Semantic Annotation of Office Documents

Boheng Fan, Ning Li and Yingai Tian

Beijing Information Science & Technology University, Beijing, China

Abstract

Semantic annotations can be used to add semantic metadata to documents, which enables their effective, automated analyses. However, only a few document formats (e.g., HTML, PDF) currently support semantic annotations. In office document formats represented by OOXML or ODF, semantic annotations are not yet supported. On the basis of an in-depth study of office document formats, this paper draws on the mainstream semantic metadata identification method of HTML and proposes OOXML-oriented semantic annotation rules. These rules support the addition of semantic metadata to office documents in a standardized way. In addition, pre-processing and post-processing methods are presented, which enable existing office software to read, edit, and save office documents with semantic metadata without modification. Experimental results demonstrate that the proposed approach can add semantic metadata to office documents, and the added semantic metadata can be parsed into RDF triple-structured data by mainstream validators such as W3C or Google. This research can provide a good foundation for tasks such as office document classification, office document information retrieval, and information extraction.

Keywords

document format, semantic annotation, RDFa

1. Introduction

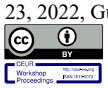
With the increasing number and widespread dissemination of various types of documents, the need to automatically analyze documents continues to grow. Their automated analyses generally require semantic annotation, which is a core document technology [1]. Semantic annotations use an Ontology or Vocabulary to add semantic markups to the specific content of a document, identify the corresponding concepts or entities, and establish the relationships between the tagged content and the vocabulary. They enable documents to have semantic metadata, which enables their automated analyses; in turn, this facilitates the sharing and effective use of document information through semantic retrieval, content extraction, and automatic classification of documents.

Recently, semantic annotations have emerged that use markup languages (e.g., Microdata [2], RDFa [3]) together with the Schema.org [4] vocabulary. RDFa is a standard developed by the W3C. It can add the following attributes to elements:

- @vocab: The default vocabulary currently in use.
- @typeof: The concept (type) of the resource.
- @property: A property that a type has.

Search engines such as Google, Yahoo!, and Yandex optimize search results based on structured markups embedded in web pages, and they can extract structured data from HTML to understand the semantics of their content. One particular area witnessing the boom in semantic annotations is e-commerce, where online shops increasingly embed Schema.org annotations in HTML-pages to describe products. This enables search engines to easily identify products, drive traffic to appropriate websites [5], and present related advertisements. Such structured product data on the web create opportunities for new services such as product classification [6], product matching [7], and recommender systems [8]; emerging research areas include product knowledge graphs [9].

ICBASE2022@3rd International Conference on Big Data & Artificial Intelligence & Software Engineering, October 21-23, 2022, Guangzhou, China



Beyond web applications, text is also present in a wide variety of documents including office, fixed-layout, and complex mixed formats.

A limited number of studies are available that focus on the semantic annotation of office documents. Tallis [10] provides a set of semi-automatic document annotation system Semantic Word for the DOC format. Carr [11] proposes the WICKOffice system. Fink [12] develops a Word plug-in that allows users to manually annotate professional terms in the biomedical field in documents. However, these annotation systems cannot support standard office document formats such as Office Open XML (OOXML) [13], Open Document Format (ODF) [14], Unified Office document Format (UOF) [15]. And they can not support multi-source semantic annotations.

When semantically annotating an office document, the following requirements need to be met: 1) Ability to record semantic metadata in document formats; 2) Ability to support document format standards such as OOXML, ODF, UOF; 3) Ability to withstand the semantic metadata editing and modification; 4) Ability to maintain consistency between document content and semantic metadata during document editing; 5) Ability to support multi-source semantic annotation, allowing the same annotation content to correspond to multiple ontologies or vocabularies. These complex requirements have impeded the development of mature tools for embedding semantic annotations in office documents.

To address the requirements summarized above, this paper proposes a method to support the semantic annotation of office documents. The method includes the extension method ofoffice document, the design of semantic annotation rules and support for semantic annotations in mainstream office software. The results of this paper have been adopted by the group standard T/CESA 1176-2021 "Information technology—Method for embedding semantic metadata into the electronic documents [16]".

2. Related Work

Although the studies has limited results on the semantic annotation of office documents, there are some related research results that can be used for reference. For example, text in HTML is stored in a similar way to office documents, and there are mature semantic annotation technologies for PDF.

HTML mainly adopts the method of embedding annotations, and the results are recorded as attributes of related elements. Tittel [17] embeds the vocabulary of RDFa-marked in web page content on medieval French. Beno [18] developed Doc2RDFa, an HTML rich text processor capable of automatically and manually annotating content. However, the system only targets documents in the legal field. Salem [19] enhances original online content with semantic annotations automatically derived from different knowledge sources. It is the transformation from a web document without semantic annotations to another document with RDFa semantics. Albukhitan [20]、Mbouadeu [21] use deep learning to automatically annotate HTML documents, and the results are stored in the original document using JSON-LD or Microdata.

PDF mainly uses a separate annotation method, and the results are stored in separate data blocks in the document. Kim [22]、Eriksson [23] establish the mapping relationship between the content and an Ontology in PDF documents by using Extensible Metadata Platform (XMP) [24].

In the above research, the embedded annotation results are easy to manage and can ensure that the semantic metadata are consistent with the document content. However, they have a certain impact on the original document format. In contrast, separate annotation results need to be stored separately, which makes it difficult to ensure that the semantic metadata are consistent with the document content, but have less impact on the original document format.

Among the embedded semantic annotation techniques, RDFa has stronger expressiveness and broader applicability; it can support multiple vocabularies [25]. Therefore, in view of the many advantages of RDFa application in HTML, this paper extends the office document format OOXML based on RDFa so that semantic metadata can be added to office documents. At present, the OOXML format is the most widely used in office documents, and it is often considered as a research object. Other XML-based office document format standards such as ODF and UOF are similar, and the method in this paper is also applicable.

3. Document Format Extension

Before using RDFa for the semantic annotation of OOXML documents, it is first necessary to analyze the document structure of HTML and OOXML. The news in Figure 1 is used as an example to illustrate the analysis.

At least 22 dead, more than 1,200 injured in Turkey earthquake
By Isil Sariyuc, Hamdi Alkhshali and Amir Vera, CNN
Updated: Sat, 25 Jan 2020 15:19:20 GMT
Istanbul (CNN) At least 22 people died and hundreds injured in eastern Turkey after an earthquake rattled the region on Friday evening, according to authorities.The 6.7-magnitude quake struck near the town of Sivrice, in eastern Elazig province, collapsing at least 10 buildings, Turkish Interior Minister Sulyman Soylu said.

Figure 1: A case of news text.

For an HTML document, the main content is displayed by combining different block-level elements and inline elements. Block-level elements such as 'div', 'p', etc. usually represent paragraphs or fragments of text. Inline elements such as 'span' are usually used to refine the segmentation of local text. Using RDFa does not affect HTML browsing. The semantic annotation of HTML using RDFa is shown in Figure 2. There are two annotations, "Elazig province" and "Sulyman Soylu", which are the two entity names representing a place (i.e., typeof="Place") and a person (i.e., typeof="Person"). The vocabulary used is "http://schema.org/".

```
<div vocab="http://schema.org/">
    <span typeof="Place" property="name">Elazig province</span>,collapsing at least 10 buildings, Turkish Interior Minister
    <span typeof="Person" property="name">Sulyman Soylu</span>
</div>
```

Figure 2: Example of RDFa annotations in HTML.

For an OOXML document, the storage is based on the ZIP compressed packaging format [26]. The main content of the office documents is recorded in the document.xml file in the document package. Paragraphs are stored as paragraph elements 'p' in the OOXML format; these are the basic units that constitute a document. The child elements of 'p' include the paragraph attribute element 'pPr' and run element 'r'. The text in the paragraph forms multiple 'r' according to different styles. The child elements of 'r' contain the sentence attribute element 'rPr' and the text element 't'. The text in the sentence is recorded in the 't' element. Semantic metadata are mainly annotated at the 'r' or 't' level.

RDFa attributes can be embedded into 't' or other elements as shown in Figure 3.

```
<w:p vocab="http://schema.org/">
    <w:r><w:t typeof="Place" property="name">Elazig province</w:t></w:r>
    <w:r><w:t>,collapsing at least 10 buildings, Turkish Interior Minister</w:t></w:r>
    <w:r><w:t typeof="Person" property="name"> Sulyman Soylu</w:t></w:r>
</w:p>
```

Figure 3: Example of RDFa annotations in OOXML.

For the general situation that is described above, this way of extending OOXML based on RDFa is feasible. However, OOXML is different from HTML. OOXML is mainly used for editing office documents, and its document data and semantic information have a relatively complex relationship. This complexity leads to the following issues:

1. The scope of semantic annotations in OOXML may be inconsistent with the scope of text elements in two cases:

a) A single text may correspond to multiple semantic annotations. In the above example, entities such as "Istanbul", "Turkey" appear under the same 't'. Since 't' is already the smallest unit of text, it is impossible to assign different RDFa properties to different entities. This is illustrated in Figure 4.

```
<w:p>
    <w:r><w:t>Istanbul (CNN) At least 22 people died and hundreds injured in eastern Turkey after an earthquake rattled the region on Friday evening, according to authorities.
    </w:t></w:r>
</w:p>
```

Figure 4: Text element contains OOXML source for multiple entities.

b) The content to be annotated may be scattered among multiple run elements 'r' or text elements 't'. In office software such as Microsoft Word, the texts of different formats are automatically divided into different run elements 'r'. It is difficult to obtain a complete annotation. "Isil Sariyuc" in the example illustrated in Figure 5 is placed in multiple run elements 'r' or text elements 't'

```
<w:p>
    <w:r><w:t xml:space="preserve">Isil </w:t></w:r>
    <w:r><w:t xml:space="preserve">Sariyuc</w:t></w:r>
</w:p>
```

Figure 5: OOXML source code for entities scattered into multiple run and text elements.

2. HTML is used for viewing, there is no need for readers to support editing functions. In contrast, OOXML supports editing, so documents with embedded semantic metadata need to be able to be opened and edited by office software. In addition, the embedded semantic metadata need to be robust and withstand repeated editing.

In summary, semantic annotation rules and office software that support semantic annotations for office documents are needed.

4. Semantic Annotation Rules

In order to ensure the consistency of semantic annotations in OOXML within the scope of text elements, this paper proposes a model for office document annotations. The annotation model is shown in Figure 6.

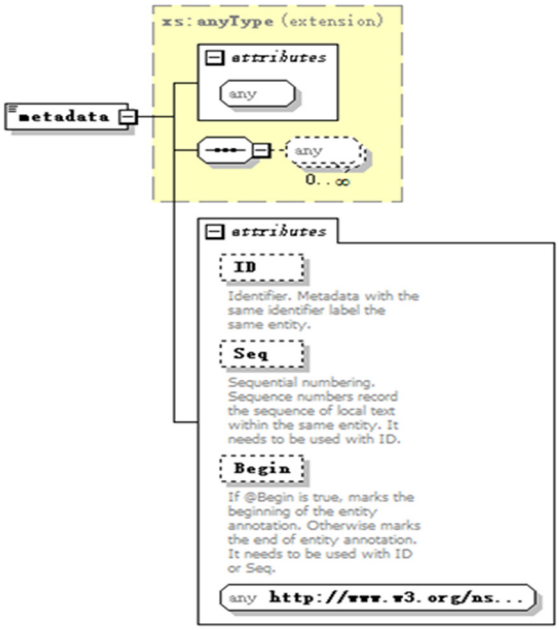


Figure 6: Annotation model.

The annotation model is described by an XML schema. The root element is "metadata", which has three attributes: @ID, @Seq, and @Begin. Any RDFa property can also be used.

For the case where a single text corresponds to multiple semantic annotation content, multiple content to be annotated in the text element 't' can be placed into multiple 'metadata' element nodes for description; the relevant text is set as 'metadata' in the content of the element. Taking "Istanbul", "Turkey", and "earthquake" in Figure 4 as examples, the results of their annotations are shown in Figure 7.

```
<w:t vocab=" http://schema.org/">
    <dsm:metadata rdfa:typeof="Place" rdfa:property="name">Istanbul</dsm:metadata>
(CNN) At least 22 people died and hundreds injured in eastern
    <dsm:metadata rdfa:typeof="Place" rdfa:property="name">Turkey</dsm:metadata>
after an
    <dsm:metadata rdfa:typeof="Event" rdfa:property="name">earthquake</dsm:metada-
ta> rattled the region on Friday evening, according to authorities.
</w:t>
```

Figure 7: Annotation method in which a single text corresponds to multiple semantic annotation contents.

In Figure 7, "Istanbul" and "Turkey" are place (rdfa:typeof="Place") names (rdfa:property="name"); "earthquake" is the name (rdfa:property="name") of the event (rdfa:typeof="Event"). These annotations utilize the vocabulary "http://schema.org/".

For the case in which the content to be annotated is scattered in multiple run elements 'r' or text elements 't', multiple semantic annotations can be combined in sequence by specifying the @ID and @Seq attributes in the 'metadata' element, where @ID is the number of the marked entity and @Seq is the sequence number of the annotation. @ID combined with @Seq can realize multi-segment annotations of the same entity. @Begin indicates the start or end of an annotation.

For example, "Isil Sariyuc" is the editor of this news. In some vocabularies, name is further refined into first and last names. Therefore, "Isil Sariyuc" is annotated as two entities, namely "Isil" and "Sariyuc". The annotation results are shown in Figure 8.

```
1 <dsm:metadata ID="a0" Seq="1" Begin="true" rdfa:vocab="http://schema.org/"
  rdfa:typeof="Person" rdfa:property="name"/>
2 <w:r><w:t xml:space="preserve">Is</w:t></w:r>
3 <w:r><w:t xml:space="preserve">il </w:t></w:r>
4 <dsm:metadata ID="a0" Seq="1" Begin="false"/>
5 <dsm:metadata ID="a0" Seq="2" Begin="true" rdfa:vocab="http://schema.org/"
  rdfa:typeof="Person" rdfa:property="name"/>
6 <w:r><w:t xml:space="preserve">Sari</w:t></w:r>
7 <w:r><w:t xml:space="preserve">yuc</w:t></w:r>
8 <dsm:metadata ID="a0" Seq="2" Begin="false"/>
```

Figure 8: An annotation method in which the content to be annotated is scattered among multiple run elements and texts.

The two metadata element nodes described in lines 1-4 and 5-8 in Figure 8 have the same @ID to indicate that they collectively annotate one person entity. The annotations are delineated with @Begin="true" in the metadata element node of line 1 and @Begin="false" in the metadata element node of line 4. They have the same @Seq attribute (Seq="1") to indicate that they are for one person entity in the vocabulary "http://schema.org/" (rdfa:vocab="http://schema.org/" rdfa:typeof="Person" rdfa:property="name"). The first part of the annotation corresponds to the text "Is", "il" in the two sentence elements 'r'. Similarly, lines 5-8 indicate that the annotation content of the second part of the person entity corresponds to the text "Sari" and "yuc".

5. Semantic Document Editing and Processing

The OOXML document that is extended by adding RDFa attributes to the document in OOXML format is called a semantic document. However, this extension prevents word processing software from opening and editing semantic documents properly. This paper proposes a solution to achieve seamless switching between semantic documents and ordinary office documents through pre-processing and post-processing methods.

5.1. Semantic Document Pre-processing and Conversion

During pre-processing, the comment mechanism in the office document is used as the carrier for recording semantic metadata in the word processing software. The semantic metadata marked in the semantic document are stored in comments. Comments in office documents are annotation information attached to document content fragments. Comments are displayed independently and associated with the original text, pictures, and other content in office documents. The comments do not impact the typesetting style of the original document; they are convenient for people to observe and operate. At the same time, users can directly edit and modify comments when editing the original text of the document, which provides a robust editing environment. Therefore, using comments to store semantic metadata allows users to edit and modify semantic metadata in the editing process. It is also possible to keep the semantic metadata consistent during editing.

In document.xml in the OOXML packaged file, comments are described by three elements: 'commentRangeStart', 'commentRangeEnd', and 'commentReference'. Among them, 'commentRangeStart' and 'commentRangeEnd' determine the starting position and ending position of the comment, indicating the range of the comment (comment.Range). The 'commentReference' element is associated with the content of the comment element in comment.xml; it is used to display the content of the comment in word processing software. In a comment, the above four elements have the same attribute @id. For example, after annotating "Istanbul" in Figure 1, document.xml appears as shown in Figure 9.

```
<w:commentRangeStart w:id="0"/>
<w:r>
    <w:t>Istanbul</w:t>
</w:r>
<w:commentRangeEnd w:id="0"/>
<w:r>
    <w:commentReference w:id="0"/>
</w:r>
```

Figure 9: Annotation reference in document.xml.

The annotation is associated with the comment.xml file structure as shown in Figure 10.

```
<w:comment w:id="0"/>
<w:p>
    <w:r><w:t>comment content</w:t></w:r>
<w:p>
</w:comment>
```

Figure 10: Annotation reference in comment.xml.

To distinguish between general comments and comments containing semantic metadata, a special username "dsm" is used for semantic metadata comments. The assumption is that the special username is only associated with semantic comments. In the future, a special type for document comments can

be investigated to record semantic metadata in office documents. However, this requires revising the relevant standards through standards development organizations.

The core idea of the pre-processing algorithm is to identify the scope of each semantic annotation. That is, the scope of the semantic elements' 'metadata' determines the scope of the office documents comment. According to the recorded semantic metadata, the content of the office documents comment is determined.

When determining the scope of comments in an office document, two cases can occur: (1) There are multiple semantic annotation entities under a single text element 't'; (2) An annotation entity is described by multiple run elements 'r' or text elements 't'. For case 1, the document structure needs to be adjusted in order to comply with the standard format of OOXML. Specifically, it is necessary to copy multiple 'r' and 't' element nodes and ensure that each 't' contains only one element 'metadata'. For case 2, the comment range can be determined directly according to the two element nodes' 'metadata'. In this paper, the following algorithms are used to pre-process semantic documents.

Algorithm1：**CreateCommentNode**
Input：*m* as metadata, *r* as range, *u* as user
Output：*c* as the comment node

```
1  Function CreateCommentNode(m, r, u)
2     Create a new comment node c;
3     c.content ← n.metadata; // n is the semantic node in r
4     c.range ← Range(t); // t is the text node in r; Range(t) means get range of
t
5     c.user ← "dsm"; // Set user of c as "dsm"
6     Output c;
7  End Function
```

Algorithm2：**ForwardTransform**
Input：*D* as the semantic document
Output：*D* as the transformed document in the standard format

```
1  Function ForwardTransform(D)
2     Let c-list be the comment list to be added into D;
3     c-list ← null;
4     For each paragraph p in D
5        For each run node r in p
6           If r contains multiple semantic nodes Then
7              Split r into multiple nodes r-list, each of which contains a single
semantic node;
8           End If
9        End For
10    End For
11    For each semantic node n in D
12       If n spans multiple text nodes Then
13          Group the text nodes of same semantics into one segment;
14          For each text segment s which n spans
15             c ← CreateCommentNode(n.metadata, Range(s), "dsm");
16             Add c into c-list;
17          End For
18       Else // n spans single text node t
19          c ← CreateCommentNode(n.metadata, Range(t), "dsm");
20          Add c into c-list;
21       End If
22    End for
23    Add c-list into D;
24    Output D;
25End Function
```

Algorithm 1 is used to generate comments, including the range and content of comments. Algorithm 2 is the flow of the pre-processing algorithm. In Algorithm 2, step 2-3 declares the set of comments to be added and assigns the value to be empty. Steps 4-22 are the process of generating semantic metadata comments. Among them, steps 4-10 are used to adjust the document structure, which corresponds to the first case of determining the range of comments described above. In steps 11-22, if the annotated entity is described by multiple run elements 'r' or text elements 't', thenperform steps 13-17. After determining the start and end positions of annotated entity, call algorithm 1 to generate semantic comment; otherwise, according to the document structure generated in steps 4-10, Algorithm 1 is directly invoked to generate semantic comment. Steps 23-25 output the converted office documents conforming to the OOXML standard.

5.2. Semantic Document Post-processing and Conversion

The purpose of post-processing is opposite to that of pre-processing. The core idea is to find each annotation with semantic metadata and its scope in office documents, so as to determine the content and scope of semantic annotations. There are also two cases: (1) the entity is described by a single text element 't'; (2) the entity is described by multiple run elements 'r' and text elements 't'. For case 1, the entity is placed into the 'metadata' element node for description, and the content of the annotation is placed into the 'metadata' attribute. For case 2, two 'metadata' element nodes need to be added. This is accomplished by setting the corresponding @Begin and inserting the nodes after the 'commentRangeStart' and before the 'commentRangeEnd'. The content of the comment is placed in the 'metadata' attribute whose @Begin value is true. In this paper, the following algorithm is used for post-processing semantic documents.

Algorithm3：**BackwardTransform**
Input：*D* as the document in the standard format
Output：*D* as the semantic document

```
1  Function BackwardTransform(D)
2    For each comment node c in D
3      If c.user is "dsm" Then
4        Let n be the semantic node;
5        If c spans multiple text segment Then
6          n.id ← GenerateID(); // Generate ID
7          n.metadata ← c.content;
8          Insert n at the first text node;
9          Add end tag of n with n.id at the end of c.range;
10       Else // c spans single text node t
11         n.id ← GenerateID();
12         n.metadata ← c.content;
13         Insert n at t;
14       End If
15       Remove c;
16     End If
17   End For
18   Output D;
19 End Function
```

In Algorithm 3, steps 2-17 are the process of adding the semantic element 'metadata'. In step 3, it is determined whether comments is a semantic metadata comments according to comments user name. If it is determined to be true, perform steps 4-15. Steps 5-14 are used to add the positions and content of semantic element 'metadata'. Among them, if the entity is described by multiple texts, perform steps 6-9 to assign the comment content to the attribute of the semantic element 'metadata' whose @Begin value is "true"; otherwise, perform steps 11-13, assign the comment content to the attribute of semantic element metadata, and insert 'metadata' under the text element 't'. Steps 18-19 output the semantic document generated by conversion.

6. Experiment

A semantic annotation tool is implemented for office documents based on C# and the winform designer. The tool has three functions: semantic annotation, pre-processing conversion, and post-processing conversion.

Taking the news in Figure 1 as an example, it is annotated according to the afore-mentioned semantic annotation rules for office documents. This example involves multiple entity objects, such as "Istanbul", "Elazig province", "earthquake", and so on. When applying the proposed method for entity annotation, it is necessary to select the appropriate type from the Schema.org vocabulary according to the nature of the object, and also select the appropriate attribute. Taking the first paragraph of the news example as an example, the annotation results are shown in Figure 11.

```
<w:p vocab="http://schema.org">
    <dsm:metadata ID="a0" Seq="1" Begin="true" rdfa:typeof="Person" rdfa:property="na-
me"/>
    <w:r><w:t xml:space="preserve">Is</w:t></w:r>
    <w:r><w:t xml:space="preserve">il </w:t></w:r>
    <dsm:metadata ID="a0" Seq="1" Begin="false"/>
    <dsm:metadata ID="a0" Seq="2" Begin="true" rdfa:typeof="Person" rdfa:property="na-
me"/>
    <w:r><w:t xml:space="preserve">Sari</w:t></w:r>
    <w:r><w:t xml:space="preserve">yuc</w:t></w:r>
    <dsm:metadata ID="a0" Seq="2" Begin="false"/>
    <w:r>
        <w:t>
            <dsm:metadata rdfa:typeof="Place" rdfa:property="name"> Istanbul</dsm:met-
adata"> (
            <dsm:metadata rdfa:typeof="Organization" rdfa:property="name">CNN</dsm:-
metadata">) At least 22 people died and hundreds injured in
            <dsm:metadata rdfa:typeof="Place" rdfa:property="name"> eastern Turkey</d-
sm:metadata> after an
            <dsm:metadata rdfa:typeof="Event" rdfa:property="name"> earthquake </ds-
m:metadata>rattled the region on
            <dsm:metadata rdfa:typeof="Event" rdfa:property="startTime">Friday evening-
</dsm:metadata>, according to authorities.
        </w:t>
    </w:r>
</w:p>
```

Figure 11: Example of an extended OOXML document with semantic annotations.

The above-mentioned extended OOXML document is pre-processed and converted into a standard OOXML document. Figure 12 shows the editing interface after the office software Microsoft Word opens the document. As can be seen in Figure 12, the seman-tic metadata in the semantic document are presented in the form of comments in the Word document. Users can readily view and edit these comments. The semantic user name is "dsm" and each semantic comment corresponds to a named entity.

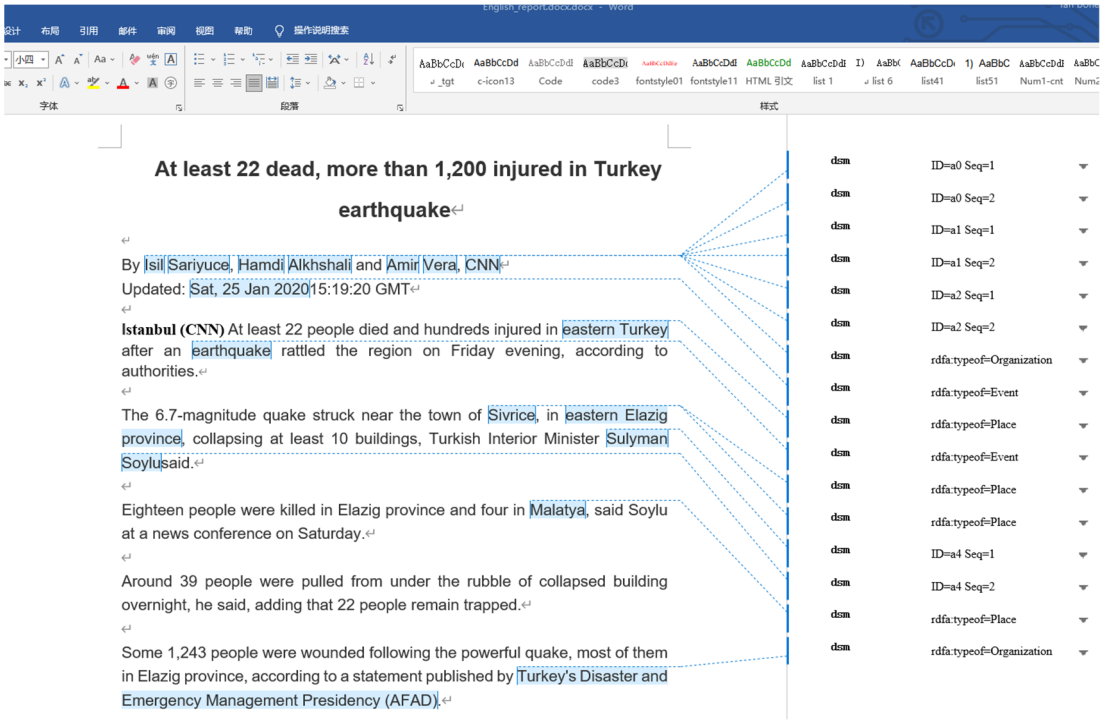


Figure 12: Open preprocessed document in Microsoft Word.

The semantic document structure obtained by the post-processing conversion of the office document shown in Fig 12 is consistent with that in Fig11.

Semantic documents can be checked for RDFa syntax correctness using the Google Structured Testing Tool or the RDFa.info[27] validator. The semantic metadata of the RDF triple structure can also be extracted using GRDDL [28] or W3C's RDFa parser to generate Turtle, RDF/XML, or N-Triples files. The parsed structured data in Figure 13 shows that the semantic document can be verified by the RDFa.info testing tool.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix schema: <http://schema.org/> .
_:1
  rdf:type schema:Person;
  schema:name "Isil Sariyuce" .
_:2
  rdf:type schema:Place;
  schema:name "Istanbul" .
_:3
  rdf:type schema:Organization;
  schema:name "CNN" .
_:4
  rdf:type schema:Place;
  schema:name " eastern Turkey " .
<http://rdfa.info/play/item5>
  rdf:type schema:Event;
  schema:name " earthquake ";
  schema:startDate "Friday evening".
```

Figure 13: Document structured data.

Based on RDF data, knowledge graphs can be further generated. Figure 14 is a knowledge graph corresponding to part of the semantic annotation content of Figure 11.

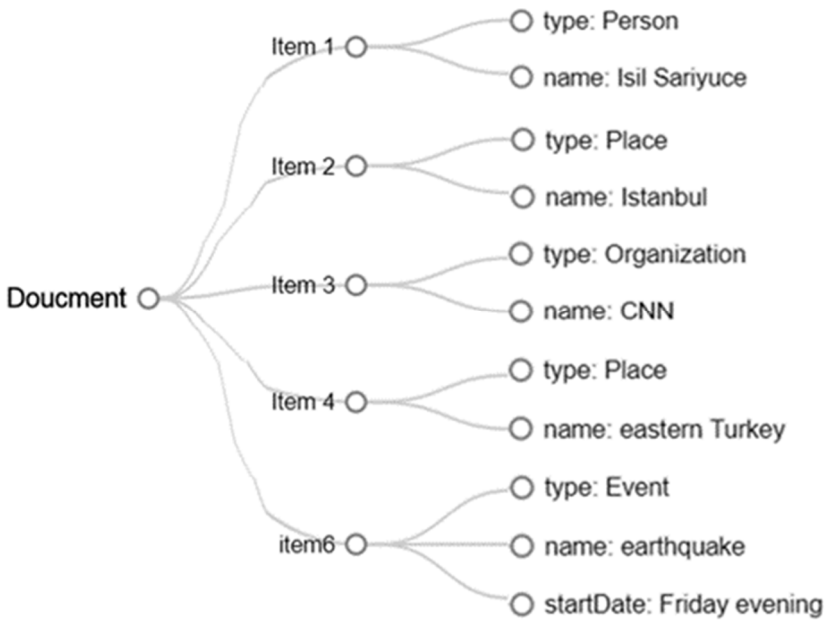


Figure 14: Knowledge graph bsed on document semantic metadata.

The above results indicate that the proposed method successfully realizes the semantic annotation of office documents, which supports the effective extraction of knowledge from documents.

7. Conclusion

This paper takes OOXML as the research object, and analyzes in detail the differences between the OOXML format and the HTML format for text descriptions. On this basis, a method for the semantic annotation of office documents is proposed. According to the format characteristics of office documents, specific semantic annotation rules are designed. At the same time, an approach to achieve seamless switching between semantic documents and office documents through pre-processing and post-processing methods is proposed. This ensures the marked documents can be supported by word processing software after pre-processing. After post-processing, office documents with semantic annotations can be reformed into semantic documents. Experiments show that the use of the proposed annotation method can enhance the semantic representation ability of office documents; in turn, this can support the automated analysis of annotated documents.

This paper mainly focuses on semantic annotation of office documents in OOXML format. The approach can be extended to ODF, UOF, and other formats in the future, which can broaden support for including semantic annotations in a wider variety of office document formats. In addition, the annotation object in this paper is text; the semantic annotation of other objects such as images, tables, formulas, and so on has not been considered. As their structure is more complex, further research and improvement of the annotation method are needed. Natural language processing technology can also be considered in the future to realize automatic or interactive semantic annotation and improve the efficiency of annotation.

8. Acknowledgements

Project supported by National Natural Science Foundation of China: The Intelligent Analysis and Optimization Method for Reflowable Documents (61672105).

9. References

[1] Fu Zhu. (2014) A Review of Semantic Annotation for Text Documents. Journal of the China Society for Scientific and Technical Information, 4:439-448.

[2] Sikos, Leslie. (2015) Mastering structured data on the Semantic Web: From HTML5 microdata to linked open data. Apress.
[3] W3C. (2015) XHTML+RDFa 1.1 — Third Edition. https://www.w3.org/TR/xhtml-rdfa/.
[4] Guha R V., Brickley D. and Macbeth S. (2016) Schema.org: Evolution of Structured Data on the Web. Communications of the ACM, 59,(2): 44-51.
[5] Zhang Z., Bizer C., Peeters R. and Primpeli A. (2020) MWPD2020: Semantic Web Challenge on Mining the Web of HTML-embedded Product Data. Proceedings of CEUR Workshop, RWTH,2020.
[6] Zhang Z. and Paramita M. (2019) Product Classification Using Microdata Annotations. International Semantic Web Conference, Springer, Cham, 2019, 716-732.
[7] Peeters R., Primpeli A., Wichtlhuber B. and Bizer C. (2020) Using Schema.org Annotations for Training and Maintaining Product Matchers. Proc of the 10th International Conference on Web Intelligence, Mining and Semantics, 2020, 195-204.
[8] Bai P., Ge Y., Liu F., and Lu H. (2019) Joint interaction with context operation for collaborative filtering. Pattern Recognition, 88: 729-738.
[9] Dong X L. (2018) Challenges and Innovations in Building a Product Knowledge Graph. Proc of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, 2869-2869.
[10] Tallis M. Semantic Word processing for content authors. Proceedings of the Knowledge Markup & Semantic Annotation Workshop, Florida, USA, 2003.
[11] Carr L., Miles-Board T., Wills G., Woukeu A. and Hall W. (2004) Towards a knowledge-aware office environment. International Conference on Practical Aspects of Knowledge Management, Springer, Berlin, Heidelberg, 2004, 129-140.
[12] Fink JL., Fernicola P., Chandran R., Parastatidis S., Wade A. AND Naim O. (2010) Word add-in for ontology recognition: semantic enrichment of scientific literature. BMC bioinformatics, 11, (1): 1-8.
[13] ISO/IEC 29500-1:2016 Information technology — Document description and processing languages — Office Open XML File Formats. (2004) .
[14] ISO/IEC 26300: 2015 Information technology — Open Document Format for Office Applications (OpenDocument) v1.2. (2015).
[15] GB/T 20916 — 2007: UOF--Unified Office document Format (2007). China Standards Press, Beijing.
[16] T/CESA 1176-2021 Information Technology Electronic Document Semantic Metadata Embedding Method. 2021.
[17] Tittel S., Bermyudez-Sabel H. and Chiarcos C. (2018) Using RDFa to link text and dictionary data for Medieval French. Proc of the Eleventh International Conference on Language Resources and Evaluation, Miyazaki, Japan, 2018, 7-12.
[18] Beno M., Filtz E. and Kirrane S. (2019) Doc2RDFa: semantic annotation for web documents.
[19] Salem H., Mazzara M. and Elnaffar S. (2021) Automatically Injecting Semantic Annotations into Online Articles. International Conference on Advanced Information Networking and Applications, Springer, Cham, 2021, 617-624.
[20] Mbouadeu S F., Keshtkar F. and Bukhari S A C. (2021) Semantically: A Framework for Structured Biomedical Content Authoring and Publishing.
[21] Albukhitan S., Alnazer A. and Helmy T. (2018) Semantic annotation of arabic web documents using deep learning. Procedia Computer Science, 130, 589-596.
[22] Kim H L., Kim H G. and Decker S. (2006) Semantic documentation using semantic web technologies and social web services. International Conference on Next Generation Web Services Practices, IEEE, 2006, 27-32.
[23] Eriksson H. (2007) The semantic-document approach to combining documents and ontologies. International journal of human-computer studies, 65, (7): 624-639.
[24] IX-ISO. (2012) ISO 16684-1-2012. Graphics Technology Extensible Metadata Platform (XMP) Specification Part 1: Data Modeling, Serialization and Core Properties.
[25] Song Yu., Zheng Yi. and Wu Yan. (2009) Overview of RDFa semantic annotation techniques. Proc of National Conference on Computer Networks and Communications, 300-306.

[26] ISO/IEC. (2021) ISO/IEC 29500-2. 2021Document description and processing languages — Office Open XML file formats — Part 2: Open packaging conventions.
[27] Buraga S C. and Panu A. (2013) A web tool for extracting and viewing the semantic markups. International Conference on Knowledge Science， Engineering and Management, Berlin, Springer, 2013, 570-579.
[28] Hazayul-Massieux D. and Connolly D. Gleaning resource descriptions from dialects of languages (grddl). World Wide Web Consortium, W3C Coordination Group Note NOTE-grddl-20040413.