

Extraction Techniques for Mining Services from Web Sources

Hasan Davulcu, Saikat Mukherjee, I.V. Ramakrishnan

Dept. of Computer Science
SUNY Stony Brook
Stony Brook, NY 11794, USA
{davulcu, saikat, ram}@cs.sunysb.edu

Abstract

The Web has established itself as the dominant medium for doing electronic commerce. Consequently the number of service providers, both large and small, advertising their services on the web continues to proliferate. In this paper we describe new extraction algorithms for mining service directories from web pages. We develop a novel propagation technique for identifying and accumulating all of the attributes related to a service entity in a web page. We provide experimental results of the effectiveness of our extraction techniques by mining a database of veterinarian service providers from web sources.

1. Introduction

A number of service providers operate their own web sites promoting their services at length while others are merely listed in a referral site. Aggregating all of the providers into a queryable service directory makes it easy for customers to locate the best suited services for their needs. An attractive solution is to create the service directory by mining the web for service providers. Such a solution has several merits. Firstly, it does not need any explicit participation by the service provider and hence is scalable. Secondly, since the process is automated and repeatable the content can always be kept current. Finally, the same process can be readily adapted to different domains. We characterize services by an on-

tology consisting of a taxonomy of service concepts, their associated attributes (such as names and addresses) and type descriptions for the attributes. In addition the ontology also associates an attribute identifier function with each attribute. Applying the function to a web page will locate all the occurrences of the attribute in that page. Each web page is parsed into a DOM (Document Object Model) tree and the identifier functions, specified in the ontology for locating occurrences of the attributes in the page, are applied. The problem is to group all the attributes corresponding to each service provider. In this paper we describe a novel ontology-directed propagation technique for identifying and accumulating all of the attributes related to each service entity in a web page. By using a concept of *scoring* and *conflict resolution* to prevent erroneous associations, our algorithm groups the attributes related to each service provider in a web document.

2. Ontology Directed Mining

A service ontology is characterized by a set of service concepts C , the IS-A relationship between concepts, a set of single-valued attributes A_s , a set of multi-valued attributes A_m , a function A that associates a set of attributes with a concept, and the extraction function $Attr_id$ associated with an attribute. Each entity is *uniquely* identified by a set of single-valued attributes. We call any such set as a *key*. e.g. for service providers two possi-

ble keys are $\{street, city\}$ and $\{street, zip\}$. Let T be the DOM tree of a page. $Parent(n)$ denotes the parent of node n and $children(n)$ denotes all its children. We are interested in identifying subtrees in T in which no single-valued attribute occurs more than once. We use the notion of a *mark* for doing so. Whenever $mark(n)$ is ϕ it means that there exists more than one occurrence of a single valued attribute in its subtree. Specifically, the subtrees rooted at a node can be merged as long as no single-valued attribute occurs in more than one subtree. A maximally marked node n is an internal node such that $mark(Parent(n))$ is ϕ . Let $\sigma(n)$ denote the concatenation of the text strings associated with the leaf nodes of the subtree rooted at n ; $Attr$ be the set of attributes of the concept c ; $\{k_1, \dots, k_n\}$ be the attributes that constitute the key of c ; $R(a_1, \dots, a_n)$ denote the tuple of attributes associated with an entity. We will extract one tuple from a home page and several such tuples from a referral page. We use $score(n)$ to denote $|mark(n)|$. Algorithm Extract takes as input the tree of the page and the set of attribute names of the concept c . It outputs either a single tuple containing the values of the attributes if it is a home page or a set of tuples if it is a referral page. Extract_Home_Page takes as input the set of attribute names whose values are to be extracted and the set of maximally marked nodes in the document tree. The intuition behind this algorithm is that the maximally marked node contains the key attributes associated with the service entity and any node in the document tree might contain occurrences of the multi-valued attributes. For referral pages we have to extract the attributes of several entities. The main problem here is associating the extracted attributes with their corresponding entities. We use the notion of a *conflicting* set that will be used in making such an association. Let Γ be as defined in Algorithm Extract. Observe that Γ is an ordered set of nodes. Let $\langle m_1, m_2, \dots, m_q \rangle$ denote the nodes in this ordered sequence. We say that Γ is *conflict-free* whenever $\exists i, m_i, m_{i+1} \in \Gamma$ such that $mark(m_i) \cup mark(m_{i+1})$ is consistent. Γ is not conflict-free if all pairs of consecutive nodes are mutually incon-

sistent. Observe that whenever Γ is not conflict-free then any maximally marked node represents a single entity. All we need to do is simply pick the attributes in it and create the tuple for that entity. If this is not the case then attributes of an entity may be distributed across neighboring nodes. In that case we will have to detect the boundaries separating each entity. In addition even if Γ is conflict-free the leaf nodes in it will have conflicts and we will have to detect boundaries separating the attributes of entities in the text string at the leaf node.

Algorithm Extract ($T, Attr$)

begin

1. **forall** nodes $n \in T$ **do**
 2. $mark(n)$
 3. **end**
 4. Let $\Gamma = \{ \text{maximally marked nodes} \} \cup \{ \text{leaf nodes marked } \phi \}$
 5. **if** $\exists m_i, m_j \in \Gamma \wedge \{Attr_id(k_1)(\sigma(m_i)), \dots, Attr_id(k_n)(\sigma(m_i))\} \neq \{Attr_id(k_1)(\sigma(m_j)), \dots, Attr_id(k_n)(\sigma(m_j))\}$ **then**
 6. T is a referral page
 7. **else**
 8. T is a home page
 9. **endif**
 10. **if** T is a home page **then**
 11. $R = \text{Extract_Home_Page}(Attr, \Gamma)$
 12. **elseif** T is a referral page **then**
 13. $\{R_1, \dots, R_n\} = \text{Extract_Referral_Page}(Attr, \Gamma)$
 14. **end**
- end**

Algorithm Extract_Home_Page ($Attr, \Gamma$)

begin

1. pick the node n in Γ with the maximum $score$
 2. **forall** $a_i \in Attr \wedge a_i \in A_s$ **do**
 3. $R[a_i] = Attr_id(a_i)(\sigma(n))$
 4. **end**
 5. **forall** $a_i \in Attr \wedge a_i \in A_m$ **do**
 6. $R[a_i] = \bigcup_{m_i \in \Gamma} Attr_id(a_i)(\sigma(m_i))$
 7. **end**
 8. return R
- end**

3. Service Directory Mining System Implementation

The system consists of three main components: an *acquisition* component, a *classification* component and an *extraction* component. The acquisition component retrieves HTML pages from the web that are likely to be relevant for the intended

domain of services. This is done by doing a keyword search for the service with a web search engine. The search engine returns a number of urls pointing to pages that match the keywords. All of these web pages are fetched by the acquisition component.

```

Algorithm Extract_Referral_Page (Attr,  $\Gamma$ )
begin
1. if  $\Gamma$  is not conflict-free then
2.   forall  $m_i \in \Gamma$  do
3.     if  $m_i$  is a leaf  $\wedge$   $mark(m_i) = \phi$  then
4.        $\{R_1, \dots, R_n\} =$  Boundary_Detection(Attr,  $m_i$ )
5.     else
6.       forall  $a_j \in Attr$  do
7.          $R_i[a_j] = Attr\_id(a_i)(\sigma(m_i))$ 
8.       end
9.     end
10.  end
11. else
12.    $\{R_1, \dots, R_n\} =$  Boundary_Detection(Attr,  $\Gamma$ )
13. end
14. return  $\{R_1, \dots, R_n\}$ 
end

```

The classification component filters the retrieved pages into a set of web pages that the classifier has judged to be actually relevant for the intended service. For training the classifier one hand picks examples of web pages that are relevant to the intended domain of service. These serve as the positive examples. One must also choose pages unrelated to the service as the negative examples. The extraction component, driven by the service ontology, does unsupervised extraction of attribute values from classified pages and builds the services directory. The mining system described above was used to create a service directory of veterinarians. For veterinarian service providers, we built an ontology consisting of two concepts: the *Service Provider* concept at the root, and the concept *Veterinarian*. The *Service Provider* concept consists of the attributes *service provider name*, *street*, *city*, *state*, *zip*, *phone*, *email*, *url*. The concept *Veterinarian* consists of the attribute *vet's name*. In addition, *Veterinarian* inherits all the attributes of *Service Provider*. The attributes *phone*, *email* and *vet's name* are multi-valued while the other attributes are single-valued. Regular expressions were used to identify *phone num-*

ber, *email*, *state* and *zip* in a page. Rules were used to identify *street*, *vet's name* and *service provider name*. We trained the classifier by picking 371 veterinarian home/referral and 303 non-veterinarian web pages. The web search yielded 13,691 distinct pages. The trained classifier was used to select the relevant pages from them. Classification identified 3400 pages as positive or relevant to the veterinarian domain. The extraction algorithm was run on all of the 3400 pages. We provide experimental results of this case study below. From these 3400 positively classified pages, 950 were identified as home pages of veterinarian service providers while 1900 were identified as referral pages by the extraction algorithm. The $\langle city, state, zip \rangle$ triple was used as the key. There were about 550 pages with missing zips that were discarded. Table 1 shows the statistics of different attribute values collected for these 2850 pages.

Attribute	Number of Records	
	Home Pages	Referral Pages
City	950	12300
State	950	12300
Zip	950	12300
Street	950	12300
Phone	806	10930
Email	938	780
Doctor Name	711	3930
Hospital Name	856	12300
URL	950	-

Table 1. Number of records extracted for each attribute for home and referral pages

For comparison we retrieved a total of 650 email addresses and 990 urls of veterinarian service providers listed in <http://vetquest.com>, <http://vetworld.com> and the yellow pages in <http://www.superpages.com>. In contrast our mining system yielded 1718 emails and 950 urls of home pages.

4. Related Work

Extraction from semi-structured sources by *wrapper generation* methods is an extensively researched topic [10, 2, 8, 1, 11, 9, 12]. Wrappers have several disadvantages: (a) a significant amount of work is required to generate the rules, and (b) they are document specific as they rely on the syntactic relationship between HTML tags and the attribute value for proper extraction. Wrappers are therefore brittle to changes in the document structure. In contrast our extraction algorithms are independent of any page specific relationships between HTML tags and attribute values. All that is needed is an ontology for the intended service domain. With such an ontology extraction from any document relevant to that domain can be carried out. Information extraction techniques as embodied in [5, 4, 3] use supervised machine learning methods. Observe that the creation of the ontology is the only supervised step in our approach. Regardless of the ontology, our algorithm for associating attribute values to their corresponding service entity in a web document is unsupervised. Supervised machine learning techniques that will handle multiple entities in a document are as yet not known. Query languages for semi-structured documents constitutes an important class of extraction techniques. They all assume that the document schema is known a priori. In our approach we make no such assumptions. The work that comes closest to ours is [7, 6]. In this work the extraction problem is formulated as one of detecting boundaries between records and hence is applicable to only referral pages. The boundary detection is based on several different heuristics which can result in incorrectly identifying the entities.

5. Conclusion

Engineering an ontology is largely dictated by the the complexity of the identifier functions. In the case study reported in this paper we used regular expressions as the identifier extraction functions. Incorporating complex rules in the ontol-

ogy can result in improving the precision of extraction considerably. This is an area worthy of investigation. Our requirement for the existence of a key to distinguish between home and referral pages resulted in misclassifying some referral pages. Relaxing this requirement is a topic of future work.

References

- [1] N. Ashish and C. Knoblock. Wrapper generation for semi-structured internet sources. *ACM SIGMOD Record*, 26(4):8–15, 1997.
- [2] P. Atzeni and G. Mecca. Cut & paste. In *ACM Symposium on Principles of Database Systems*, pages 117–121, Arizona, June 1997. ACM.
- [3] M. E. Califf and R. J. Mooney. Relational learning of pattern-match rules for information extraction. In *Working Notes of AAAI Spring Symposium on Applying Machine Learning to Discourse Processing*, pages 6–11, Menlo Park, CA, 1998. AAAI Press.
- [4] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. M. Mitchell, K. Nigam, and S. Slattery. Learning to construct knowledge bases from the world wide web. *Artificial Intelligence*, 118(1-2):69–113, 2000.
- [5] M. Craven, D. DiPasquo, D. Freitag, A. K. McCallum, T. M. Mitchell, K. Nigam, and S. Slattery. Learning to extract symbolic knowledge from the World Wide Web. In *Proceedings of AAAI-98, 15th Conference of the American Association for Artificial Intelligence*, pages 509–516, Madison, US, 1998. AAAI Press, Menlo Park, US.
- [6] D. Embley, Y. Jiang, and Y.-K. Ng. Record-boundary discovery in Web documents. In *ACM SIGMOD Conference on Management of Data*, pages 467–478. ACM, 1999.
- [7] D. W. Embley, D. M. Campbell, R. D. Smith, and S. W. Liddle. Ontology-based extraction and structuring of information from data-rich unstructured documents. In *Proceedings of the International Conference on Knowledge Management*. ACM, 1998.
- [8] J. Hammer, H. Garcia-Molina, S. Nestorov, R. Yerneni, M. M. Breunig, and V. Vassalos. Template-based wrappers in the tsimmis system. In *ACM SIGMOD Conference on Management of Data*, pages 532–535. ACM, 1997.
- [9] N. Kushmerick, D. S. Weld, and R. B. Doorenbos. Wrapper induction for information extraction. In *Intl. Joint Conference on Artificial Intelligence*, volume 1, pages 729–737, Nagoya, Japan, 1997.
- [10] I. Muslea. Extraction patterns for information extraction tasks: A survey. In *AAAI-99 Workshop on Machine Learning for Information Extraction*.
- [11] M. Perkowski, R. Doorenbos, O. Etzioni, and D. Weld. Learning to understand information on the internet: An example-based approach. *Journal of Intelligent Information Systems*, 8(2):133–153, March 1997.
- [12] S. Soderland. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34(1-3):233–272, 1999.