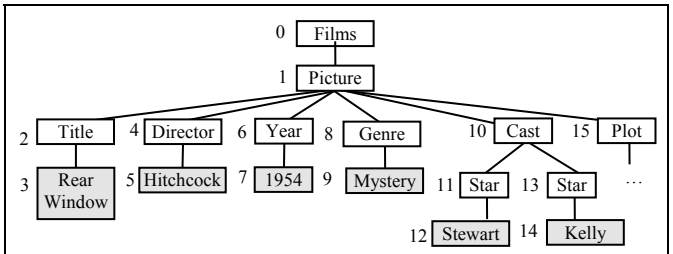
- Element/attribute tag names consisting of individual words,
- Element/attribute tag names consisting of compound words, usually made of two individual terms (t_1 and t_2)[4] separated by special delimiters (namely the underscore character, e.g., "*Directed_By*"), or the use of upper/lower case to distinguish the individual terms (e.g., "*FirstName*"),
- Element/attribute text values consisting of sequences of words separated by the space character.

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

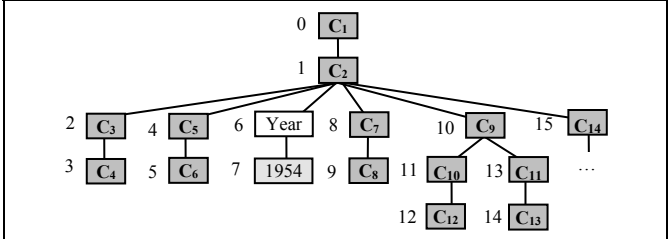
[2] While the order of attributes (unlike elements) is irrelevant in XML [1], yet we adopt an ordered tree model to simplify processing [44, 58].

[3] *Knowledge base* & *semantic network* are used interchangeably hereafter.

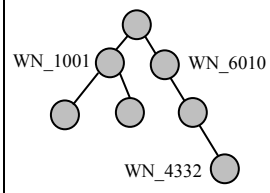
[4] More than two terms per XML tag name is unlikely in practice [59].



a. Input syntactic XML tree (representing Doc 1 in Figure 1).



b. Output semantic XML tree, where nodes 6 and 7 were not targeted for disambiguation.



c. Input semantic network (WordNet), with symbolic concepts & identifiers.

Concept-Doc Index

Concept : NodeID(s)
C1 : 0
C2 : 1
…
C_{14} : 15

Concept-WN Index

Concept : WN_ID(s)
C1 : WN_1001
C2 : WN_1043
…
C_{14} : WN_6010

d. Output inverted concept indexes, generated after disambiguation along with the output semantic tree.

Figure 4. Sample input (syntactic) XML tree and output (semantic) XML trees.

Considering the first case, no significant pre-processing is required, except for stemming (when the word is not found in the reference semantic network). Considering the second case (i.e., compound words, usually disregarded in existing methods), if t_1 and t_2 match a single concept in the semantic network (i.e., a synset in WordNet, e.g. *first name*), they are considered as a single token. Otherwise, they are considered as two separate terms, and are processed for stop word removal and stemming. Yet, they are kept within a single XML node label (ℓ) in order to be treated together afterward, i.e., one sense will be finally associated to ℓ, which is formed by the best combination of t_1 and t_2's senses (in contrast with studies in [29, 56] which process token senses separately as distinct labels). As for the third case, we apply traditional tokenization (i.e., the text value sentence is broken up into a set of word tokens t_i), processed for stop word removal and stemming, and then represented each as an individual node (x_i) labeled with the corresponding token ($x_i.\ell = t_i$), and appearing as children of the containing element/attribute node.

3.3 Node Selection for Disambiguation

Given an input XML tree, the first step is to select target nodes to disambiguate, which (we naturally assume) are the most ambiguous nodes in the document tree. Thus we aim to provide a mathematical definition to quantify an XML node ambiguity degree which can be used to select target nodes for disambiguation (answering *Motivation 1*). To do so, we start by clarifying our assumptions about XML node ambiguity:

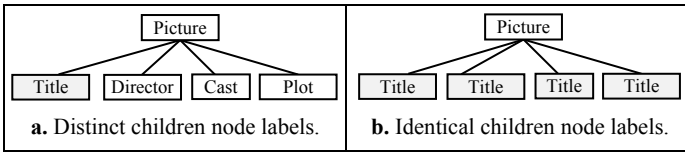
- **Assumption 1:** The semantic ambiguity of an XML node is related to the number of senses of the node's label: i) the more senses it has, the more ambiguous the node is, ii) the less senses it has, the less ambiguous the node is.
- **Assumption 2:** The semantic ambiguity of an XML node is related to its distance from the root node of the document tree: i) the closer it is to the root, the more ambiguous it is, ii) the farther it is from the root, the less ambiguous it is.

Assumption 2 follows the nature of XML and semi-structured data, where nodes closer to the root of the document tend to be more descriptive of the whole document, i.e., having a broader meaning, than information further down the XML hierarchy [8, 61]. In other words, as one descends in the XML tree hierarchy, information becomes increasingly specific, consisting of finer details [54], and thus tends to be less ambiguous.

- **Assumption 3:** The semantic ambiguity of an XML node is related to its number of children nodes having distinct labels: i) the lesser the number of distinct children labels, the more ambiguous the node is, ii) the more the number of distinct children labels, the less ambiguous the node is.

Assumption 3 is highlighted in the sample XML trees in Figure 5. One can clearly identify the meaning of root node label *"Picture"* (i.e., *"motion picture"*) in Figure 5.a., by simply looking at the node's distinct children labels. Yet the meaning of *"Picture"* remains ambiguous in the XML tree of Figure 5.b (having several children nodes but with identical labels). Hence, we believe that distinct children node labels can provide more hints about the meaning of a given XML node, making it less ambiguous.

- **Assumption 4:** An XML node which label has only one possible sense is considered to be unambiguous (i.e., semantic ambiguity is minimal), regardless of its distance from the tree root and its number of distinct children labels.



a. Distinct children node labels. **b.** Identical children node labels.

Figure 5. Sample XML document trees.

While our goal is to quantify XML semantic ambiguity, yet this can be done in many alternative ways that would be consistent with the above assumptions. Hence, we first provide a set of propositions that map to the above assumptions, which we will utilize to derive our ambiguity degree measure.

Proposition 1: The ambiguity degree of an XML node x in tree T increases when the number of senses of $x.\ell$ is high in the reference semantic network SN, or else it decreases such that:

$$\text{Amb}_{Polysemy}(x.\ell, SN) = \frac{senses(x.\ell) - 1}{\text{Max}(senses(SN)) - 1} \in [0,1] \quad \textbf{(1)}$$

where $Max(senses(SN))$ is the maximum number of senses of a word/expression in SN (e.g., in WordNet 2.1 [14], $Max_{polysemy}$ = 33, for the word *"head"*) □

Proposition 2: The ambiguity degree of an XML node x in tree T increases when the distance in number of edges between x and $R(T)$ is low, or else it decreases such that:

$$\mathrm{Amb}_{Depth}(x, T) = 1 - \frac{x.d}{\mathrm{Max}(depth(T))} \in [0,1] \quad \textbf{(2)}$$

where $Max(depth(T))$ is the maximum depth in T □

Proposition 3: The ambiguity degree of an XML node x in tree T increases when the number of children nodes of x having distinct labels, designated as $\overline{x.f}$, is low, or else it decreases:

$$\mathrm{Amb}_{Density}(x, T) = 1 - \frac{\overline{x.f}}{\mathrm{Max}(\overline{fan\text{-}out(T)})} \in [0,1] \quad \textbf{(3)}$$

where $Max(\overline{fan\text{-}out(T)})$ is the maximum number of children nodes with distinct node labels in T. We identify this factor as node density factor to distinguish it from traditional node fan-out: number of children nodes (regardless of label, cf. Definition 1) □

From the above propositions, we can derive a general definition for XML ambiguity degree:

Definition 3 – XML Node Ambiguity Degree: Given an XML tree T, a node $x \in T$, and a reference semantic network SN, we define the ambiguity degree of x, $Amb_Deg(x)$, as the ratio between $Amb_{Polysemy}(x.\ell, SN)$ on one hand, and the sum of $1\text{-}Amb_{Depth}(x, T)$ and $1\text{-}Amb_{Density}(x, T)$ on the other hand:

$$\mathrm{Amb_Deg}(x, T, SN) = \frac{w_{\mathrm{Polysemy}} \times \mathrm{Amb}_{Polysemy}(x.\ell, SN)}{w_{\mathrm{Depth}} \times (1 - \mathrm{Amb}_{Depth}(x, T)) + w_{\mathrm{Density}} \times (1 - \mathrm{Amb}_{Density}(x, T)) + 1} \in [0,1] \quad \textbf{(4)}$$

where $w_{Polysemy}$, w_{Depth}, $w_{Density} \in [0, 1]$ are independent weight parameters allowing the user to fine-tune the contributions of polysemy, depth, and density factors respectively ●

Lemma 1: The ambiguity degree measure Amb_Deg in Definition 3 varies in accordance with Propositions 1-3, and conforms to Assumptions 1-4 □

Proofs of Propositions 1-3 and Lemma 1 have been omitted for space limitations, and can be found in [38].

Special case: When the label of node x consists of a compound word made of tokens t_1 and t_2, we compute $Amb_Deg(x)$ as the average of the ambiguity degrees of t_1 and t_2.

Amb_Deg is computed for all nodes in the input XML tree. Then, only the most ambiguous nodes are selected as targets for disambiguation following an ambiguity threshold $Thresh_{Amb}$ automatically estimated or set by the user, i.e., nodes having $Amb_Deg(x, T, SN) \geq Thresh_{Amb}$, whereas remaining nodes are left untouched. Note that the user can disregard the ambiguity degree measure: i) by setting $w_{Polysemy} = 0$ so that all nodes end up having $Amb_Deg = 0$ regardless of constituent polysemy, depth, and density factors, or ii) by setting $Thresh_{Amb} = 0$ so that all nodes are selected for disambiguation regardless of their ambiguity degrees.

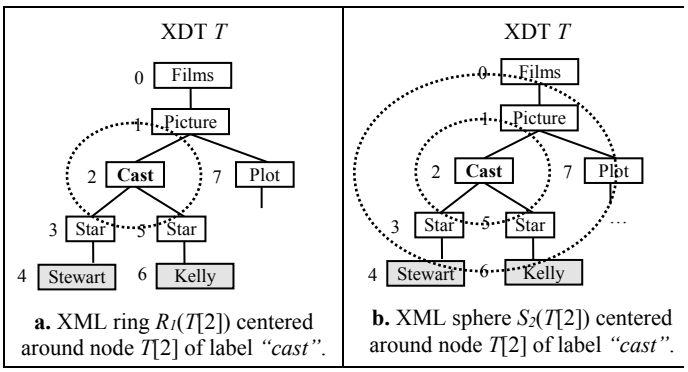
Note that the fine-tuning of parameters is an optimization problem such that parameters should be chosen to maximize disambiguation quality (through some cost function such as *f-measure*, cf. Section 4). This can be solved using a number of known techniques that apply linear programming and/or machine learning in order to identify the best weights for a given problem class, e.g., [19, 30, 37]. Providing such a capability, in addition to manual tuning, would enable the user to start from a sensible choice of values (e.g., identical weight parameters to consider all ambiguity features equally, i.e., $w_{Polysemy} = w_{Depth} = w_{Fan\text{-}out} = 1$, with a minimal threshold $Thresh_{Amb} = 0$ to consider all results initially) and then optimize and adapt the disambiguation process following the scenario and optimization (cost) function at hand. We do not further address the fine-tuning of parameters here since it is out of the scope of this paper (to be addressed in an upcoming study).

3.4 Context Definition and Representation

3.4.1 XML Sphere Neighborhood

For each target node selected from the previous phase, node contexts have to be defined and processed for disambiguation. While current approaches only partly consider the semi-structured nature of XML in defining disambiguation contexts (*Motivation 2*), we introduce the XML *sphere neighborhood* context model, inspired from the sphere-search paradigm in XML IR [17][5], taking into account the whole structural surrounding of an XML target node (including its ancestors, descendants, and siblings) in defining its disambiguation context. We define the notion of XML ring as the set of nodes situated at a specific distance from the target node. An XML sphere would encompass all rings included at distances less or equal to the size (radius) of the sphere.

Definition 4 – XML Ring: Given an XML tree T and a target node $x \in T$, we define an XML ring with center x and radius d as the set of nodes located at distance d from x, i.e., $R_d(x) = \{x_i \in T \mid Dist(x, x_i) = d\}$ ●



a. XML ring $R_1(T[2])$ centered around node $T[2]$ of label *"cast"*.

b. XML sphere $S_2(T[2])$ centered around node $T[2]$ of label *"cast"*.

Figure 6. Sample XML (ring and) sphere neighborhoods.

The distance between two XML nodes in an XML tree, $Dist(x_i, x_j)$, is typically evaluated as the number of edges separating the nodes. For instance, in tree T of Figure 6.a, the distance between nodes $T[2]$ and $T[6]$ of labels "*cast"* and *"Kelly"* respectively is equal to 2. Hence, the XML ring $R_1(T[2])$ centered around node $T[2]$ (*"cast"*) at distance 1 consists of nodes: $T[1]$ (*"Picture"*), $T[3]$ (*"star"*) and $T[5]$ (*"star"*). Note that our approach can be straightforwardly extended to consider different kinds of tree node distance functions (including edge weights, density, or direction, etc. [16, 21]). Yet, we restrict ourselves to the most intuitive notion of node distance here, and report the investigation of other distance functions to a dedicated study.

[5] While comparable with the concept of *XML sphere* exploited in [19], the latter consists of an XML retrieval paradigm for computing TF-IDF scores to rank XML query answers, which is orthogonally different, in its use and objectives, from our disambiguation proposal.

Definition 5 – XML Sphere[6]**:** Given an XML tree T, a target node $x \in T$, and a set of XML rings $R_{d_j}(x) \subseteq T$, we define an XML sphere with center x and radius d as the set of nodes in the rings centered around x at distances less or equal to d, i.e., $S_d(x) = \{x_i \in T \mid x_i \in R_{d_j}(x) \wedge d_j \leq d\}$ ●

In Figure 6.b, the XML sphere $S_2(T[2])$ centered around node $T[2]$ of label *"cast"* with radius 2 consists of: ring $R_1(T[2])$ of radius 1 comprising nodes $T[1]$ (*"picture"*), $T[3]$ (*"star"*) and $T[5]$ (*"star"*), and ring $R_2(T[2])$ of radius 2 comprising nodes $T[0]$ (*"Films"*), T[4] (*"Stewart"*), T[6] (*"Kelly"*), and $T[7]$ (*"Plot"*). The size (radius) of the XML sphere context is tuned following user preferences and/or the nature of the XML data at hand (e.g., XML trees might contain specialized and domain-specific data, and thus would only require small contexts to achieve good disambiguation, whereas more generic XML data might require larger contexts to better describe the intended meaning of node labels and values, cf. experiments in Section 4).

3.4.2 Context Vector Representation

Having identified the context of a given XML target node, we need to evaluate the impact of each of the corresponding context nodes in performing semantic disambiguation (in contrast with existing methods using the *bag-of-words* paradigm where context is processed as a set of words/nodes disregarding XML structure: *Motivation 3*). Here, we introduce a *relational information* solution based on the general vector space model in information retrieval [32] (in comparison with the specific decay function used in [29]), designed to consider the structural proximity/relations among XML nodes in computing disambiguation scores following our sphere neighborhood model. Our mathematical formulation follows two basic assumptions:

- **Assumption 5:** XML context nodes closer to the target node should better influence the latter's disambiguation, whereas those farther away from the target node should have a smaller impact on the disambiguation process.

This is based on the structured nature of XML, such as nodes closer together in the XML hierarchy are typically more related than more separated ones.

- **Assumption 6:** Nodes with identical labels, occurring multiple times in the context of a target node, should better influence the latter's disambiguation in comparison with nodes with identical labels occurring a lesser number of times.

This is based on the notion of context in WSD, where words occurring multiple times in the context of a target word have a higher impact on the target's meaning. Therefore, we represent the context of a target XML node x as a weighed vector, which dimensions correspond to all distinct node labels in its sphere neighborhood context, weighted following their structural distances from the target node.

Definition 6 – XML Context Vector: Given a target node $x \in$ XML tree T, and its sphere neighborhood $S_d(x) \in T$, the corresponding context vector $\overrightarrow{V_d(x)}$ is defined in a space which dimensions represent, each, a single node label $\ell_r \in S_d(x)$, such as $1 < r < n$ where n is the number of distinct node labels in $S_d(x)$. The coordinate of a context vector $\overrightarrow{V_d(x)}$ on dimension ℓ_r, $w_{\overrightarrow{V_d(x)}}(\ell_r)$, stands for the weight of label ℓ_r in sphere $S_d(x)$ ●

Definition 7 – XML Node Label Weight: The weight $w_{\overrightarrow{V_d(x)}}(\ell_r)$ of node label ℓ_r in context vector $\overrightarrow{V_d(x)}$ corresponding to the sphere neighborhood $S_d(x)$ of target node x and radius d, consists of the structural frequency of nodes $x_i \in S_d(x)$ having label $x_i.\ell = \ell_r$. It is composed of a normalized occurrence frequency factor $Freq(\ell_r, S_d(x))$ (based on Assumption 6) defined using a structural proximity factor $Struct(x_i, S_d(x))$ (based on Assumption 5). Formally, given $|S_d(x)|$ the cardinality (in number of nodes) of $S_d(x)$:

$$w_{\overrightarrow{V_d(x)}}(\ell_r) = \frac{\mathrm{Freq}(\ell_r, \mathrm{S_d(x)})}{\mathrm{Max_{Freq}}} = \frac{2 \times \mathrm{Freq}(\ell_r, \mathrm{S_d(x)})}{|\mathrm{S_d(x)}| + 1} \in [0,1] \quad \textbf{(5)}$$

$Freq(\ell_r, S_d(x))$ underlines the total number of occurrences of nodes $x_i \in S_d(x)$ having label $x_i.\ell = \ell_r$, weighted w.r.t. structural proximity, formally:

$$\mathrm{Freq}(\ell_r, \mathrm{S_d(x)}) = \sum_{\substack{x_i \in S_d(x) / \\ x_i.\ell = \ell_r}} \mathrm{Struct}(x_i, \mathrm{S_d(x)}) \in \left[\frac{1}{d+1}, \frac{|\mathrm{S_d(x)}|+1}{2}\right] \quad \textbf{(6)}$$

$Struct(x_i, S_d(x))$ underlines the structural proximity between each context node $x_i \in S_d(x)$ having $x_i.\ell = \ell_r$, and the target (sphere center) node x, formally:

$$\mathrm{Struct}(x_i, \mathrm{S_d(x)}) = 1 - \frac{\mathrm{Dist}(x, x_i)}{d+1} \in \left[\frac{1}{d+1}, 1\right] \quad \textbf{(7)}$$

The denominator in $Struct(x_i, S_d(x))$ is incremented by 1 (i.e., d+1) to allow context nodes occurring in the farthest ring of the sphere context $S_d(x)$, i.e., the ring $R_d(x)$ of radius d, to have a non-null weight in $\overrightarrow{V_d(x)}$, and thus a non-null impact on the disambiguation of target node x ●

For instance, given the XML tree in Figure 6, Figure 7 shows context vectors of sphere neighborhoods $S_1(T[2])$ and $S_2(T[2])$ centered around node $T[2]$ of label *"Cast"*. One can realize that label weights in Figure 7 increase as nodes occur closer to the target node (Assumption 5), and as the number of node label occurrences increases in the sphere context (Assumption 6, e.g., in $\overrightarrow{V_1(T[2])}$, $w_{\overrightarrow{V_1(\text{"}Cast\text{"})}}(\text{"}Star\text{"}) = 2 \times w_{\overrightarrow{V_1(\text{"}Cast\text{"})}}(\text{"}Picture\text{"})$ since node label "*Star*" occurs twice in $S_1(T[2])$ while "*Picture*" occurs once; also in $\overrightarrow{V_2(T[2])}$). Formally:

Lemma 2: The context vector weight measure $w_{\overrightarrow{V_d(x)}}(\ell_r)$ in Definition 7 varies in accordance with Assumptions 5 and 6 □

The lemma's proof is omitted here, and can be found in [38], along with detailed computation examples.

In short, context nodes are weighted based on their labels' occurrences as well as the sizes (radiuses) of the sphere contexts to which they correspond, varying context node weights and thus their impact on the target node's disambiguation accordingly.

[6] The notion of *sphere* here is equivalent to that of a *disk* in 2D space. Yet, we adopt the *sphere* nomination for clearness of presentation.

	Cast	*Picture*	*Star*
$\overrightarrow{V_1(T[2])}$	0.4	0.2	0.4

	Cast	*Picture*	*Star*	*Films*	*Stewart*	*Kelly*	*Plot*
$\overrightarrow{V_2(T[2])}$	0.25	0.1667	0.3334	0.0835	0.0835	0.0835	0.0835

Figure 7. Sample sphere context vectors based on the sphere neighborhoods in Figure 6.

3.5 XML Semantic Disambiguation

Once the contexts of XML nodes have been determined, we process each target node and its context nodes for semantic disambiguation. Here, we propose to combine two strategies: the *concept-based* approach and the *context-based* approach. The former is based on semantic concept comparison between target node senses (concepts) and those of its sphere neighborhood context nodes, whereas the latter is based on context vector comparison between the target node's sphere context vector in the XML tree and context vectors corresponding to each of its senses in the reference semantic network. The user will be able to combine and fine-tune both approaches (answering *Motivation 4*).

3.5.1 Concept-based Semantic Disambiguation

It consists in comparing the target node with its context nodes, using a combination of semantic similarity measures (*edge-based*, *node-based*, and *gloss-based*, cf. Section 2.1) in order to compare corresponding semantic concepts in the reference semantic network. Then, the target node sense with the maximum similarity (relatedness) score, w.r.t. context node senses, is chosen as the proper target node sense. To do so, we propose an extension of context-based WSD techniques (cf. Section 2.2.3) where we:

- Build upon the sphere neighborhood context model, to consider XML structural proximity in evaluating the semantic meanings of context nodes (in comparison with the traditional *bag-of-words* context model),
- Allow an extensible combination of several semantic similarity measures, in order to capture semantic relatedness from different perspectives (in comparison with most existing methods which exploit pre-selected measures).

Definition 8 – Concept-based Semantic Score: Given a target node $x \in$ XML tree T and its sphere neighborhood $S_d(x) \in T$, and given s_p as one possible sense for $x.\ell$ in a (weighted) reference semantic network $\overline{SN}$, we define *Concept_Score*(s_p, $S_d(x)$, $\overline{SN}$) to quantify the semantic impact of s_p as the potential candidate for the intended sense (meaning) of $x.\ell$ within context $S_d(x)$ in T w.r.t. $\overline{SN}$, computed as the average of the weighted maximum similarities between s_p and context node senses:

$$\text{Concept_Score}(s_p, S_d(x), \overline{SN}) = \sum_{x_i \in S_d(x)} \frac{\underset{s_j^i \to x_i.\ell}{\text{Max}} \left(\text{Sim}(s_p, s_j^i, \overline{SN})\right) \times w_{\overline{V_d(x)}}(x_i.\ell)}{|S_d(x)|} \in [0,1] \quad \textbf{(8)}$$

where s_j^i designates the jth sense of context node $x_i.\ell \in S_d(x)$, and $Sim(s_p, s_j^i, \overline{SN})$ designates the semantic similarity measure between senses s_p and s_j^i w.r.t. $\overline{SN}$ ●

Definition 9 – Semantic Similarity Measure: It quantifies the semantic similarity (relatedness) between two concepts (i.e., word senses) c_1 and c_2 in a reference (weighted) semantic network $\overline{SN}$[7], computed as the weighted sum of several semantic similarity measures[8]. Formally:

$$\text{Sim}(c_1, c_2, \overline{SN}) = w_{Edge} \times \text{Sim}_{Edge}(c_1, c_2, SN) + w_{Node} \times \text{Sim}_{Node}(c_1, c_2, \overline{SN})) + w_{Gloss} \times \text{Sim}_{Gloss}(c_1, c_2, SN)) \quad \in [0, 1] \quad \textbf{(9)}$$

where:

- $w_{Edge} + w_{Node} + w_{Gloss} = 1$ and $(w_{Edge}, w_{Node}, w_{Gloss}) \geq 0$
- Sim_{Edge} is a typical edge-based measure from [59],
- Sim_{Node} is a typical node-based measure from [27],
- Sim_{Gloss} is a normalized extension of a typical gloss-based measure from [6] ●

Special case: If the target node label $x.\ell$ is a compound word consisting of two tokens t_1 and t_2 for which no single match was found in the reference semantic network $\overline{SN}$ (cf. Section 3.2), an average score for each possible combination of senses (s_p, s_q) corresponding to each of the individual token senses (s_p for token t_1, and s_q for t_2) is computed to identify the sense combination which is most suitable for the compound target node label:

$$\text{Concept_Score}((s_p, s_q), S_d(x), \overline{SN}) = \sum_{x_i \in S_d(x)} \frac{\underset{s_j^i \to x_i.\ell}{\text{Max}} \left(\text{Sim}((s_p, s_p), s_j^i, \overline{SN})\right) \times w_{\overline{V_d(x)}}(x_i.\ell)}{|S_d(x)|} \in [0,1] \quad \textbf{(10)}$$

where

$$\text{Sim}((s_p, s_p), s_j^i, \overline{SN}) = \frac{\text{Sim}(s_p, s_j^i, \overline{SN}) + \text{Sim}(s_q, s_j^i, \overline{SN})}{2} \in [0,1]$$

Note that a compound context node label $x_i.\ell$ which tokens t_1^i and t_2^i do not match any single concept in $\overline{SN}$, is processed similarly to a compound target node label.

3.5.2 Context-based Semantic Disambiguation

It consists in comparing the target node sphere neighborhood in the XML tree with each of its possible sense (concept) sphere neighborhoods in the reference semantic network. To do so, we adopt the same notions of sphere neighborhood and context vector (Definitions 4-7) defined for XML nodes in an XML tree, to build the sphere neighborhood and context vector of a semantic concept in the semantic network. The only difference here is that sphere rings in the semantic network are built using the different kinds of semantic relations connecting semantic concepts (e.g., hypernyms, hyponyms, meronyms, holonyms, cf. Definition 2), in contrast with sphere rings in an XML tree which are built using XML structural containment relations (Definition 1). Here, given

[7] $\overline{SN}$ designates a semantic network SN weighted with corpus statistics needed to compute a *node*-based similarity measure [29, 48]. Yet, the original non-weighted semantic network SN is sufficient to compute typical *edge*-based and *gloss*-based measures (cf. Section 2.1).

[8] Here, we use three typical semantic similarity measures, yet any other semantic similarity measure can be used, or combined with the latter.

a reference semantic network *SN*, a semantic concept $c \in SN$, and a radius d, we designate by $R_d(c)$, $S_d(c)$, and $\overrightarrow{V_d(c)}$ the ring, sphere, and context vector of radius d corresponding to concept c in *SN* respectively. Note that linguistic pre-processing (cf. Section 3.2) can be applied to concept labels (when needed[9]) before building context vectors and computing vector weights. Formally:

Definition 10 – Context-based Semantic Score: Given a target node $x \in$ XML tree T, its sphere neighborhood $S_d(x) \in T$ and context vector $\overrightarrow{V_d(x)}$, and given s_p as one possible sense for x.ℓ in a reference semantic network *SN*, with its sphere neighborhood $S_d(s_p) \in SN$ and context vector $\overrightarrow{V_d(s_p)}$, we define *Context_Score*(s_p, $S_d(x)$, *SN*) to quantify the semantic impact of s_p as the potential candidate designating the intended sense (meaning) of $x.\ell$ within context $S_d(x)$ in T w.r.t. *SN*, computed using a vector similarity measure between $\overrightarrow{V_d(x)}$ and $\overrightarrow{V_d(s_p)}$:

$$\text{Context_Score}(s_p, S_d(x), SN) = \cos(\overrightarrow{V_d(x)}, \overrightarrow{V_d(s_p)}) \in [0, 1] \quad \textbf{(11)}$$

where *cos* designates the *cosine* vector similarity measure[10] ●

Special case: If the target node label $x.\ell$ is a compound word consisting of tokens t_1 and t_2 for which no single match was found in the reference semantic network *SN*, an integrated score for each possible combination of senses (s_p, s_q) corresponding to each of the individual token senses (s_p for token t_1, and s_q for token t_2) is computed. Here, the sphere neighborhoods and context vectors of individual senses s_p and s_q are combined together to represent the context sphere of the combination of senses (s_p, s_q) in *SN*:

$$\text{Concept_Score}((s_p, s_q), S_d(x), SN) = \cos(\overrightarrow{V_d(x)}, \overrightarrow{V_d(s_p,s_q)}) \in [0,1] \quad \textbf{(12)}$$

where $\overrightarrow{V_d(s_p,s_q)}$ is a compound context vector generated based on compound sphere neighborhood $S_d(s_p, s_q) = S_d(s_p) \bigcup S_d(s_q)$.

3.5.3 Combined Semantic Disambiguation

While *concept-based* and *context-based* disambiguation can be applied separately as described in the above sections, yet we allow the user to combine and fine-tune both approaches (answering *Motivation 4*), producing a combined score as the weighted sum of concept-based and context-based scores:

$$\begin{aligned} &\text{Concept_Score}(s_p, S_d(x), \overline{SN}) = \\ &w_{Concept} \times \text{Concept_Score}(s_p, S_d(x), \overline{SN}) + \\ &w_{Context} \times \text{Context_Score}(s_p, S_d(x), SN) \quad \in [0, 1] \end{aligned} \quad \textbf{(13)}$$

where $w_{Concept} + w_{Context} = 1$ and $(w_{Concept}, w_{Context}) \geq 0$

Note that disambiguation algorithms have been omitted for space limitations. Overall complexity simplifies to the sum of the complexities of *concept-based* and *context-based* disambiguation processes, i.e., $O(|senses(x.\ell)| \times |S_d(x)| \times |senses(x_i.\ell)|)$, and $O(|senses(x.\ell)| \times (|S_d(x)| + |S_d(s_p)|)$ respectively (cf. details in [38]).

[9] This depends on the semantic network (not required with WorldNet).

4. EXPERIMENTAL EVALUATION

We have developed a prototype titled *XSDF*[11] to test and compare our approach with its most recent alternatives. We have evaluated two criteria: i) *semantic ambiguity* and ii) *disambiguation quality*.

4.1 Experimental Test Data

We used a collection of 80 test documents gathered from several data sources having different properties (cf. Table 3), which we describe and organize based on two features: i) *node ambiguity*, and ii) *node structure* (cf. Table 1). The former feature highlights the average amount of ambiguity of XML nodes in the XML tree, estimated using our *ambiguity degree* measure, *Amb_Deg* $\in$ [0, 1]. The latter feature describes the average amount of structural richness of XML nodes, in terms of node *depth*, *fan-out*, and *density* in the XML tree, estimated as the sum of normalized node depth (1-Amb_{Depth}), fan-out, and density (1-$Amb_{Density}$) factors, averaged over all nodes in the XML tree, formally:

$$\text{Struct_Deg}(x, T) = \frac{w_{Depth} \times x.d}{\text{Max}(depth(T))} + \frac{w_{Fan\text{-}out} \times x.f}{\text{Max}(fan\text{-}out(T))} + \frac{w_{Density} \times \overline{x.f}}{\text{Max}(fan\text{-}out(T))} \in [0,1] \quad \textbf{(14)}$$

where $w_{Depth} + w_{Fan\text{-}out} + w_{Density} = 1$ and (w_{Depth}, $w_{Fan\text{-}out}$, $w_{Density}$) $\geq$ 0. In other words, high node depth, fan-out, and/or density here indicate a highly structured XML tree, whereas low node depth, fan-out, and/or density indicate a poorly structured (relatively flat) tree. In our experiments, we set equal weights $w_{Depth} = w_{Fan\text{-}out} = w_{Density} = 1/3$ when measuring *Struct_Deg* (cf. Table 1). In this study, we do not address the issue of assigning weights, which could be performed using optimization techniques (e.g., linear programming and/or machine learning [19, 30, 37]) to help fine-tune input parameters and obtain optimal results (cf. Section 3.3). Such a study would require a thorough analysis of the relative effect of each parameter on disambiguation quality, which we report to a dedicated subsequent study.

Table 1. XML test documents organized based on average node *ambiguity* and *structure*.

	Structure +	**Structure –**
Ambiguity +	**Group 1** *Amb_Deg* = 0.1127 & *Struct_Deg* = 0.6803	**Group 2** *Amb_Deg* = 0.1378 & *Struct_Deg* = 0.6621
Ambiguity –	**Group 3** *Amb_Deg* = 0.0625 & *Struct_Deg* = 0.612	**Group 4** *Amb_Deg* = 0.0447 & *Struct_Deg* = 0.5515

4.2 XML Ambiguity Degree Correlation

We compare XML node ambiguity ratings produced by human users with those produced by our system (i.e., via our *ambiguity degree* measure, *Amb_Deg*, cf. Section 3.3), using *Pearson's correlation coefficient*, $pcc = \delta_{XY}/(\sigma_X+\sigma_Y)$ where: x and y designate user and system generated ambiguity degree ratings respectively, σ_X and σ_Y denote the standard deviations of x and y respectively, and δ_{XY} denotes the covariance between the x and y variables. The values of $pcc \in$ [-1, 1] such that: -1 designates that one of the variables is a decreasing function of the other variable (i.e., values deemed ambiguous by human users are deemed unambiguous by the system, and visa-versa), 1 designates that one of the variables is an increasing function of the other variable

[10] We adopt *cosine* since it is widely used in IR [35]. Yet, other vector similarity measures can be used, e.g., Jaccard, Pearson corr. coeff., etc.

[11] Available online at http://sigappfr.acm.org/Projects/XSDF/

(i.e., values are deemed ambiguous/unambiguous by human users and the system alike), and 0 means that the variables are not correlated. Five test subjects (two master students and three doctoral students, who were not part of the system development team) were involved in the experiment. Manual ambiguity ratings (integers $\in$ [0, 4], i.e., $\in$ [*min*, *max*] ambiguity) where acquired for 12-to-13 randomly pre-selected nodes per document, i.e., a total of 1000 nodes (during an average 10 hours rating time per tester) and then correlated with system ratings, computed with variations of *Amb_Deg*'s parameters to stress the impact of its factors ($Amb_{Polysemy}$, Amb_{Depth}, and $Amb_{Density}$): i) Test #1 considers all three factors equally ($w_{Polysemy} = w_{Depth} = w_{Density} = 1$), ii) Test #2 focuses on the polysemy factor ($w_{Polysemy} = 1$ while $w_{Depth} = w_{Density} = 0$), iii) Test #3 focuses on the depth factor ($w_{Depth} = 1$ while $w_{Polysemy} = 0.2$ and $w_{Density} = 0$), iv) Test #4 focuses on the density factor ($w_{Density} = 1$, $w_{Polysemy} = 0.2$ and $w_{Depth} = 0$).

Results compiled in Table 2 highlight several observations. First, XML ambiguity seems to be perceived and evaluated similarly by human users and our system – obtaining maximum positive correlation between human and *Amb_Deg* scores – when highly ambiguous and highly structured XML nodes are involved (e.g., Group 1). Second, ambiguity seems to be evaluated differently by users and our system when less ambiguous and/or poorly structured XML nodes are involved, attaining: negative or close to null correlation when low ambiguity and/or poorly structured XML nodes are evaluated (e.g., Groups 2, 3, and 4). This might be due to the intuitive understanding of semantic meaning by humans, in comparison with the intricate processing done by our automated system. For instance, in the case of documents of Dataset 9 of Group 4 (conforming to the *personnel.dtd* grammar of the Niagara XML document collection, cf. [38]), the meaning of child node label "*state*" under node label "*address*" was obvious for our human testers (providing an ambiguity score of 0/4). Yet, the interpretation of the meaning of "*state*" is not so obvious for a machine, especially using a rich lexical dictionary such as WordNet where word "*state*" has 8 different meanings. Here, a label considered relatively unambiguous form the user's point of view was assigned a higher ambiguity score by the system based on the expressiveness of the lexical reference.

Concerning *Amb_Deg*'s parameter weight variations (for $w_{Polysemy}$, w_{Depth}, and $w_{Density}$) with tests 2, 3, and 4, all three parameters seem to have comparable impacts on ambiguity evaluation. Note that evaluating XML node ambiguity is not addressed in existing approaches (they do not select target nodes, but simply disambiguate all of them, which can be complex and needless).

4.3 XML Semantic Disambiguation Quality

In addition to evaluating our XML ambiguity degree measure, we ran a series of experiments to evaluate the effectiveness of our XML disambiguation approach. We used the same test datasets described previously. Target XML nodes were first subject to manual disambiguation (12-to-13 nodes were randomly pre-selected per document yielding a total of 1000 target nodes, allowing the same human testers to manually annotate each node by choosing appropriate senses from WordNet, which required an average 22 hours per tester) followed by automatic disambiguation. We then compared user and system generated senses to compute *precision*, *recall* and *f-value* scores.

4.3.1 Testing with Different Configurations

We first tested the effectiveness of our approach considering its different features and possible configurations, considering: i) the properties of XML data (w.r.t. ambiguity and structure), ii) context size (sphere neighborhood radius), and iii) the disambiguation process used (concept-based, context-based, and the combined approach). We only show *f-value* levels in Figure 8 for space limitation (*precision* and *recall* levels follow similar patterns). Several interesting observations can be made here.

Table 2. Correlation between human ratings and system generated ambiguity degrees (cf. graphs in [38]).

		Test #1 *All factors*	**Test #2** *Polysemy*	**Test #3** *Depth*	**Test #4** *Density*
Group 1	Doc 1	0.394	0.411	0.335	**0.439**
Group 2	Doc 2	**0.017**	0.181	0.243	0.139
Group 3	Doc 3	-0.087	-0.139	-0.071	-0.138
	Doc 4	0.408	0.438	0.390	0.398
	Doc 5	-0.184	-0.185	-0.131	-0.235
Group 4	Doc 6	-0.284	-0.291	-0.243	-0.316
	Doc 7	-0.177	-0.190	-0.254	-0.143
	Doc 8	-0.119	-0.025	0.033	-0.156
	Doc 9	-0.452	-0.301	-0.251	**-0.456**
	Doc 10	-0.258	0.180	0.412	0.276

1) Considering ***XML data properties***, one can realize that our approach produced consistent *f-value* levels $\in$ [0.55, 0.69] over all the tested configurations. The highest levels were reached with Dataset 1 of Goup1 having *high ambiguity* and *rich structure*, which resonates with the node ambiguity results discussed in the previous section (highly ambiguous and structurally rich XML nodes seem to be most effectively processed by our approach).

2) Considering ***context size*****,** optimal *f-value* levels are obtained with the smallest sphere neighborhood radius d=1 with Group 1 (*high ambiguity* and *rich structure* XML nodes), whereas optimal levels are obtained with larger contexts having d=3 with Groups 2, 3, and 4 (*low ambiguity* and/or *poor structure*). This is expected since increasing context size with highly ambiguous/structure rich XML would 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.

3) Considering the ***disambiguation process***, one can realize that the *concept-based* approach[12] generally produces higher *f-value* levels in comparison with the *context-based* approach, the latter appearing to be more sensitive to context size. This is expected since the context-based approach primarily depends on the notion of context and context nodes, in both the XML document and semantic network, and thus increasing/decreasing context size would disturb its effectiveness. The effect of context size here could be aggravated when using a rich semantic network (such as WordNet) where a small increase in sphere neighborhood radius could include a huge number of concepts (synsets) in the semantic network context vector, thus adding considerable noise.

To sum up, the above results emphasize the usefulness and need for a flexible approach (such as ours), allowing the user to fine-tune the disambiguation process in order to optimize disambiguation following the nature and properties of the data.

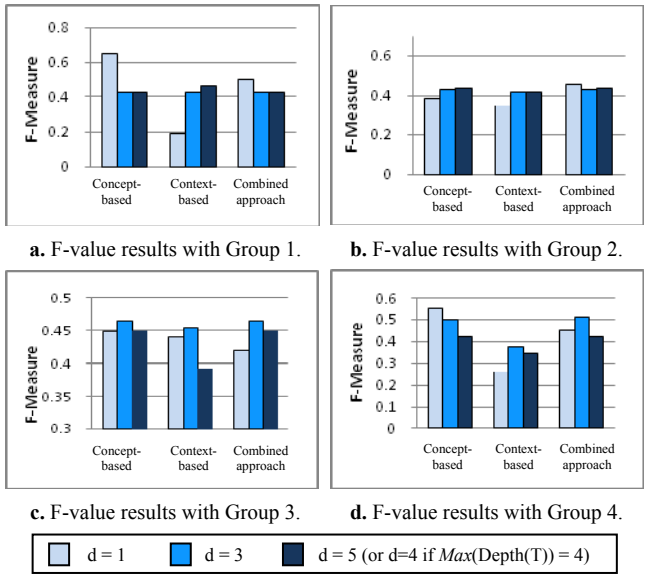
[12] When applying the concept-based approach, semantic similarity measures were considered with identical parameter weights ($w_{Edge} = w_{Node} = w_{Gloss} = 1/3 = 0.3334$), since evaluating the effectiveness of different semantic similarity measures is out of the scope of this paper.

Table 3. Characteristics of test documents.

Groups	Datasets	Source dataset	Grammar	N# of docs	Avg. N# of nodes per doc	Node label polysemy (N# of senses)		Node Depth		Node Fan-out		Node Density	
						Avg.	Max.	Avg.	Max.	Avg.	Max.	Avg.	Max.
Group 1	1	Shakespeare collection[1]	shakespeare.dtd	10	192.054	7.052	30	3.687	6	0.604	20	0.38	6
Group 2	2	Amazon product files[2]	amazon_product.dtd	10	113.333	8.407	72	4.309	7	0.539	13	0.38	6
Group 3	3	SIGMOD Record[3]	ProceedingsPage.dtd	6	39.375	4.615	16	2.743	6	0.692	9	0.692	9
	4	IMDB database[4]	movies.dtd	6	15.475	4	10	2.666	5	1.066	5	1	5
	5	Niagara collection[5]	bib.dtd	8	26.5	4.384	13	2.961	5	0.884	5	0.884	5
Group 4	6	W3Schools[6]	cd_catalog.dtd	4	16.5	3.937	10	2.312	3	0.812	6	0.812	6
	7	W3Schools	food_menu.dtd	4	16	2.375	7	2.437	3	0.562	4	0.562	4
	8	W3Schools	plant_catalog.dtd	4	11.675	3.454	15	2	3	1.181	6	1.181	6
	9	Niagara collection	personnel.dtd	4	19	3.947	9	2.368	5	1.157	4	1.157	4
	10	Niagara collection	club.dtd	4	15.5	4.533	10	2.266	4	1.4	5	1.4	5

Table 4. Comparing our method with existing approaches

Approaches	Considers linguistic pre-processing	Considers tag tokenization (compound terms)	Addresses XML node ambiguity	Integrates an inclusive XML structure context	Flexible w.r.t. context size	Adopts *relational information* approach	Combines the results of various semantic similarity measures	Straightforward mathematical functions	Disambiguates XML structure and content
RPD [50]	√	x	x	x	x	x	x	x	x
VSD [29]	√	√	x	√	√	√	x	x	x
XSDF (our approach)	√	√	√	√	√	√	√	√	√



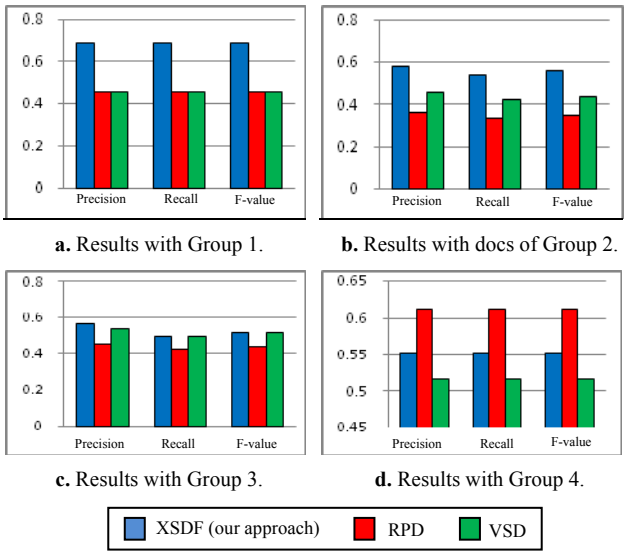
a. F-value results with Group 1. **b.** F-value results with Group 2.

c. F-value results with Group 3. **d.** F-value results with Group 4.

Figure 8. Average *f-value* scores considering different features and configurations of our approach.

4.3.2 Comparative Study

In addition, we evaluated the effectiveness of our approach in comparison with two of its most recent alternatives: *RPD* (Root Path Disambiguation) [50], and *VSD* (Versatile Structure Disambiguation) [29]. A qualitative comparison is shown in Table 4. We ran a battery of tests considering the different features and configurations or our approach. Here, we provide a compiled presentation considering optimal input parameters for our approach[19] (i.e., context size *d*=1 when processing Group 1, *d*=3 when processing Groups 2, 3, 4, using the *concept-based* disambiguation process with all groups) and its alternatives (as indicated in corresponding studies). Results in Figure 9 show that our method yields *precision, recall,* and *f-value* levels higher than those achieved by its predecessors, with almost all test groups except with Group 4 where *RPD* produces better results. In fact, XML nodes in Group 4 are less ambiguous and poorly structured in comparison with remaining test groups. Hence, choosing a simple context made of root path nodes has proven to be less noisy in this case, in comparison with the more comprehensive context models used with our approach and with VSD.



a. Results with Group 1. **b.** Results with docs of Group 2.

c. Results with Group 3. **d.** Results with Group 4.

Figure 9. Average *PR*, *R* and *F-value* scores comparing our approach with RPD [50] and VSD [29].

One can also realize that our method produces highest *precision*, *recall*, and *f-value* levels with Group 1 (*high ambiguity* and *rich structure* XML nodes), with an average 35% improvement over *RDP* and *VSD* (Figure 9.a), in comparison with average 25%, 5%, and almost 0% improvements with Groups 2, 3, & 4 respectively. This concurs with our results of the previous section: our method is more effective when dealing with highly ambiguous nodes within a rich XML structure, in comparison with less ambiguous/poorly structured XML.

[13] Available at http://metalab.unc.edu/bosak/xml/eg/shaks200.zip
[14] Available at simply-amazon.com/content/XML.html
[15] Available at http://www.acm.org/sigmod/xml
[16] Data extracted from http://www.imdb.com/ using a wrapper generator.
[17] Available at http://www.cs.wisc.edu/niagara/
[18] Available from http://www.w3schools.com
[19] Manually identified from repeated tests with different parameter values.

5. CONCLUSION

This paper introduces a novel XML Semantic Disambiguation Framework titled *XSDF*, to semantically annotate XML documents with the help of machine-readable lexical knowledge base (e.g., WordNet), which is a central pre-requisite to various applications ranging over semantic-aware query rewriting [11, 40], XML document classification and clustering [49, 53], XML schema matching [13, 55], and blog analysis and event detection in social networks [2, 7]. *XSDF* covers the whole disambiguation pipeline from: i) linguistic pre-processing of XML node labels to handle compound words (neglected in most existing solutions), to ii) selecting ambiguous nodes for disambiguation using a dedicated *ambiguity degree* measure (unaddressed in most solutions), iii) representing target node contexts as comprehensive and flexible (user chosen) *sphere neighborhood* vectors (in contrast with partial and fixed context representations, e.g., parent node or sub-tree context), and iv) running a hybrid disambiguation process, combining two (user chosen) methods: *concept-based* and *context-based* (in contrast with static methods). Experimental results w.r.t. user judgments reflect our approach's effectiveness in selecting ambiguous XML nodes and identifying node label senses, in comparison with existing solutions.

We are currently investigating different XML tree node distance functions (including edge weights, density, direction, etc. [16, 21]), to define more sophisticated neighborhood contexts. Fine-tuning user parameters using dedicated optimization techniques [19, 30] is another work in progress. We are also investigating the use of additional/alternative lexical knowledge sources such as Google [22], Wikipedia [12], and FOAF [2] to acquire a wider word sense coverage, and thus explore our approach in practical applications, namely semantic blog and wiki document clustering.

6. REFERENCES

[1] Abiteboul S. *et al.*, *Data on the Web: From Relations to Semistructured Data and XML.* Morgan Kaufmann Publisher, 1999. pp. 258.

[2] Aleman-Meza B. *et al.*, *Scalable Semantic Analytics on Social Networks for Addressing the Problem of Conflict of Interest Detection.* TWeb, 2008. 2(1):7.

[3] Amitay E. *et al.*, *Multi-Resolution Disambiguation of Term Occurrences.* Proc. of the Inter. CIKM Conf., 2003. pp. 255-262.

[4] Artiles J. *et al.*, *Word Sense Disambiguation based on Term to Term Similarity in a Context Space.* In SensEval-3:, 2004. pp. 58-63.

[5] Banerjee S. and Pedersen T., *An adapted Lesk algorithm for word sense disambiguation using WordNet.* In Proc. of CICLing'02, 2002.

[6] Banerjee S. and Pedersen T., *Extended Gloss Overlaps as a Measure of Semantic Relatedness.* Inter. IJCAI'03 Conf., 2003. p. 805-810.

[7] Berendt B. *et al.*, *Bridging the Gap: Data Mining and Social Network Analysis for Integrating Semantic Web and Web 2.0.* JWS, 2010. 8(2-3): 95-96.

[8] Bertino E. *et al.*, *Measuring the structural similarity among XML documents and DTDs.* J. of Intel. Info. Systems, 2008. 30(1):55-92.

[9] Budanitsky A. and Hirst G., *Evaluating WordNet-based Measures of Lexical Semantic Relatedness.* Compt. Linguistics, 2006. 32(1): 13-47.

[10] Chan Y. S. *et al.*, *NUS-PT: Exploiting parallel texts for word sense disambiguation in the English all-words tasks.* SemEval, 2007, 253-256

[11] Cimiano P. *et al.*, *Towards the Self-Annotating Web.* WWW, 2004, 462-471.

[12] Dandala B. *et al.*, *Sense Clustering using Wikipedia.* RANLP, 2013, 164-171.

[13] Do H. and Rahm E., *Matching Large Schemas: Approaches and Evaluation.* Information Systems, 2007. 32(6): 857-885.

[14] Fellbaum C., *Wordnet: An Electronic Lexical Database.* MIT Press, 1998, 422.

[15] Francis W. N. and Kucera H., *Frequency Analysis of English Usage.* Houghton Mifflin, Boston, 1982.

[16] Ganesan P. *et al.*, *Exploiting Hierarchical Domain Structure To Compute Similarity.* ACM TOIS, 2003. 21(1):64-93.

[17] Graupmann J. *et al.*, *The SphereSearch Engine for Unified Ranked Retrieval of Heterogeneous XML and Web Documents.* VLDB, 2005, 529-540.

[18] Guo Y. *et al.*, *HIT-IR-WSD: A WSD System for English Lexical Sample Task.* SemEval 2007, ACL.

[19] Hopfield J. and Tank D., *Neural Computation of Decisions in Optimization Problems.* . Biological Cybernetics, 1985. 52(3):52–141.

[20] Ide N. and Veronis J., *Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art.* Compt. Ling., 1998. 24(1):1-40.

[21] Jiang J. and Conrath D., *Semantic Similarity based on Corpus Statistics and Lexical Taxonomy.* Inter. COLING Conf., 1997.

[22] Klapaftis I. and Manandhar S., *Evaluating Word Sense Induction and Disamiguation Methods.* Language Resources and Eval., 2013.

[23] Krovetz R. and Croft W. B., *Lexical Ambiguity and Information Retrieval.* ACM Trans. on Info. Systems, 1992. 10(2):115-141.

[24] Leacock C. and Chodorow M., *Combining Local Context and WordNet Similarity for Word Sense Identification.* MIT Press, 1998. pp. 265-283.

[25] Lee J. *et al.*, *Information Retrieval Based on Conceptual Distance in IS-A Hierarchies.* J. of Documentation, 1993. 49(2):188-207.

[26] Lesk M., *Automatic Sense Disambiguation using Machine Readable Dictionaries.* Inter. SIGDOC'86 Conf., 1986.

[27] Lin D., *An Information-Theoretic Definition of Similarity.* ICML, 1998.

[28] Maguitman A. *et al.*, *Algorithmic Detection of Semantic Similarity.* Proc. of the Inter. WWW Conf., 2005. pp. 107-116.

[29] Mandreoli F. *et al.*, *Versatile Structural Disambiguation for Semantic-Aware Applications.* In Proc. of Inter. CIKM Conf., 2005. pp. 209-216.

[30] Marie A. and Gal A., *Boosting Schema Matchers.* OTM 2008, 283-300.

[31] Marquez L. *et al..*, *Supervised corpus-based methods for WSD. In Word Sense Disambiguation: Algorithms and Applications.* E. Agirre and P. Edmonds, Eds. Springer, New York, NY, 2006. pp. 167-216.

[32] McGill M., *Introduction to Modern IR,* 1983. McGraw-Hill, New York.

[33] Mihalcea R., *Knowledge-based Methods for WSD.* In Word Sense Disambiguation: Algorithms and Applications, 2006. pp. 107–131.

[34] Mihalcea R. and Faruque E., *Senselearner: Minimally supervised word sense disambiguation for all words in open text.* SensEval-3, 2004, 155-158.

[35] Miller G., *WordNet: An On-Line Lexical Database.* Inter. Journal of Lexicography, 1990. 3(4).

[36] Miller G.A. *et al.*, *A Semantic Concordance.* In Proc. of ARPA Workshop on Human Language Technology, 1993. pp. 303–308.

[37] Ming M. *et al.*, *A Harmony Based Adaptive Ontology Mapping Approach.* In Proc. of the Inter. SWWS'08 Conf., 2008. pp. 336-342.

[38] Charbel N. *et al.*, *Resolving XML Semantic Ambiguity - Technical Report.* Available at http://sigappfr.acm.org/Projects/XSDF/, 2014.

[39] Navigli R., *Word Sense Disambiguation: a Survey.* CSUR, 2009. 41(2):1–69.

[40] Navigli R. and Velardi P., *An Analysis of Ontology-based Query Expansion Strategies.* In proc. of the Inter. IJCAI'03 Conf., 2003.

[41] Nierman A. and Jagadish H. V., *Evaluating structural similarity in XML documents.* Proc. of WebDB, 2002. pp. 61-66.

[42] Patwardhan S. *et al.*, *Using Measures of Semantic Relatedness for Word Sense Disambiguation.* In Proc. of CICLing'03, 2003. pp. 241-257.

[43] Patwardhan S. *et al.*, *SenseRelate:TargetWord – A Generalized Framework forWord Sense Disambiguation.* AAAI'05, 4, 1692-1693.

[44] Pradhan S. *et al.*, *Semeval-2007 task-17: English lexical sample, SRL and all words.* In Proc. of SemEval'07, 2007. pp. 87-92.

[45] Resnik P., *Disambiguating Noun Groupings with Respect to WordNet Senses.* 3rd Workshop on Large Corpora, 1995. pp. 54-68.

[46] Richardson R. and Smeaton A., *Using WordNet in a Knowledge-based approach to information retrieval.* BCS-IRSG Colloquium on IR, 1995.

[47] Stanford Center for Biomedical Informatics Research. *Protégé Ontology Editor*. [cited January 2015].

[48] Tagarelli A. *et al.*, *Word Sense Disambiguation for XML Structure Feature Generation.* In Proc. of the ESWC, 2009, pp. 143–157.

[49] Tagarelli A. and Greco S., *Semantic Clustering of XML Documents.* ACM Transactions on Information Systems (TOIS), 2010. 28(1):3.

[50] Tagarelli A. *et al.*, *Word Sense Disambiguation for XML Structure Feature Generation.* In Proc. of ESWC, 2009, pp. 143–157.

[51] Taha K. and Elmasri R., *CXLEngine: A Comprehensive XML Loosely Structured Search Engine.* Proc. of the EDBT DataX'08, 2008, 37-42.

[52] Taha K. and Elmasri R., *XCDSearch: An XML Context-Driven Search Engine.* IEEE TKDE, 2010. 22(12):1781-1796.

[53] Tekli J. *et al.*, *A Novel XML Structure Comparison Framework based on Sub-tree Commonalities and Label Semantics.* Elsevier JWS, 2012. 11: 14-40.

[54] Tekli J. *et al.*, *An Overview of XML Similarity: Background, Current Trends and Future Directions.* Elsevier CS Review, 2009. 3(3):151-173.

[55] Tekli J. *et al.*, *Minimizing User Effort in XML Grammar Matching.* Elsevier Information Sciences Journal, 2012. 210:1-40.

[56] Theobald M. *et al.*, *Exploiting Structure, Annotation, and Ontological Knowledge for Automatic Classification of XML Data.* WebDB, 2003, pp. 1-6.

[57] Tratz S. *et al.*, *PNNL: A supervised maximum entropy approach to word sense disambiguation.* In Proc. of SemEval, 2007. pp. 264-267.

[58] W3C. *The Document Object Model*, 2005, http://www.w3.org/DOM.

[59] Wu Z. and Palmer M., *Verb Semantics and Lexical Selection.* 32nd Annual Meeting of the ACL, 1994. pp. 133-138.

[60] Yaworsky D., *Word-Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora.* COLING, 1992, pp. 454-460.

[61] Zhang Z. *et al.*, *Similarity Metric in XML Documents.* Knowledge Management and Experience Management Workshop, 2003.