Cross-Fertilizing Deep Web Analysis and Ontology Enrichment

Marilena Oita
INRIA Saclay – Île-de-France
Télécom ParisTech; CNRS LTCI
Paris, France
marilena.oita@telecom-paristech.fr

Antoine Amarilli
École normale supérieure
Télécom ParisTech; CNRS LTCI
Paris, France
antoine.amarilli@ens.fr

Pierre Senellart
Institut Mines–Télécom
Télécom ParisTech; CNRS LTCI
Paris, France
pierre.senellart@telecom-paristech.fr

ABSTRACT

Deep Web databases, whose content is presented as dynamically-generated Web pages hidden behind forms, have mostly been left unindexed by search engine crawlers. In order to automatically explore this mass of information, many current techniques assume the existence of domain knowledge, which is costly to create and maintain. In this article, we present a new perspective on form understanding and deep Web data acquisition that does not require any domain-specific knowledge. Unlike previous approaches, we do not perform the various steps in the process (e.g., form understanding, record identification, attribute labeling) independently but integrate them to achieve a more complete understanding of deep Web sources. Through information extraction techniques and using the form itself for validation, we reconcile input and output schemas in a labeled graph which is further aligned with a generic ontology. The impact of this alignment is threefold: first, the resulting semantic infrastructure associated with the form can assist Web crawlers when probing the form for content indexing; second, attributes of response pages are labeled by matching known ontology instances, and relations between attributes are uncovered; and third, we enrich the generic ontology with facts from the deep Web.

1. ONTOLOGIES AND THE DEEP WEB

The *deep Web* consists of dynamically-generated Web pages that are reachable by issuing queries through HTML forms. A form is a section of a document with special *control* elements (e.g., checkboxes, text inputs) and associated labels. Users generally interact with a form by modifying its controls (entering text, selecting menu items) before submitting it to a Web server for processing.

Forms are primarily designed for human beings, but they must also be understood by automated agents for various applications such as general-purpose indexing of response pages, focused indexing [13], extensional crawling strategies (e.g., Web archiving), automatic construction of ontologies [29], etc. However, most existing approaches to automatically explore and classify the deep Web crucially rely on domain knowledge [10, 12, 30] to guide form understanding. Moreover, they tend to separate the steps of form interface understanding and information extraction from result pages, although both contribute [27] to a more *authentic* vision on the backend database schema. The form interface exposes in the input schema some attributes describing the query object, while response pages present this object instantiated in Web records that outline the form output schema. In this paper, we determine a mapping between the input and output schemas which associates the data types corresponding to form elements in the input schema to instances aligned in the output schema.

A harder challenge is to understand the *semantics* of these data types and how they relate to the object of the form. The input–output schema mapping may give us hints, such as the input schema labels, but this information cannot suffice by itself. This has been addressed in related work using heuristics [26] or an assumed domain knowledge [19] which is either manually crafted or obtained by merging different form interface schemas belonging to the same domain. Domain knowledge is, however, not only hard to build and maintain, but also often restricted to a choice of popular domain topics, which may lead to biased exploration of the deep Web.

We present a new way to deal with this challenge: we initially probe the form in a domain-agnostic manner and transform the information extracted from response pages into a labeled graph. This graph is then aligned with a *general-domain* ontology, YAGO [23], using the PARIS ontology alignment system [22]. This allows us to infer the semantics of the deep Web source, to obtain new, representative query terms from YAGO for the probing of form fields, and to possibly enrich YAGO with new facts.

VLDS'12 August 31, 2012. Istanbul, Turkey.

2. RELATED WORK

Merging input schemas of deep Web interfaces has been used to acquire domain ontologies automatically [29] and perform Web database classification and query routing [3]. The main drawback of these approaches is that data integration dramatically relies on the interface schema, whose shallow features (the form structure and labels) are neither complete, nor representative enough for the actual response records [4].

To obtain response pages, the form has to be filled in and submitted first. Most approaches described in the literature are domain-specific and use dictionary instances [19]. *Domain-agnostic probing* approaches are more powerful because they do not make such assumptions and incrementally build knowledge that tends to improve the probing and the quality of response pages. However, existing domain-agnostic techniques do not aim at understanding the intensional purpose of the form, but at extensional crawling [5].

Deep Web response pages are an extremely rich source of semi-structured information. Works dealing with response pages assume the form probing mechanism understood and focus on information extraction (IE) from *Web records* [8]. Extracting the schema from response pages [15] is possible due to the structural similarity of records. Because this schema has been obtained by probing the form and analyzing the response pages, it is called the *output schema* of the form.

The data extracted from deep Web sources through IE processing is typically used to build and/or enrich ontologies [2, 21, 24], gazzetters [11] or to expand sets of entities [28]. ODE [21] in particular gets closer to our work by its holistic approach, but still needs a domain ontology built by matching different deep Web interfaces. A more important difference appears in the annotation of the extracted data from response pages using heuristic rules for label assignment, similar to [26]. Comparatively, we use PARIS alignment algorithm.

The next step is the discovery of the semantic relationships between the entity of the form and the record attributes; for this, several techniques are proposed in the literature. Traditionally, statistical and rule-based methods use the instances in a *textual context* in order to infer the relation between them [9]. Another option [20] is to match the terminology of a given term with a *known* concept using semantic resources such as DBpedia or WordNet [18]. Yet another trend is to use classifiers that can predict specific relations (e.g., *subClassOf*) given enough training and test data [6]. The closest work to ours may be [14], an approach relying on supervised learning that uses a generic ontology to infer types and relations among the data in a Web table. We deal with the more general setting of deep Web interfaces here, and we propose a fully automatic approach that does not require human supervision.

3. ENVISIONED APPROACH

We now present our vision of a holistic deep Web semantic understanding and ontology enrichment process, which is summarized in Figure 1: a Web form is analyzed and probed, record attribute values are extracted from result pages, and their types are mapped to input fields. While these steps are rather standard and we follow the well-established best practices, they have never been analyzed in a holistic manner without the assumption of domain knowledge that describes the form interface. The novelty of studying these steps in connection comes from their contribution to the formation of a labeled graph which encompasses data values of unknown types and *implicit* semantic relations. This graph is further aligned with a generic ontology for knowledge discovery using PARIS.

Form Analysis and Probing. The form interface is presented as an *input schema* which gives a prescriptive description of the object that the user can query through the form. The input schema is the ordered list of labels corresponding to form elements, possibly together with constraints and possible values (for drop-down lists and other non-textual input fields). Important data constraints or properties of the backend Web database can be discovered through well-designed probing and response page analysis. Some may be precious for a crawler that interacts with the form: Are stop words indexed? Which Boolean connectors are used (conjunctive or disjunctive)? Is the search affected by spelling errors? We perform form probing in an agnostic manner (i.e., without domain knowledge) following [5]. We try to set non-textual input elements or to fill in a textual input field with stop words or with *contextual terms* extracted from non-textual input controls (e.g., drop-down list entries) or surrounding text (e.g., indications to the user). We rely on the fact that many sites provide a generous index (i.e., a response page can be obtained by inputting a single letter). A more elaborate idea is to use AJAX auto-completion facilities.

Record Identification. If the form has been filled in correctly, we obtain a result page. Otherwise, to identify possible error pages, our method infers a characteristic XPath expression by submitting the form with a nonsense word and tracing its location in the DOM of the response page. This approach uses the fact that the nonsense word will usually be repeated in the error page to present the erroneous input to the user. If not, techniques such as those of [19] can be applied. If the probing yields a response page which does not contain the error pattern, then we determine the generic XPath location of Web records using [16].

Output Schema Construction. A way to build the output schema is to use the reflection of a given domain knowledge in response pages [25]. Another method is to perform *attribute alignment* [1] for records obtained from different pages. Since Web records represent subtrees which are structurally similar at DOM level, we extract the values of their textual leaf nodes and cluster these values based on their DOM path. The rationale is that the values found under the same record internal path are attributes of the same type. For instance, "Great Expectations" and "David Copperfield" in Figure 1 both represent literals of the *title* attribute of a *book* and have a common location pattern. We define a *record feature* as the association between a relevant record internal path and its cumulated bag of instances. The *output schema* for a response page is then defined by the ordered sequence of record features. In practice, we remove uninformative record features from the output schema by restricting ourselves to paths which contain *different* instances across various response pages.

Input and Output Schema Mapping. We align input fields of the form with record features of the result pages in the following fashion. For non-textual form elements such as drop-down lists, we check if their values do not trivially match one of the record features of the output schema. For textual form elements, we use a more elaborate idea. Due to binding patterns, query instances which appear at a certain record internal path should appear again at the same location when they are submitted in the "right" input field for this path. If we submit them in an unrelated field, however, we should obtain an error page or unsuitable results. Formally, given a record feature f of the output schema, we can see if it maps to a textual input t by filling in t with one of the initial instances of f (say i) and submitting the form. Either we obtain an error page, which means f and t should not be mapped, or we obtain a result page in which we can use f's record internal path to extract a new bag I of instances for f. In this case, we say that t and f are mapped if all instances in I are equal to i or contain it as a substring (i.e., i appears again at f's location pattern). We obtain the mapping by performing these steps for all couples (f,t).

Most of the time, the input–output schemas do not match exactly. The attributes that cannot be matched are usually explicit in the input schema (e.g., given by non-textual inputs, like drop-down lists), or only present in the output schema (e.g., the price of a book).

Graph Generation. We represent the data extracted from the Web records as RDF triples [17], in the following manner:

1. each record is represented as an *entity*;
2. all records are of the same *class*, stated using *rdf:type*;
3. the attribute values of records are viewed as *literals*;
4. each record links to its attribute values through the relation (i.e., *predicate*) that corresponds to the record internal path of the attribute type in the response page;

Since the triples form a labeled directed graph, it is possible to add much more information to the representation, provided that we have the means to extract it. An idea would be to include a more detailed representation of a record by following the hyperlinks that we identify in its attribute values and replacing them in the original response page with the DOM tree of the linked page. In this way, the extraction can be done on a more complete representation of the backend database. We can also add complementary data from various sources, e.g., Web services or other Web forms belonging to the same domain.

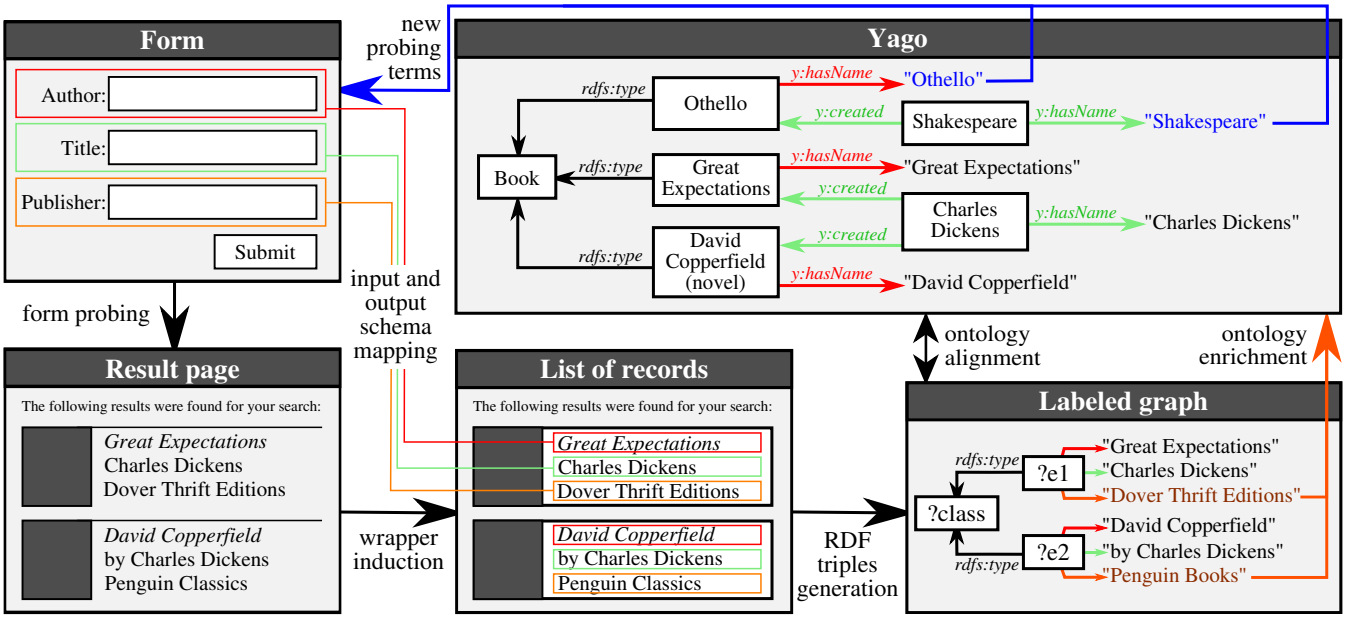

Figure 1: Overview of the envisioned approach

Ontology Alignment. The ontology that we compile from the result pages is aligned with a large reference ontology. We use YAGO [23], though our approach can be applied to any reference ontology. We use PARIS [22] to perform the ontology alignment. Unlike most other systems, PARIS is able to align both entities and relations. It does so by bootstrapping an alignment from the matching literals and propagating evidence based on *relation functionalities*. Through the alignment, we discover the class of entities, the meaning of record attributes and the actual relation that exists between them. Two main adaptations are needed to use PARIS in the deep Web data alignment process. First, extracted literals usually differ from those of YAGO because of alternate spellings or surrounding stop words. A typical case on Amazon is the addition of related terms, e.g., "Hamlet (French Edition)" instead of just "Hamlet". To mitigate this problem we normalize the literals, eliminate punctuation and stop words. Pattern identification in the data values of the same type could increase the probability of extracting cleaner values. We are working on a way to index YAGO literals in a manner that is resilient to the small differences we wish to ignore. A promising approach to do this is *shingling* [7].

Second, an entity-to-literal relation in the labeled graph may not necessarily correspond to a single edge in the reference ontology, but to a sequence of edges. This amounts to a *join* of the involved relations; a typical case in our prototype is the "author" attribute which is linked to a record entity through a two-step YAGO path "y:created y:hasPreferredName". To ensure that the alignment with joins, typically costly, can be performed in practice, we limit the maximal length of joins. A consequence is that PARIS will explore a smaller fraction of YAGO in the search for relations relevant to the data of our labeled graph. In addition to the use of record attribute values as literals, PARIS could use the form labels (through the input–output mappings) to guide the alignment and favor YAGO relations with a similar name. Some record instances do not align with any literal of the ontology. The cause is that they represent information which is unknown to YAGO.

Form Understanding and Ontology Enrichment. Ontology alignment gives us knowledge about the data types, the domains and ranges of record attributes, and their relation to the object of the form (in our case, a book). The propagation of this knowledge to the input schema through the input–output mapping (for the form elements that have been successfully mapped) results in a better understanding of the form interface. On the one hand, we can infer that a given field of the Amazon advanced search form expects author names, and leverage YAGO to obtain representative author names to fill in the form. This is useful in intensional or extensional automatic crawl strategies of deep Web sources. On the other hand, we can generate *new* result pages for which data location patterns are already known and enrich YAGO through the alignment that we once determined.

There are three main possibilities to enrich the ontology. First, we can add to the ontology the *instances* that did not align. For instance, we can use the Amazon book search results to add to YAGO the books for which it has no coverage. Second, we can add *facts* (triples) that were missing in YAGO. Third, we can add the *relation types* that did not align. For instance, we can add information about the publisher of a book to YAGO. This latter direction is more challenging, because we need to determine if the relation types contain valuable information. One safe way to deal with this *relevance problem* is to require attribute values to be mapped to a form element in the input schema. We can then use the *label* of the element to annotate them.

4. PRELIMINARY EXPERIMENTS

We have prototyped this approach for the Amazon book advanced search form[1]. Obviously, we cannot claim any statistical significance of the results we report here, but we believe that the approach, because it is generic, can be successfully applied to other sources of the deep Web.

Our preliminary implementation performed agnostic probing of the form, wrapper induction, and mapping of input–output schemas. It generated a labeled graph with 93 entities and 10 relation types out of which 2 (title and author) are recognized by YAGO. Literals underwent a semi-heuristic normalization process (lowercasing, removal of parenthesized substrings). We then replaced each extracted

[1] `http://www.amazon.com/gp/browse.html?node=241582011`

literal with a similar literal in YAGO, if the similarity (in terms of the number of common 2-grams) was higher than an arbitrary threshold.

We aligned this graph with YAGO by running PARIS for 15 iterations, i.e., a run time of 7 minutes (most of it was spent loading YAGO, the proper computation took 20 seconds). Though the vast majority of the books from the dataset were not present in YAGO, the 6 entity alignments with best confidence were books that had been correctly aligned through their title and author. To limit the effect of noise on relation alignment, we recomputed relation alignments on the entity alignments with highest confidence; the system was thus able to properly align the title and author relations with "y:hasPreferredName" and "y:created y:hasPreferredName", respectively. These relations were associated to the record internal paths of the output schema attributes and propagated to form input fields.

5. DISCUSSION

Our vision is that of a holistic system for deep Web understanding and ontology enrichment, where each stage of the process (form analysis, information extraction, schema matching, ontology alignment, etc.) would benefit of every other part. This is an ambitious project, but our current prototype already exhibits promising results.

Many challenges remain to be tackled: resilience to outliers and noise resulting from imperfect literal matching and information extraction; proper management of the confidence in the results of each automatic task, especially when they are used as the input of another task; identification of new relation types of interest among those extracted from a Web source; integration of the information contained in several different deep Web sources of the same domain.

Acknowledgments

We acknowledge Fabian Suchanek for initial discussions on this topic. The research has been funded by the European Union's seventh framework programme, in the setting of the European Research Council grant Webdam, agreement 226513, and the FP7 grant ARCOMEM, agreement 270239.

6. REFERENCES

[1] M. Alvarez, A. Pan, J. Raposo, F. Bellas, and F. Cacheda. Extracting lists of data records from semi-structured Web pages. *Data and Knowledge Engineering*, 64(2), 2008.

[2] Y. J. An, S. A. Chun, K.-C. Huang, and J. Geller. Enriching ontology for deep Web search. In *Proc. DEXA*, 2008.

[3] Y. J. An, J. Geller, Y.-T. Wu, and S. A. Chun. Semantic deep Web: automatic attribute extraction from the deep Web data sources. In *Proc. SAC*, 2007.

[4] R. Balakrishnan and S. Kambhampati. SourceRank: Relevance and trust assessment for deep Web sources based on inter-source agreement. In *Proc. WWW*, 2011.

[5] L. Barbosa and J. Freire. Siphoning hidden-Web data through keyword-based interfaces. *J. Information and Data Management*, 1(1), 2004.

[6] E. Beisswanger. Exploiting relation extraction for ontology alignment. In *Proc. ISWC*, 2010.

[7] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. Syntactic clustering of the Web. *Computer Networks*, 29(8-13), 1997.

[8] J. Caverlee, L. Liu, and D. Buttler. Probe, cluster, and discover: Focused extraction of QA-pagelets from the deep Web. In *Proc. ICDE*, 2004.

[9] P. Cimiano, G. Ladwig, and S. Staab. Gimme' the context: Context-driven automatic semantic annotation with C-PANKOW. In *Proc. WWW*, 2005.

[10] T. Furche, G. Gottlob, G. Grasso, X. Guo, G. Orsi, and C. Schallhart. Real understanding of real estate forms. In *Proc. WIMS*, 2011.

[11] T. Furche, G. Grasso, G. Orsi, C. Schallhart, and C. Wang. Automatically learning gazetteers from the deep Web. In *Proc. WWW*, 2012.

[12] B. He, K. C.-C. Chang, and J. Han. Discovering complex matchings across Web query interfaces: A correlation mining approach. In *Proc. KDD*, 2004.

[13] S. Kumar, A. K. Yadav, R. Bharti, and R. Choudhary. Accurate and efficient crawling the deep Web: Surfacing hidden value. *International J. Computer Science and Information Security*, 9(5), 2011.

[14] G. Limaye, S. Sarawagi, and S. Chakrabarti. Annotating and searching Web tables using entities, types and relationships. *Proc. VLDB*, 3(1), 2010.

[15] S. Nestorov, S. Abiteboul, and R. Motwani. Extracting schema from semistructured data. In *ACM International Conference on Management of Data (SIGMOD 1998)*, 1998.

[16] M. Oita and P. Senellart. Own work undergoing double-blind reviewing, 2012.

[17] Resource Description Framework (RDF): Concepts and abstract syntax. W3C Recommendation. `http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/`.

[18] C. Reynaud and B. Safar. Exploiting WordNet as background knowledge. In *Proc. ISWC Ontology Matching (OM-07) Workshop*, 2007.

[19] P. Senellart, A. Mittal, D. Muschick, R. Gilleron, and M. Tommasi. Automatic wrapper induction from hidden-Web sources with domain knowledge. In *Proc. WIDM*, 2008.

[20] G. Stoilos, G. B. Stamou, and S. D. Kollias. A string metric for ontology alignment. In *Proc. ISWC*, 2005.

[21] W. Su, J. Wang, and F. H. Lochovsky. ODE: Ontology-assisted data extraction. *ACM Trans. Database Syst.*, 34(2), 2009.

[22] F. M. Suchanek, S. Abiteboul, and P. Senellart. PARIS: Probabilistic alignment of relations, instances, and schema. *Proc. VLDB Endow.*, 5(3), 2011.

[23] F. M. Suchanek, G. Kasneci, and G. Weikum. YAGO: A core of semantic knowledge unifying WordNet and Wikipedia. In *Proc. WWW*, 2007.

[24] M. Thiam, N. Pernelle, and N. Bennacer. Contextual and metadata-based approach for the semantic annotation of heterogeneous documents. In *Proc. SeMMA*, 2008.

[25] N. Tiezheng, Y. Ge, S. Derong, K. Yue, and L. Wei. Extracting result schema based on query instances in the deep Web. *Wuhan University J. Natural Sciences*, 12(5), 2007.

[26] J. Wang and F. H. Lochovsky. Data extraction and label assignment for Web databases. In *Proc. WWW*, 2003.

[27] J. Wang, J.-R. Wen, F. Lochovsky, and W.-Y. Ma. Instance-based schema matching for Web databases by domain-specific query probing. In *Proc. VLDB*, 2004.

[28] R. C. Wang and W. W. Cohen. Language-independent set expansion of named entities using the Web. In *Proc. ICDM*, 2007.

[29] W. Wu, A. Doan, C. Yu, and W. Meng. Bootstrapping domain ontology for semantic Web services from source Web sites. In *Proc. VLDB Workshop on Technologies for E-Services*, 2005.

[30] X. Yuan, H. Zhang, Z.-Y. Yang, and Y. Wen. Understanding the search interfaces of the deep Web based on domain model. In *Proc. ICIS*, 2009.