To appear in the proceedings of the *10th European Conference on Machine Learning.*

First-Order Learning for Web Mining*

Mark Craven, Seán Slattery and Kamal Nigam

School of Computer Science, Carnegie Mellon University
Pittsburgh, PA 15213-3891, USA
e-mail: ⟨firstname⟩.⟨lastname⟩@cs.cmu.edu

Abstract. We present compelling evidence that the World Wide Web is a domain in which applications can benefit from using first-order learning methods, since the graph structure inherent in hypertext naturally lends itself to a relational representation. We demonstrate strong advantages for two applications – learning classifiers for Web pages, and learning rules to discover relations among pages.

1 Introduction

In recent years, there has been a large body of research centered around the topic of learning first-order representations. Although these representations can succinctly represent a much larger class of concepts than propositional representations, to date there have been only a few problem domains in which first-order representations have demonstrated a decided advantage over propositional representations. The graph-like structure provided by pages on the World Wide Web is one domain that seems natural for first-order representation, yet has not been previously studied in this context. Cohen [1] has used first-order methods for text classification, but the focus was on finding relations between words rather than between documents. The lower half of Figure 1 illustrates the notion of the Web as a directed graph where pages correspond to the nodes in the graph and hyperlinks correspond to edges. Using this representation, we address two types of learning tasks: *learning definitions of page classes*, and *learning definitions of relations between pages*. In contrast to related efforts on similar Web tasks, our work focuses on learning concepts which represent relational generalizations of the inherent graph structure.

Our work on these two learning tasks has been conducted as part of a larger effort aimed at developing methods for automatically constructing knowledge bases by extracting information from the Web [2]. Given an ontology defining classes and relations of interest, such as that shown in the top half of Figure 1, along with training examples consisting of labeled Web pages, the system learns a set of information extractors for the classes and relations in the ontology, and then populates a knowledge base by exploring the Web. The task of recognizing class instances can be framed as a page-classification task. For example, we can

* This research was supported in part by the Darpa HPKB program under contract F30602-97-1-0215.

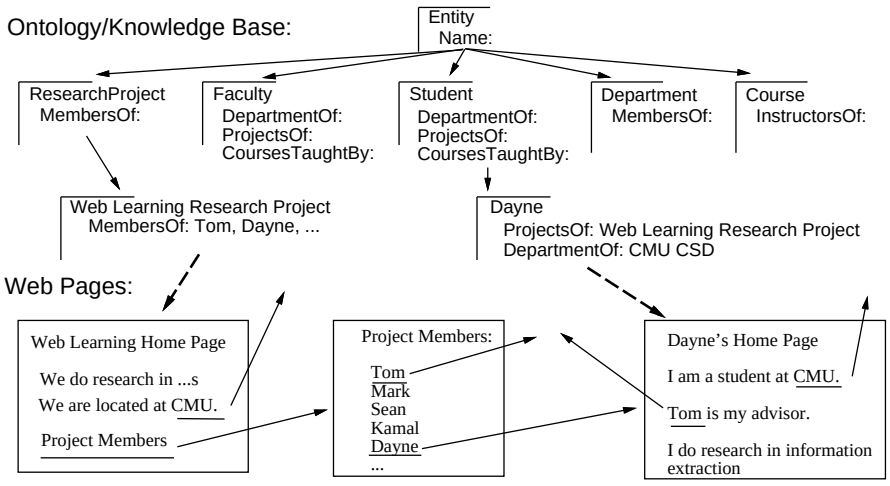

Fig. 1. The top part of the figure shows part of an ontology that defines page classes and relations between pages. The bottom part shows the Web represented as a directed graph, which forms examples of the classes and relations in the ontology.

extract instances of the Faculty class by learning to recognize the home pages of faculty members. Similarly, we can identify relations that exist among objects by recognizing prototypical patterns of hyperlink connectivity among pages. Consider the lower half of Figure 1, which shows an example of an instance of the MembersOfProject relation: Dayne is a member of the Web Learning project.

The applicability of these two learning tasks (learning page classes and learning page relations) extends beyond the setting of building knowledge bases from the Web. A variety of applications, including information filtering systems and browsing assistants, have used trainable page classifiers. As an example of learning relations among pages, consider the task of automatically extracting job listings by starting from company Web sites and finding the "employment opportunities" page. This task can be framed as one of learning a concept definition that specifies search-control rules for navigating the Web. In general, there are many potential applications of such search-control rules for finding a particular Web resource from a given class of starting points.

2 Learning First-Order Definitions of Page Classes

This section presents four classification problems and reports that a first-order learner can perform better than a more traditional document classifier which ignores document relationships. We first present two classification algorithms, a conventional text learning algorithm (Naive Bayes) which ignores document relationships, and a first-order learner (FOIL) which can use such information. More complete details of the data set, algorithms and experiments can be found elsewhere [2].

Table 1. Recall (R) and precision (P) percentages on each binary classification task using Naive Bayes, FOIL with words only, and FOIL with words and links.

	Student		Course		Faculty		Project	
method	R	P	R	P	R	P	R	P
Naive Bayes	51.4 %	42.6 %	46.4 %	28.2 %	23.0 %	16.8 %	1.3 %	3.6 %
FOIL (words)	25.3	50.3	34.5	44.3	43.3	48.4	6.1	10.0
FOIL (words & links)	73.0	70.2	39.5	53.8	58.2	61.2	10.2	21.3

For these experiments, we use a data set assembled for our research in extracting knowledge bases from the Web. It contains 4,127 pages and 10,945 hyperlinks drawn from the Web sites of four computer science departments. Each page was labeled as belonging to one of the classes Department, Faculty, Student, ResearchProject, Course, or Other. We also labeled relation instances consisting of pairs of pages. For example, each instance of the InstructorsOfCourse relation consists of a Course home page and a Person home page. Our data set of relation instances comprises 251 InstructorsOfCourse instances, 392 MembersOfProject instances, and 748 DepartmentOfPerson instances.

As a representative conventional text classifier, we use the Naive Bayes classifier [4]. To classify a document with n words $(w_1, w_2, \ldots, w_n)$ into one of a set of classes C we simply calculate:

$$\arg\max_{c_j \in C} \Pr(c_j) \prod_{i=1}^{n} \Pr(w_i|c_j) \quad \text{where} \quad \Pr(w_i|c_j) = \frac{N(w_i, c_j) + 1}{N(c_j) + T} \ .$$

$N(w_i, c_j)$ is the number of times word w_i appears in training set examples from class c_j, $N(c_j)$ is the total number of words in the training set for class c_j and T is the total number of unique words in the corpus.

We use version 4.2 of Quinlan's FOIL [5] system for learning first-order clauses, and two types of background relations to describe the data:

• has_*word*(Page) : This set of relations indicates that *word* occurs on Page. To reduce the number of predicates, standard text categorization techniques were used, leaving between 500 and 800 word predicates per training set.

• link_to(Page, Page) : This relation represents hyperlinks between web pages in our corpus. The first argument is the page on which the link occurs, and the second is the page to which it is linked.

Using leave-one-university-out cross validation we build and test binary classifiers for four classes. We assess the classifiers using recall (R) and precision (P) defined as:

$$R = \frac{\text{\# correct positive examples}}{\text{\# of positive examples}} \ , \ P = \frac{\text{\# correct positive examples}}{\text{\# of positive predictions}} \ .$$

To test the value of a first-order representation, we run FOIL using two different sets of background relations, one which has only the word predicates (words) and one which has both the word and link predicates (words & links). Table 1

Table 2. Some rules induced by FOIL with good test set performance. Also shown are the number of positive and negative test set examples covered by each rule.

student_page(A) :- link_to(B,A), has_*michael*(B), has_*graduat*(B), has_*richard*(B), not(has_*depart*(B)).	151 ⊕	27 ⊖
course_page(A) :- has_*instructor*(A), not(has_*good*(A)), link_to(A,B), not(link_to(B,_1)), has_*assign*(B).	31 ⊕	3 ⊖
faculty_page(A) :- has_*professor*(A), has_*ph*(A), link_to(B,A), has_*faculti*(B).	18 ⊕	3 ⊖

shows the precision and recall results averaged over the four test sets. Using descriptions of the words and links, FOIL outperformed the other methods on these binary classification tasks. The power of using a relational description is evident from the difference in performance between the two FOIL runs.

Table 2 shows three of the rules induced by FOIL with high test-set accuracy. After analyzing the results, we found that the B variable in the sample student_page rule binds to directory pages of graduate students (the literals test for common names and the stemmed version of the word *graduate*). In effect, the rule states that A is a student home page if it is linked to by a directory page of graduate students. On average, FOIL using words and links induced 20 clauses for Student, 19 clauses for Course, 12 for Faculty and 7 for ResearchProject.

The results of this experiment lead us to believe that first-order representations and algorithms are well suited to Web page classification tasks. They use hyperlink information easily and can outperform traditional text-classification approaches which have no means to use such information effectively.

3 Learning First-Order Definitions of Page Relations

In this section, we discuss the task of learning to recognize relations of interest that exist among pages. An assumption underlying our approach is that relations among pages are represented by *hyperlink paths* in the Web. Thus, the learning task is to characterize the prototypical paths of the target relations. We learn definitions for the following target relations from the ontology shown in Figure 1: department_of_person(Page, Page), instructors_of_course(Page, Page), and members_of_project(Page, Page). In addition to the positive instances for these relations, our training sets include approximately 300,000 negative examples.

The problem representation we use consists of the following background relations:

• *class*(Page) : For each *class* from the previous section, the corresponding relation lists the pages that represent instances of *class*. These instances are determined using actual classes for pages in the training set and predicted classes for pages in the test set.

• link_to(Hyperlink, Page, Page) : This relation represents the hyperlinks that interconnect the pages in the data set.

• has_*word*(Hyperlink) : This set of relations indicates the words that are found in the anchor (i.e., underlined) text of each hyperlink. The vocabulary for this

Table 3. Recall (R) and precision (P) results for the relation learning tasks. The symbols $\diamond$ and $\star$ precede each result that is uniformly superior (i.e. better than on all four test sets) to the same measure for FOIL, and m-estimate FOIL, respectively.

	DepartmentOfPerson		InstructorsOfCourse		MembersOfProject	
method	R	P	R	P	R	P
FOIL	26.9%	97.1%	53.8%	84.9%	32.1%	80.8%
FOIL w/m-estimates	26.9	97.1	59.8	89.3	$\diamond$ 44.9	83.8
PATH-MCP	$\diamond\star$ 78.5	98.0	$\diamond\star$ 64.9	89.1	$\diamond\star$ 49.7	82.6

set of relations includes about 350 words for each training set.

• all_words_capitalized(Hyperlink) : The instances of this relation are hyperlinks in which the words in the anchor text all start with a capital letter.

• has_alphanumeric_word(Hyperlink) : The instances of this relation are hyperlinks which contain a word with both alphabetic and numeric characters.

• has_neighborhood_*word*(Hyperlink) : This set of relations indicates the words that are found in the "neighborhood" of each hyperlink. A neighborhood is the paragraph, list item, table entry, title or heading in which a hyperlink is contained. The vocabulary for this set includes 200 words.

The algorithm we use for learning page relations augments FOIL's hill-climbing search with a deterministic variant of Richards and Mooney's *relational pathfinding* method [6]. The basic idea underlying this method is that a relational problem domain can be thought of as a graph in which the nodes are the domain's constants and the edges correspond to relations which hold among constants. The algorithm tries to find a small number of prototypical paths in this graph that connect the arguments of the target relation. Richards and Mooney's algorithm is nondeterministic in that it randomly selects an uncovered positive instance to use as a seed. We have developed a deterministic variant (PATH-MCP) that finds the *most common path* among the uncovered positive instances. Once such a path is found, an initial clause is formed from the relations that constitute the path, and the clause is further refined by a hill-climbing search. Like Džeroski and Bratko's m-FOIL [3], PATH-MCP uses m-estimates of a clause's error to guide its construction. We have found that this evaluation function results in fewer, more general clauses than FOIL's information gain measure.

We evaluate our approach using the same four-fold cross-validation methodology we used in Section 2. Table 3 shows precision and recall results for learning the three target relations using basic FOIL, FOIL with m-estimates, and PATH-MCP. The results in this table indicate several interesting conclusions. First, all of the methods are able to learn accurate (i.e., high-precision) rules for all three tasks. The primary differences are in terms of coverage. A second interesting result is that the PATH-MCP method achieves higher levels of recall than the non-pathfinding methods. This result is due to the fact that both versions of FOIL fail to learn any clauses describing paths of more than one hyperlink, whereas PATH-MCP is able to learn clauses characterizing multiple-hyperlink paths.

Table 4. Two of the clauses learned by PATH-MCP. Also shown are the number of positive and negative test-set examples covered by each clause.

department_of_person(A,B)	:- person(A), department(B), link_to(C,A,D), link_to(E,D,F), link_to(G,F,B), has_neighborhood_*james*(E).	371 ⊕ 4 ⊖
members_of_project(A,B)	:- research_project(A), person(B), link_to(C,A,D), link_to(E,D,B), has_neighborhood_*people*(C).	18 ⊕ 0 ⊖

Finally, Table 4 shows two of the interesting clauses learned by PATH-MCP. Both of them describe relations represented by multiple hyperlinks, and the DepartmentOfPerson clause is similar to the Student clause shown in Section 2 in that it has learned to exploit directory pages of people (referenced by the variable E) in order to find the people associated with a given department. On average, PATH-MCP learned 3 clauses for the DepartmentOfPerson relation, 7 clauses for InstructorsOfCourse, and 5 clauses for MembersOfProject.

4 Conclusions

We have presented experiments in two real-world learning problems that involve mining information from the Web, an interesting testbed for first-order learning. Our experiments in learning page classifiers show that, in some cases, first-order learning algorithms learn definitions that have higher accuracy than commonly used statistical text classifiers. When learning definitions of page relations, we demonstrate that first-order learning algorithms can learn accurate, non-trivial definitions that necessarily involve a relational representation.

Finally, we note that although the background relations used in our experiments represent the graph structure of hypertext, we could also use first-order representations that describe the internal layout of Web pages. In future work, we plan to investigate the value of learning with this additional relational structure.

References

1. W. W. Cohen. Learning to classify English text with ILP methods. In L. De Raedt, editor, *Advances in Inductive Logic Programming*. IOS Press, 1995.
2. M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. Learning to extract symbolic knowledge from the World Wide Web. Technical report, Department of Computer Science, Carnegie Mellon Univ., 1998.
3. S. Džeroski and I. Bratko. Handling noise in inductive logic programming. In S.H. Muggleton and K. Furukawa, editors, *Proc. of the 2nd International Workshop on Inductive Logic Programming*.
4. T. Mitchell. *Machine Learning*. McGraw Hill, 1997.
5. J. R. Quinlan and R. M. Cameron-Jones. FOIL: A midterm report. In *Proc. of the European Conf. on Machine Learning*, pages 3–20, Vienna, Austria, 1993.
6. B. Richards and R. Mooney. Learning relations by pathfinding. In *Proc. of the 10th Nat. Conf. on Artificial Intelligence*, pages 50–55, San Jose, CA, 1992. AAAI Press.