Using Data-Extraction Ontologies to Foster Automating Semantic Annotation

Yihong Ding
Department of Computer Science
Brigham Young University
Provo, Utah 84602
ding@cs.byu.edu

David W. Embley
Department of Computer Science
Brigham Young University
Provo, Utah 84602
embley@cs.byu.edu

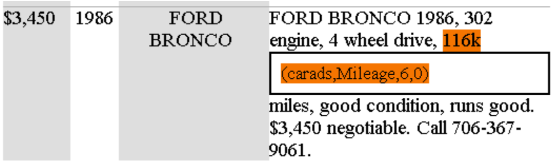
Abstract

Semantic annotation adds formal metadata to web pages to link web data with ontology concepts. Automated semantic annotation is a primary way of enabling the semantic web. A main drawback of existing automated semantic annotation approaches is that they need a post-extraction mapping between extraction categories and ontology concepts. This mapping requirement usually needs human intervention, which decreases automation. Our approach uses data-extraction ontologies to avoid this problem. To automate semantic annotation, the new approach uses an ontology-based data recognizer that fosters automated semantic annotation, optimizes the system performance, provides support for ontology assembly, and is compatible with semantic web standards.

1 Introduction

The semantic web is the web containing machine-processable web data [2]. Semantic annotation research is one of the basic research problems for the semantic web. Automated semantic annotation is a primary way of enabling the semantic web.

A *semantic annotation* process adds formal metadata to web pages. This metadata links data in a web page to defined concepts in an *ontology*, which is an explicit, formal specification of conceptualizations [7]. Figure 1 shows a simple example of semantic annotation, which is produced by our online demo [4]. In the figure, the annotated content is the string "116k." The metadata shows that it is an instance of the concept "Mileage," which is defined in the ontology "carads." Without annotation, "116k" may have on interpretable meaning or may have many different meanings, such as price and length, besides mileage. With this annotation, machine agents can precisely interpret this "116k" with respect to the ontology "carads." Hence the data "116k" becomes machine-processable.



$3,450	1986	FORD BRONCO	FORD BRONCO 1986, 302 engine, 4 wheel drive, 116k (carads,Mileage,6,0) miles, good condition, runs good. $3,450 negotiable. Call 706-367-9061.

Figure 1. Sample Semantic Annotation

In existing automated semantic annotation approaches (e.g. [1, 8, 10]), a typical process employs a data extraction engine to extract data instances from web pages. Since none of the currently adapted data extraction engines are ontology-based, the typical process performs "a set of heuristics for post-processing and mapping of the IE [information extraction] results to an ontology" after extracting data [10]. Semantic mapping from the IE results to the ontology typically needs much human intervention [14], which decreases automation. Kiryakov et. al. also pointed out that such a "post-processing and mapping" requirement is, as their words, "the main drawback" for these existing automated annotation approaches [10]. They argued that "such heuristics are not sufficient for large-scale, domain-independent semantic annotation."

In this paper, we propose a new automated semantic annotation system using data-extraction ontologies which has addressed the mentioned problem. Along with several other improvements, Figure 2 shows the architecture of our system. There are four unique components that distinguish our approach from the existing ones. First, the ontology-based data recognizer directly uses data-extraction ontologies to do data extraction [6]. Since the extraction process is based on ontologies, it avoids the overhead of aligning extraction categories with ontology concepts after extracting data. Second, the conceptual annotator and the structural annotator compose the two-layer annotation model that can both be resilient in general situation and run fast when annotating web pages with a specified layout. Therefore the system can achieve an optimized performance in total. Third,

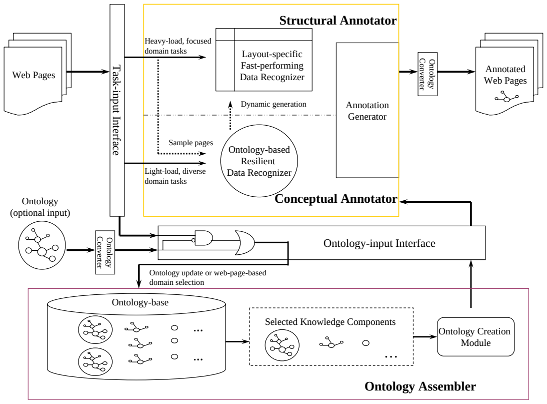


Figure 2. System Architecture

the ontology assembler provides functions that allow users to assemble a task-oriented ontology through an interactive process. Its goal is to maximize the reuse of existing knowledge and minimize the load of manual ontology creation. At last, the ontology converter transforms ontological representations between a semantic web standard OWL (Web Ontology Language) and our data-extraction ontology language OSMX (Object-oriented Systems Model in XML). This converter makes our work be compatible to the semantic web standard.

In Section 2 we describe the encountered problems and our solution approaches when developing the new annotation system. In Section 3, we discuss how our work advances related work. In Section 4, we summarize our contributions and present the future plan of our work.

2 System Description

Our goal is to foster automating semantic annotation so that the new system is practical for real-world applications. To satisfy the goal, we need to figure out solutions of four sub-problems. In this section, we introduce the four sub-problems and their solution approaches, which in sequence are the ontology-based data recognizer, the two-layer annotation model, the ontology assembler, and the ontology converter. Figure 2 illustrates the architecture of the whole system. In short, it inputs a set of web pages and an ontology, and it outputs the annotated web pages with respect to the input ontology. When discussing each sub-problem, we are going to explain its corresponding part in the figure.

2.1 Ontology-based Data Recognizer

To increase the degree of automation, we have focused on two specific factors—resiliency and adaptiveness. The property of *resiliency* allows a system to be continuously applicable on different web-page layouts; and the property of *adaptiveness* allows a system to be continuously applicable on different domains. Less resiliency or adaptiveness for an annotation system means that the system needs more human intervention, i.e. less degree of automation, when annotating web pages either with a different layout or for a different domain.

Relying on web-page layout causes extraction engine not to be resilient. When there is a new page layout or a layout changes, a layout-based data extraction tool needs to perform a wrapper regeneration process so that it can continuously work. Although we can automate the regeneration process, it is too difficult to detect a layout mismatch automatically. In majority it relies on humans to check, which obviously decreases the system's degree of automation. The reason of less adaptiveness is that the system encodes domain-specific knowledge in program implementations, instead of using declarative semantics in ontologies. Hence when a domain changes, users may have to modify program implementations, which decreases the system's degree of automation.

Our solution is to adapt the ontology-based data recognizer that uses data-extraction ontologies. Like the other ontologies, we can specify the intensions of a domain, such as object sets, relationship sets, and hierarchical structures, in a data-extraction ontology. Beyond, our data-extraction ontology language—OSMX—defines formalized semantics that allow users to declare extention recognition semantics of any concept, which we called a data frame [6]. Syntactically we present declarative semantics in data frames by regular expressions. The ontology-based data recognizer matches data frames to web content to find candidate instances, and then uses a set of heuristics to solve ambiguous matchings.

With this methodology, the input data-extraction ontologies contain all the required domain-specific information. So this ontology-based data recognizer is adaptive. In addition, OSMX does not allow users to declare layout-specific semantics in data frames. The extracting process is thus totally resilient. More details about OSMX and our resilient data extraction and semantic annotation using OSMX ontologies can be found in our previous publications [5, 6].

2.2 Two-Layer Annotation Model

Although the use of ontology-based data recognizer improves the system's degree of automation, its run-time performance is comparatively lower than the layout-based data extraction tools [12]. Our ontology-based data extraction engine requires many computational cycles to enumerate all the extracted candidates and resolve ambiguities, which slows down the execution speed. Also, although the extraction engine is designed to achieve good accuracy in general cases, it does not take into account local structural patterns, which can lead to higher extraction accuracy.

We propose the two-layer annotation model to solve this speed and accuracy problem. As Figure 2 shows, the model is composed by two different types of annotators—the conceptual annotator and the structural annotator. On the lower-layer the conceptual annotator uses the ontology-based data recognizer we have just discussed. It provides a resilient and adaptive base for the entire annotation system. On the upper-layer the structural annotator uses a dynamically created, layout-specific data recognizer. When the system needs to annotate large numbers of web pages, and especially if these web pages are for a focused domain and hold a common layout,[1] such a layout-specific recognizer can perform very fast and accurate data extraction on these web pages. With the combination of these two annotators, the system can achieve an optimized performance in total.

We claim that this solution is sound. First, according to the survey made by Laender et. al., among all types of data-extraction tools the layout-specific ones have the best speed of execution. They can also carry out very high accuracy on extraction results when the layout of target web pages perfectly matches the specified layout in their wrappers [12]. Second, it appears possible to dynamically generate a layout-specific data recognizer using annotated sample web pages produced by the conceptual annotator. These annotated web pages constitute an automatically collected training set. With the training set, the recognizer generation becomes a classic machine learning process for retrieving a common layout pattern based on a training set. We can apply a similar wrapper generation method as in [11] and [3] to build layout-specific data recognizers.

2.3 Ontology Assembler

Semantic annotation relies on ontologies. Although we may have built a well-performed annotation system, it is useless when no ontology is available. To be a practical solution, we need the annotation system to be able to help users build an ontology when no existing ones are applicable. So we propose the ontology assembler.

As in Figure 2, the ontology assembler consists of two parts—an ontology-base and an ontology creation module. The ontology-base consists of pre-used and pre-constructed ontologies, snippets of ontology, and single concept recognizers. The ontology creation module can be as simple as an ontology editor that users can view and manually create and modify ontologies. The theme of the ontology assembler is to maximize the reuse of existing ontologies and minimize the work of constructing new ontologies.

When there are no input ontologies, as Figure 2 shows, the ontology-input interface takes a set of descriptive web pages into the ontology assembler. The assembler performs a knowledge-selection process to look for relevant ontology components in the ontology-base using the descriptive web pages. As the dashed box in Figure 2, these selected ontology components could be ontologies, snippets of ontology, or single concept recognizers. The users can thus watch and assemble these selected components through the ontology creation module.

2.4 Ontology Converter

The reason we annotate pages in the semantic web is so we can use them. Any system that does not conform to semantic web standards will not be interoperable, and thus will not be used. Our last remaining problem is that OSMX is not a widely recognized ontology language in the semantic web community.

The solution is straightforward. We need to make an ontology converter that does transformations between OSMX and OWL, the current standard ontology langauge for the semantic web. The converter has been implemented in Java and used Jena [9]. An ontology in either language is first mapped onto the Jena API, through which the converter outputs it in the opposite ontology language.

As in Figure 2, we have placed the converter on both the input and output sides of the system. When users input an OWL ontology, the converter transforms it to its OSMX representations so that the ontology-based data recognizer can process it. On the other hand, if the input is an OSMX ontology, the system converts it to OWL when outputting annotations so that the annotated contents are always with respect to the concepts in an OWL ontology.

3 Discussion and Related Work

From several perspectives our proposed approach advances related work. There have been several automated semantic annotation approaches and each of them has adapted an existing data-extraction engine (e.g. [1, 8, 10]). As we mentioned earlier, "the main drawback" of these systems is that after extracting, they need to perform "post-processing and mapping of the IE results to an ontology." The intrinsic reason for this problem is that "none of these approaches expects an input or produces output with respect to ontologies" [10]. Except for ontology-based data-extraction tools, all the other automated data-extraction approaches do not use ontologies [12]. It is not trivial to integrate ontologies into these non-ontology-based approaches. Therefore, in [10] Kiryakov et. al. suggested that the best solution was "to use the ontology more directly during the process of extraction." Arlotta et. al. [1] also proposed as future work

[1] This is quite common in the ordinary web, e.g., the auto-generated web pages within many large commercial web sites such as amazon.com.

combining their work with an ontology-based data extraction approach such as our data-extraction ontology. The use of data-extraction ontologies in our system does match their suggestion. To the best of our knowledge, our work is the first attempt of using directly ontology-based data extraction tools for semantic annotation.

In the literature, there are several data extraction approaches that use machine learning methods to build layout-specific wrappers, and there are ontology-based data extraction approaches [12]. As we have discussed, these two data extraction approaches are complementary. Until now, however, no previous work has been done. A reason of this lack-of-references is because of the cost of building ontologies. Ontology is not a mandatory requirement in the traditional data extraction paradigm. It is debateable whether the augmentation of ontologies may bring more values for data extraction than the cost of building them. This is, however, unlikely to be a problem for semantic annotation because it is widely accepted that we need ontology to help with building the semantic web.

The ontology assembler research is a mixture of text classification and ontology reuse. Text classification is a traditional machine learning topic that has been studied for many years. Instead of categorizing text with pre-defined labels, the significance of our work is to label them with existing ontology components. Previous work has categorized text using concepts within a taxonomy (e.g. [13]). To the best of our knowledge, however, none of them have thought of retrieving identified ontology components to reuse them to compose a new domain ontology. When more ontologies become available due to the emergence of the semantic web, this type of ontology reuse will likely become more valuable.

4 Conclusions and Future Work

Our vision is to foster automating semantic annotation so that the new system is practical for real-world applications. The new approach uses an ontology-based data extraction engine to avoid the overhead of aligning extraction categories in extraction wrappers with concepts in domain ontologies. This research also combines the resiliency of ontology-based data extraction techniques with the fast and highly accurate layout-based data extraction techniques using a novel two-layer annotation model. The dynamic domain ontology assembler helps maximize the reuse of existing knowledge and minimize the load of manual ontology creation. The conversions between OSMX and OWL link this semantic annotation work to the rest of the semantic web community.

Upon to the time we submit this paper, this is an ongoing project. We have successfully built the ontology-based semantic annotation prototype system that is resilient and has good accuracy. There is also an online demo shows the current status of our resilient annotation tool [4]. Based on approximately 20 domains with which we have experimented, our preliminary results show that we can typically achieve close to 100% precision and recall in the simple, unified domains such as automobile sales and apartment rentals. However, in more complicated or loosely unified domains, the precision and recall for some fields falls off dramatically. For example, on the domain of obituaries we were only able to achieve about 74% precision on annotating relatives of the deceased and only about 82% recall on annotating funeral addresses. In the future, the implementation of the two-layer annotation model can improve the accuracies with the use of local structural patterns.

In the meantime, we have finished the implementation of the ontology converter, and we are preparing a paper on this topic. The implementations of the two-layer annotation model and the ontology assembler are under way, and they constitute future work we plan to accomplish.

References

[1] L. Arlotta, V. Crescenzi, G. Mecca, and P. Merialdo, “Automatic annotation of data extracted from large web sites,” *Proc. Sixth International Workshop on the Web and Databases (WebDB 2003)*, pp. 7-12, San Diego, California, June 2003.

[2] T. Berners-Lee, J. Hendler, and O. Lassila, “The Semantic Web,” *Scientific American*, vol. 36, no. 25, pp. 34-43, May 2001.

[3] V. Crescenzi, G. Mecca, and P. Merialdo, “RoadRunner: Towards Automatic Data Extraction from Large Web Sites,” *Proc. 27th International Conference on Very Large Data Bases (VLDB 2001)*, pp. 109–118, Roma, Italy, September, 2001.

[4] Simple Ontology-based Semantic Annotator Demo, Data Extraction Group, Brigham Young University, URL: http://dithers.cs.byu.edu/deg/demos /annotationdemo/simple-demo.php.

[5] Y. Ding, “Annotating Documents for The Semantic Web Using Data-Extraction Ontologies,” *PhD Dissertation Proposal*, Brigham Young University, Provo, Utah, December 2005.

[6] D.W. Embley, D.M. Campbell, Y.S. Jiang, S.W. Liddle, D.W. Lonsdale, Y.-K. Ng, and R.D. Smith, “Conceptual-model-based data extraction from multiple-record web pages,” *Data & Knowledge Engineering*, vol. 31, no. 3, pp. 227-251, November 1999.

[7] T.R. Gruber, “A translation approach to portable ontology specifications,” *Knowledge Acquisition*, vol. 5, no. 2, pp. 199-220, 1993.

[8] S. Handschuh, S. Staab, and F. Ciravegna, “S-CREAM Semi-automatic CREAtion of Metadata,” *Proc. European Conference on Knowledge Acquisition and Management (EKAW-2002)*, pp. 358–372, Madrid, Spain, October, 2002.

[9] Jena, A Semantic Web Framework for Java, URL: http://jena.sourceforge.net/.

[10] A. Kiryakov, B. Popov, I. Terziev, D. Manov, and D. Ognyanoff, “Semantic Annotation, Indexing, and Retrieval,” *Journal of Web Semantics*, vol. 2, no. 1, pp. 49–79, December 2004.

[11] N. Kushmerick, D. S. Weld, and R.B. Doorenbos, “Wrapper induction for information extraction,” *Proc. Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97)*, pp. 729–737, NAGOYA, Aichi, Japan, August, 1997.

[12] A.H.F. Laender, B.A. Ribeiro-Neto, A.S. da Silva, and J.S. Teixeira, “A brief survey of web data extraction tools,” *SIGMOD Record*, vol. 31, no. 2, pp. 84-93, June 2002.

[13] S. Tiun, R. Abdullah, and T.E. Kong, “Automatic Topic Identification Using Ontology Hierarchy,” *Proc. Second International Conference on Computational Linguistics and Intelligent Text Processing*, pp. 444–453, Mexico-City, Mexico, February, 2001.

[14] E. Rahm and P. Bernstein, “A survey of approaches to automatic schema matching,” *The VLDB Journal*, vol. 10, no. 4, pp. 334-350, December 2001.