Automatically Extracting Form Labels

Hoa Nguyen, Eun Yong Kang, Juliana Freire

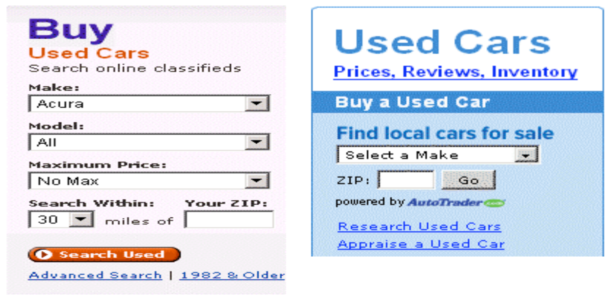
School of Computing, University of Utah, Salt Lake City, UT, USA
{thanhhoa, ekang, juliana}@cs.utah.edu

***Abstract*— We describe a machine-learning-based approach for extracting attribute labels from Web form interfaces. Having these labels is a requirement for several techniques that attempt to retrieve and integrate data that reside in online databases and that are hidden behind form interfaces, including schema matching and clustering, and hidden-Web crawlers. Whereas previous approaches to this problem have relied on heuristics and manually specified extraction rules, our technique makes use of learning classifiers to identify form labels. Our preliminary experiments show this approach is promising and has high accuracy.**

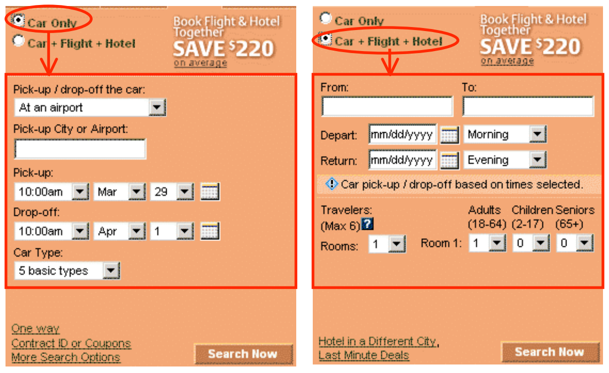
I. Introduction

The problem of retrieving and integrating information that resides in online databases has received a lot of attention in both the research and industrial communities [1], [2], [3], [4]. This interest is driven both by the quality of the information and the growing number of online databases—it is estimated that there are several million online databases [5]. As most online databases can only be accessed through Web form interfaces, to automatically retrieve and integrate their contents, their form interfaces must be understood [6], [7], [8], [9]. The ability to identify the labels for different form elements is key to many important applications. For example, to integrate multiple online databases, correspondences among the elements in different forms need to be identified [2], [10]. Labels are also needed for retrieving data hidden behind the form interfaces: in an integration system, queries over the global schema must be translated into sets of attribute-value assignments over forms of individual databases [11]; and hidden-Web crawlers must derive valid attribute-value assignments to siphon the database contents [12], [13].

In this paper, we propose a new approach for automatically parsing and extracting element labels of form interfaces that are designed primarily for human consumption. Although the HTML standard provides a `label` attribute to associate descriptive information with individual form elements, it is not widely used. Instead, text representing attribute names are interspersed with the HTML markup. As Figure 1 illustrates, several different layouts are possible—labels can be placed in many different positions: on top, in the bottom, to the left of, to the right of, and even inside a form element. For example, in Figure 1(a), while the label `Make` is above its corresponding selection list in the form on the left, in the form on the right it is actually one of the values in the selection list. Figure 1(b) shows a dynamic form whose content and layout change based on which radio button is selected: if the `Car only` button is selected, all labels are place on top of



(a) Forms in the *Auto* domain



(b) Dynamic form in the *Airfare* domain

Fig. 1. Examples of web search interfaces illustrating the variability in layout design within and across different domains.

the form elements, and if the `Car + Flight + Travel` is selected, some of the labels are placed to the left. A given label may also correspond to multiple form elements, and some elements may have no associated label. The label `Depart` in Figure 1(b) (right) is associated both with the text field for inputting the date and with the selection list for specifying the time. Last, but not least, labels may also be placed outside the form tags. This wide variation in form design layouts and in the nesting relationship between form elements and labels makes the problem of automatically identifying element-label mappings particularly challenging.

Related Work. Based on the assumption that there are common patterns for label placement, previous approaches to label extraction relied either on heuristics (*e.g.,* based on the textual layout of the page) to *guess* the appropriate label for a given form attribute [14], [13] or on manually-specified extraction rules [6]. These approaches, however, require substantial human input—heuristics and rules must be manually crafted.

The form analysis component of HiWE, the hidden-Web crawler proposed by Raghavan and Molina, uses a layout engine to identify labels that are close to an element. They then generate a series of candidate mappings which are ranked using a set of pre-defined rules (*e.g.,* position, number of words, font size). Zhang et al. [6] cast the problem of label extraction as parsing. They developed a novel grammar that allows users to declaratively specify extraction rules which capture both topological patterns (*e.g.,* alignment and adjacency) and proximity information. They deal with the problem of ambiguities in label-element assignments as well as with incomplete grammars, by encoding not only patterns but also their precedence in the grammar. Instead of relying on the visual layout of form elements, He et al. [14] examine the HTML structure of the form. They capture the textual layout of labels and elements as an interface expression (IEXP). The IEXP of a given search interface consists of three basic items t, e and |, where t denotes a label/text, e denote a form element, and | denotes a row delimiter, which corresponds to HTML tags such as `<p>` and `<br>`. They define rules which determine the association between label/text and elements based on their positions in the IEXP expression.

Ours is the first work to make use of learning-based techniques to automatically infer the extraction patterns. This automation makes our approach more scalable, reducing the need for human input both for the creation and maintenance of the extraction rules. Our experiments show that this automated approach is effective and able to obtain high extraction accuracy. Furthermore, unlike previous works which are restricted to static forms, our approach is able to handle forms which use client-side scripting (*e.g.,* JavaScript) to dynamically render its elements, which are widely used on the Web.

II. Solution Overview

Given a Web form, our goal is to determine a mapping between a form element and its corresponding textual description. Instead of relying on manually specified rules and heuristics, our approach uses classifiers to identify element labels by learning structural patterns of forms. The use of a *learning-based approach makes our approach adaptable*: to handle new form layout patterns, the forms just need to be added to the training sets for the classifiers.

The label extraction process has three phases which are described below: candidate mapping generation, classification, and reconciliation. Given a Web form, we generate a series of candidate mappings between form elements and textual labels in the neighborhood of the elements. The *candidate mapping generator must identify a set of possible labels for each form element*. It utilizes JavaScript-aware parser and an HTML rendering engine, this makes it possible for the extractor to handle dynamic forms. It also extracts a set of features that are later used by the classifiers, such as for example: the form element types, font, size, and location.

As discussed above, there is a wide variation in how forms are designed, even within a single online database domain. Thus, an important challenge is how to construct accurate classifiers. To simplify the learning task, we have designed a *hierarchical framework that performs classification in two steps*: the initial step uses a coarse (high-recall) classifier whose goal is to eliminate mappings that are clearly incorrect; the second step uses a more specific (high-precision) classifier to select mappings that are the most likely to be correct. These classifiers are described in detail in [15], where we also show that: the combination of classifiers leads to accuracies higher than single-classifier configurations; and general classifiers can be built which obtain high accuracy across multiple online database domains.

Because forms may use different and conflicting layout choices, the classifiers may be unable to correctly identify the label for certain elements. Take for example the forms in Figure 1(a). Both belong to the auto domain, but whereas in the form on the left labels are placed on the top of elements, labels are place inside and to the left of elements in the form on the right. This leads both to ambiguities (*i.e.,* multiple labels assigned to an element) and dangling elements (*i.e.,* elements with no associated labels). To deal with this problem, we apply a *mapping reconciliation* step, where matches obtained by the classification process are used to help identify the correct labels for the unmatched form elements. Previous works have observed that the set of terms used for form elements is usually small—form interfaces tend to use a small set of terms to represent attributes within a domain [10]. The intuition behind the effectiveness of our mapping reconciliation is that terms which occur frequently in mappings across different forms are more likely to be element labels than terms that appear less frequently. Thus, when multiple candidate labels exist for an element, we choose the label which has the highest frequency and that has not been used in the form.

III. Experiments

We have evaluated our approach over a dataset containing forms that belong to four online database domains: "Airfare", "Auto", "Books" and "Movies". This dataset consists of 2,884 web forms automatically collected by the Form-Focused Crawler (FFC) [3], [16].

Effectiveness of Our Approach. Figure 2 shows F-measure scores obtained by our approach both for the combined classifiers used in isolation (CC) and together with the mapping reconciliation step (CC+MR). The F-measure is the harmonic mean between precision and recall [17]. A high F-measure means that both recall and precision have high values—a perfect classification would result in an F-measure with value equal 1. The scores for CC+MR vary between 83% and 95%, indicating that the approach is effective: it is able to automatically learn the layout patterns and accurately identify label-element mappings. This figure also shows that the mapping reconciliation component always leads to increased F-measure scores—the improvements vary between 4.5% and 14.4%.

The lowest score obtained by the classification process was for the *Airfare* domain (0.76), and the highest was for the *Movie* domain (0.91). A possible explanation for this difference lies in the layout heterogeneity for *Airfare*. Examining the

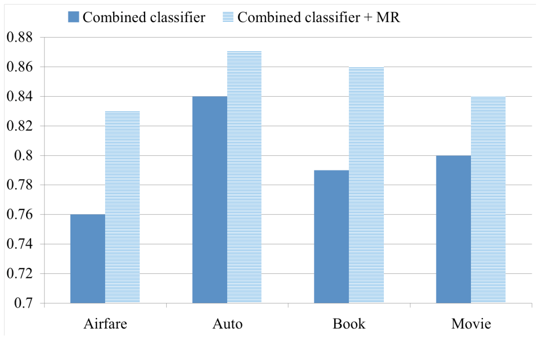


Fig. 2. Effectiveness of our approach. This figure shows the F-measure scores for the classification process run in isolation, and of the combination of the classification with the mapping reconciliation step.

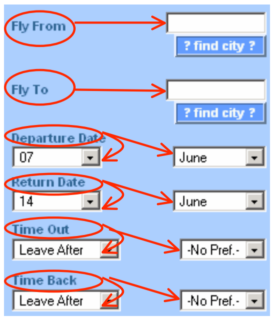


Fig. 3. Variation in the distance between labels and elements

structure of the forms in these two domains, we observe that in *Airfare*, there is a large variation in distance between labels and elements, even within a single form. This is illustrated in the form shown Figure 3. In this form, whereas some labels are placed very close to the corresponding elements (*e.g.,* `Departure date`), others are relatively further (*e.g.,* `Fly from`). In contrast, for *Movie*, the labels are more consistently placed close to the corresponding elements. Consequently, it is harder to correctly predict labels for the more heterogeneous *Airfare* domain than for the *Movie* domain.

Domain-Specific versus Generic Classifiers. The previous results relied on the construction of combined classifiers for each domain—four combined classifiers were constructed and each was trained using forms from a single domain. An interesting question is whether it is possible to construct a generic classifier that obtains high accuracy for forms across different domains.

We built the generic configuration as follows. First, we trained a combined classifier using a mix of forms from all four domains to obtain the generic classifier. The generic training set is generated by randomly selecting 10% forms from each domain. In the testing phase, we ran that classifier and mapping reconciliation process on each domain to measure the resulting accuracy. The domain-specific classifier combination always outperforms the generic, obtaining accuracies that are between 1.8% and 12.3% higher than the generic configuration. This shows that some layout patterns are indeed present in multiple domains, and that a generic classifier can be effective. Thus, the cost of training the combined classifier can be amortized by using it on multiple domains.

IV. Conclusion

We have described a learning-based approach for automatically extracting labels of Web form elements. Our preliminary experiments, using over two thousand forms automatically retrieved by a focused crawler, show that this approach is effective and obtains high extraction accuracy. There are several avenues we plan to explore in future work. Besides performing a detailed comparison between our approach and previous works on label extraction, we will investigate the use of active learning to incrementally improve the accuracy of our classifiers.

Acknowledgment

This work was funded by the National Science Foundation under grants IIS-0713637, IIS-0534628, OCE-0424602, IIS-0513692, and CNS-0524096.

References

[1] "Google Base," http://base.google.com/.

[2] W. Wu, C. Yu, A. Doan, and W. Meng, "An Interactive Clustering-based Approach to Integrating Source Query interfaces on the Deep Web," in *Proceedings of ACM SIGMOD*, 2004, pp. 95–106.

[3] L. Barbosa and J. Freire, "Searching for Hidden-Web Databases," in *Proceedings of WebDB*, 2005, pp. 1–6.

[4] K. C.-C. Chang, B. He, and Z. Zhang, "Toward Large-Scale Integration: Building a MetaQuerier over Databases on the Web," in *Proceedings of CIDR*, 2005, pp. 44–55.

[5] J. Madhavan, S. Cohen, X. L. Dong, A. Y. Halevy, S. R. Jeffery, D. Ko, and C. Yu, "Web-scale data integration: You can afford to pay as you go," in *Proceedings of CIDR*, 2007, pp. 342–350.

[6] Z. Zhang, B. He, and K. Chang, "Understanding web query interfaces: best-effort parsing with hidden syntax," in *Proceedings of ACM SIGMOD*, 2004.

[7] K. Chang, B. He, and Z. Zhang, "Metaquerier over the deep web: Shallow integration across holistic sources," in *In Proceedings of the VLDB Workshop on Information Integration on the Web*, 2004.

[8] W. Wu, A. Doan, and C. Yu, "Merging interface schemas on the deep web via clustering aggregation," in *Proceedings of IEEE ICDM*, 2005, pp. 801–805.

[9] W. Wu, A. Doan, and C. T. Yu, "Webiq: Learning from the web to match deep-web query interfaces," in *Proceedings of IEEE ICDE*, 2006, p. 44.

[10] B. He and K. C.-C. Chang, "Statistical Schema Matching across Web Query Interfaces," in *Proceedings of ACM SIGMOD*, 2003, pp. 217–228.

[11] P. Wu, J.-R. Wen, H. Liu, and W.-Y. Ma, "Query selection techniques for efficient crawling of structured web sources," in *Proceedings of IEEE ICDE*, 2006.

[12] L. Barbosa and J. Freire, "Siphoning Hidden-Web Data through Keyword-Based Interfaces," in *Proceedings of SBBD*, 2004, pp. 309–321.

[13] S. Raghavan and H. Garcia-Molina, "Crawling the Hidden Web," in *Proceedings of VLDB*, 2001, pp. 129–138.

[14] H. He, W. Meng, C. Yu, and Z. Wu, "Automatic extraction of web search interfaces for interface schema integration," in *Proceedings of WWW*, 2004.

[15] H. Nguyen, E. Y. Kang, and J. Freire, "Automatically extracting form labels," University of Utah, Tech. Rep., 2007, extended version.

[16] L. Barbosa and J. Freire, "An adaptive crawler for locating hidden-web entry points," in *Proceedings of WWW*, 2007, pp. 441–450.

[17] B. Larsen and C. Aone, "Fast and effective text mining using linear-time document clustering." in *Proceedings of KDD*, 1999, pp. 16–22.