University of
BRISTOL

Price, S. (2004). *A review of the state of the art in Machine Learning on the Semantic Web: Technical Report CSTR-05-003*. Department of Computer Science, University of Bristol.
http://www.cs.bris.ac.uk/Publications/Papers/2000978.pdf

Early version, also known as pre-print

Link to publication record on the Bristol Research Portal
PDF-document

University of Bristol – Bristol Research Portal

General rights

A review of the state of the art in Machine Learning on the Semantic Web

Technical Report

Simon Price

Department of Computer Science, University of Bristol, Bristol BS8 1UB, UK

simon.price@bristol.ac.uk

October 12, 2004

Abstract

This paper reviews the current state of the art of machine learning applied to the Semantic Web. It looks at the Semantic Web and its languages, including RDF and OWL, from a machine learning perspective. Trends in the Semantic Web are mentioned throughout and the relationship with Web Services is examined. Applications are discussed with recent examples and pointers to data sets. Finally, the emerging field of Semantic Web Mining is introduced.

1   Introduction

This paper reviews the current state of the art of machine learning applied to the Semantic Web. The intended readership is researchers and practitioners in the machine learning and computational intelligence community. No substantial prior knowledge of the Semantic Web is assumed as the paper includes a brief tutorial introduction to the Semantic Web, given from a machine learning perspective.

In terms of focus, the review exhibits a distinct bias towards my own personal interest in logical approaches to machine learning and, in particular, learning from structured data. However, I believe that this bias is appropriate given the structured nature of the Semantic Web and its own inbuilt logics bias.

The rest of this paper is divided into three parts: an introduction to the Semantic Web with occasional comments relating to machine learning; a review of existing machine learning applications with discussion of potential applications; and finally, some concluding remarks.

2   Introduction to the Semantic Web

The Semantic Web initiative was set up by the World Wide Web Consortium (W3C) to enable, "... an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation" [1].

Given the requirement for backward compatibility with the existing Web and the exponential growth of Web data, this is a massive engineering task by any measure. Nevertheless, to consider the Semantic Web only as an engineering task would be to ignore the substantial scientific challenges that need to be addressed in order to fulfil the goals of the initiative.

The Semantic Web, following in the tracks of the original Web, is a continuously evolving system rather than a static entity. This is evident from the W3C's definition [1] given below.

> **Definition:** *The Semantic Web is the representation of data on the World Wide Web. It is a collaborative effort led by W3C with participation from a large number of researchers and industrial partners. It is based on the Resource Description Framework (RDF), which integrates a variety of applications using XML for syntax and URIs for naming.*

Although the design for the Semantic Web is based on RDF and URIs, there are also a series of other language layers that sit on top of this foundation layer. A brief introduction to each of these is given below along with observations from the perspective of machine learning.

2.1   Uniform Resource Identifier (URI)

The Uniform Resource Identifier (URI) addressing scheme is well known as the means of locating documents on the Web. Typical URIs are short strings that start with scheme names like "http:", "mailto:" or "ftp:". In their traditional usage, each URI refers to a resource, or a specific point within a resource. Most users would expect this resource to be located somewhere on the Web. In fact, the URI specification [2] does not require this and most Web users would be surprised to discover that URIs support references to entities that are not even network retrievable. The Semantic Web makes good use of this global referencing feature of URIs to allow statements to be made about anything that has an identity. So, as well as continuing to support references to Web resources, such as HTML pages and other online documents, URIs also permit references to entities such as human beings, corporations, bound books in a library or more ethereal entities such as concepts and relations.

URI's observe a syntax that is familiar to any Web user and, while that syntax is governed by [2], the ownership and creation is delegated: anyone can create a URI. To allow decentralised growth of the Web, there is no central repository or clearing house for URIs. Consequently, multiple URIs can refer to a single entity. This clearly poses some interesting problems in testing for equality (or equivalence) between URIs.

2.2   Resource Description Framework (RDF)

Resource Description Framework (RDF) [3][4] is a language that utilises triples of URIs. An RDF statement is a {subject, predicate, object} triple of URIs. As a graph model, this corresponds to a directed graph with subject and object as labelled nodes connected by the labelled directed arc predicate. Literal values may be used in an RDF triple in place of the object; any literal values are treated by an RDF parser as an anonymous URI. An example triple for the statement,

```
"http://www.example.org/index.html has a creator whose value is the literal John Smith"
```

could be represented as the following informal plain text triple.

subject http://www.example.org/index.html
predicate http://purl.org/dc/elements/1.1/creator
object John Smith

The formal, standard serialization of RDF is in XML, which makes it easy for machines to process but is not very human-readable. For a tutorial on the details of RDF and its XML serialization and RDF graph models, see [5]. A Prolog notation is both more human-readable than XML and more likely to be familiar to the machine learning community. The above triple can be represented in Prolog as:

```
rdf( 'http://www.example.org/index.html',
     'http://purl.org/dc/elements/1.1/creator',
     literal('John Smith') ).
```

More complex statements can be made in RDF by using multiple triples - i.e. graphs of triples. Figure 1, shows a graph of four triples describing the editor of the "RDF/XML Syntax Specification (Revised)" document, giving his name and his homepage. One initially confusing peculiarity of RDF is the widespread use of blank nodes (sometimes called anonymous nodes in the literature) to bind triples together without creating a public URI. Using just URIs, literals and blank nodes it is possible to model a wide range of data structures whilst retaining the practical advantage that graphs consisting of triples from diverse sources can be easily merged.
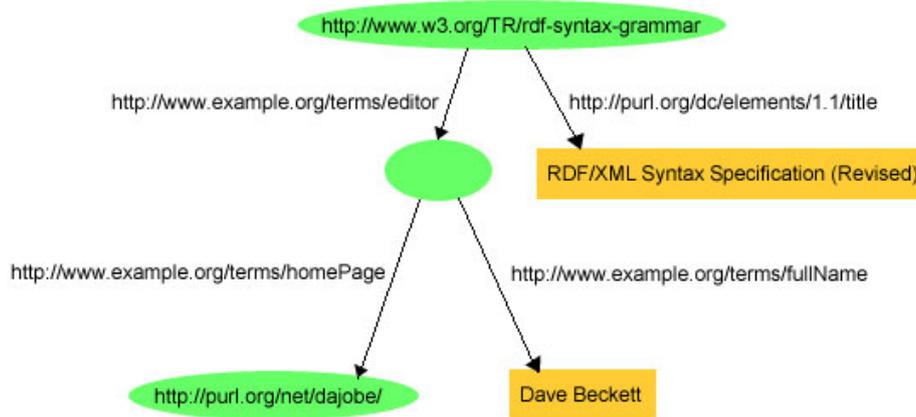
Figure 1 - Semantic Web Layers [5]

There is no shortage of stable and mature tools for parsing, storing and querying RDF. Some popular open source examples for procedural programming languages include Jena (JAVA), Redland (C++) and Sesame (JAVA). See [44] for a more complete list of RDF tools and, so called, triplestores. RDF is also supported in higher-level languages such as Haskell and Prolog, the latter being particularly relevant for many areas of machine learning. Support for RDF in Prolog is currently exemplified by the open source SWI-Prolog [6] with its accompanying RDF library, designed to support efficient storage and in-memory querying of up to 40 million triples (requiring circa 100MB RAM per million triples).

One word of caution with regard to representing structured data in Prolog is that, as reported by [7] for learning first order rules, no single representation is best in all cases. So a transformation away from SWI-Prolog's optimised `rdf/3` format may be necessary in order to optimise the learner.

2.3 Semantic Web Layers

The diagram below, reproduced from [8], shows the language layers currently envisaged by the W3C Semantic Web Activity. Only the layers from RDF down are mentioned in the W3C's definition of the Semantic Web. Useful applications can be found even using only these lower levels (e.g. RSS 1.0 news feeds and Weblogs [9] and Dublin Core metadata [10]). On the Web today, the lower down the layers one looks, the more applications one finds. Conversely, the higher one looks above the RDF layer, the fewer applications of these technologies there are to be found (at present).
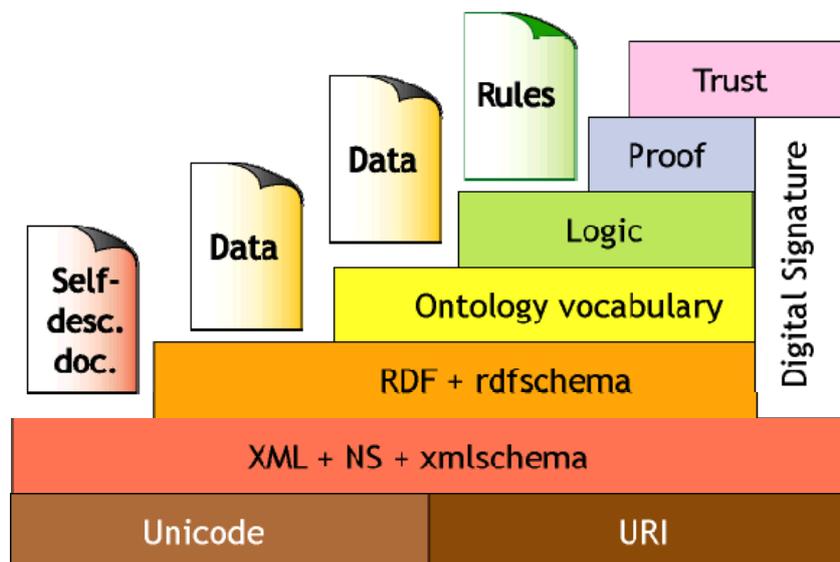

Figure 2 - Semantic Web Layers [8]

3

2.4 RDF Schema

Closely related to RDF is RDF Schema [11]. RDF Schema is a language for describing properties and classes of RDF resources, with semantics for generalization-hierarchies of such properties and classes. It is a simple datatyping model for RDF that introduces classes, *is-a* relationships and properties as well as some range and domain restrictions.

RDF Schema has recently been renamed as "RDF Vocabulary Description Language", but the older name is used in this paper for consistency with the literature. It is also useful to note that some of the literature collectively refers to RDF plus RDF Schema as RDF(S).

2.5 Ontology Vocabulary

In practice much of the ongoing Semantic Web research activity concentrates on the ontology vocabulary layer. As with the Semantic Web layers themselves, it is much easier to find examples of basic ontology applications than it is to find examples of complex ontology applications. Historically, the digital library community [12] has been at the forefront of developments in the creation and adoption of basic ontological forms including taxonomies and thesauri. Many of these pre-date the digital era and are migrations to the Web of older paper-based schemes. At the more complex end of the ontological spectrum, much of the literature on this layer is historically based around the DAML+OIL language [13], which has more recently been superseded by the OWL language [14]. Like its predecessors, OWL goes further than RDF Schema by adding vocabulary that describes relations between classes, cardinality, equality, richer typing, characteristics of properties, and enumerated classes. The tables below list the vocabulary for classes and properties used in OWL. They are included here in order to give a feel for the nature of the language without delving into the specific syntax and semantics. For full details, see the W3C OWL Web Ontology Language Specification [14] and its supporting documents.

Classes in the OWL vocabulary	Properties in the OWL vocabulary
AllDifferent AnnotationProperty Class DataRange DatatypeProperty DeprecatedClass DeprecatedProperty FunctionalProperty InverseFunctionalProperty Nothing ObjectProperty Ontology OntologyProperty Restriction SymmetricProperty Thing TransitiveProperty	allValuesFrom backwardCompatibleWith cardinality complementOf differentFrom disjointWith distinctMembers equivalentClass equivalentProperty hasValue imports incompatibleWith intersectionOf inverseOf maxCardinality minCardinality oneOf onProperty priorVersion sameAs someValuesFrom unionOf versionInfo

OWL comes in three flavours: Lite, DL and Full. Each of these sublanguages is an extension of its simpler predecessor, both in what can be legally expressed and in what can be validly concluded. RDF documents will generally be in OWL Full, unless they are specifically constructed to be in OWL DL or Lite.

OWL Lite - Hierarchical classification (ideal for thesauri and other taxonomies); abides by all the restrictions of OWL DL; also, forbids the use of oneOf, unionOf, complementOf, hasValue, disjointWith, DataRange; and has other restrictions on expressiveness to aid the efficiency of complete reasoners.

OWL DL - *description logics* (computationally complete but inference services are restricted to classification and subsumption); requires pairwise separation of classes, properties and individuals (e.g. a class cannot at the same time be an individual); forbids the use of inverseOf, InverseFunctionalProperty,

SymmetricProperty, TransitiveProperty; most RDF(S) vocabulary cannot be used; and other restrictions to assure, based on current research, that a decidable reasoning procedure for OWL DL can exist.

OWL Full - full syntactic freedom of RDF(S) (no computational guarantees); currently no implementations of OWL Full reasoners although this is an ongoing area of research.

From a machine learning perspective, information expressed in RDF(S) and ontology languages like the OWL family can be viewed as background knowledge. Therefore, machine learning approaches that are able to exploit background knowledge expressed in a predominantly logical format would seem to be most appropriate for applications in the Semantic Web's ontology vocabulary layer. Multi-strategy learning approaches involving a mix of logical and non-logical methods may also be relevant.

2.6   Logic, Digital Signature, Proof and Trust

These highest levels of the Semantic Web have yet to be fully defined. Some work has been done in this area with CWM [31] and with Euler [32] but these applications do not yet exploit machine learning techniques.

The broad idea for these layers is that applications will be able to reason about whether a statement is true and to provide evidence that supports their decisions. One interesting change that the application of machine learning may bring to trust is reasoning with probability-based degrees of trust rather than the conventional deductive absolutes normally used today. The current Semantic Web has yet to fall prey to spam and false information in the way that email has but it is only going to be a matter of time before reasoning about trust becomes an essential prerequisite to any kind of proof using Semantic Web data.

2.7   Web Services and the Semantic Web

Recently, there has been a movement in the W3C community to bring together the Semantic Web and Web Services [15]. These are currently two different user communities with only modest overlap and interchange. Web Services are XML-based interfaces to programs accessible via the Web; they offer an operating system neutral Remote Procedure Call (RPC) protocol for the Web. Today's Web Services are predominantly business orientated and frequently offer simple, short-transaction services based around domain-specific XML vocabularies.

Apart from interoperability and extensibility, Web Services allow the combination of simple services to achieve complex operations. To facilitate discovery and use of such services, the Web Service Description Language (WSDL) [16] has been created in XML. Curiously, WSDL is not an RDF vocabulary although this may change [15][17]. A competing specification, DAML-S and OWL-S [18], based on RDF and OWL, fits more naturally with the Semantic Web layer model.

Fundamentally, the Semantic Web has a strong emphasis on data whereas Web Services have a strong emphasis on programs. Machine learning also spans both data and programs. Many future applications of machine learning to the Semantic Web are likely to require support for Web Services and may well present themselves to the outside world through Web Services. Machine learning may also have a role to play in service discovery, selection and pipelining of conventional e-Commerce Web Services. For example, in [19] an ontology-based repository is used to connect web miners and application agents.

2.7.1   The Semantic Grid and the Semantic Web

Although not normally listed under the heading of Web Services, Grid computing and the Semantic Grid, being equally as concerned with data as with programs, fall somewhere between the Semantic Web and Web Services. Having evolved somewhat independently to the Web and the Semantic Web, Grid computing uses its own protocols for service discovery and invocation. Arising out of computationally demanding research fields as experimental physics and bioinformatics, Grid computing is characterised by support for stateful computation, large volumes of data and high throughput. The current core application area of Grid computing is e-Science but the Grid community is well aware of the Semantic Web and work is under way to develop the Semantic Grid. The Semantic Grid aims to enable seamless integration with the Semantic Web and Web Services. Exactly how this will turn out is, as yet, unknown but it seems likely that the Semantic Grid will at least incorporate RDF and OWL.

Machine learning is already used to mine large science databases and there is every reason to expect that this practice will make its way into the eventual Semantic Grid as some form of Grid Service. In many ways, the application of machine learning to mining e-Science databases is not too far removed from current techniques for mining large databases and should fit in well with the Semantic Grid vision [36].

3   Applications of Machine Learning

Machine learning has a significant role to play at each of the levels of the Semantic Web. However, in reviewing the joint literature of the Semantic Web and machine learning it turns out that each of the Semantic Web levels is, at the current time, effectively divided in two. One part is concerned with the application of machine learning in *creating* the Semantic Web; the other is concerned with the application of machine learning in *using* the Semantic Web. On that basis, the following review is divided into two sections: Creating the Semantic Web and Using the Semantic Web.

4   Creating the Semantic Web

The Semantic Web is in its early stages and as of yet barely touches the estimated billions of Web pages/documents that already exist. The machine learning research community has been quick to identify a role for machine learning in the creation of RDF metadata and ontologies as a way of retrospectively bringing the existing Web into the Semantic Web. Indeed, the literature of the Information Extraction and Knowledge Engineering research communities constitutes the bulk of this section of the review. When compared to literature from the Semantic Web community it becomes clear that, to-date, the machine learners have focused their efforts almost exclusively on the annotation of Web pages, which, as it turns out, is only one part of the overall Semantic Web vision. The annotation of the existing Web is a top priority and needs to be addressed as a matter of urgency but this is just one part of the Semantic Web. When reading the following section it should be born in mind that the Semantic Web is about exposing **all** data in RDF format, not just data about Web pages. This includes data from relational databases, data from mobile devices, data locked in proprietary file formats, data about images, calendars, events, transactions, software, security, trust, copyright and so on. Intuitively, machine learning could have a role in creating and using many of these RDF data sources but these are not currently covered in the joint machine learning and Semantic Web literature and so are not covered in this review.

- Metadata Generation
- Examples of Machine Learning in Metadata Generation
- Ontology Creation
- Examples of Machine Learning in Ontology Creation
- Ontology Mapping

4.1   Metadata Generation

Machine learning can be used to semi-automate an otherwise manual Web content annotation process. However, for reasons discussed below this augmentation of an existing manual process is unlikely to adequately address the needs of the Semantic Web now or at any time in the future. To address the shortcomings of manual annotation, researchers have explored a variety of experimental approaches to automating the creation of metadata using a range of machine learning approaches to information extraction. A representative sample of these generally, small-scale exploratory investigations is discussed later in this section along with details of large-scale attempt to bootstrap the annotation of the entire Web.

4.1.1   Semi-Automated Annotation

Semi-automated annotation uses software tools to assist a human to select and apply appropriate metadata for Web content. The degree and method of automation varies from tool to tool. Unfortunately, this process remains essentially a manual process and suffers from the same limitations as entirely manual annotation. So, before discussing the undoubted advantages that machine learning driven automation can bring to an otherwise manual annotation process, it is necessary to look more closely at manual annotation itself. In so doing, an argument can be made that semi-automated annotation will play only a marginal role in the creation of the Semantic Web.

Manual annotation, the creation of RDF metadata describing Web content, can be performed by the author of the content embedding RDF statements into the content itself prior to its publication. Alternatively or additionally, externally stored RDF statements may be created to describe the document during its creation or at any subsequent time. This external metadata can be created by the original content author but may equally well be created by other people. The metadata can refer to characteristics of the document in its entirety (e.g. keywords, copyright, author) or to specific elements within the document (e.g. image, hyperlink, named entity, news item). In either case, a URI namespace reference to each of the ontology vocabularies (e.g. schema,

thesaurus, taxonomy) used in the RDF description must be specified within the document or as part of the RDF statements themselves so that software can correctly interpret the semantics of the metadata.

For professional cataloguers, such as digital librarians, this manual creation of metadata is a daily part of their job. They will have been trained in the use of software tools to describe Web resources and over time will have acquired a familiarity with the necessary cataloguing schemes, metadata syntax and semantics. Unfortunately, the same is not true of the vast majority of people who create Web content either as part of their job or in any other capacity. Consequently, manually created metadata is currently and, for other reasons given below, is likely to always remain a rare and expensive commodity. Unlike digital librarians, the majority of Web content authors are completely unaware of metadata and its importance in the Semantic Web. As the Semantic Web grows in size and utility this lack of awareness might become less of a problem but in the meantime it is a given. So, at this present point in time, lack of awareness is probably the single main reason why content authors are not creating metadata. However, even when awareness rises, further factors described below will come to the fore and place limits on the manual creation of metadata.

The first factor concerns people's unwillingness to spend time creating metadata. Although various metadata standards have existed for most of the Web's life, few people are willing to spend time adding metadata to their Web or non-Web content because doing so is time consuming (and often boring). Supporting evidence for this line of argument is the infrequency with which users of wordprocessor add even basic metadata like title, author, category, keywords or comments to their documents. This is in spite the fact that this can be simply done using the built-in document properties form available on any modern wordprocessor. Also, unlike creating metadata for the Semantic Web, there is a clear, local advantage to organisations in terms of knowledge management: classifying and locating documents would be much easier with this simple metadata. One speculation as to why this behaviour may be the norm is that a content author's motivation for adding data that may not be of direct immediate benefit to themselves is probably not that high.

Other limiting factors on manual annotation relate to the ability of humans to consistently interpret and apply the ontology vocabulary. It is self evident that RDF, particularly in its XML serialisation, is not easily read by humans and only software professionals or enthusiasts are ever likely to want to or be able to edit RDF directly. This problem can, if you ignore the aforementioned motivational issue, be overcome by the availability of good annotation tools for entering metadata. A more serious problem that afflicts both technical and non-technical creators of metadata is the need to identify, learn and interpret one or more ontology vocabularies in order to describe a Web resource. The aforementioned identification and learning problems can be sidestepped by building this information into the tools themselves. What cannot be sidestepped in manual or semi-automated annotation tools is the need for the person using the tool to make a judgement about which ontology terms to use in describing the Web resource.

Arguably, it is in addressing this last point about the human annotator having to make a judgement that machine learning can be most usefully applied to assist manual annotation; by suggesting the most appropriate ontology labels a machine learning system might be able to enhance the productivity, objectivity and consistency of the human(s) who will ultimately compose the metadata. Semi-automated annotation alone is wholly inadequate for the creation of the Semantic Web, but useful systems have been created and have an important role to play in the provision of the high quality metadata.

4.1.2   Automated Annotation

Machine learning is an obvious candidate for creating RDF metadata to retrospectively apply a layer of semantics onto the existing Web. Information extraction from unstructured data, such as flat-file databases and natural language text, is a well-established discipline in which machine learning is a useful component. Although, intuitively, this unstructured approach does not take advantage of semi-structured and structured information implicit in HTML and XML mark-up. Web Mining [37], on the other hand, does include techniques that take advantage of this structure through the analysis of document structure, directed links, site structure and server logs.

Even in the case of abundant semantic data, where RDF metadata about a resource already exists, it is possible that specific applications will require their own metadata views of the same data for reasons of trust and context. For instance, a content rating Web Service that returns a rating and chain of trust for a given Web page (e.g. suitability for viewing by children) may not want to depend on the metadata supplied by the Web content provider.

The automated generation of metadata from Web pages, also known as *screen scraping* or *metadata harvesting*, is often considered a poor alternative to human-created metadata. Given the available technology today, this is probably true. However, leaving aside the issues of cost, that view ignores the unavoidable fact that humans are inconsistent both over time and from person to person [20]. So, in terms of quality, perfection is not required in order to match the capabilities of human metadata creators. In terms of performance, humans could not hope to meet the expected demand for metadata for the existing, let alone the future, Web.

4.2   Examples of Machine Learning in Metadata Generation

The following is a sample of different approaches to the creation of metadata from online text by information extraction using machine learning. The coverage is neither complete nor a recommendation of the effectiveness of the methods; it merely presents a number of interesting directions currently under investigation.

4.2.1   General purpose Information Extraction

In [22] a Hidden Markov Model (HMM) approach is used for term identification and classification in previously unseen, untagged plain texts based on previously seen, manually marked-up XML example texts. The texts were drawn from online community documents produced in two different domains: the Message Understanding Conferences (MUCs) electronic news and the MEDLINE molecular biology abstracts. In each case, domain experts applied XML mark-up to a body of texts which were then used as training data. The training used raw text strings of words, broken into sentences but with no deep linguistic analysis. Also, the XML structure was essentially *flattened* by excluded nesting of terms from the mark-up scheme. Figure 3 is fragment of an example of a marked-up XML text from the MUC corpus. The mark-up shows the class named "ORGANIZATION" associated with the sequence of words, "Harvard", "Law", "School" and "PERSON" associated with "Washington". Such associations might possibly have been made using only a dictionary approach for term look-up. However, the HMM approach also generalised its knowledge using orthogonal features in the surface forms of the texts - including character features such as punctuation, capitalisation, and font (e.g. Greek). This was felt to have been particularly important in the case of the technical language of the molecular biology corpus.

```
A graduate of <ENAMEX TYPE="ORGANIZATION">Harvard Law School</ENAMEX>,
Ms. <ENAMEX TYPE="PERSON">Washington</ENAMEX> worked as a lawyer in the corporate
finance division...
```

Figure 3 - Annotated text from MUC corpus [22]

The purpose of the training the model was to learn the most likely sequence of name classes (*C*) for a given sequence of words (*W*). Using the Markov assumption, a HMM was implemented to estimate the maximized Pr(C|W) from bigrams of name classes. The algorithm created a frequency list of words and name classes which was then used to create metadata: applying XML mark-up to unseen plain text from the same corpora.

Based on the experimental results, the paper argues that automated learning approaches are the more promising way forward for automated information extraction because the alternatives, hand-built dictionary-based systems, cannot be expected to be easily ported to new domains. Also, that the latter ignore a potentially valuable source of the domain expert's knowledge: marked up texts.

The paper's conclusion points out that the HMM approach cannot easily model large feature sets due to fragmentation of the probability distribution. However, the paper suggests that other machine learning approaches, such as Support Vector Machines, may well overcome this limitation.

4.2.2   Incorporating domain-specific background knowledge

Inductive Logic Programming (ILP) is used in [21] to learn information extraction rules from manually marked-up XML natural language texts taken from the popular science publications *Nature* and *New Scientist*.

The domain is that of chemical compounds and other concepts related to global warming. The mark-up for the training data is based on a domain-specific ontology that was manually constructed by merging relevant parts of multiple pre-existing ontologies. Six predicates representing quantitative and qualitative relationships of interest were chosen. 30 texts were marked-up on a sentence by sentence basis with an XML representation of these six relationships as shown in figure 4. (Unusually for an XML representation, the elements do not enclose the text of interest but instead precedes the punctuation at the end of the sentence). The mark-up formed both the target relations required during the supervised learning and the valid set of statements that can be made about the texts during the test procedure.

```
The global experiment of increasing atmospheric CO2 concentrations by burning fossil
fuels has neither a control nor replicates
<target name="cAC(CO2,increase)"/>.
So it is difficult to quantify how much faster the world's forests might be growing
under high CO2 conditions
<target name="aCQ(CO2,high)"/>.
Higher levels of CO2 can clearly make plants grow better
<target name="aCQ(CO2,high)"/>.
```

Figure 4 - Annotated sentences from *Nature* [21]

The ILP FOIL learner was used in the experiment with the closed-world flag set so that only positive instances of the target relation needed to be provided. NLP (Natural Language Processing) techniques were used to enrich or filter the input to FOIL. The techniques optionally included part of speech (POS) tagging, word stemming, POS filtering/convergence, frequency analysis, named-entity recognition and immediate successor context. The sentence representation used a bag-of-words approach supplemented by simple context as shown below.

```
hasWord(Sentence-ID,POS,WORD)
hasWord(Sentence-ID,ne,NamedEntity)
context(Sentence-ID,Word-1,Word-2)
```

Further background knowledge was provided for FOIL by encoding selected aspects of the ontology to represent the concept class hierarchy and map between concept and word forms as follows.

```
isa(Class,Class)
txtform1(Class,Word)
txtform2(Class,Word,Word)
```

Ontology types (e.g. all classes below *Gas*), symbol types (e.g. *increase*, *decrease*, *none*) and numerical types were also provided as input to the learner.

F-scores were calculated for rules sets generated by multiple runs of FOIL; each time, including or excluding elements of the aforementioned background knowledge (e.g. with ontology; without ontology; with ontology and named-entity; with ontology and text mapping; etc.). The best combined performance across relations gave $0.54 \geq F \geq 0.68$ and $0.70 \geq precision \geq 1.00$. This compared favourably with a manually created rule set produced by a domain expert (reported as $0.57 \geq F \geq 0.67$ and precision not given). The paper concludes that the shallow structural analysis used in the experiment was adequate, even without the inclusion of grammatical information in the sentence representation. The author conjectures that a two-stage information extraction process, where grammatical information is acquired on-demand for sentences whose interpretation requires it may produce superior results.

4.2.3   Exploiting existing digital library classifications

In [23] a novel strategy is used to work around a glaring problem with scaling-up automated metadata generation to span the entire Web. Machine learning based information extraction generally assumes that the training cases are pre-labelled by a human indexer. This may well be acceptable for extraction from resources within a specific domain or limited vocabulary and with a more-or-less conventional structure. So when scaling this up to the broad and diverse categories found on the Web, the number of training cases needed explodes, the acquisition of text fragments becomes difficult, and their manual labelling becomes infeasible. However, this paper exploits a promising resource of web data that has already undergone a process of human indexing: web directories such as *Open Directory* or *Yahoo!*

The assumption is that the directory category headings (such as `.../Manufacturing/Materials/Metals/Steel/...`) coincide with *informative terms* (e.g. the names of products or services offered by the owners of a page in that directory category). By matching the directory category headings with the page fulltext, the software described in the paper obtains sentences that contain these *informative terms*. Other terms situated near the *informative terms* in the structure of the sentence are candidates for *indicator terms*, provided they occur frequently on pages from various domains. The resulting collection of *indicator terms* is then used in the role of extraction patterns for discovering *informative terms* in previously unseen pages: the metadata production stage of the process.

The paper describes further work in progress that aims to enhance this approach by incorporating manually produced ontological knowledge about directories, particularly relating to context that may be assumed within Web pages in a given directory category.

4.2.4   Large-scale classification against a simple ontology

In a large-scale experiment [33] aimed at bootstrapping the Semantic Web, SemTag is used in conjunction with the TAP ontology on a collection of 264 million Web pages to automatically generate 434 million disambiguated semantic tags. This body of metadata is published on the Web as a label bureau. SemTag classifies documents to label them with TAP ontology terms based on a simple lexical match. Machine learning comes into play in order to deal with ambiguous words/terms that occur at more than one place in the TAP taxonomy. How SemTag's disambiguation algorithm works is described below.

SemTag begins with a *spotting pass* over the document collection in which it finds all instances of the 72K labels that appear in the TAP ontology. For each match it also extracts the 10 words before and 10 words after the label, using this as contextual information which the developers call *spots*. A representative sample of the

data is then scanned to determine the corpus-wide distribution of terms at each (internal) node in the taxonomy, expressing this as both a 'probability' (term frequency) and TF-IDF (term frequency - inverted document frequency). The TDB algorithm then uses this information plus a manually generated metadata training set to learn a similarity function between term contexts. Nearest neighbour and Bayes are compared as the learners, with only minor differences in performance (labelling accuracy around 80%). Interestingly, in the reported experiments TBD turns out to be fairly insensitive to the presence of the manually generated metadata and accuracy is maintained with 'extremely minimal metadata'.

SemTag is an important example application of machine learning to metadata generation and demonstrates one practical approach to scaling up automatic metadata generation for the current Web. However, as the authors themselves point out, few people have the necessary hardware required to snap-shot the Web and so it is difficult to repeat this approach for every ontology a Semantic Web user may wish to apply to the Web corpus. The SemTag approach is viable as a broad coverage, shallow complexity metadata creator; a different approach, such as domain-specific, focused crawling, might be required for the creation of more complex metadata.

4.3   Ontology Creation

Many people believe that the ontology vocabulary layer will form the core of future Semantic Web applications. This seems likely in terms of simple ontologies and small ontologies. However, large and complex ontologies have serious usability problems for humans in terms of human inability to use them consistently over time or from person to person. As with simple metadata, it is hoped that machine learning will be able to supply or at least augment consistency of use.

Most ontologies are hand-crafted and are the result of collaboration between domain experts and knowledge engineers. Creating an ontology is far more complex than extracting individual metadata elements as RDF. Despite this, some progress has been made.

- Ontology Maintenance
- Semi-Automated Ontology Creation
- Automated Ontology Creation

4.4   Examples of Machine Learning in Ontology Creation

4.4.1   Ontology Maintenance

In [24] the Aleph ILP learner is used, not to create an ontology but, to revise and maintain an existing ontology. The paper recommends an ILP method for assertional (A-Box) mining human marked up examples to generate new concepts and that these generated concepts be used to revise a DAML+OIL ontology by inserting them into an appropriate position in the ontology's terminological (T-Box) concept hierarchy.

As well as growing an ontology by adding new concepts, machine learning can be used to prune redundant concepts that are either underused or overly specific for the intended application. Pruning is used in [35] to adjust an existing ontology for a specific application domain. Here concepts that are less frequent within the domain-specific corpus than in the generic corpus are pruned from the ontology.

4.4.2   Semi-Automated Ontology Creation

[25] describes work to semi-automate the production of ontologies using a variety of learning techniques, including statistical text processing for concept extraction, hierarchical concept clustering, and association rule learning. Unfortunately, the process is complicated and still requires a highly skilled human knowledge engineer in order to fully exploit the toolset presented in the paper. The work is focused on ontologies that can be (almost) completely contained in RDF(S) and new methods for improved ontology engineering will be required to support higher layers of the Semantic Web tower. Furthermore, in concluding, the authors highlight an important challenge for the role of ontologies on the Semantic Web: ontology boundaries become less well defined because the XML namespace mechanism allows ontologies to point to and include each other in an "amoeba-like" structure.

In [38] small, highly domain-specific ontologies are individually extracted from HTML examples under user direction. These small ontologies are then pairwise merged into a combined ontology under the guidance of a user and supported by a quality metric based on weighted precision and recall. The example given is merging ontologies for different car rental Web sites and concentrates on describing Web forms for rentals. The automatic extraction of candidate ontologies by learning from a user's Web browsing activity demonstrates a promising direction for machine learning support for the creation of some of the ontologies required for the

Semantic Web. However, the authors note the desirability of a fully automated ontology creation process and view their current work as a step towards that goal.

4.4.3   Automated Ontology Creation

A survey of ontology learning for the Semantic Web [39] concluded that, "Until recently ontology learning per se, i.e. for comprehensive construction of ontologies, has not existed. However, much work in a number of disciplines - computational linguistics, information retrieval, machine learning, databases, software engineering - has actually researched and practiced techniques for solving part of the overall problem". The survey authors present an array of techniques drawn from the above disciplines, a superset of those used in [25] for semi-automation, and position them within their own architecture for ontology learning centred around their *OntoEdit* workbench software. An architecture of this nature will undoubtedly form a key tool in the near future for information professionals such as ontology engineers or digital librarians. Although other frameworks for automatically creating ontologies will clearly also be required to fulfil the Semantic Web vision and these are more likely to be based around software agents and Web Services than the described framework. That said, the framework provides a useful vehicle for discussing how automated ontology creation might work and it does not require a huge leap of the imagination to map the described framework onto a distributed implementation or an agent architecture.

Fully automated construction of complex ontologies is still an active area of research. However, some considerable headway has been made at the lower end of ontological complexity in [26] with XTRACT, a novel system for inferring a Document Type Descriptor (DTD) schema for a database of XML documents. DTDs are not mandatory for XML documents and it is frequently the case that no DTD exists for a given document collection. Hence being able to create a DTD automatically provides potentially useful structural relationship information that may be used by machine learners. The inference algorithms in XTRACT use a three-step process: (1) finding patterns in the input sequences and replacing them with regular expressions to generate "general" candidate DTDs, (2) factoring candidate DTDs using adaptations of algorithms from the logic optimization literature, and (3) applying the Minimum Description Length (MDL) principle to find the best DTD among the candidates. The system performed well, identifying DTDs which were fairly complex and contained factors, metacharacters and nested regular expression terms. Notably, the resultant DTDs compared well with human created ones for the same data, the learnt ones often being equally as readable by humans as the manually created ones.

More recently, and at the more complicated end of the ontological spectrum, the OntoLearn system [40] has been used to create domain-specific linguistic ontologies using the WordNet ontology as a starting point. Most other approaches to automated ontology construction extend existing ontologies with unknown words and concepts. OntoLearn on the other hand cuts down WordNet to more tightly fit a specific domain and extends WordNet by compositing exist terms in the ontology to create complex concepts (e.g. "bus" + "service" = "bus service"). Decision tree rule learners (C4.5 and TiMBL) were applied to learn relations between pairs of concepts and so link the complex concepts into the ontology. An example rule learnt by C4.5 is shown below (words with a # in are concept identifiers).

```
If in modifier [conveyance#3,transport#1] = 1 and in head [act#1,humanact#1] = 1 then
relation MANNER (92.2%)
e.g. bus service, coach tour
```

Around 20 classification rules were produced from a manually tagged training set, with the best experiment reported as having a 6% error rate over 405 examples. However, the evaluation of the resultant ontologies is an open problem and the developers of OntoLearn advocate evaluation *within* some existing application.

At present OntoLearn is unable to analyse totally unknown terms although this is an ongoing area of research. The OntoLearn approach is complemented by a more typical, but equally interesting, approach [41] of extending WordNet with unknown concepts.

4.5   Ontology Mapping

Ontologies are a balance between generality and specificity. The ideal ontology for a given domain is highly specific to that domain. By contrast, general ontologies may be widely applicable but are unable to capture all the details of a specific domain. Therefore, there is a natural tendency for the proliferation of domain-specific and application-specific ontologies. This introduces a new problem: if the Semantic Web is to work across domains and applications, i.e. across ontologies, then there must be some way of mapping between and merging ontologies. Once again, this appears to be a fertile application area for machine learning and [27] presents a survey of tools for the purpose.

Ontology mapping has a long tradition in the guise of relational database schema alignment. Schema alignment is not an entirely solved problem and is somewhat less complex than the general ontology mapping problem by virtue of the restricted structure of relational databases. See [42] for a survey of automated schema matching methods. This database alignment literature is likely to make useful contributions to automated ontology mapping for the Semantic Web. In fact it also seems likely that techniques from a wide range of research communities will be required, ranging from classical computer science (e.g. graph matching and graph isomorphism), digital libraries (e.g. thesauri mapping), information retrieval (e.g. cross-searching) right through to domain expertise from the intended application area. Machine learning is far from the complete answer to ontology mapping but it is shaping up as an important component.

A noteworthy example of work in this area is the GLUE system and its more recent extension CGLUE [45]. Given two ontologies, for each concept in one ontology GLUE finds the most similar concept in the other ontology. CGLUE does likewise but also identifies complex mappings where a concept in one ontology maps to multiple concepts in the other. GLUE estimates the joint probability distribution of the concepts from each ontology in order to calculate the pairwise similarity of concepts. The estimate is arrived at by training classifier A on ontology A and then applying that classifier to ontology B; then by repeating the exercise in reverse, training classifier B on ontology B and using that to classify ontology A. The *Jaccard* coefficient (the ratio of overlap between two concepts with respect to the total number of concepts) is then computed as a function of the joint probabilities. "Most general child" and "Most specific parent" similarity measures are also used, based on background knowledge of the domain expressed as a taxonomy. Finally, relaxation labelling is used to incorporate features of neighbouring taxonomy nodes into the labelling of each node in the taxonomies, exploiting the intuition that, for instance, the parent of node is likely to match if its children all match. Two learners are used as classifiers: the Name Learner and the Content Learner. The former uses the unique identifier names of the taxonomy node as the instance whereas the latter uses the textual content of the instance (as an unstructured bag of tokens). Naive Bayes is used in both learners. The combined predictions of the learners are combined via a simple weighted sum. GLUE matched nodes with an accuracy of 66-97% on several real-world domains. CGLUE is shown to find complex matchings between ontologies. However, the mapping process is currently only semi-automated, in that it suggests mappings for consideration by a human.

5   Using the Semantic Web

Predictably, given the ongoing research effort to establish and populate the Semantic Web, there are currently fewer applications of machine learning that use the Semantic Web than there are for contributing to its creation. That said, most of the applications mentioned in the previous section make some use of the data they extract and so might also have qualified to appear here. Evidence for the current rarity of machine learning usage on the Semantic Web comes from a wide-ranging survey [43] briefly describes 85 Semantic Web applications under the twelve categories below.

- Catalogue/thesaurus management
- Data dependent agents
- Data integration
- Knowledge formation
- Knowledge management
- Metadata for annotation and enrichment
- Metadata for discovery and selection
- Metadata for media and content
- Ontology management
- Personal information management
- Semantic indexing
- Syndication

This body of applications demonstrates that the Semantic Web is not short of applications per se. However, searching through the application descriptions, none mention machine learning explicitly although five expressly mention text mining, information extraction, metadata extraction or semi-automated ontology creation. Unfortunately, these applications are primarily examples of machine learning helping to create the Semantic Web rather than to use it.

Given that machine learning is still locked into the creation stages of the Semantic Web, this section describes a number of potentially relevant, publicly available, RDF datasets and gives pointers to the emerging field of Semantic Web Mining.

5.1 Datasets

The Semantic Web is in its infancy and so most of the Web is not yet semantically described. Therefore, obtaining RDF datasets suitable for machine learning experiments can be problematic [28] and it is not uncommon for data preparation to begin by generating an RDF dataset from other data formats. Such a conversion is performed automatically by DAML on a number of non-RDF datasets, including WordNet, NYSE+NASDAQ stock symbols and CIA World Fact Book. The resultant RDF is made publicly available at http://www.daml.org/data/

Another approach to obtaining RDF is to harvest it from the Web using Web crawling software. This is frequently done for one of the most widely used RDF vocabularies: the RSS content syndication format used in thousands of news feeds and Weblogs [9]. Once data is in RSS format it can be viewed as RDF either directly, in the case of RSS 1.0 or indirectly, by transformation, for other versions of RSS.

As yet, there do not appear to be any synthetic dataset generators that are specifically designed to produce Semantic Web metadata. There are, however, numerous software applications available to produce RDF from various data formats. Details of these can be found in [43].

5.2 Semantic Web Mining

Semantic Web Mining, as described in [29], aims to combine the Semantic Web with Web Mining. The idea is to improve the results of Web Mining by exploiting the new semantic structures in the Web and to help build the Semantic Web through the use of Web Mining. In the context of this section we will only cover what the paper has to say about mining the Semantic Web. The paper advocates the use of Relational Data Mining (RDM) techniques [30] to mine both Semantic Web content and structure. RDM looks for patterns that involve multiple relations in a relational database, doing so directly, without first transforming the data into a single table. ILP is the most developed area in relational data mining which comprises techniques for classification, regression, clustering and association analysis. In [29], the authors suggest that these same RDM techniques can be easily adapted to deal with data described in RDF and ontologies. However, the size and distributed nature of Semantic Web data presents a substantial scientific challenge to their successful application.

For the mining of Web server log files, the paper notes that by registering references to concepts from an ontology, usage mining with semantics becomes possible. A system for creating such semantic log files from a knowledge portal is cited in the paper.

6   Concluding Remarks

The Semantic Web is a rapidly evolving enhancement to the existing Web, bringing richly structured metadata to the previously unstructured and semi-structured Web. Publicly available RDF datasets are available through repositories like daml.org and globally successful applications like RSS 1.0 newsfeeds and Weblogs. This global abundance of metadata is shaping up as a fruitful research and application area for machine learning and, arguably, may one day become the dominant application area for machine learning.

The key languages of the Semantic Web are RDF and the vocabularies built on top of RDF. RDF triples map well onto Prolog databases, as well as relational databases, to enable the application of machine learning techniques for learning from structured data.

The emerging discipline of Semantic Web Mining is a natural progression from the established discipline of Web Mining, but with a much greater focus on structured data. Additionally, the "packaging up" of machine learning in the form of Web Services that interoperate amongst each other and the rest of the Semantic Web may well require new techniques.

References

1. Berners-Lee, T., Hendler, J., Lassila, O. (2001). The Semantic Web, *Scientific American*, May
2. IETF, RFC 2396, URI Specification

3. Lassila, O., and Swick, R. (1999). Resource description framework (RDF). W3C proposed Recommendation, http://www.w3.org/TR/1999/REC-rdf-syntax-19990222

4. Beckett, D. (2003). RDF/XML Syntax Specification (Revised) http://www.w3.org/TR/2003/WD-rdf-syntax-grammar-20030123

5. Manola, F. & Miller, E. (2003). RDF Primer, http://www.w3.org/TR/2003/WD-rdf-primer-20030123/

6. Wielemaker, J., Schreiber, G., Wielinga, B. (2003). Prolog-based Infrastructure for RDF: Scalability and Performance, http://www.swi.psy.uva.nl/projects/SWI-Prolog/articles/iswc-03.pdf

7. Flach, P.A. & Lavrac, N. (2003). Rule Induction, *Intelligent Data Analysis* (2E), Berthold, M. & Hand, D., editors, Springer-Verlag.

8. Koivunen M. & Miller, E. (2001). W3C Semantic Web Activity, http://www.w3.org/2001/12/semweb-fin/w3csw

9. Brickley, D. et al (2000). RDF Site Summary (RSS) 1.0, http://web.resource.org/rss/1.0/spec

10. Beckett, D., Miller, E., & Brickley, D. (2002). Expressing Simple Dublin Core in RDF/XML, http://dublincore.org/documents/2002/07/31/dcmes-xml/

11. Brickley, D. (2003). RDF Vocabulary Description Language 1.0: RDF Schema, 23 January 2003 http://www.w3.org/TR/2003/WD-rdf-schema-20030123/

12. IFLANET (2003). Digital Libraries Metadata Resources, International Federation of Library Associations and Institutions, http://www.ifla.org/II/metadata.htm

13. Conolly, D., et al (2001). DAML+OIL Reference Description, http://www.w3.org/TR/daml+oil-reference

14. McGuinness, D. L. & van Harmelen, F. (2003). OWL Web Ontology Language Overview http://www.w3.org/TR/2003/WD-owl-features-20030331/

15. Berners-Lee, T. (2003). Web Services - Semantic Web, WWW-2003, Budapest, keynote, http://www.w3.org/2003/Talks/0521-www-keynote-tbl/

16. Chinnici, R., et al (2003). Web Services Description Language (WSDL) Version 1.2, http://www.w3.org/TR/2003/WD-wsdl12-20030611/

17. Ogbuji, U. (2000). Supercharging WSDL with RDF, IBM developerWorks, http://www-4.ibm.com/software/developer/library/ws-rdf/

18. Martin, D., et al (2003). DAML-S (and OWL-S) 0.9 Beta, http://www.daml.org/services/daml-s/0.9/

19. Haustein, S. (2001) Utilising an Ontology Based Repository to Connect Web Miners and Application Agents, Semantic Web Mining Workshop, ECML-2001, Freiburg, Germany.

20. Markey, K., (1984). Interindexer Consistency Tests: A Literature Review and Report of a Test of Consistency in Indexing Visual Materials, *Library and Information Science Research*, **6**, 155-77.

21. Aitken, J. S. (2002). Learning Information Extraction Rules: An Inductive Logic Programming Approach, van Harmelen, F., Ed. Amsterdam, IOS Press.

22. Collier, N. (2001). Machine Learning for Information Extraction from XML marked-up text on the Semantic Web, Semantic Web Workshop, WWW-10, Hong Kong 1-5 May, 29-36.

23. Kavalec, M., Svatek, V., & Strossa, P. (2001). Web Directories as Training Data for Automated Metadata Extraction, Semantic Web Mining Workshop, ECML-2001, Freiburg, Germany.

24. Nakabasami, C. (2002). An Inductive Approach to Assertional Mining for Web Ontology Revision, *International Semantic Web Conference (ISWC-2002)*, Sardinia, Italy.

25. Maedche, A., & Staab, S. (2001). Learning ontologies for the semantic web. *Semantic Web Workshop 2001*, WWW-10, Hong Kong 1-5 May

26. Garofalakis, M. et al. (2003). XTRACT: Learning Document Type Descriptors from XML Document Collections, *Data Mining and Knowledge Discovery*, **7**, 23–56.

27. Noy, N.F., Musen, M.A. (2002) Evaluating Ontology-Mapping Tools: Requirements and Experience. *Workshop on Evaluation of Ontology Tools at EKAW'02 (EON2002)*.

28. Edwards, P., Grimnes, G.A., Preece, A. (2002). An Empirical Investigation of Learning from the Semantic Web, Information Research, Semantic Web Mining Workshop, ECML-2002, Sydney, Australia.

29. Berendt, B., Hotho, A., Stumme, G. (2002). Towards Semantic Web Mining, *International Semantic Web Conference (ISWC-2002)*, Sardinia, Italy.

30. Dzeroski, S. & Lavrac, N. editors (2001). Relational Data Mining, Springer.

31. Palmer, B. (2001). CWM - Closed World Machine, http://infomesh.net/2001/cwm/

32. De Roo, R. (2003). Euler proof mechanism, http://www.agfa.com/w3c/euler/

33. Dill, S. et al (2003) SemTag and Seeker: Bootstrapping the semantic web via automated semantic annotation, WWW2003, Budapest, Hungary.

34. Noy, N. F. et al (2001) Creating Semantic Web Contents with Protege-2000, *IEEE Intelligent Systems*, 16(2), 60-71.

35. Kietz, J., Volz, R., Maedche, A. (2000) A method for semi-automatic ontology acquisition from a corporate intranet text, EKAW
36. De Roure, D. (2003) The Semantic Grid Vision, http://www.semanticgrid.org/vision.html
37. Kosala, R. and Hendrik Blockeel, H. (2000) Web Mining Research: A Survey, ACM SIGKDD Explorations, **2**(1), 1-15
38. Modica, A. et al (2001) The use of machine generated ontologies in dynamic information seeking, *Proceedings of Cooperative Information Systems, CoopIS 2001*, 433–448.
39. Maedche, A. & Staab, S. (2001). Ontology learning for the semantic web. *IEEE Intelligent Systems*, 16(2)
40. Navigli, R. & Velardi, P. (2004) Learning Domain Ontologies from Document Warehouses and Dedicated Web Sites, to appear in *Computational Linguistics*, MIT Press
41. Alfonseca, E & Manandhar, S. (2002) Improving an Ontology Refinement Method with Hyponymy Patterns, Language Resources and Evaluation (LREC-2002), Las Palmas
42. Rahm, E. & Bernstein, A. (2001) A Survey of Approaches to Automatic Schema Matching, VLDB Journal **10**(4), 334-350
43. Reynolds, D. et al (2002) Semantic web applications - analysis and selection, http://www.w3.org/2001/sw/Europe/reports/chosen_demos_rationale_report/hp-applications-survey.html
44. Beckett, D. (2004) Dave Beckett's RDF Resource Guide, http://www.ilrt.org/discovery/rdf/resources
45. Doan, A. et al. (2003) Learning to Match Ontologies on the Semantic Web, VLDB Journal **12**(4), 303-319