

A Machine Learning Framework for Automatically Annotating Web Pages with Simple HTML Ontology Extension (SHOE)

QingFeng Lin¹, Stephen D. Scott², and Sharad C. Seth²

¹J.D. Edwards & Company, One Technology Way, Denver, CO 80237, USA

E-mail: qingfeng_lin@jdedwards.com

²Dept. of Computer Science, Ferguson 115, University of Nebraska, Lincoln, NE 68588-0115, USA

E-mail: {[sscott](mailto:sscott@cse.unl.edu), [seth](mailto:seth@cse.unl.edu)}@cse.unl.edu

Abstract

With enormous amounts of information injected into the Internet every second, manual maintenance of the knowledge base on the Internet is a hopeless task. A reasonable remedy for this problem is to create a “machine understandable” Internet. To achieve this, Heflin et al. proposed an HTML-based knowledge representation language called Simple HTML Ontology Extension (SHOE). SHOE can be used in many application domains, but it requires users to manually annotate the web pages. To overcome the shortages of SHOE, we created a machine learning framework called AutoSHOE for automatically annotating web pages with SHOE annotations. With this framework, users can easily collect SHOE-annotated pages as training data, experiment with different feature selection methods and learning algorithms to find the best approach for learning a particular ontology, and automatically annotate new web pages with trained classifiers and rule sets. In addition, AutoSHOE allows new feature selectors and learners to be easily plugged into the system and run anywhere through the web. We present the AutoSHOE architecture and then discuss experimental results of our proof-of-concept design.

1 Introduction and Motivation

It is well known that getting a computer to understand an HTML document is very difficult, if not impossible. This is because HTML is primarily a formatting tool for graphical displays intended to be read by humans. HTML does not allow other processing capabilities. In order to help an intelligent agent to “understand” the knowledge on web pages, Heflin et al. [1] proposed an HTML-based knowledge representation language called Simple HTML Ontology Extension (SHOE). SHOE is a small extension of HTML. It allows web page authors to annotate their web documents with respect to one or more predefined *ontologies*. An ontology defines the *categories (or classes)* and the *relations* between the categories. A web page can subscribe to one or more ontologies, declare data categories, and make assertions about those categories under the relations defined in the ontologies. SHOE makes intelligent agent software on the web possible by annotating HTML using XML-style notation. As a first step toward creating a machine-understandable Internet, SHOE is useful in many ways [2]. However, using SHOE requires humans to manually annotate the web pages. This approach is tedious and labor-intensive. Moreover, SHOE is of no use without such annotations and the SHOE annotations depend entirely on the ontology that the web page author used.

To overcome these shortages, we propose automatic annotation of web pages with machine learning techniques. Machine learning techniques have been successfully applied to information extraction, text document classification, relation rule learning and many other information processing areas [3]. However, machine-learning systems for ontologies are difficult to build for several reasons: (a) They require the designer to have background knowledge in machine learning; (b) training a learner on an ontology requires time consuming steps of gathering large amounts of labeled training data, selecting features, translating features

into the learner's file format, and training the learners; and (c) many different feature selection methods must be tried in order to find one that works best for a specific ontology. On the other hand, once training is done, annotating new pages is relatively easy and cheap. *AutoSHOE* bypasses the above difficulties by obtaining trained learners through the Internet to annotate unlabeled web pages.

AutoSHOE is a machine learning framework that simplifies the training process and the sharing of training data and trained learners. It integrates machine learning systems to learn SHOE ontologies in a seamless way. With this framework, the users can collect online SHOE-annotated pages as training data, experiment with different feature selection methods and learning algorithms to find the best approach for learning a particular ontology, and automatically annotate new web pages with SHOE's annotations. As a framework, AutoSHOE is highly extensible, sharable and customizable. AutoSHOE can be accessed at <http://autoshoe.unl.edu> and a more detailed description is available from Lin [4].

2 Machine Learning Approach for SHOE

A well-defined learning system includes a task, training data and a performance measure. In AutoSHOE, our training data will be SHOE-annotated web pages and our task will be to automatically label a new web page with SHOE's annotation tags. In other words, AutoSHOE takes two inputs as training data: an ontology that specifies the classes and relations of interest and the web pages that are the instances of the ontology. AutoSHOE will output a *classifier* that can classify new web pages and *rule sets* that can deduce the relations among the new pages.

2.1 Learning Classification Rules for SHOE Ontologies

Learning SHOE's classification rules in our system is equivalent to looking for a classifier that can label new web pages with predefined ontology categories. In order to find such a classifier, we need a set of labeled instances and induction algorithms. In Figure 1, we see that AutoSHOE collects the labeled instances by first parsing the *category* tags in SHOE-annotated documents. The attribute *name* in the *category* tag provides the label for an instance. The attribute *for* in the *category* tag gives the URL of the HTML document. This HTML document will be transformed into attribute-value pairs by three steps: first the *feature text* (the piece of text that reflects the contents of the page, such as body text, title/header text etc) will be extracted from an HTML document, then *feature keywords* will be extracted from text, finally the text will be represented as a *feature vector* consisting of keywords. This feature vector along with the label will be combined together as an *instance* and inserted into the database. Later, the labeled instances will be extracted from the database as training data for an *induction algorithm*. This induction algorithm will produce a *classifier* as well as the *performance measure* of this classifier. These outputs will be inserted back to the database for further use.

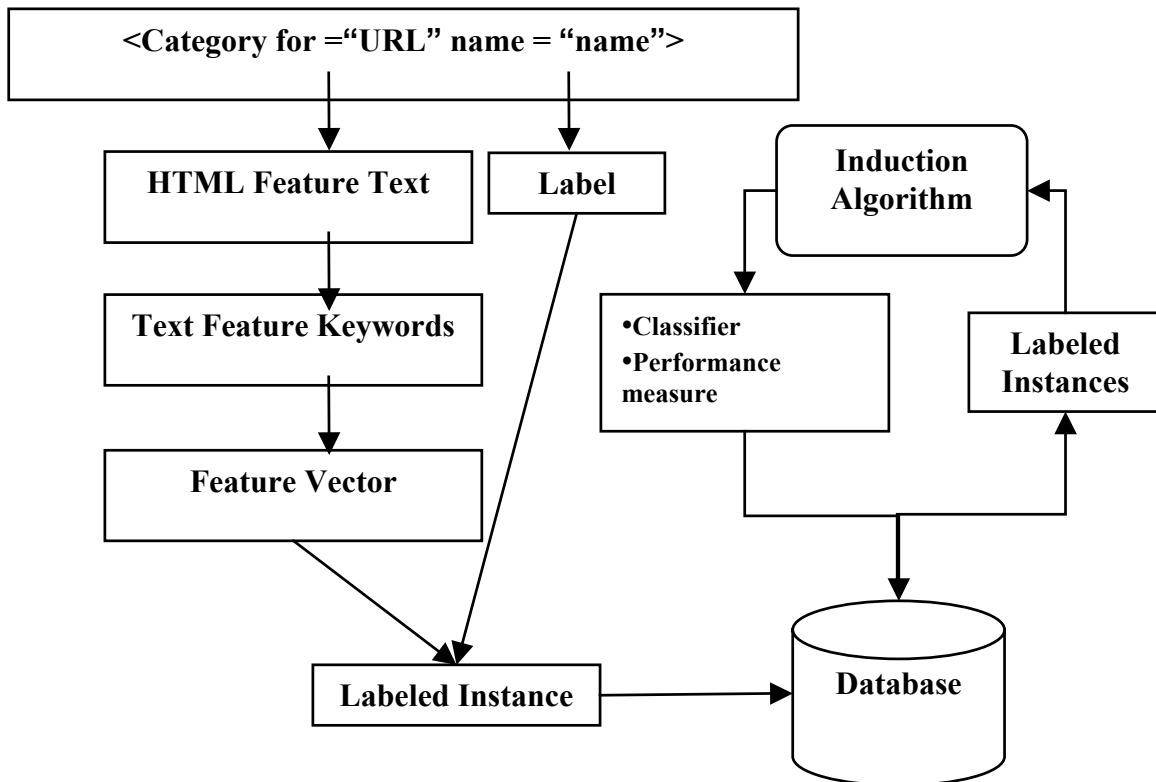


Figure 1: Process of Learning SHOE's Categorization Rules

2.2 Learning SHOE's Relation Rules

The problem of Learning SHOE's relation rules can be divided into two sub-problems. When the relation rules describe the relationship between categories, it can be viewed as a binary relation learning problem. When the relation rules describe a relationship between a category and other data, such as a number, date, or string (e.g. a string representing an address), it can be viewed as a text field extraction problem. In this section, we will focus on how to learn the binary relations that describe relationship between pairs of web pages.

SHOE's binary relation can be learned by a *first-order logic* learner such as FOIL [5]. In order to let the first-order rule learner produce a definition of the target relation, we need positive tuples and negative tuples of the target relation and background knowledge. In Figure 2, AutoSHOE gets the positive tuples and background knowledge by parsing SHOE's *relation* tags and gets the negative tuples by randomly enumerating two pages that do not belong to the positive tuples.

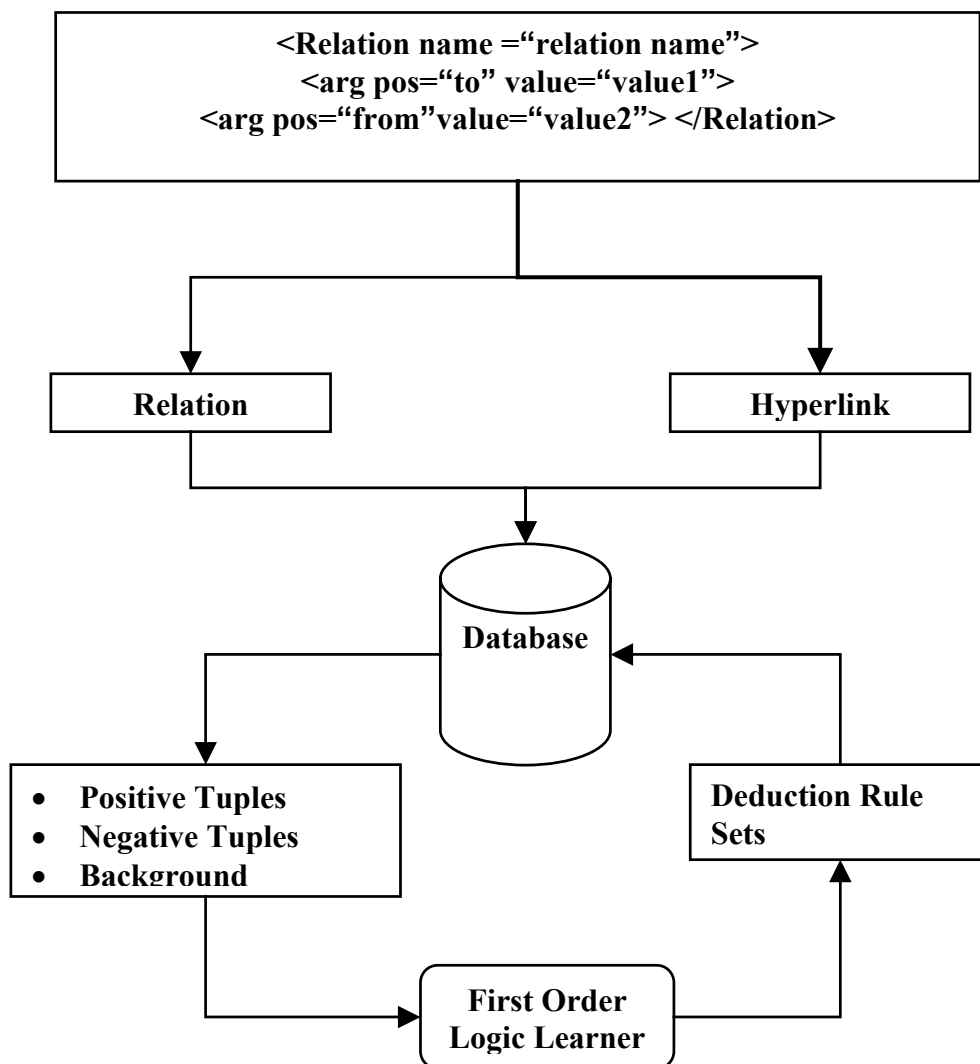


Figure 2: Process of Learning SHOE's Relation Rules

2.3 Annotating New Web Pages

After the training is done, we can use the learned classifiers and rule sets to annotate new web pages (Figure 3). The annotation process is similar to training. First we collect a set of new HTML documents that we wish to label with respect to the ontology that we trained our classifier for. Then we follow the same feature selection procedure to represent the HTML document as a feature vector. This feature vector will be used as an unlabeled instance. We can also parse the hyperlinks in the web pages, which will be used as the background relations to deduce the target relation. The unlabeled instances and the hyperlinks will be inserted into the database. In order to annotate the web pages with SHOE's classification rules, we send the unlabeled instances from the database to the classifier. The classifier then labels these instances. In order to annotate the web pages with SHOE's relation rules, we extract the rule sets and the background relations from the database and send them to a deduction program such as Prolog. The deduction program will discover the target relation according to the rule sets and the background relations.

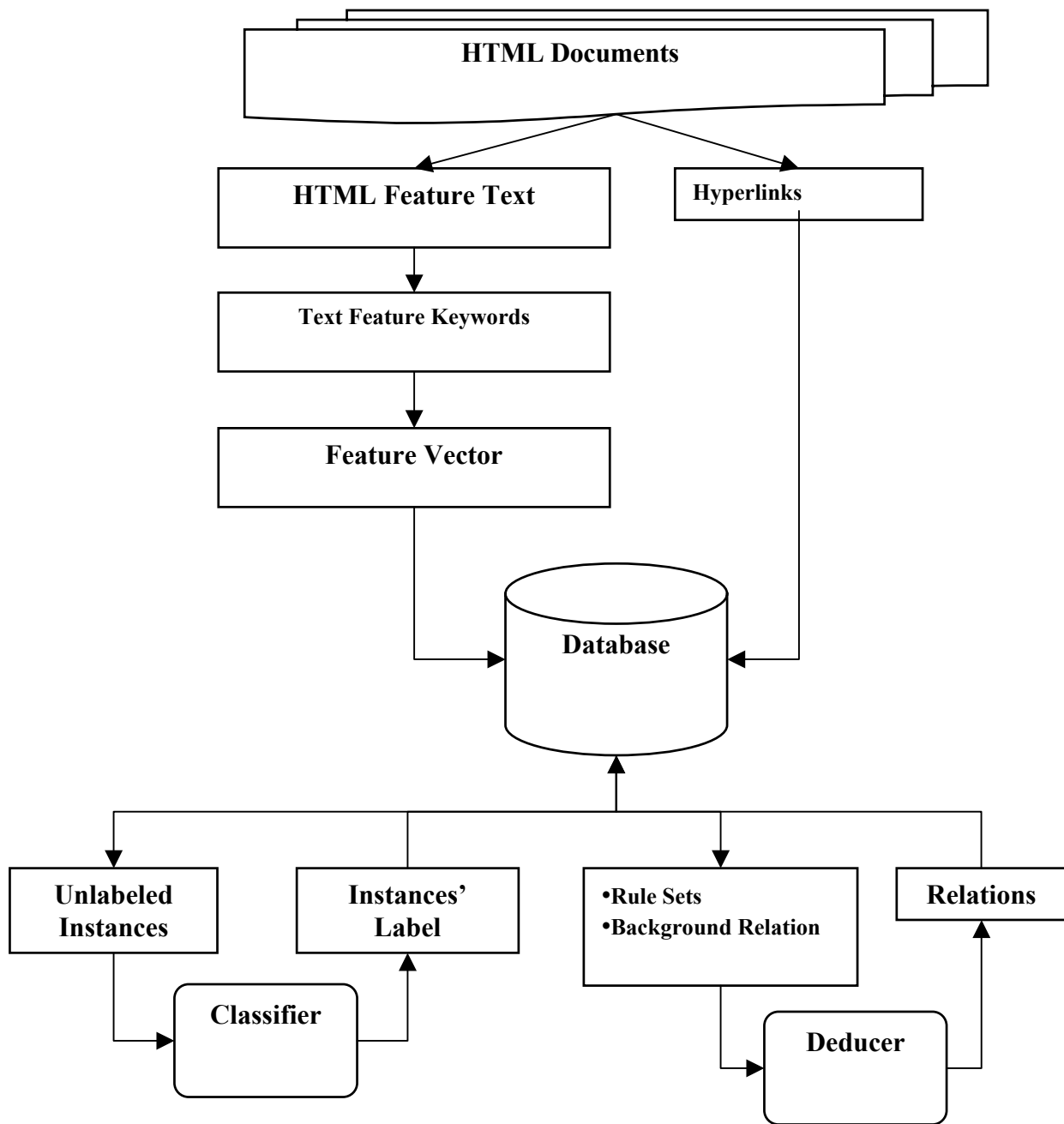


Figure 3: Web Page Annotation

2.4 AutoSHOE Architecture

AutoSHOE is a three-tiered architecture (Figure 4). The first layer is the presentation layer, which is a set of HTML and Java Server Pages. The end users use the presentation layer to collect data, train classifiers, and annotate new web pages. The middle layer is the tools layer, which is a set of Java Objects. These tools are responsible for searching, collecting, filtering, storing and maintaining data. On the back end are the third party machine learning algorithms, which are used for feature selection and learning SHOE's classification and relation rules. These systems can be written in any language and can run on any server. They plug into the AutoSHOE framework by using system adapters. These adapters talk to the broker in AutoSHOE and map the training data to its format.

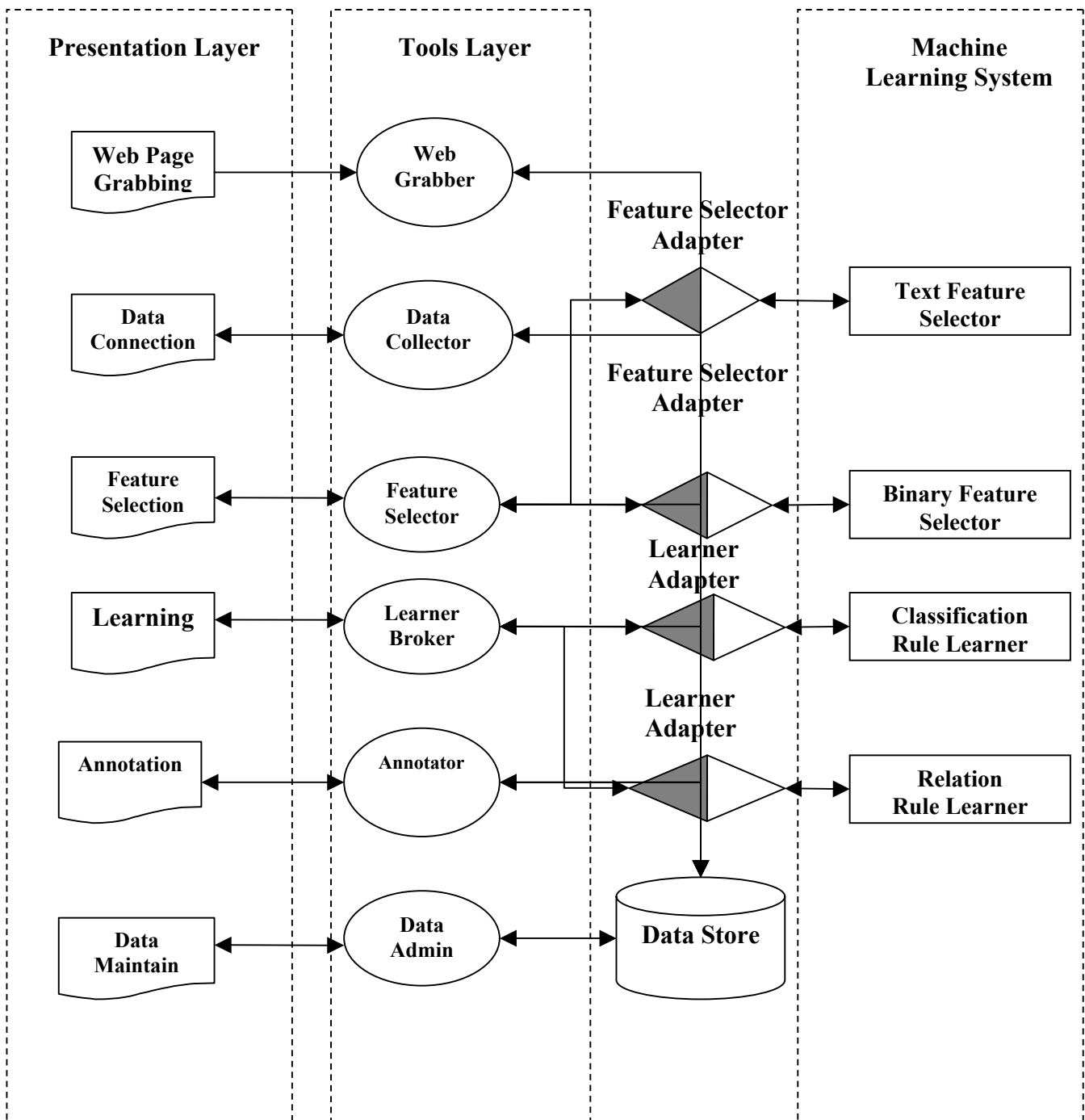


Figure 4: System Architecture of AutoSHOE

3 A Case Study: Learning cs-depart-ontology

As a proof of concept, we used this framework to set up an experiment to learn the computer science department ontology (cs-depart-ontology), as defined by Heflin et al. We first used our framework to collect the SHOE-annotated web pages from four universities (University of Maryland at College Park, University of Washington, University of Texas at Austin, and University of Wisconsin-Madison) as training data. Then we plugged in three different learning systems: Rainbow [6], MLC++ [7] and FOIL6 [8]. *Rainbow* is a software package that performs statistical text classification. It classifies the text using learning algorithms such as naïve Bayes and expectation maximization (EM). MLC++ is a machine learning C++ library that provides many commonly used machine-learning algorithms, such as C4.5, ID3, instance-

based learning, and naïve-Bayes. FOIL6 is a rule learning system that produces Horn clauses from relational data.

In order to learn the classification rules of the cs-depart-ontology, we first applied several HTML and text feature selection methods to preprocess the raw data. Then we used Rainbow's naïve Bayes algorithm and MLC++'s implementations of ID3 and k-nearest-neighbor to find a classifier for the web pages according to the categories defined by cs-depart-ontology. Our experiments show that a very accurate classifier can be produced by extracting the body text from the HTML, then using the text feature selector to prune out the terms that occur in fewer than six documents, then using the rest of text to train the naive-Bayes algorithm.

In order to learn the relation rules of the cs-depart-ontology, we first parsed the hyperlinks among the web pages, then used FOIL to learn the relation rule `teacherOf(Page,Page)` and `subOrganization(Page,Page)`. The learned rule set is as follows.

```
cs.teacherOf(A,B) :- cs.course(B), cs.faculty(A), linkTo(C,A,B).
cs_subOrganization(A,B) :- cs_ResearchGroup(A), cs_Department(B), linkTo(C,A,B).
cs_subOrganization(A,B) :- cs_ResearchGroup(A), cs_University(B), linkTo(C,A,B).
```

At the end of the experiment, we used the trained classifier and learned rules set to automatically annotate the web pages of the Computer Science Department of the University of Nebraska-Lincoln. The results are in Table 1.

Table 1: Confusion Matrix for Automatic Annotation of CS-UNL Web Pages

	Faculty	Student	Course	ResearchGroup	Total
Faculty	10	6	0	0	62.5%
Student	2	66	2	0	94.28%
Course	0	2	26	0	92.85%
ReseachGroup	0	1	0	0	0%

As we can see, the results are good in that we attain over 90% accuracy for the classes *Student* and *Course*. The class *Faculty* has a low accuracy rate due to a small data set available for training. Also, for some faculty web pages the context is too simple: for example, some pages only contain a resume, which confuses the classifier. Finally, since the class *ResearchGroup* has only one test page, evaluation of the classification accuracy for this class is meaningless.

4 Conclusion

We presented an online machine learning center that can automatically grab SHOE-annotated pages as training data and use machine learning techniques to learn classifiers and rule sets for a SHOE-defined ontology. As the middleware between the complex machine learning systems and the end users, AutoSHOE provides a uniform and easy to use environment so that even inexperienced users can use this framework to learn an ontology and annotate new pages. These labeled pages can then be manually verified and used to train new classifiers. As a framework, AutoSHOE is highly extensible: it allows new learning algorithms and feature selection algorithms to be dynamically added into the system. AutoSHOE is highly sharable: it allows the training data, feature selectors, learners to be allocated anywhere on the web, and the

training results can be accessed from anywhere through web. AutoSHOE is also highly customizable: it allows users to tailor the interface. Finally, our AutoSHOE prototype successfully proved the concept of an online machine learning center.

Acknowledgments

The authors thank the reviewers for their helpful comments. This work was supported in part by NSF grant CCR-9877080 with matching funds from CCIS and a Layman Foundation grant. QingFeng Lin performed this work at the University of Nebraska.

References

- [1] Jeff Heflin, James Hendler, and Sean Luke, SHOE: A knowledge representation language for Internet applications. Technical Report CS-TR-4078, University of Maryland, 1999.
- [2] Jeff Heflin, James Hendler, and Sean Luke, Applying Ontology to the Web: A Case Study. In *Proceedings of the International Work-Conference on Artificial and Natural Neural Networks, IWANN'99*, 1999.
- [3] Tom Mitchell, *Machine Learning*, McGraw-Hill Publishing Company, 1997.
- [4] QingFeng Lin, A Machine Learning Framework for Automatically Annotating Web Pages with Simple HTML Ontology Extension (SHOE). Master's thesis, University of Nebraska-Lincoln, 2000.
- [5] J. Ross Quinlan and R. M. Cameron-Jones, FOIL: A midterm report, 3-20. In *Proceedings of the European Conference on Machine Learning*, Springer-Verlag, 1993.
- [6] Andrew K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. <http://www.cs.cmu.edu/~mccallum/bow>, 1996.
- [7] MLC++ home page: <http://www.sgi.com/tech/mlc/>. Last reviewed March 13, 2001.
- [8] FOIL6: Produces Horn clauses from data expressed as relations. <ftp.cs.su.oz.au/pub/foil6.sh>