

---

## Categorisation of web documents using extraction ontologies

---

Li Xu\*

Department of Computer Science,  
University of Arizona South,  
1140 N Colombo Ave., Sierra Vista, AZ 85635, USA  
E-mail: lxu@email.arizona.edu

\*Corresponding author

David W. Embley

Department of Computer Science,  
Brigham Young University,  
Provo, Utah, USA  
E-mail: embley@cs.byu.edu

**Abstract:** Automatically recognising which HTML documents on the Web contain items of interest for a user is non-trivial. As a step toward solving this problem, we propose an approach based on information-extraction ontologies. Given HTML documents, tables, and forms, our document recognition system extracts expected ontological vocabulary (keywords and keyword phrases) and expected ontological instance data (particular values for ontological concepts). We then use machine-learned rules over this extracted information to determine whether an HTML document contains items of interest. Experimental results show that our ontological approach to categorisation works well, having achieved  $F$ -measures above 90% for all applications we tried.

**Keywords:** web document categorisation; web document classification; extraction ontologies.

**Reference** to this paper should be made as follows: Xu, L. and Embley, D.W. (2008) 'Categorisation of web documents using extraction ontologies', *Int. J. Metadata, Semantics and Ontologies*, Vol. 3, No. 1, pp.3–20.

**Biographical notes:** Li Xu received a BS in Computer Science (1994) from Shandong University in China and an MS in Computer Science (1997) from the University of Science and Technology of China. In 2003, she earned a PhD in Computer Science from Brigham Young University. Since 2003, she has been a tenure-track Assistant Professor at the University of Arizona South. She has worked extensively in the areas of data and metadata representation, heterogeneous data integration, and semantic search over data and metadata.

David W. Embley received a BA in Mathematics (1970), and an MS in Computer Science (1972), both from the University of Utah. In 1976, he earned a PhD in Computer Science from the University of Illinois. From 1976 to 1982, he was a faculty member at the University of Nebraska, where he was tenured in 1982. Since then he has been a faculty member in the Department of Computer Science at Brigham Young University. His research interests include conceptual modelling and extraction and integration of web data. His research is supported in part by the National Science Foundation.

---

### 1 Introduction

The World Wide Web contains abundant repositories of information in HTML documents – indeed, it contains so much that locating information of interest is a huge challenge. Even sorting through a tiny subset of HTML documents for particular items of interest is overwhelming. How can we automatically select just those documents that have the needed information for a user?

In this paper, we focus on the specialised subproblem of servicing users seeking highly specific information about items of interest either for themselves or for their customers. In both cases, users only want data-rich Web documents in a narrow domain of interest, either for manual perusal for themselves or for subsequent automatic information extraction for their customers. Often, the users we consider have a long-term need for gathering information within a particular domain of

interest. Examples include users wanting to purchase a car, rent an apartment, or find genealogical information about their ancestors.

In our work we apply techniques from data extraction (Laender et al., 2002), information retrieval (Baeza-Yates and Ribeiro-Neto, 1999), and machine learning (Quinlan, 1993). Using these techniques, we exploit the content of HTML documents and construct automated processes to do categorisation (i.e., to classify a document as belonging or not belonging to the specified domain of interest). The HTML documents we consider include semistructured HTML documents, HTML tables, and HTML forms. Given a set of HTML documents, we use an extraction ontology to find expected ontological vocabulary (keywords and keyword phrases) and expected ontological instance data (particular values for ontological concepts). We then give extracted vocabulary and instance data to machine-learned rules to determine whether the HTML documents contain items of interest for the application.

The basic motivation for our ontological approach is the belief that if we can encode a formal conceptualisation (Gruber, 1993) of the items of interest to a user, we can use this formal conceptualisation to identify Web documents that contain them. A disadvantage of this approach, of course, is the cost of constructing extraction ontologies (<http://www.deg.byu.edu/multidemos.html>).<sup>1</sup> The approach is therefore not useful for ad hoc, one-time queries, but rather for applications requiring repeated searches or requiring informational repository construction. An advantage, on the other hand, is that the precision and recall can be high, making the effort worthwhile. The approach is particularly useful for applications where subsequent extraction is wanted or where semantic annotation is needed.

This paper expands on our previous work (Embley et al., 2001) by

- extending our coverage – adding single-record documents, forms, and documents with linked subdocuments in addition to multiple-record documents
- redoing earlier experiments and expanding the applications used for experiments
- completely rewriting the description of our approach
- improving our heuristics leading to better results.

This paper presents our contributions as follows. Section 2 discusses related work. Section 3 illustrates our assumptions about HTML documents and provides an example to which we refer throughout the paper to illustrate our ideas. Section 4 describes extraction ontologies, on which we base our work. Given an extraction ontology and a set of HTML documents, Section 5 explains how we automatically obtain statistical measures for determining document relevance using a set of heuristics including:

- a density heuristic
- an expected-values heuristic
- a grouping heuristic.

Section 6 presents the architecture of the framework we have built. Section 7 provides an empirical evaluation of our approach including our experimental results. Section 8 gives concluding remarks and our plans for future work.

## 2 Related work

An abundance of related work has been done, but outside of our lab, none of this related work applies extraction ontologies to do categorisation. Although related work is not based on extraction ontologies, it is nevertheless relevant because it provides for comparison of alternatives and because it suggests techniques applicable to the goal of finding items of interest in Web documents. Representative of the abundant related work, we cite the following.

### *Search engines*

For our application, search engines (e.g., Google ([www.google.com](http://www.google.com)) and Swoogle (Ding et al., 2004)) are clearly not precise enough.

### *Information retrieval*

More general Information Retrieval (IR) techniques (Baeza-Yates and Ribeiro-Neto, 1999) and even Web IR techniques (Kobayashi and Takeda, 2000) typically do not solve the problem well either. Traditional IR systems retrieve documents based on exact matching of terms in user queries and documents. Fang and Zhai (2006) address this issue by incorporating semantic term matching techniques into IR models. Lin and Demner-Fushman (2006) apply conceptual knowledge in a restricted domain such as Clinical Medicine to achieve a better performance gain. IR systems usually only facilitate retrieving information units quickly for relatively short-term information needs of a diverse large group of users, whereas our application facilitates specific interests of a particular user or a group of users and supports long-term information needs. Nevertheless, some semantic-specific IR techniques like (Chu-Carroll et al., 2006) use XML fragments to conceptualise, restrict, or relate terms in user queries in order to achieve a high-precision search over semantically annotated text. Semantic annotation, however, is still only being explored and there are no encouraging results that are good enough to be practical (Dill et al., 2003; Kiryakov et al., 2004).

### *Text categorisation*

Text categorisation systems assign text documents to one or more predefined categories based on their content (Sebastaini, 2002). Most of these systems are based on

machine learning; they use pre-classified documents to train classifiers, as we do. They do not use specifically identified and extracted features as we do, however, and they therefore lack support to explicitly specify items of user interest in a precise way. Instead, each document is represented using a feature vector, which is typically a vector of terms that appear in document categories. The number of terms is often high, resulting in feature vectors of high dimensionality. Processing large feature sets, however, is not reliable because of the typical lack of sufficient training data. Thus, to obtain optimal cost-effectiveness, a pass of dimensionality reduction is

required. In removing features to represent documents, the reduction process must be performed with care to avoid removing potentially useful information about the meaning of the documents. Some approaches (Schapire et al., 2002; Wu and Srihari, 2004; Dayanik et al., 2006) combine domain knowledge, informally represented as keywords, pseudo training data, and domain knowledge text respectively, to learn from a handful training examples. In recent work on aligning ontologies with real-world data sets, researchers suggest extracting a finite set of words as a context to describe a document, generalising this context for a concept given training

**Figure 1** A car-ads Web document and a non-car-ads Web document: (a) car ads from www.elktribune.com and (b) items for sale from www.crookstontimes.com

Last Updated  
Monday, January 24, 2000 12:19pm Cars for Sale

---

DEPENDABLE CAR  
1989 Subaru SW. Auto, AC, \$1900 OBO. Call (336)835-8579.

---

FACTORY WARRANTY  
1998 Elantra. Black 4 door w/ tinted windows. Auto, pb, ps, cruise, am/fm cassette stereo, a/c. Excellent condition pay off OBO. Call (336)526-5444 anytime & leave message.

---

1994 HONDA ACCORD EX  
Auto, power everything, jade green w/gold package. Under 100K miles. Call (336)526-1081 after 7pm.

---

1999 GRAND AM  
27,000 miles, silver, auto, still under warranty. \$14,000 OBO. Call (336)366-4996 anytime.

---

'53 Chevy Bel Aire. All original, looks like new. Serious inquiries only. \$8500. Call (336)468-8924 after 4 pm.

---

TWO GREAT CARS  
1973 MGB convertible. British racing green. Mags, new tires, 4-speed, 1 owner, excellent running condition. \$4500.  
1977 Olds Cutlass Supreme. New white paint job w/ 1/2 red Landau top, original mags & new tires. Auto., 1 owner, low mileage, loaded. Call (336)984-2843.

---

95 FORD CONTOUR  
5-speed, great condition, one owner, \$5300. Call (336)526-8853 & leave message if no answer.

---

SEIZED CARS FROM \$500  
Sports, luxury & economy cars, trucks, 4x4's utility and more. For current listings, call 1-800-311-5048 ext. 10012.

---

1996 VW JETTA GL  
26,000 miles. 4 door, 5-speed, AC, sunroof, 1 owner. \$11,000. Call (336)874-7317 anytime.

---

'85 Buick Park Avenue. \$500. Head may be cracked. Will run. Body good condition. Call (336)526-2768.

---

'95 Ford Thunderbird. Loaded, V-8, 45K, \$6995. Call S&J Motors at (336)874-3403.

---

'96 Mercury Tracer. 4 door, 5 speed, 34K, \$4995. Call S&J Motors at (336)874-3403.

---

'88 Firebird. V8, 5.0, fuel injected, T-tops, 109,000 miles, red, runs great. \$1880. Call (336)526-1164 anytime.

---

1990 CONVERSION VAN  
350 motor, auto, new tires, TV, VCR, captain chairs, front & rear AC. \$4,995. Call (336)320-2658 anytime.

---

COMMERCIAL WORK VAN  
'95 Chevy Astro, V6, w/ac & fully equipped utility shelves. \$9400. Call (336)526-2675 & leave message.

(a)

Last update: Wednesday, December 22, 1999

Select a category

Apartment For Rent For Sale or Rent Lost or Found  
For Rent Help Wanted  
For Sale House For Rent

---

Apartment For Rent  
ONE EFFICIENCY, 2 & 3 bdrm, all utilities paid.  
Call 281-2051 -

---

For Rent  
HOUSING SOLUTIONS - Free TV cable furn. \$60/wk - \$210/mo. 281-4060. -

---

For Sale  
1998 JD 455 mower, 60' deck. Call for price. Also, homemade Go-Cart. Call after 5:30 pm 218-281-1128. -

---

For Sale or Rent  
10,000 SQ. FT. office building. Handicap accessible.  
Call 281-3631. -

---

Help Wanted  
NOW HIRING full time and part time customer service representatives. Advancement possible and weekly pay. Must be able to work weekends and holidays. Apply at Superamerica, 411 N Main St., Crookston, MN EOE -

---

PART-TIME AND weekend help working with developmentally disabled adults. Call Melissa or Karen at 281-3872. -

---

REM-NORTHWEST Services, Inc. has a full time Program Coordinator/Coordinator position open in Crookston working with four developmentally disabled adults. Duties include hiring, staff supervision, scheduling, oversight of most areas of the home's operation. Applicant must be 18 years of age or older. Must have a high school diploma or equivalent. One year experience serving people with developmental disabilities preferred. Must have a valid driver's license and driving record that meets REM's insurability requirements. Insurance and benefits available. If interested call for application at 218-281-5113. E.O.E. -

---

REM-NORTHWEST Services, Inc. has full and part time Coordinator positions available in Crookston, MN, working with citizens who are developmentally disabled. Excellent benefits are offered including health, dental, life, 401K and profit sharing for full and part time employees working 20 hours a week or more. Exceptional training is provided. Applicants must be 18, have a valid driver's license and high school diploma or GED. Apply by calling for application at 218-281-5113 or 1-800-532-7655. E.O.E. -

---

House For Rent  
3 BDRM HOUSE \$450/mo. 281-1970. 22 STEEL BUILDINGS, NEW, must sell. 40x60x14 was \$17,500 now \$10,971; 50x100x16 was \$27,850 now \$19,990; 80x135x16 was \$79,850 now \$42,990; 100x175x20 was \$129,650 now \$78,850. 1-800-406-5126. -

---

Lost or Found  
FOUND: Golden retriever about 4 months old. Found 7 miles south of Crookston. Call after 5:30 pm 281-1128. -

(b)

data, and then providing a ranking method that ranks ontology concepts according to their relevancy to a given context extracted from a document (Segev and Gal, 2007). Compared with this work on text categorisation, our work avoids the high-dimensionality problem associated with machine learning by carefully choosing a few, specific, heuristic-based features, all of which are made available through extraction ontologies.

*Focused crawling*

Focused crawlers (Chakrabarti, 1999; Chakrabarti et al., 1999) aim to search and retrieve only the subset of the Web that pertains to a pre-defined set of topics. Like most text-categorisation systems, the topics are specified using exemplary documents. The essential idea in focused crawling is the hypothesis that there is a short range of topical locality on the Web. Focused crawlers exploit topic locality based on the linkage structure of the Web. To effectively discover resources, focused crawlers start at a few well chosen points and only crawl within the range of documents satisfying a pre-defined set of topics. The topic of a crawled page is determined by both its content and its linked neighbours on the Web. Our system does not

crawl, per se; instead it evaluates any page given to it with respect to specified items of user interest. Similar to focused crawling, we do look at linked documents, but only in specialised cases where we seek additional information about the initial page from which we came.

*Extraction ontologies*

Within our lab, we and other colleagues have tried alternative ways to make use of results obtained by applying extraction ontologies to do text categorisations (Euzenat and Shvaiko, 2007; Xu and Embley, 2006).<sup>2</sup> We have tried Vector Space Model (VSM) and Clustering Models (Kwong and Ng, 2003), logistic regression (Wang and Ng, 2003), statistical multivariate analysis (Ng et al., 2001), and machine learning (Embley et al., 2001). In all this work, we only considered multiple-record Web documents – documents such as classified ads with one entry after another all from the same domain.


**3 HTML documents**

The HTML documents we consider in our work include semistructured HTML documents such as the lists of ads

**Figure 2** HTML page with table from www.bobhowardhonda.com (see online version for colours)

Year	Make and Model	Price	Miles	Exterior	Photo
<input type="checkbox"/>	1999 <a href="#">Pontiac Firebird</a>	Contact Us	32,883	Blue	
<input type="checkbox"/>	2000 <a href="#">Acura RL 3.5</a>	\$23,988	36,657	Silver	
<input type="checkbox"/>	2002 <a href="#">Honda Accord EX</a>	\$21,988	13,875	White	
<input type="checkbox"/>	2002 <a href="#">Honda Passport</a>	\$20,998	10,410	Black	
<input type="checkbox"/>	2002 <a href="#">Acura RSX Type-S</a>	\$20,988	14,208	Red	
<input type="checkbox"/>	2000 <a href="#">Chevrolet Camaro</a>	\$13,995	45,297	White	
<input type="checkbox"/>	2001 <a href="#">Honda Accord Value Package</a>	\$13,995	31,710	Silver	
<input type="checkbox"/>	2001 <a href="#">Chevrolet Silverado C1500</a>	\$13,988	28,022	Pewter	

**Figure 3** HTML form and submission results from www.wheels.com: (a) filled in HTML form and (b) retrieved car ads after submitting the form in Figure 3(a) (see online version for colours)



Year:  to

Make:


Model:

Color:

Price:  to


(a)

**2003 BMW 325 I PREMIUM BLACK**  
w/ BLACK int. 6 CYLINDER  
43,157mi. AUTO BALANCE BMW  
WARRANTY, LEATHER, PWR  
MOONROOF, ALLOY WHEELS,  
\$19,995 Stock No. AS1012 [AUTO  
SHOWCASE OF CAROL STREAM](#)  
[\[More Detail\]](#)




---

**2002 BMW 525 BLACK 6**  
CYLINDER 61,496mi. AUTO  
PREMIUM PACKAGE, HEATED  
SEATS, MOONROOF, CD, NO  
HASSLE SALE PRICE!! \$19,995  
Stock No. BM8551 [BENSENVILLE  
MOTORS](#) [\[More Detail\]](#)




---

**2001 BMW 530i BLACK 6 CYLINDER GASOLINE**  
74,015mi. 5 SPEED MANUAL 16Inch Wheels, Headlight  
Washers, Privacy Glass, Moonroof, Leather, Wood Trim,  
Heated Front Seats, In-Dash CD, Premium Audio \$17,995  
Stock No. 26056A [BENTLEY DOWNERS GROVE](#) [\[More  
Detail\]](#)

(b)

in Figure 1, HTML tables such as the one in Figure 2, and HTML forms such as the one in Figure 3(a), which when filled-in and processed, yields a table or a semistructured list such as the one in Figure 3(b).

We assume that the HTML documents are data-rich and narrow in ontological breadth Embley et al. (1999a). A document is *data rich* if it has a number of identifiable constants such as dates, names, account numbers, ID numbers, part numbers, times, currency values, and so forth. A document is *narrow in ontological breadth* if we can describe its application domain with a relatively small ontological model. The documents in Figures 1–3 are all data rich and narrow in ontological breadth.

When evaluating relevancy, we recognise and consider three kinds of HTML documents: *multiple-record* documents, *single-record* documents, and *application forms*. The documents in Figures 1, 2, and 3(b) are all multiple-record documents because they contain similar descriptions of several different items. Figure 4 shows a car ad linked from *Honda Accord EX* in Figure 2, which we call a single-record document because it declares the various features of only one item – the *Honda Accord EX* for sale. Figure 3(a) is an application form. When considering a form, we may have, in addition to the labelled form fields,

- selection lists with possible values of interests
- the results returned, if we can automatically fill in and submit the form using default values as discussed in Liddle et al. (2001, 2002).

Given an HTML document in our approach, we collect two kinds of text components:

- text that appears in the whole document, which we call the *document text component*
- text fragments that appear within individual forms in the document, each of which we call a *form text component*.

A form text component includes the text that labels form fields and values in selection lists. If the document contains no form, the set of form text components is empty. Note that the document text component of an HTML document subsumes all the form text components that appear within the forms as well as the text that is outside the forms in the document. We therefore formalise a document  $d$  as a sequence of text components, written  $d = [t_d, t_{f_1}, \dots, t_{f_n}]$ , where  $t_d$  is the document text component,  $t_{f_i}$  ( $1 \leq i \leq n$ ) is a form text component, and  $n$  is the number of forms in the document  $d$ . (Note that  $d = [t_d]$  if the document  $d$  contains no form.)

#### 4 Extraction ontologies

We define an extraction ontology to be a conceptual-model instance that describes a real-world application in a narrow, data-rich domain of interest. Each of our extraction ontologies consists of two components:

- an *object/relationship-model instance*, which describes sets of objects, sets of relationships among objects, and constraints over object and relationship sets<sup>3</sup>
- for each object set, a *data frame*, which defines the potential contents of the object set (Embley et al., 1992).

**Figure 4** Linked page with additional information from [www.bobhowardhonda.com](http://www.bobhowardhonda.com) (see online version for colours)


2002 Honda Accord EX		\$21,988
<b>Features</b>		
<ul style="list-style-type: none"> <li>• Air Conditioning</li> <li>• Driver Side Air Bag</li> <li>• Passenger Side Air Bag</li> <li>• Anti-Lock Brakes</li> <li>• AM/FM Cassette</li> <li>• Security Features</li> <li>• Alloy Wheels</li> <li>• Automatic Transmission</li> <li>• Bucket Seats</li> <li>• Compact Disc Player</li> <li>• Cruise Control</li> <li>• Front Wheel Drive</li> <li>• Intermittent Wipers</li> </ul>	 <p>www.BobHowardAuto.com</p> <p>Click on photo to enlarge</p>	
	<b>Price</b>	\$21,988
	<b>Mileage</b>	13,875 miles
	<b>Body Type</b>	Car
	<b>Body Style</b>	Coupe
	<b>Exterior</b>	White
	<b>Transmission</b>	Automatic
	<b>Engine</b>	3.0L 6 cyl Fuel Injection
	<b>Fuel Type</b>	Gas
	<b>Stock Number</b>	350291A
	<b>VIN</b>	1HGCG22562A018644

Figure 5 Car-ads extraction ontology (partial)

1. Car [-> object];
2. Car [0:0.975:1] has Year [1:\*];
3. Car [0:0.925:1] has Make [1:\*];
4. Car [0:0.908:1] has Model [1:\*];
5. Car [0:0.45:1] has Mileage [1:\*];
6. Car [0:0.8:1] has Price [1:\*];
7. Car [0:2.1:\*] has Feature [1:\*];
8. PhoneNr [1:\*] is for Car [0:1.15:\*];
9. Year matches [4]
10.     constant {extract "\d{2}";
11.         context "\b'[7-9]\d\b";
12.     ...
13. Mileage matches [8]
14.     ...
15.     keyword "\bmiles\b", "\bmi\.", "\bmi\b", "\bmileage\b", "\bmilage\b";
16. ...

A data frame for an object set defines the textual appearance of constant objects for the object set (instance values for ontological concepts) and establishes appropriate keywords and keyword phrases (ontological vocabulary words and phrases) that are likely to appear in a document when objects in the object set are mentioned. Figure 5 shows part of our car-ads extraction ontology, including object and relationship sets and cardinality constraints (Lines 1–8) and a few lines of the data frames (Lines 9–16). The full ontology for car ads is about 600 lines long.

An object set in an extraction ontology represents a set of objects which may either be lexical or nonlexical. Data frames with declarations for constants that can potentially populate the object set represent lexical object sets, and data frames without constant declarations represent nonlexical object sets. *Year* (Line 9) and *Mileage* (Line 13) are lexical object sets whose character representations have a maximum length of 4 and 8 characters respectively. *Make*, *Model*, *Price*, *Feature*, and *PhoneNr* are the remaining lexical object sets in our car-ads application; *Car* is the only nonlexical object set. The notation ‘[-> object]’ in Line 1 designates *Car* as the main object set of interest for the extraction ontology.

We describe the constant lexical objects and the keywords for an object set by regular expressions using Perl syntax. When applied to a textual document, the **extract** clause in a data frame causes a string that matches a regular expression to be extracted, but only if the **context** clause also matches the string and its surrounding characters. Thus, in Figure 5 one of the several patterns for *Year* looks for and extracts two digits (the **extract** clause in Line 10), but only in the context of a leading apostrophe preceded by a word boundary and only if the first of the two digits is 7–9 (the **context** clause in Line 11). The **keyword** clause specifies potential words, abbreviations, or phrases likely to occur close to a value and likely to disambiguate the value from other similar values that actually belong to other ontological concepts. Thus, for a car mileage value, we might find words such

as ‘miles’ or ‘mileage’, the spelling variant ‘mileage’, and the abbreviation ‘mi’ with or without a terminating period (Line 15).

We denote a relationship set by a name that includes its participant object set names (e.g., *Car has Year* and *PhoneNr is for Car*). The *min:max* pairs and *min:ave:max* triples in the relationship-set name are *participation constraints*: *min* designates the minimum number of times an object in the object set can participate in the relationship set; *ave* designates the average number of times an object is expected to participate in the relationship set; and *max* designates the maximum number of times an object can participate, with \* designating an unknown maximum number of times. The participation constraint on *Car* for *Car has Feature*, for example, specifies that a car need not have any listed features, that a car has 2.1 features on the average, and that there is no specified maximum for the number of features listed for a car.

For our car-ads extraction ontology, we obtained participation constraints as follows. We selected several car-ads pages, chosen to be representative of typical car-ads pages. From each of these pages we selected several individual car-ads, each chosen to be representative of a typical car-ad. We then simply obtained minimum, average, and maximum values for each object set in each relationship set and normalised the values for a single car ad. For other applications, we similarly obtain participation constraints. In general, with some expertise in the application and a sampling of Web documents, extraction ontology developers can establish reasonable participation constraints. These cardinalities need not be exact and should be treated as expected rather than exact values.

Extraction ontologies are the key ingredient in our categorisation methodology. Extraction ontologies allow us to move beyond keyword approaches by allowing the methodology to also use recognised instances and measures of appropriate organisation over collections of keyword and instance values. Together these techniques support higher precision classification.



## 5 Heuristics

Given the information we are able to obtain from extraction ontologies, we consider three kinds of heuristics: *density* heuristics, an *expected values* heuristic, and a *grouping* heuristic. For a document  $d$ , we consider an extraction ontology  $O$  and a text component  $t$ , which is either a document text component or a form text component. The set of density heuristics measure the densities of constants and keywords defined in  $O$  that appear in  $t$ . The expected-values heuristic uses the VSM (Baeza-Yates and Ribeiro-Neto, 1999), a common information-retrieval document representation model, to compare the number of constants expected for each object set, as declared in  $O$ , to the number of constants found in  $t$  for each object set. The grouping heuristic measures the occurrence of groups of textual values found in  $t$  with respect to expected groupings of textual values implicitly specified in  $O$ . The heuristics are not at odds. And we design them to supplement each other.

The next three subsections define these heuristics, explain the details about how we provide a measure for each heuristic, and give examples to show how they work. When reading these subsections, bear in mind that in creating these heuristics, we favoured simplicity. More sophisticated measures can be obtained. For example, for density measures we could account for uncertainty in constant and keyword matches (Embley et al., 1999b). For expected values, we could more accurately match object sets with recognised values by using additional heuristic rules (Embley et al., 1999a; Embley and Xu, 2000). For grouping, we could first compute record boundaries (Embley et al., 1999c) and rearrange record values (Embley and Xu, 2000). However, more sophisticated measures are more costly. We have chosen to experiment with less costly heuristics, and, as will be shown, our results bear out the seeming correctness of this choice.

### 5.1 Density heuristics

A text component  $t$  parsed from an HTML document  $d$  that is relevant to a particular extraction ontology  $O$  should include many constants and keywords for object sets defined in the ontology. Based on this observation, we define a set of density heuristics. We compute the density heuristics with respect to the strings recognised by the data-frame recognisers of an extraction ontology as a whole and also of each object set individually. We compute the density of  $t$  with respect to the ontology  $O$  as a whole as follows:

$$\text{Density}(t, O) = \text{number of matched characters in } t \\ \text{for } O / \text{number of characters in } t.$$

When counting characters in documents and in recognised strings for density measures, we exclude characters in HTML tags. We compute the density of  $t$  with respect to

an individual object set  $o$  in  $O$  as follows:

$$\text{Density}(t, o) = \text{number of matched characters in } t \\ \text{for } o / \text{number of characters in } t.$$

We must be careful, of course, not to count characters more than once. For example, in the phrase ‘asking only 18K’, a car-ads extraction ontology might recognise ‘18K’ as potentially both a price and a mileage. Nevertheless, we should only count the number of characters as three, not six. Further, we need determine whether we count the value ‘18K’ for a price or for a mileage.

Consider the document text component  $t_{d_a}$  in the multiple-record document  $d_a$  in Figure 1(a). Recall that the nonlexical object set of the car-ads extraction ontology is *Car* and that the lexical object sets are *Year*, *Make*, *Model*, *Mileage*, *Price*, *Feature*, and *PhoneNr*. Some of the lexical values found in  $t_{d_a}$  include ‘1989’ (*Year*), ‘\$1900’ (*Price*), ‘100K’ (*Mileage*), ‘Auto’ (*Feature*), ‘Cruise’ (*Feature*), ‘(336)835-8579’ (*PhoneNr*), ‘Subaru’ (*Make*), and ‘SW’ (*Model*). The keywords ‘Cars for Sale’ for the object set of interest *Car*, ‘miles’ and ‘mileage’ for *Mileage*, and ‘Call’ for *PhoneNr* appear in  $d_a$ . The total Number of characters in  $t_{d_a}$  is 1992, whereas the number of matched characters is 696. Hence, the  $\text{Density}(t_{d_a}, O)$  is 0.3493 (=696/1992). For each object set in the car-ads extraction ontology  $O$ , there is also a density measure. For example, the number of matched characters for *Make* is 47. Therefore,  $\text{Density}(t_{d_a}, \text{Make})$  is 0.0236.

When we apply the density heuristics for the car-ads extraction ontology to the document text component  $t_{d_b}$  of the document  $d_b$  in Figure 1(b), the densities are much lower. Although no makes, models, or car features appear, there are years, prices, and phone numbers and the ontology (mistakenly) recognises ‘10,000’ (in ‘10,000 SQ. FT.’) and ‘401K’ (the retirement plan) as potential mileages. Altogether, 229 characters of 2627 are recognised by the car-ads ontology. Thus, the density with respect to the car-ads extraction ontology  $\text{Density}(t_{d_b}, O)$  is 0.0871. There are also eight other density measures, one for each object set. For example, the document text component of  $d_b$  contains keywords and values for *PhoneNr*, and the density for *PhoneNr* is 0.0533. The density for *Car* is 0.0 since the document text component does not contain any keywords for the object set of interest, *Car*, in the car-ads extraction ontology.

### 5.2 Expected-values heuristic

Density alone is likely to be a good determiner of relevancy in many cases, but it is not hard to imagine a page in which a large fraction of values would be recognised and yet the page is not relevant. Consider, for example, a list of hot tubs and prices. Price recognisers in our car-ads ontology would recognise the prices, and since nearly half the list consists of prices, the density would be high. The expectations about what values should appear, however, are nowhere close to being in line with the expectations for car ads – except for an unexpected occasional appearance,

there would be no years, no makes or models, no car features, no mileage numbers, and no contact information.

We apply the VSM model to measure whether a text component  $t$  parsed from an HTML document  $d$  has the number of values expected for each lexical object set of an extraction ontology  $O$ . Based on the lexical object sets and the participation constraints in  $O$ , we construct an ontology vector  $V_O$ . Based on the same lexical object sets and the number of constants recognised for these object sets in  $t$ , we construct a document vector  $V_t$ . We measure the relevance of  $t$  to  $O$  with respect to our expected-values heuristic by observing the cosine of the angle between  $V_t$  and  $V_O$ .

To construct the ontology vector  $V_O$ , we

- identify the lexical object-set names (these become the names of the coefficients of  $V_O$ )
- determine the average participation (i.e., the expected frequency of occurrence) for each lexical object set with respect to the object set of interest specified in  $O$  (these become the values of the coefficients of  $V_O$ ).

For example, the ontology vector for the car-ads extraction ontology is  $\langle Year:0.975, Make:0.925, Model:0.908, Mileage:0.45, Price:0.8, Feature:2.1, PhoneNr:1.15 \rangle$ , where these values are the average participation-constraint values obtained as explained in Section 4. Thus, for a typical single car ad we would expect almost always to find a year, make, and model, but we only expect to find the mileage about 45% of the time and the price about 80% of the time. Further, we expect to see a list of features that on the average has a couple of items in it, and we expect to see a phone number and sometimes more than one phone number.<sup>4</sup>

To construct the document vector  $V_t$ , we use the names of the coefficients of  $V_O$ , and obtain the value of each coefficient of  $V_t$  by automatically counting the number of appearances of constant values in  $t$  that belong to each lexical object set. Table 1 shows the values of the coefficients of the document vector for the document text component of the car-ads document in Figure 1(a), and Table 2 shows the values of the coefficients of the document vector for the document text component of the non-car-ads document in Figure 1(b).

We have discussed the creation of a document vector as though correctly detecting and classifying the lexical values in a text in a document were easy – but sometimes it is not easy. We identify potential lexical values for an object set as explained in Section 4, but identification can be erroneous. After initial identification, we must decide which of the potential object-set/constant pairs to accept. We could use sophisticated heuristics based on keyword proximity, application-ontology cardinalities, record boundaries, and missing-value defaults to best match object sets with potential constants, as explained in Embley et al. (1999a, 1999c), Embley and Xu (2000). Instead, however, we use techniques that are far less sophisticated and thus also far less costly. In our less costly procedures

we consider three cases: a string of characters within a document is recognised as

- belonging to only one object set
- belonging to two or more object sets all of which are specialisations of the same object set
- belonging to two or more object sets all of which are not specialisations of the same object set.

**Table 1** Lexical values found in the multiple-record car advertisements in Figure 1(a)

Name of lexical object set	Corresponding lexical values found in the document	Number of lexical values
Year	1989, 1998, 1994, 1999, '53, 1973, 1977, 95, 1996, ...	16
Make	Subaru, HONDA, Chevy, Olds, FORD, VW, Buick, Mercury, ...	10
Model	SW, Elantra, ACCORD, GRAND AM, Cutlass, CONTOUR, JETTA, ...	12
Mileage	100K, 27000, 26000, 45K, 34K, 109000	6
Price	\$1900, \$14,000, \$8500, \$4500, \$5300, \$11,000, \$6995, \$4995, \$1880, ...	11
Feature	Auto, Black, 4 door, pb, ps, cruise, am/fm, cassette, stereo, green, ...	29
PhoneNr	(336)835-8579, (336)526-5444, (336)526-1081, (336)366-4996, ...	15

**Table 2** Lexical values found in the multiple-record *Items for Sale* document in Figure 1(b)

Name of lexical object set	Corresponding lexical values found in the document	Number of lexical values
Year	1999, 1998, 60, 40, 50, 80	6
Make		0
Model		0
Mileage	10,000, 401K	2
Price	\$17,500, \$10,971, \$27,850, \$19,990, \$79,850, \$42,990, \$129,650, \$78,850	8
Feature		0
PhoneNr	281-2051, 281-4060, 218-281-1128, 281-3631, 281-3872, 218-281-5113, 218-281-5113, 800-532-7655, 281-1970, 800-406-5126, 281-1128	11

For Case 1, we simply accept the string as belonging to the object set. As an example of Case 2, there are several dates in obituaries: *Death Date*, *Birth Date*, *Funeral Date*, *Viewing Date*, and *Burial Date*. All dates are specialisations of *Date* in the obituary extraction ontology. The appearance of keywords (e.g., ‘born on’ and ‘died on’) makes it possible to distinguish the multiple kinds of dates. Thus, for Case 2, we reject a recognised string if keywords are not present. For Case 3, recognisers for different object sets include a common string of characters as part of a value belonging to their object set. For example, the string ‘23,000’ may be recognised as being either a price or



a mileage. To resolve these ambiguities, we consider three subcases:

- exact match
  - subsumption
  - partial overlap.
- 1 If a value  $v$  is recognised as potentially belonging to more than one object set, we use the closest keyword that appears before or after  $v$  to determine which object set to choose. For example, “. . . asking only 23,000 . . .” would be a price whereas “. . . 23,000 miles on it . . .” would be a mileage. If no applicable keyword is found, we choose one of the object sets arbitrarily.
  - 2 If a value  $v$  is a proper substring of lexical value  $w$ , we retain  $w$  and discard  $v$ .
  - 3 If lexical value  $v$  and lexical value  $w$  appear in a Web document, such that a suffix of  $v$  is a prefix of  $w$ , we retain  $v$  and discard  $w$ .

As mentioned, we measure the similarity between an ontology vector  $V_O$  and a document vector  $V_t$  by measuring the cosine of the angle between them. In particular, we calculate

$$Similarity(t, O) = \cos \theta = P/N,$$

where  $P$  is the inner product of the two vectors, and  $N$  is the product of the lengths of the two vectors. When the distribution of values among the object sets in  $V_t$  closely matches the expected distribution specified in  $V_O$ , the angle  $\theta$  will be close to zero, and  $\cos \theta$  will be close to one.

Consider the car-ads extraction ontology  $O$  in Figure 5 and the Web document  $d_a$  in Figure 1(a). The coefficients

of  $V_O$  for  $O$  are 0.975, 0.925, 0.908, 0.45, 0.8, 2.1, and 1.15, which are the expected frequency values of lexical object sets *Year*, *Make*, *Model*, *Mileage*, *Price*, *Feature*, and *PhoneNr*, respectively for a single ad in the car-ads extraction ontology. The coefficients of  $V_{t_{d_a}}$  for  $d_a$  are 16, 10, 12, 6, 11, 29, and 15 (see the last column of Table 1), which are the actual number of appearances of the lexical values in  $d_a$ . We thus compute  $Similarity(t_{d_a}, O)$  to be 0.996.<sup>5</sup> Now consider the car-ads extraction ontology  $O$  again and the Web document  $d_b$  in Figure 1(b). The coefficients of  $V_O$  are the same, but the coefficients of  $V_{t_{d_b}}$  for  $d_b$  are 6, 0, 0, 2, 8, 0, and 11 (see the last column of Table 2). We thus compute  $Similarity(t_{d_b}, O)$  to be 0.567.

### 5.3 Grouping heuristic

An HTML document may have a high density measure for an extraction ontology  $O$  and may also have a high expected-values measure for  $O$  but may still not be considered relevant to  $O$ . Consider, for example, consumer report pages for cars. The proportion of recognised values may be large, and it is likely that the proportions of recognised values would correspond reasonably well with the expected-value proportions for car ads. Individual car descriptions and buying potential, however, would be missing, and a person wanting to purchase a car would be disappointed.

In the common case when documents have multiple records, each of which is relevant to an extraction ontology  $O$ , we would like to know that the text is organised into groups that can be recognised as records. Each record should, by itself, be relevant for  $O$ . As a simple heuristic to determine whether the recognised values are interleaved in a way that could be considered consistent with potential records of  $O$ , we consider the group of values in a document

**Figure 6** Groups of 1-max values extracted from HTML documents: (a) first four groups of 1-max values extracted from Figure 1(a) and (b) first four groups of 1-max values extracted from Figure 1(b)

Year: 2000 Year: 1989 Make: Subaru Model: SW N r. of distinct 1-m ax object sets: 3	Year: 1999 Year: 1998 Year: 60 Mileage: 10,000 N r. of distinct 1-m ax object sets: 2
Price: 1,900 Year: 1998 Model: Elantra Year: 1994 N r. of distinct 1-m ax object sets: 3	Mileage: 401K Year: 40 Price: 17,500 Price: 10,971 N r. of distinct 1-m ax object sets: 3
Make: HONDA Model: ACCORD Mileage: 100K Year: 1999 N r. of distinct 1-m ax object sets: 4	Year: 50 Price: 27,850 Price: 19,990 Year: 80 N r. of distinct 1-m ax object sets: 2
Model: GRAND AM Mileage: 27,000 Price: 14,000 Year: '53 N r. of distinct 1-m ax object sets: 4	Price: 79,850 Price: 42,990 Price: 129,650 Price: 78,850 N r. of distinct 1-m ax object sets: 1

(a)

(b)

that should appear at most once in each record and measure how well they are grouped.

We refer to an object set whose values should appear at most once in a record as a *1-max object set*. Maximum participation constraints in an ontology constrain the values of the 1-max object sets to appear at most once in a record. For example, in the car-ads extraction ontology, the 1-maximum on *Car* in the relationship set *Car [0:0.975:1]* has *Year [1:\*]* specifies that *Year* is a 1-max object set. Other 1-max lexical objects in the car-ads ontology are *Make*, *Model*, *Mileage*, and *Price*.

Instead of counting the number of 1-max object sets in an extraction ontology  $O$ , a more accurate way to obtain the number of 1-max object-set values expected to appear in each record is to sum the average expectation numbers for the 1-max object sets in  $O$ . Since the average expectation numbers for *Year*, *Make*, *Model*, *Mileage*, and *Price* in the car-ads ontology are 0.975, 0.925, 0.908, 0.45, and 0.8, respectively, the anticipated number of lexical values from these object sets in a car advertisement is 4.058. To obtain an expected group size, we truncate the decimal value of the sum.

The expected group size  $n$  is an estimate of the number of 1-max object-set values that should be in each single record in a multi-record document. Thus, if we list all recognised 1-max object-set values in the order they occur in a document and divide this sequence into groups of  $n$ , each group should have  $n$  values from  $n$  different object sets. The closer a document comes to this expectation, the better the grouping measure should be. For the multiple-record car-ads Web document in Figure 1(a), Figure 6(a) shows the first four groups of 1-max object-set values extracted from the document. Similarly, Figure 6(b) shows the first four groups of 1-max object-set values extracted from the document in Figure 1(b).

We measure how well the groups match the expectations with a grouping factor (denoted *Grouping*), which is calculated as follows:

$$Grouping(t, O) = \frac{\text{Sum of Distinct Lexical Values in Each Group}}{\text{Number of Groups} \times \text{Expected Number of Values in a Group}}$$

Considering just the part of the document  $t_{upT053}$  in Figure 1(a) that corresponds to the groups in Figure 6(a), we obtain the following grouping factor:

$$Grouping(t_{upT053}, O) = \frac{3 + 3 + 4 + 4}{4 \times 4} = 0.875.$$

For the entire document text component  $t_{d_a}$  of HTML document  $d_a$  in Figure 1(a), the grouping factor is 0.865; whereas for the entire document text component  $t_{d_b}$  of HTML document  $d_b$  in Figure 1(b), the grouping factor is 0.500.

## 6 Web document categorisation

For a particular categorisation application, we provide an extraction ontology. We formalise the evaluation of HTML-document relevancy to a category  $c$  as follows. An extraction ontology  $O$  specifies the application representing  $c$ . We represent an HTML document  $d$  as a sequence of text components, written  $d = [t_d, t_{f_1}, \dots, t_{f_n}]$ , where  $t_d$  is the document text component,  $t_{f_i}$  ( $1 \leq i \leq n$ ) is a form text component, and  $n$  is the number of forms in the document  $d$ . If the document  $d$  contains no form,  $d = [t_d]$ . Given the extraction ontology  $O$  and an HTML document  $d = [t_d, t_{f_1}, \dots, t_{f_n}]$ , we use  $m$  heuristic rules (see Section 5) to compute  $m$  confidence measures  $H_{t_d} = (h_1, h_2, \dots, h_m)$  for the document text component  $t_d$  and use the same  $m$  heuristic rules to compute  $H_{t_{f_i}} = (h_{i1}, h_{i2}, \dots, h_{im})$  for each form text component  $t_{f_i}$  ( $1 \leq i \leq n$ ). Thus we describe the similarity between the HTML document  $d$  and the extraction ontology  $O$  as a heuristic vector  $d_H = \langle H_{t_d}, H_{t_{f_1}}, \dots, H_{t_{f_n}} \rangle$  over  $n + 1$   $m$ -tuples of confidence measures. To categorise the document, we attempt to assign  $d_H$  to the category  $c$  obtaining either  $c_P$ , which represents a positive example (one relevant to the application), or  $c_N$ , which represents a negative example (one irrelevant to the application).

### 6.1 Training phase

For our work we use the popular machine learning algorithm C4.5 Quinlan (1993). C4.5 is a rule post-pruning decision-tree algorithm. The learning task is to check the suitability of documents for a given extraction ontology (i.e., to do categorisation by returning ‘Y’ (yes) when a document is suitable and returning ‘N’ (no) otherwise). The bias of C4.5 favours the shortest rule, so that if several rules are equally accurate, a decision tree with the fewest branches is chosen. We chose C4.5 because C4.5 learns decision trees that are human readable and meaningful in terms of the features of the classification model. Every decision tree is structured as a nested hierarchical sequence of simple and easy to understand classification rules. The accuracy of the learning algorithm is generally comparable to the accuracy of other methods. Other learning algorithms are, of course, possible. Indeed, as mentioned earlier, our research group has tried multivariate statistical analysis Ng et al. (2001), logistic regression Wang and Ng (2003), and VSM and Clustering Models Kwong and Ng (2003) as alternative learning algorithms.

Considering all three patterns (multiple-record documents, single-record documents, and application forms), we divide the learning task into three subtasks:

- suitability of a document text component that describes multiple records for one extraction ontology
- suitability of a document text component that represents an individual singleton record for one extraction ontology

- suitability of a form that yields information for one extraction ontology.

C4.5 learns a decision tree for each of the three subtasks.

We use supervised learning to train the classifier. For each application, a human expert selects a set of training documents for the application. To cover the three subtasks (assuming they all exist for the application), the expert should select training documents of each type. For the car-ads application, for example, we selected semistructured HTML documents (e.g., Figure 1(a)) as well as HTML tables containing multiple car ads (e.g., Figure 2) so that the classifier can obtain the knowledge it needs to classify a multiple-record car-ads document. We also selected single car-ad documents (e.g., Figure 4) and documents with HTML forms (e.g., Figure 3(a)). Further, the expert should also select negative training examples. In choosing those that do not belong to the application domain, expert should find documents that include some of the concepts in the ontology (e.g., Figure 1(b) for our car-ads application), so that classifiers can learn the difference between documents that are inside the domain and documents with overlapping concepts that are outside the domain.

For each training document  $d$ , the human expert creates a training example either for the document text component in  $d$  or for one of the form text components in  $d$ , if any.<sup>6</sup> A training example,  $e = (H, c)$ , is a list of values  $H$ , one for each heuristic rule, plus a concept class  $c$ , which is either  $c_P$  for a positive training example or  $c_N$  for a negative training example. Given an extraction ontology  $O$  specifying an application, the values  $H$  include ‘density’, ‘vsm’ and ‘grouping’ measures computed based on the heuristic formulas for density,  $Density(t, O)$ , expected-values,  $Similarity(t, O)$ , and grouping,  $Grouping(t, O)$  of the text component  $t$ . For each individual object set  $o$  (e.g., ‘Make’, ‘Model’ and ‘Price’ in car ads), the values  $H$  also include a density measure computed based on  $Density(t, o)$ . The training data contains three groups of training examples, each of which is for one of the three subtasks. If a training document  $d$  contains a form  $f_i$  relevant to  $O$ , the expert uses the list of heuristic values  $H_{t_{f_i}}$  obtained from the form text component  $t_{f_i}$  to construct a positive training example  $(H_{t_{f_i}}, c_P)$ . Otherwise, if the form  $f_i$  is not relevant to  $O$ , the expert builds a negative training example  $(H_{t_{f_i}}, c_N)$ . If the document is a single-record document relevant to  $O$ , the expert uses the list of heuristic values  $H_{t_d}$  obtained from the document text component  $t_d$  to build a positive training example  $(H_{t_d}, c_P)$ , or vice versa a negative training example. Similarly, the expert uses  $H_{t_d}$  to build a training example for the relevancy of a multiple-record document with respect to  $O$ .

The C4.5 classifier builds a classification model as the output of the training phase. The classification model contains three decision trees: one for single-record document relevancy, one for multiple-record document relevancy, and one for HTML-form relevancy. Figure 7 shows a classification model the classifier built for car ads. Within one tree in the classification model, a node denotes

a predicate using heuristics measures. For example, ‘density’, ‘vsm’ and ‘grouping’ are measures for a text component  $t$  computed based on the heuristic formulas for density, expected-values, and grouping over the car ad extraction ontology. An object-set name (e.g., ‘Model’, ‘Make’, and ‘Price’ in Figure 7) denotes the density measure for an individual object set within the ontology. Each leaf node has a ‘Y’ to designate that a document is relevant to the ontology or an ‘N’ to designate that it is not relevant. The parenthetical numbers  $(x/y)$  following ‘Y’ and ‘N’ for a decision-tree leaf  $L$  give the total number of training examples  $x$  classified for  $L$  and the number of incorrect training examples  $y$  classified for  $L$ .

**Figure 7** Classification model for car ads: (a) form decision tree; (b) multiple-record tree and (c) single-record tree

```
Model <= 0.151389
| Model<=0.000483: N (21.0/2.0)
| Model > 0.000483
| | Make <= 0.0011: N (5.0)
| | Make > 0.0011: Y (6.0/1.0)
Model > 0.151389: Y (18.0)
```

(a)

```
density <= 0.160838
| grouping <= 0.782609: N (24.0)
| grouping > 0.782609: Y (3.0/1.0)
density > 0.160838: Y (23.0)
```

(b)

```
vsm <= 0.760121: N (25.0/3.0)
vsm > 0.760121
| Price <= 0.001826: N (5.0/2.0)
| Price > 0.001826: Y (20.0)
```

(c)

## 6.2 Test phase

Given the classification model built in the training phase, we test the relevancy of HTML documents. The test data consists of heuristic vectors computed according to the heuristic rules in Section 5 for the test documents. The classification model has three decision trees at its disposal and classifies a document with a positive prediction if the classifier classifies the document as positive based on any one of the three decision trees. It is possible that the classifier classifies a test document as both a single-record document and a multiple-record document relevant to the extraction ontology based on the document text component in the document. Moreover, it also could predict that one relevant document contains both relevant forms as well as a singleton record or multiple records for the application. We are, however, only interested in a prediction, and thus we declare a document to be relevant if any one of the three trees in the classification model returns  $c_P$ , a positive result.

In addition to the text components within the document, the classifier can check available linked pages

and pages returned from form filling to further evaluate the document. The HTML table in Figure 2 contains several links, some of which lead to more detailed descriptions of the car ads in the document. Figure 4 is one of the linked pages from the top page in Figure 2. If we can determine that a linked page (e.g., the document in Figure 4) is relevant to the application, we can use this information to help classify the top page (e.g., the document in Figure 2). Intuitively, if a top page leads to multiple relevant linked pages, we have more confidence that the top page contains multiple records that are of interest. Because of the expense of retrieving potentially many dozens of linked pages, however, the algorithm does not explore all linked pages from the top-page HTML document  $d$ . Instead, we first locate a group of potentially useful links from  $d$ . Since we believe that the useful links in a multiple-record document are likely to all be together in a common repository, the procedure to locate useful links first groups links in  $d$  by (longest) common URL prefix and then sorts the groups of links in descending order based on the number of links in each group. The number of the links that are likely of interest is the largest in a relevant multiple-record document. To both discard spurious groups of links with only one or two members and to avoid processing all the links in a group, we choose a small threshold  $N$  experimentally (we chose  $N = 5$ ). Then, if the number of the links in a group is less than  $N$ , we ignore the evaluation of the group, and if the number of links is greater than  $N$ , we only evaluate  $N$  of them. We evaluate the links in a group by checking the relevancy of the top-level document  $d$  with the text of the linked pages inserted into  $d$ .

Besides linked pages, we also use information on pages returned by automatic form filling (Liddle et al., 2001; Yau, 2001).<sup>7</sup> For a document  $d$  containing a form  $f$  and having associated documents  $s$  retrieved by automatic form filling, we use one of two strategies, choosing between them based on user preference. If a user favours recall, we evaluate  $s$  when the classifier classifies  $d$  as irrelevant based on  $f$  alone; otherwise, if a user favours precision, we evaluate  $s$  when the classifier classifies  $d$  as relevant based on  $f$  alone.

## 7 Empirical evaluation

We experimented with four applications: car ads (our running example in this paper), obituaries, real-estate house properties, and university faculty pages. Our goals were to evaluate categorisation performance over multiple kinds of HTML documents for real-world applications.

### 7.1 Applications and HTML documents

The car-ad and faculty applications are representative of many simple applications, whereas the real-estate and obituary applications are representative of more complex applications. These two more complex applications have more object sets and relationship sets than the simple applications, and the object and relationship sets are

configured in more complex ways. In the obituary application, for example, one relationship set is ternary, more than one object set is nonlexical, and several relationship sets are specialisations—for example, both *Birth Date* and *Death Date* are specialisations of *Date*.

For the car-ad and real-estate applications, we collected four sets of HTML documents: semi-structured HTML documents, HTML tables, HTML forms, and negative documents. For the obituary application, we collected only semistructured documents, HTML form documents, and negative documents. (Obituaries rarely, if ever, appear as tables having attributes such as *Deceased Name*, *Age*, *Death Date*, etc.) For the faculty application, we took a sampling of documents covering items such as faculty, staff, project, course, and department from the University WebKB data set Craven et al. (1998). We divided the documents for each application into two sets: training documents and test documents. For each of the car-ad, obituary, and real-estate applications, there were three sets of training examples, one each for single-record documents, multiple-record documents, and application-forms. For the faculty application, there was one set of training examples for single-record documents. Each training set consisted of 50 examples, half of which were positive, and half of which were negative.

Table 3 shows the distributions of semistructured HTML documents, HTML tables, and HTML forms in the test documents. Each test set consisted of 100 documents, half of which were negative. For the car-ads application, 10 of the semistructured HTML documents contained multiple-record car ads, and 10 contained single-record car ads. All the HTML tables contained multiple-record car ads. For the obituary application, the semistructured HTML documents contained 25 multiple-record documents and 15 single-record documents. Among the 25 multiple-record obituary documents, 15 contained only partial obituaries. These 15 documents led to linked pages, some of which contained complete obituary records. For the real-estate application, the semistructured HTML documents contained 13 multiple-record documents and 13 single-record documents. Among the 13 multiple-record real-estate documents, 6 contained only partial house-property records. These six documents led to linked pages where detailed house-property information was provided. The one HTML table contained multiple-record house properties. The HTML form documents for each application contained application forms but no single- or multiple-records of interest. For the faculty application, all the semistructured HTML documents contained single-record faculty home pages.

The negative documents collected for each application contained documents with applications similar to those of the extraction ontologies. For example, we included forms to retrieve used auto parts, car reviews, and motorcycle sales to test the classifier trained for car-ads; we included birth and marriage announcements, genealogy records and forms, and bibliographies to test the classifier trained for obituaries; we included house inspection, house mortgage quotes, and real-estate buying guides to test the

classifier trained for house properties; and we included research projects, university staff, courses, and department pages to test the classifier trained for faculty pages. We assumed that if our categorisation methodology could sort out differences among closely related applications, it could easily reject vastly different application documents (e.g., country descriptions, rail and flight schedules, and molecular biology data). To verify this assumption, we used some of the pages collected for one application as negative examples for the other three applications.

Even though some of the semistructured HTML documents and HTML table documents in Table 3 contained additional irrelevant application forms (e.g., user registration forms), we expected that the classifiers would produce appropriate predictions based on the document text components that appear in the documents rather than the irrelevant form text components. For the HTML form documents, since they did not contain application records, we expected that the classifiers would produce the positive predictions using the application-form decision trees based on form text components that appear within forms.

**Table 3** Test documents for car ads and obituaries

Application	Semistructured	HTML table	HTML form	Negative
Car	20	20	10	50
Obituary	40	0	10	50
Real Estate	26	1	23	50
Faculty	50	0	0	50

**Figure 8** Classification model for obituaries: (a) form decision tree; (b) multiple-record tree and (c) single-record tree

```

DeceasedPerson <= 0.004638
| DeathDate <= 0.00402: N (27.0/2.0)
| DeathDate > 0.00402: Y (4.0)
DeceasedPerson > 0.004638: Y (19.0)
(a)

```

```

density <= 0.069183: N (23.0)
density > 0.069183
| grouping <= 0.2: N (2.0)
| grouping > 0.2: Y (25.0)
(b)

```

```

DeceasedPerson <= 0
| FuneralDate <= 0.003459: N (19.0/1.0)
| FuneralDate > 0.003459: Y (2.0)
DeceasedPerson > 0
| DeathDate <= 0
| | Age <= 0.000489
| | | RelativeName <= 0.001017: Y (5.0/1.0)
| | | RelativeName > 0.001017: N (2.0)
| | | Age > 0.000489: Y (18.0)
| | DeathDate > 0: N (4.0)
(c)

```

## 7.2 Classification models

Figures 7–10 respectively show the classification models the classifiers built for car ads, obituaries, house properties, and faculty pages. For an extraction ontology  $O$ , each classification model except for

the faculty application contains three decision trees: one for multiple-record documents, one for single-record documents, and one for application forms. The classification model for the faculty application contains one decision tree for single-record documents.

Given the decision trees for the four applications in Figures 7–10, we can see that the classifiers use different combinations of heuristics to check the relevancy, and that all are useful. Over all four applications the classifiers largely exploit density heuristics. Figure 8 shows that ‘vsm’, the expected-values heuristic, was not useful for the classifier of the obituary application. Figures 9 and 10 show that ‘grouping’, the grouping heuristic, was not useful for the classifiers of the real-estate and faculty applications.

**Figure 9** Classification model for house properties: (a) form decision tree; (b) multiple-record tree and (c) single-record tree

```

Price <= 0.007065
| MLS <= 0.014925
| | Fireplace <= 0.003282: N (26.0/1.0)
| | Fireplace > 0.003282: Y (2.0)
| | MLS > 0.014925: Y (3.0)
| Price > 0.007065: Y (19.0)
(a)

```

```

Price <= 0.003259: N (24.0/1.0)
Price > 0.003259
| vsm <= 0.15202
| | density <= 0.150538: N (2.0)
| | density > 0.150538: Y (2.0)
| vsm > 0.15202: Y (22.0)
(b)

```

```

Bathroom <= 0.001235: N (24.0/1.0)
Bathroom > 0.001235
| Price <= 0: N (2.0)
| Price > 0: Y (24.0)
(c)

```

**Figure 10** Classification model for faculty pages

```

vsm <= 0.414434: N (11.0)
vsm > 0.414434
| Title <= 0.003236
| | ResearchInterest <= 0.009385: N (9.0)
| | ResearchInterest > 0.009385: Y (5.0/1.0)
| Title > 0.003236
| | ResearchProject <= 0.014448
| | | density <= 0.036107: N (2.0)
| | | density > 0.036107: Y (21.0)
| | ResearchProject > 0.014448: N (2.0)

```

## 7.3 Experiments

For each application, we performed four sets of experiments.

- We measured the basic precision, recall, and the  $F$ -measure for each application by simply taking a page, analysing it, and categorising it as relevant or irrelevant to the application

- As a baseline for comparison against this simple categorisation, we applied the Naive Bayes classifier (Baker and McCallum, 1998) implemented in Rainbow (McCallum, 1996)
- We then added to our basic experimentation the potential to consider linked pages and pages returned from filled-in forms. For this part of the experiment, we considered both precision-biased and the recall-biased methods.
- We conducted studies to evaluate the contributions of individual heuristics and pairs of heuristics over our three basic heuristics: *Densities*, *Expected Values*, and *Grouping*.

For all experiments, we evaluated the performance of the classifiers on the test documents described in Table 3.

### 7.3.1 Basic results

Given

- the number of relevant documents  $N$  determined by a human expert, which is listed in Column 2 in Table 4
- the number of correct relevant documents  $C$  selected by our approach, which is listed in Column 3
- the number of incorrect relevant documents  $I$  selected by our approach, which is listed in Column 4, we computed the recall ratio as  $R = C/N$ , the precision ratio as  $P = C/(C + I)$ , and the  $F$ -measure as  $F = 2/(1/R + 1/P)$ .

We report all these values as percentages in Table 4.

**Table 4** Basic results

Application	Number relevant	Number correct	Number incorrect	Recall (%)	Precision (%)	$F$ -measure (%)
Car Ad	50	48	4	96	92	94
Obituary	50	49	3	98	94	96
Real Estate	50	49	2	98	96	97
Faculty	50	45	4	90	92	91

As a basic result, observe that the  $F$ -measures ranged from a low of 91% to a high of 97%. By far, the system correctly categorised the vast majority of the 400 documents. It did, however, categorise some relevant documents as irrelevant: 2 for *Car Ad*, 1 for *Obituary*, 1 for *Real Estate*, and 5 for *Faculty*, and some irrelevant documents as relevant: 4 for *Car Ad*, 3 for *Obituary*, 2 for *Real Estate*, and 4 for *Faculty*.

### 7.3.2 Basic baseline comparison

Rainbow is an executable program that does document classification. We used its Naive Bayes classifier to classify all test documents for the four applications. For the Naive Bayes classifier, we trained a categorisation system for each application with the same collection of training documents

as for our approach. We reported the evaluation measures as percentages in Table 5.

Observe that the  $F$ -measure values for our categorisation method are uniformly better than for the Naive Bayes classifier. The Naive Bayes classifier, however, achieved 100% precision for the car-ad application, correctly classifying every single negative example. It, however, only classified 72% of the positive examples as relevant, resulting in an  $F$ -measure of 84% compared to 94% for our method.

**Table 5** Experimental results of Rainbow

Application	Number relevant	Number correct	Number incorrect	Recall (%)	Precision (%)	$F$ -measure (%)
Car Ad	50	36	0	72	100	84
Obituary	50	46	7	92	87	89
Real Estate	50	49	23	98	68	80
Faculty	50	35	8	70	81	75

### 7.3.3 Form filling and linked pages

As explained earlier, in addition to the text components that appear in a document, we can also exploit auxiliary information such as linked pages or retrieved documents obtained by form filling. First, we applied the strategy that favours recall. Our classifiers re-evaluated the negative responses using retrieved documents obtained by applying form filling. By this strategy, our classifier was able to improve its recall results from 96% to 98% for the car-ad application and from 98% to 100% for the real-estate application. Second, we applied a strategy that favours precision. After filling in forms and obtaining results, the classifiers re-evaluated the application forms that they had incorrectly classified as positive responses. By this strategy, our classifier was able to improve its precision results from 96% to 98% for the real-estate application.

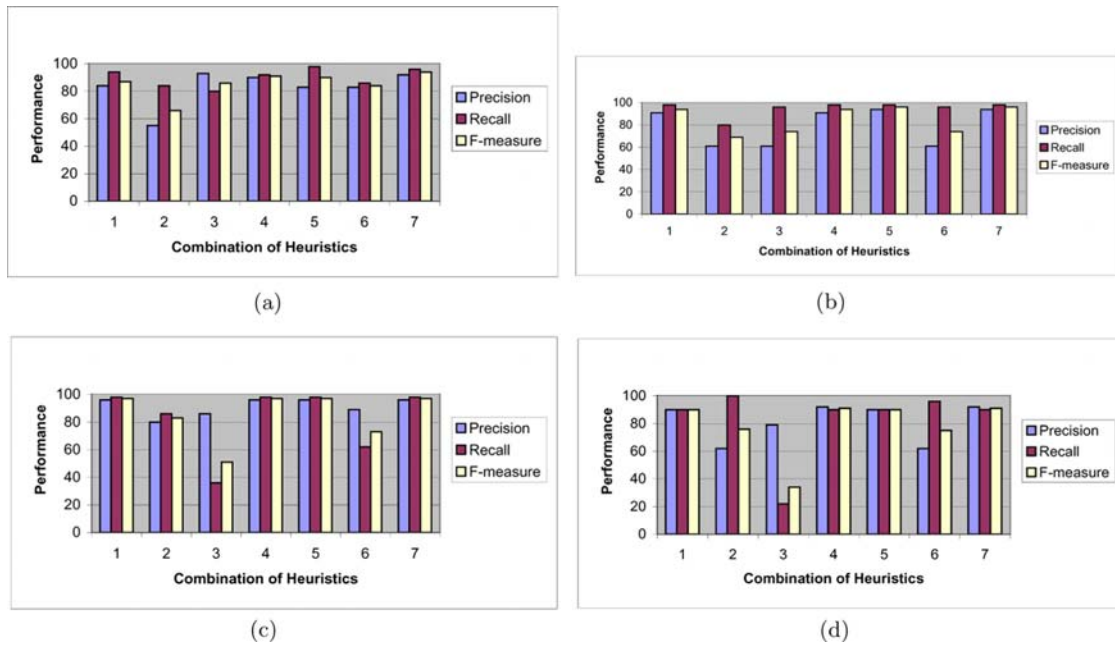
The other auxiliary information we use in our system is information on linked pages. By this strategy, our classifier for the obituary application was able to improve its recall from 98% to 100%. Note that we included 15 HTML documents containing only partial obituaries and 6 HTML documents containing only partial house properties. The test results show that the classifier for obituaries missed only one of these pages without analysing linked pages and that the classifier for house properties classified all partial house property pages correctly. This means that the partial obituaries and house properties pages usually provided enough informative information for the classifier to correctly classify them as relevant documents.

**Table 6** Combination of three kinds of heuristics

Combination	Densities	Expected-values	Grouping
1	+		
2		+	
3			+
4	+	+	
5	+		+
6		+	+
7	+	+	+



**Figure 11** Performance comparison of heuristics combinations: (a) car ad application; (b) obituary application; (c) real estate application and (d) faculty application (see online version for colours)



#### 7.3.4 Contribution of heuristics

To determine the individual and pair-wise contribution of the various heuristics, we evaluated performance by applying different combinations of heuristics. Table 6 shows the seven combinations, where ‘+’ denotes the heuristic or heuristics in the corresponding column that are in use. Figure 11 shows the contribution of the three kinds of heuristics (densities, expected-values, and grouping) to the overall performance in the four applications. The  $x$ -axis lists the seven combinations of the three kinds of heuristics. The  $y$ -axis shows the performance as a percentage.

For all four applications, Figure 11 shows that the density measures are important (Columns 1, 4, 5, and 7 are generally the best). When the classifiers exploited only the density heuristic, to evaluate the test documents for the applications (Column 1), the classifiers for the obituary, real-estate, and faculty applications achieved about 90% for all measures (precision, recall, and  $F$ -measure). For the car-ad application, however, the classifier applying only densities achieved a precision 84% and a recall 94%. Using the expected-values heuristic alone (Column 2), the classifiers achieved an  $F$ -measure of less than 70% for the car-ad and obituary applications, achieved an  $F$ -measure of 76% for the faculty application, and achieved an  $F$ -measure of 83% for the real-estate application. Using the grouping heuristic alone (Column 3), the classifiers achieved  $F$ -measures less than 50% for both the real-estate and faculty applications. The classifiers achieved an  $F$ -measure of 74% for the obituary application and achieved an  $F$ -measure of 85% for the car-ad application. Even when the classifiers used both the expected-values heuristic and the grouping heuristic together (Column 6), performance for none of the four applications improved

significantly. Table 6 shows that the classifiers for all four applications achieved the highest  $F$ -measures by applying all heuristics together.

Even though the classifiers performed well by exclusively applying density heuristics, we observed that the grouping and expected-values heuristics were helpful in classifying two kinds of documents:

- negative documents that contain abundant values and keywords specified for ontology object sets
- positive documents that contained a large amount of irrelevant data.

By applying only density heuristics, classifiers can incorrectly classify the first kind of negative documents as relevant because of their high density measures and incorrectly rejected the second kind of documents because of their low density measures. For example, the classifier applying only densities for the car-ad application incorrectly classified a car-news page as relevant since it contained many car-make and car-model values. Compared with the performance achieved by applying all heuristics, the classifiers applying only density heuristics incorrectly classified five more irrelevant documents as car-ads, incorrectly rejected one more car-ad document, incorrectly classified two more irrelevant documents as obituaries, and incorrectly classified one more irrelevant document as a faculty page.

It is clear that the density heuristics are dependent on and sensitive to the specification of extraction ontologies. The other two heuristics, expected values and grouping, are also mainly determined by the specification of extraction ontologies. Thus, when porting to a new application domain, as long as the extraction ontologies are well defined, our empirical evaluation shows that our approach

should be able to recognise relevant HTML documents with both high precision and high recall.

## 8 Conclusions and future work

We presented an approach to categorise HTML documents with respect to extraction ontologies. The approach applies to applications in which users have long-term needs for specific information from HTML documents in a particular domain of interest. The methodology proceeds in several steps:

- 1 Users describe information of interest by giving an extraction ontology.
- 2 Given an HTML document  $d$  to categorise as relevant or not relevant to an extraction ontology  $O$ , the system extracts the data it recognises in  $d$  with respect to  $O$ .
- 3 Given the extracted data, the system calculates several heuristic measures to characterise the HTML document.
- 4 Based on these heuristic measures, the system uses machine-learned rules to categorise the document as relevant or not relevant.

Results for the tests we conducted showed that the recognition  $F$ -measures were above 90% for all four of our applications. Recall ranged from 90% (faculty) on the low end to 96% (car ads) and 98% (obituaries and real estate) on the high end. Precision ranged from 92% (car ads and faculty) on the low end to 94% (obituaries) and 96% (real estate) on the high end. Our experiments also showed that we can further improve performance by considering linked pages and documents retrieved by submitting default forms.

Our approach is reliable. It classifies documents with high precision and recall. It is also flexible. The heuristics, learning algorithms, and training documents used in the approach are extensible. If new heuristics appear useful, we can immediately use them without having to change our fundamental approach. Our approach has good performance characteristics. Execution only requires a single linear-time pass over a document. Porting the system to a new domain requires a one-time cost to build an extraction ontology and train the classifiers. The effort involved is relatively expensive, and thus we suggest our methodology only for cases in which a user has a long-term interest in a particular application.

Our future work can expand in several different directions.

- We can test our approach on more applications. After fixing our methodology using the car-ads and obituaries application domains, we ported our methodology to two additional application domains (the real-estate and faculty application domains). The methodology ported as expected. To find the

limits of our methodology and characterise more precisely the kinds of domains to which it applies, we can explore applications on the fringes of our expectations – applications whose domains are not so narrow and not so data rich.

- We can investigate ways to enhance the methodology. Can additional or replacement heuristics improve our methodology? We can, for example, experiment with sophisticated record-separation heuristics (Embley et al., 1999c), sophisticated record-reconfiguration heuristics (Embley and Xu, 2000), and sophisticated value/object-set disambiguation heuristics (Embley et al., 1999a). We can also experiment with a meta-learning strategy to train a high-level classifier over several different low-level classifiers including C4.5 as described in this paper, multivariate analysis (Ng et al., 2001), logistic regression (Wang and Ng, 2003), and VSM and Clustering Models (Kwong and Ng, 2003).
- We can do a deeper level analysis of an HTML document and check relevancy based on appropriate subparts of the document rather than the entire document. Many Web pages include material, such as advertisements, that have little, if anything, to do with their main content. Our heuristics may be overwhelmed by the volume of irrelevant material on a page. If we can subdivide these types of pages and test component parts of pages individually, our generated decision-tree classifiers should be able to identify relevant component parts and thus not be overwhelmed by irrelevant material.

Although we have offered an interesting approach to categorisation based on extraction ontologies and have shown that it has good performance results in an experimental setting, much can still be done to refine our methodology and much is still required to take it beyond its experimental setting.

## Acknowledgements

We appreciate George Nagy taking the time to read and comment on a draft of our paper. His comments and insights were particularly helpful.

## References

- Baeza-Yates, R. and Ribeiro-Neto, B. (1999) *Modern Information Retrieval*, Addison Wesley, Menlo Park, California.
- Baker, L.D. and McCallum, A.K. (1998) 'Distributional clustering of words for text classification', *Proceedings of the 21st Annual International Conference on Research and Development in Information Retrieval (SIGIR'98)*, Melbourne, Australia, pp.96–103.

- Chakrabarti, S. (1999) 'Recent results in automatic web resource discovery', *ACM Computing Surveys*, Vol. 31, No. 4, December, Article No. 17.
- Chakrabarti, S., van den Berg, M. and Dorn, B.E. (1999) 'Focused crawling: a new approach for topic-specific resource discovery', *Computer Networks*, Vol. 31, pp.1623–1640.
- Chu-Carroll, J., Prager, J., Czuba, K., Ferrucci, D. and Duboue, P. (2006) 'Semantic search via xml fragments: a high-precision approach to ir', *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'06)*, Seattle, Washington, USA, 6-11 August, pp.445–452.
- Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K. and Slattery, S. (1998) 'Learning to extract symbolic knowledge from the World Wide Web', *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)*, Madison, Wisconsin, July, pp.509–516.
- Dayanik, A., Lewis, D.D., Madigan, D., Menkov, V. and Genkin, A. (2006) 'Constructing informative prior distributions from domain knowledge in text classification', *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'06)*, Seattle, Washington, USA, 6–11 August, pp.493–500.
- Dill, S., Eiron, N., Gibson, D., Gruhl, D., Guha, R., Jhingran, A., Kanungo, T., McCurley, K.S., Rajagopalan, S., Tomkins, A., Tomlin, J.A. and Zien, J.Y. (2003) 'A case for automated large scale semantic annotations', *Journal of Web Semantics*, Vol. 1, No. 1, December, pp.115–132.
- Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V.C. and Sachs, J. (2004) 'Swoogle: a search and metadata engine for the semantic web', *Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management*, Washington DC, November, pp.652–659.
- Embley, D.W., Kurtz, B.D. and Woodfield, S.N. (1992) *Object-oriented Systems Analysis: A Model-Driven Approach*, Prentice Hall, Englewood Cliffs, New Jersey.
- Embley, D.W., Campbell, D.M., Jiang, Y.S., Liddle, S.W., Lonsdale, D.W., Ng, Y-K. and Smith, R.D. (1999a) 'Conceptual-model-based data extraction from multiple-record web pages', *Data and Knowledge Engineering*, Vol. 31, No. 3, November, pp.227–251.
- Embley, D.W., Fuhr, N., Klas, C-P. and Roelleke, T. (1999b) 'Ontology suitability for uncertain extraction of information from multi-record web documents', *Datenbank Rundbrief*, Vol. 24, pp.48–53.
- Embley, D.W., Jiang, Y.S. and Ng, Y-K. (1999c) 'Record-boundary discovery in web documents', *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data (SIGMOD'99)*, Philadelphia, Pennsylvania, May–June, pp.467–478.
- Embley, D.W. and Xu, L. (2000) 'Record location and reconfiguration in unstructured multiple-record web documents', *Proceedings of the Third International Workshop on the Web and Databases (WebDB2000)*, Dallas, Texas, May, pp.123–128.
- Embley, D.W., Ng, Y-K. and Xu, L. (2001) 'Recognizing ontology-applicable multiple-record web documents', *Proceedings of the 20th International Conference on Conceptual Modeling (ER2001)*, Yokohama, Japan, November, pp.555–570.
- Euzenat, J. and Shvaiko, P. (2007) *Ontology Matching*, Springer-Verlag, Heidelberg, Germany.
- Fang, H. and Zhai, C.X. (2006) 'Semantic term matching in axiomatic approaches to information retrieval', *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'06)*, Seattle, Washington, USA, 6–11 August, pp.115–122.
- Gruber, T.R. (1993) 'A translation approach to portable ontology specifications', *Knowledge Acquisition*, Vol. 5, No. 2, pp.199–220.
- Kiryakov, A., Popov, B., Terziev, I., Manov, D. and Ognyano, D. (2004) 'Semantic annotation, indexing, and retrieval', *Journal of Web Semantics*, Vol. 2, No. 1, December, pp.49–79.
- Kobayashi, M. and Takeda, K. (2000) 'Information retrieval on the web', *ACM Computing Surveys*, Vol. 32, No. 2, pp.144–173.
- Kwong, L.W. and Ng, Y-K. (2003) 'Performing binary-categorization on multiple-record web documents using information retrieval models and application ontologies', *World Wide Web: Internet and Web Information Systems*, Vol. 6, No. 3, September, pp.281–303.
- Laender, A.H.F., Ribeiro-Neto, B. and Da Silva, A.S. (2002) 'DEByE - data extraction by example', *Data and Knowledge Engineering*, Vol. 40, No. 2, pp.121–154.
- Liddle, S.W., Yau, S.H. and Embley, D.W. (2001) 'On the automatic extraction of data from the hidden web', *Proceedings of the International Workshop on Data Semantics in Web Information Systems (DASWIS-2001)*, Yokohama, Japan, November, pp.106–119.
- Liddle, S.W., Scott, D.T., Embley, D.W. and Yau, S.H. (2002) 'Extracting data behind web forms', *Proceedings of the Joint Workshop on Conceptual Modeling Approaches for E-business: A Web Service Perspective (eCOMO 2002)*, *Lecture Notes in Computer Science (LNCS 2784)*, Tampere, Finland, October, pp.402–413.
- Lin, J. and Demner-Fushman, D. (2006) 'The role of knowledge in conceptual retrieval: a study in the domain of clinical medicine', *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'06)*, Seattle, Washington, USA, 6-11 August, pp.99–106.
- McCallum, A. (1996) *Bow: A Toolkit for Statistical Language Modeling, Text Retrieval, Classification and Clustering*, [www.cs.cmu.edu/~mccallum/bow](http://www.cs.cmu.edu/~mccallum/bow)
- Ng, Y., Tang, J. and Goodrich, M. (2001) 'A binary-categorization approach for classifying multiple-record web documents using application ontologies and a probabilistic model', *Proceedings of the 7th International Conference on Database Systems for Advanced Applications (DASFAA 2001)*, Hong Kong, April, pp.58–65.
- Quinlan, J.R. (1993) *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, California.
- Sebastaini, F. (2002) 'Machine learning in automated text categorization', *ACM Computing Surveys*, Vol. 34, No. 1, March, pp.1–47.
- Schapiro, R.E., Rochery, M., Rahim, M.G. and Gupta, N. (2002) 'Incorporating prior knowledge into boosting', *Proceedings of the Nineteenth International Conference on Machine Learning (ICML'02)*, San Francisco, CA, USA, 8–12 July, pp.538–545.

Segev, A. and Gal, A. (2007) ‘Putting things in context: a topological approach to mapping contexts to ontologies’, *Journal on Data Semantics*, August, pp.113–140.

Wang, Q. and Ng, Y. (2003) ‘An ontology-based binary-categorization approach for recognizing multiple-record web documents using a probabilistic retrieval model’, *Journal of Information Retrieval*, Vol. 6, Nos. 3–4, September–December, pp.295–332.

Wu, X. and Srihari, R. (2004) ‘Incorporating prior knowledge with weighted margin support vector machines’, *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '04)*, Seattle, WA, USA, 22–25 August, pp.326–333.

Xu, L. and Embley, D.W. (2006) ‘A composite approach to automating direct and indirect schema mappings’, *Information Systems*, Vol. 31, No. 8, pp.697–732.

Yau, S.H. (2001) *Automating the Extraction of Data Behind web Forms*, Master’s Thesis, Brigham Young University, Provo, Utah, December.

**Websites**

Websites Demos page for BYU data extraction group, <http://www.deg.byu.edu/multidemos.html>

Google, [www.google.com](http://www.google.com)

OWL Web Ontology Language Reference Manual, <http://www.w3.org/TR/owl-ref/>, W3C (World Wide Web Consortium).

**Notes**

<sup>1</sup>Experience has shown that computer science students can build an extraction ontology for a data-rich, narrow domain of interest in a few dozen person hours. Students have built extraction ontologies for a wide variety of applications including such diverse areas such as digital cameras, prescription drugs, campgrounds, and computer jobs (<http://www.deg.byu.edu/multidemos.html>).

<sup>2</sup>Extraction ontologies are akin to techniques in wrapper technology (Laender *et al.*, 2002), automated semantic annotation (Dill *et al.*, 2003; Kiryakov *et al.*, 2004), and ontology matching (Euzenat and Shvaiko, 2007; Xu and Embley, 2006). We thus point out the possibility that these technologies may also be useful for text categorisation. To the best of our knowledge, no other efforts outside of colleagues within our research group make direct use of these technologies to do text categorisation.

<sup>3</sup>We mention in passing that the ontology language on which extraction ontologies are based has been fully formalised and is equivalent to predicate calculus (see Appendix A of Embley *et al.* (1992)). Further, the subset of the ontology language we typically use is equivalent in power and complexity to OWL-DL (<http://www.w3.org/TR/owl-ref/>).

<sup>4</sup>It is easy to see that the variance might be useful, as well, but we found that the expected numbers were sufficient to get good results for the examples we tried.

$$^5 0.996 = \frac{(0.975 \times 16 + 0.925 \times 10 + 0.908 \times 12 + 0.45 \times 6 + 0.8 \times 11 + 2.1 \times 29 + 1.15 \times 15)}{(\sqrt{0.975^2 + 0.925^2 + 0.908^2 + 0.45^2 + 0.8^2 + 2.1^2 + 1.15^2}) \times (\sqrt{16^2 + 10^2 + 12^2 + 6^2 + 11^2 + 29^2 + 15^2})}$$

<sup>6</sup>Typically, an informational HTML document contains its primary information either directly on the page or behind one of its forms. The human expert should train the classifier by selecting the component that contains the primary information for the document.

<sup>7</sup>We point out that automatic form filling does not always yield results as explained in Liddle *et al.* (2001) and Yau (2001). Thus, we can only apply this technique when automatic form filling does yield results.