Hierarchical Concept Description and Learning for Information Extraction

Luo Xiao, Dieter Wissmann
CT SE 5 Siemens AG
Paul-Gossen-str. 100 91052 Erlangen
(Luo.Xiao, Dieter:Wissmann)@erls.siemens.de

Michael Brown
SemanticEdge GmbH. Berlin, Germany
Mike@SemanticEdge.com

Stefan Jablonski
Dept. of Computer Sciences VI (IMMD VI) University of Erlangen-Nuremberg, Germany
Stefan.Jablonski@informatik.uni-erlangen.de

Abstract

This paper addresses the problem of extracting information from textual documents, either normal documents or web pages. A new approach for extracting complicate information from semi-structured documents is introduced that exploits a successive hierarchical rule-learning algorithm. Through evaluation it is shown that this approach can extract complicate concepts with a much higher precision than the equivalent rule learning applied to flat text. In addition, the rate of learning is significantly higher for the hierarchical approach.

1. Introduction

This paper addresses the problem of extracting information from textual documents. This problem generally involves the transformation of information in an informal (i.e. non-machine-readable) format into a formal format. This conversion is a necessary pre-condition for the automated management of knowledge within a modern organization. The information contain in a disparate body of textual documents can be gathered together into a central knowledge resource. Once the information owned by a corporation is made explicit, a wide range of automated facilities are enabled, such as; document generation and summarisation [DIXO97,RAJM98], automatic filtering and analysis of WWW pages [MUSL98, SODE97], etc.

The main problem for Information Extraction (IE) is that in informal text there is a high proliferation of the number of patterns that are used to express a particular concept. Hence, an automated learning technique requires a lot of training to cover all variations. The problem is compounded for complex concepts that involve a combination of more basic concepts: the number of potential surface level variations is the product of the number of variations of the more basic concepts. This effectively limits the complexity of concepts that can be learnt directly from surface level text.

In this paper we present, a new approach for learning and extracting complicate information from semi-structured documents is introduced. Through evaluation it is shown that this new hierarchical concept learning algorithm can extract complicate concepts stepwise with improved overall precision.

2. System Architecture

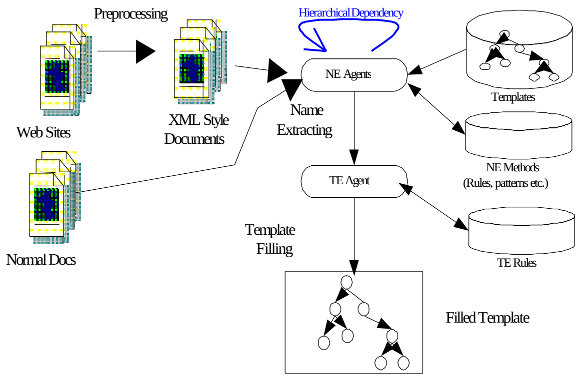

Fig. 1. System Architecture

We partition the system into three components. The system architecture is shown above. The NE-Agent (to do Name Extraction, MUC-6 [CUNN99]) extracts the contents from the pre-processed documents using structured information and classical IE methods. Afterwards the extracted contents can be filled into the predefined template by using TE-Agent (to do Template Element, MUC-6 [CUNN99]).

The focus of this paper is on the NE-Agents and the level of precision/recall that can be achieved with these elements alone. In particular, the central interest here is how much benefit can be attain by building dependencies between the individual NE-Agents. That NE-Agent X is dependent on NE-Agent Y means that the rules that are learnt by X exploit the results of the information extraction carried out by X. In this way we build a hierarchical rule learning system for the IE task.

2.1 Pre-processing

The input to the NE-Agents can either be normal (e.g. ASCII, Word, etc.) documents or Web pages. For Web pages, Although the tags of a HTML document imply its structure, we can not use this information reliably, because we can not follow the structure exactly. By contrast, XML documents have a complete hierarchical structure specified by start tags and end tags. To analyze the structure we need to convert HTML sites to well formed XML-style documents, which is the task of Preprocessing. We use the standard open source TIDY provided by W3C organization (http://www.w3.org/People/Raggett/tidy) to automate the conversion.

2.2 NE-Agent

As well as structured rules, NE-Agents extract the contents using various IE methodologies. Figure 2 presents the construction of the standard NE-Agent.

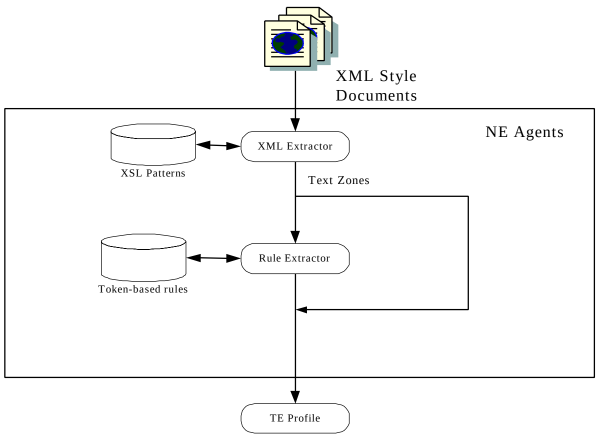

Fig. 2. Construction of NE-Agent

If the input documents are in XML format, structural information can be used to localise relevant text zone [XIAO01a]. Structure is described by XSL-patterns. With the help of patterns, specific text zones for specific concepts to be extracted can be identified. This may be sufficient for the IE task (i.e. to be extracted information corresponds to a whole text zone). Alternatively, the identified text zone is forwarded to traditional IE components can extract the exact information within the text zones.

The system uses Rule Extractor to perform traditional IE task. The extractor uses token-based rules for extracting information[1], where tokens are either alphabetic characters, numeric digits or other syntactic characters. The token-based rules consist of three parts: precore, core and postcore. While the core describes the underlying content, the pre- and postcore define information before and after the content.

For example, the rule; *sf [1] {c <int> c} bed;* extracts a number (between "{c" and "c}" is the core to be extracted), which has the string "sf" within two tokens of the start of the core and the string "bed" immediately after the core.

An inductive supervised training algorithm is used to create such rules automatically based on

[1] The initial version of token-based Rule Extractor is provided by DFKI: The German Research Center for Artificial Intelligence GmbH

a number of training examples. The basic Rule Extractor can generate some token-based patterns based on the surface level (tokenised) text but is unable to exploit the results of other NE-Agents directly.
In general there can be multiple Extractors acting as NE-Agents e.g. representing different machine learning techniques. Each provides solution for a definite problem. The Extractors typically work separately, but could also cooperate with each other. In this sense, the NE-Agents part of the system is a multi-Agents system itself. Hence, the extension to the basic architecture proposed here is to allow the representation of hierarchical dependencies between the NE-Agents. Note that other types of collaboration between NE-Agents are left as an issue for future research.

2.3 TE-Agent

In Figure 1, a TE-Agent is shown that receives the results generated by the individual NE-Agents. The TE-Agent has the general role of accounting for global constraints, such as relative positions of pieces of information within a document, in order to increase the accuracy of IE. The TE-Agent is however not discussed further in this paper and the reader is referred to [XIAO01b] for more details.

3. A New Approach: Hierarchical Concept Learning And Extracting

Traditional Information Extraction systems try to use rules or NLP algorithm to extract contents. Rules (such as regular expressions) can find pieces of fine information. However, they are not able to extract some complicate contents. Hence, while an individual date may be easy to extract, the sequence of events spread across a long narrative of news article may be very hard to extract.
Typically, complex content can be decomposed into strong patterns involving more basic concepts. This is the basic assumption underlying he idea of the hierarchical learning methodology. Firstly, rules are learnt for simple/basic concepts (such as place name, date, distance value, etc.). Such basic concepts are then replaced in the initial document with uniform token strings. This normalised text can then be used to derive more complex rules on the basis of trained simpler rules and thus to extract more complex concepts. In principle, this process could be performed successively until highly abstract concepts are extracted or until the whole document content has been captured. In practice, the process of deriving complex rules from more basic rules may cause a gradual amplification of noise that limits the degree of nesting that can be achieved in practice. However, in general, hierarchical learning allows more complex concepts to be extracted than would be possible using flat rules that are derived solely from the original source text. The algorithm for hierarchical learning is described in the following section.

3.1 LEARNING

Hand-coding patterns and rules is time consuming and error-prone. The system illustrated in Figure 1 provides various learning algorithms to semi-automated the construction of an IE System. Learning techniques include; generation of XSL-patterns (wrapper induction), induction token-based rules, statistical keywords extraction, and statistical derivation of special constraints. In the training phase, the input is some template examples, typically filled by a non-skilled manual worker. We use a GUI tool to create the examples interactively. With Drag & Drop operations user can fill the examples into predefined empty templates. The examples are defined by begin and end positions in original document. Training NE-Agents proceeds in a stepwise manner for each separate element/slot defined in a template. For each element the system gathers all corresponding examples and generates patterns and rules for this element. After the training the results are saved into an empty template, named profile. In the extracting phase the system can load the trained profile, extract values and fill them in template.
Hierarchical learning extends the basic algorithm by using the NE-Agent dependencies to govern the order in which NE-Agents is trained. Once each lower level NE-Agent has been trained it is applied to the source text and each occurrence found is replaced by a normalised representation of the following form: $ConceptName$. Note, this symbolic normalisation is superimposed on top of the original document text, thus the original surface text is always accessible.

The input for learning in set of examples defined by start positions and lengths. The learning step of each feature is given by user. For each iteration the learning algorithm contains three steps: At first the system extracts all information specified by the learned rules from the sample document. The results are defined by start position and length. At the second step the results are replaced by some unique tokens. All position and length information are updated to adjust on the current processed document. At the last step the system uses token-based Rule Extractor to learn new rules based on processed document. The algorithm is shown in figure 3.

```
For each step
        i) Get example set at this step
ii) Extract all information defined by learned rules
iii) Replace the results with unique token string of form
$ConceptName$
iv) Update the position and length in example set
v) Train the example based on processed document
vi) IF trained rule contain unique token string
      Replace token string as trained rule name
End if
        vii) Give the rule a name and set it in the trained rules
set
END For
```

Fig. 3. Hierarchical Concept Learning

1.1.1 WORKED EXAMPLE

In following we give an example in figure 3 to illustrate how rules are generated hierarchically based on a semi formal text of English Football Results archived from the 1996/1997 season. To be extracted are teams and match results. An example of the original document is:

NOTTINGHAM FOREST 0-1 EVERTON 19,892 Match Report
Full-time score and scorers for Sunday, 10. 27 1996
LIVERPOOL 2-1 DERBY COUNTY 39,515 Match Report
Full-time scores and scorers for Saturday, 10 26 1996
ARSENAL 0-3 LEEDS UNITED 38,076 Match Report
CHELSEA 3-1 TOTTENHAM HOTSPUR 28,373 Match Report…

The goal is to extract from each match, the teams playing and the result (more fine grained analysis such as differentiating home and away teams and number of goals each scored could obviously be achieved in a similar manner, but things are kept simple here for illustration purposes). Firstly, at level 1 the team names are learnt (this is most easily learnt using the keyword extraction agent):

Step 1: Train all teams, Trained Rules Set: empty
Step 2: Train the result, Trained Rules Set: TEAM
Replace all instances of TEAM in the document with token string "$TEAM$".

Thus, after the concept of TEAM has been learnt and replaced in the original document, the document content appears as follows:

$TEAM$ 0-1$TEAM$. 19,892 Match Report
Full-time score and scorers for Sunday, 10. 27 1996
$TEAM$ 2-1 $TEAM$ 39,515 Match Report
$TEAM$ 0-3 $TEAM$ 38,076 Match Report
$TEAM$ 3-1 $TEAM$ 28,373 Match Report…

Thus in this case, the concept of a RESULT is dependent on the concept of a TEAM. Thus, *training RESULT based on the above normalised document yields the rule, "$TEAM {c <int> - <int> c} $TEAM". In a similar way to the concept TEAM, each instance of this rule is replaced in the document with the symbol $RESULT$. This yields the following normalised document:*

$TEAM$ $RESULT$ $TEAM$. 19,892 Match Report
Full-time score and scorers for Sunday, 10. 27 1996
$TEAM$ $RESULT$ $TEAM$ 39,515 Match Report
$TEAM$ $RESULT$ $TEAM$ 38,076 Match Report
$TEAM$ $RESULT$ $TEAM$ 28,373 Match Report…

Finally, the concept of a MATCH can be learnt based on this normalised document. Training yields the rule for MATCH as "{c $TEAMS$ $RESULT$ $TEAMS$ c}". Instances of this rule can be replaced with the symbol $MATCH$ thus allowing yet more complex concepts to be learnt while avoiding the proliferation of surface level variations. Thus the above worked example illustrates the fundamental principles of the hierarchical learning. While this result is manually configured, the results of applying the automated hierarchical learning to this data set generated a very similar set of rules to those illustrated above (see section 4.1)

3.2 EXTRACTION

In the extraction phase a trained profile is loaded. The extraction function must be call recursively if the rule for a complex function contains rules for other more basic concpets. The algorithm for extraction is described as follows:

Extracting_Rule (ConceptName)
Get each MatchPattern from search profile of ConceptName
While MatchPattern contains references to other more basic concepts:
Get the ContainedConceptName
Extracting_Rule (ContainedConceptName)
Replace all result with $ContainedConceptName$ in document
Make same replacement in the MatchPattern
End While
Do Extraction with processed MatchPattern in the processed document
Align positions and length of results to get the right location in original document
End Extracting_Rule

Fig. 4. Extraction Algorithm

4. EXPERIMENTS

As a preliminary trial of the hierarchical learning system, it was applied to three document collections of varying informality; the football data (see above), a collection of rental ads [SODE99] and travel descriptions from Lonelyplanet WorldGuide. The Rental Ads domain was collected from the Seattle Times online-classified ads. We downloaded the test set from RISE Repository (http://www.isi.edu/~muslea/RISE/repository.html). The Lonelyplanet WorldGuide is collected from the Web sites in Lonelyplanet.com (http://www.Lonelyplanet.com)

4.1 Football result

Football results data archive contains actual football scores and related information of Premiership in England for the 1996/97 season. The format of this domain is described already in section 3.1. To be extracted are name of home team, name of away team, result and total match sentence. We used a training set with 40 examples and a test set with 400 examples. We trained concepts with three steps: at first step we trained all team names as the concept TEAM, at the second step, concept RESULT is trained with training result. Finally, the total MATCH is trained at step three. We got the concept for MATCH. The training results are a set of rules similar to those reported in the proceeding worked example. Evaluation of the system yielded results of 99% precision and 98.6% recall. These near-perfect results are not unexpected given the highly structured nature of the data – hence this experiment demonstrated that the hierarchical learning passes the minimal performance criteria.

4.2 Rental Ads

The motivation behind the Rental Ads domain was to enable a direct comparison to experiments previously carried out on the WHISK [SODE99] system. The template for information to be extracted consists of three content elements: information about neighbourhoods, about price and about the number of bedrooms. Below is given a sample of the description with an indication of what information should be extracted.

Ballard 2 BR/2 ba, w/d. 1500 sf, Penthse, d/w. 7533 24 th Av NW. $1195 Wu Prop 206-522-8172 #371
Belltown CONCEPT ONE, 1 BDRM, $775-$895, Lake Union & Sound Views, Fplc, W/D, Gar Prkg Available 206-728-9515 …

Two experiments are made with this data set. In the first experiment, the hierarchical concept learning is disabled – i.e. learning is performed based on the base text. In the second experiments hierarchical learning is enabled. Indeed, a new concept is introduced, BEDROOM, which is first trained to represent the synonyms of bedrooms (such as bd, bds, and bdrs ect.). With this concept in place the rules were trained to extract the three desired content elements.

As for WHISK [SODE99] a training set of 400 examples was used and a test set of a further 400 examples. In the first experiment recall begins at 88.4% from 50 examples and climbs to 95.0% when 400 examples. Precision starts at 87.0% from 50 examples and reaches to 97.7% when 400 examples. In the second experiment the precision ranges slightly between 96.7% and 97%, while the recall becomes from 89.8% at 50 examples to 96.0% when 400 examples. The results (after NE-Agent) are shown in comparison with WHISK as following[2]:

[2] The result of precision and recall in this table is not the final result. The final result after TE-Agent is not shown because it is not relevant to hierarchical learning approach.

	Examples	Precision	Recall
WHISK	50	94%	73%
	400	98%	94%
Experiment 1	50	73.0%	91.4%
	400	90.2%	97.0%
Experiment 2	50	94.2%	91.3%
	400	95.0%	97.0%

Table 1. Result of Rental Ads (After NE-Agent)

At the first experiment, there is good recall and poor precision after NE-Agent processing. At the second experiment, the precision already reaches to 94% when 50 examples after NE-Agent. Fig. 5 shows the total learning curves for the Rental Ad domain.

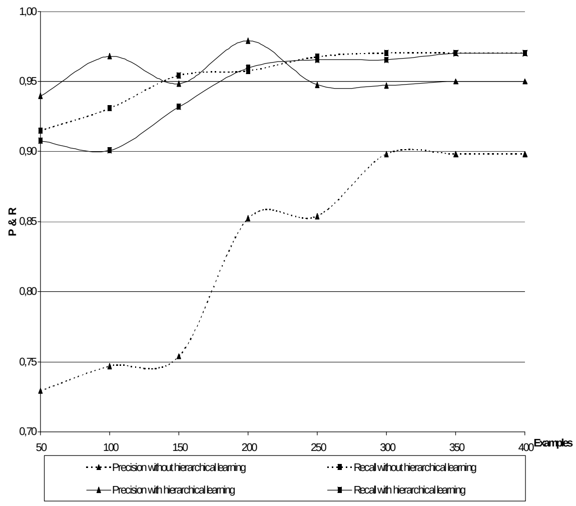

Fig. 5. Learning Curve for Rental Ads (After NE-Agent processing)

For this straightforward data set, what is demonstrated is that hierarchical learning and flat learning have little difference in terms of the eventual accuracy (precision and recall) that can be attained, but that the hierarchical learning algorithm reaches this high level of accuracy very much more rapidly than the flat learning approach.

4.3 Lonelyplanet

Lonelyplanet WorldGuide describes exiting travel information in the world. We gathered web pages about attractive activities of the tourist areas, which describe the destination of the area, types of activities (such as fishing, swimming, hiking etc.). As a first experiment on processing this less well structured data, information concerning hiking, e.g. the distance of a specific hike, was extracted. The descriptions of such information are highly varied: some are absolute distance values (e.g. "*The most popular hike on the continent is the 33km (20mi) Inca Trail, west of Cuzco*".), some are described as walking hours (such as "*The hike to the top of Mt Finkol takes 8 to 10 hours* "), some are only fussy values (e.g. "*There are dozens of day hikes and multi-day hikes throughout the island.*"). This information is embedded in documents with a lot of other distance information (distance between towns, height of mountains, etc.). To be extracted are value of distance, unit of distance and also the short text description about hiking distance.

The training set of this experiment is 50 examples, while the testing set contains 260 Lonely Planet WorldGuide descriptions. Again, two experiments were performed. In the first hierarchical learning was disabled. The input was a set of examples with position information, the output was a set of training rules about the contents.

In the second experiment hierarchical learning was performed in four steps: Step 1, the distance unit (such as *km*, *ft* etc.) and keywords about hiking (such as *Hike*, *Hiking*, *Hikers* etc.); Step 2, the distance value (such as *10*, *3000* etc.); Step 3, the combined distance information containing both distance value and distance unit (e.g. *4km*, *13 miles* etc.); Step 4, rules to extract the total sentence about hiking distance are trained. The following tables show evaluation result of two experiments.

Concept	Level	Precision	Recall	Example of Rules
HikingTerm (HIKING)	1	0,252	1,00	*1.00 {c hiking c}* *1.00 {c hikers c}*
DistanceUnit (DISU)	1	0,139	1,00	*1.00 {c miles c}* *1.00 <int> {c km c}*
DistanceValue (DSV)	1	0,321	1,00	*1.00 trail [2] {c <int> c}* *1.00 {c <int> c} [1] (*
Distance (DISTANCE)	1	0,352	1,00	*1.00 {c <int> miles c}* *1.00 {c <int> - [1] c}*
HikingDistanceInfo	1	0.150	0,321	*0.80 {c the [5] <int> [1] c}*

Table 2. Result of Lonely Planet without Hierarchical Learning

Concept	Level	Precision	Recall	Example of Rules
HikingTerm (HIKING)	1	0,252	1,00	*1.00 {c hiking c}* *1.00 {c hikers c}*
DistanceUnit (DISU)	1	0,139	1,00	*1.00 {c miles c}* *1.00 <int> {c km c}*
DistanceValue (DSV)	2	0,365	1,00	*1.00 {c <int> c} -* *1.00 {c <int> c} <DISU>*
Distance (DISTANCE)	3	0,459	1,00	*1.00 {c <DISV> - <DISU> c}* *1.00 {c <DISV> <DISU> c}*
HikingDistanceInfo	4	1,000	0,92	1.00 to [10] {c <HIKING> [10] <DISTANCE> c} 0.40 {c <HIKING> [5] <DISTANCE> c} 0.40 {c <HIKING> [10] <DISTANCE> c}

Table 3. Result of Lonely Planet with Hierarchical Learning

Clear benefits in terms of increased Recall/Precision are demonstrated in these results.

5. DISCUSSION

In this section, the key points raised by the above experiments will be discussed. The evaluation clearly demonstrates that hierarchical concept learning generally improves overall precision and, to a lesser extend, overall recall (e.g. HikingDistanceInfo). Furthermore, the Rental Ads experiment clearly shows an accelerated rate of hierarchical learning with respect to flat learning.

For the Rental Ads domain, both the performance of both the hierarchical learning approach and the WHISK benchmark are relatively flat – i.e. reach high levels of precision and recall with relatively few training examples. In the case of WHISK background knowledge is introduced before extracting, as a so called semantic class. For instance, in the Rental Ad domain the semantic class Bedroom has been defined as the disjunction:

$$Bedroom = \{brs|br|bds|bdrm|bd|bedrooms|bedroom|bed\}$$

The introduction of this background knowledge trivialises the extraction task. The high performance of the hierarchical learning can be explained in the same way: the pre-trained concept BEDROOM used as a first step makes the other content elements much easier to extract. The difference is that the BEDROOM concept is represented as a set of examples rather than being manually encoded. Thus the conclusion is that in order to enable high accuracy information extraction by hierarchical learning, care should generally be taken to select and define (through example) basic concepts that will simplify the learning task through the normalisation of the raw text.

Another advantage of hierarchical learning is that we can get satisfactory performance (precision and recall) with less training examples than traditional learning algorithm. By replacing some contents with formatted token strings the machine learning system can have standardized information as training input. These standardized inputs can reduce the variation of contents and perform a better and quicker learning result.

The above discussion shows that higher levels of accuracy can be attained more rapidly through hierarchical learning because of the normalising effect of the basic concepts used to define more complex concepts (e.g. HikingDistanceInfo defined in terms of DISTANCE). This also introduces another methodological advantage, namely that of reuse. The same basic concepts can be used to define related complex concepts. Hence, DISTANCE can be reused to define DistanceBetweenCities, etc. In this way, the knowledge engineering required for large-scale information extraction projects can be minimised.

The hierarchical learning approach also provides a generic framework by which different information extraction techniques can be combined – the basic assumption being that any information extraction technique carries out a normalisation operation on the raw text. Taking this idea one step further, the use of NLP to carry out, for example POS tagging or stemming as a pre-processing operation for hierarchal concept learning. The investigation of how different machine learning techniques and/or NLP techniques can be most favourably combined in this manner is left to future work.

6. CONCLUSION AND FUTURE WORK

In this paper we introduced a new methodology for extracting information from semi-structured documents. A new approach for hierarchical concept description is introduced. By using a simple inductive rule extractor a hierarchical concept learning algorithm is performed. From the evaluation we showed that the hierarchical concepts can improve overall precision and recall of the total system. With hierarchical concept learning less examples than normal are need in machine learning, because the machine learning algorithm receives more standardised information as input. Furthermore the hierarchical concepts can reduce the amount of rules needed to cover all surface-level variations. This feature enables user to create complicate concepts in a simple way.

In the future we will investigate the combination of different ML techniques, such as a fuzzy synonym extractor or an NLP component, within the hierarchical learning framework. Further extensions will be to investigate the use of hierarchical concept learning beyond standard information extraction tasks, for example, to summarize and classify documents.

With the introduction of our hierarchical learning system, and other IE systems like it, a solution is advanced for the difficult and currently crucial problem of designing artificially intelligent agents for extracting and formalising the content of large informal document archives and from the internet. Such approaches are vital if the vision of the Semantic Web is to be made reality.

REFERENCES

[BROW98] Combining Information Retrieval and Case-Based Reasoning for 'Middle Ground' Text Retrieval Problems - Brown M, Förtsch C, Wißmann D, Proceedings AAAI-98 Workshop on Textual Case-Based Reasoning

[CALI97] Relational Learning of Pattern-Match Rules for Information Extraction, M E Califf and R J Mooney, Proceedings ACL-97: Workshop on Natural Language Learning

[CUNN99] Information Extraction a User Guide , H Cunningham, CS-99-07, Dept. Of Comp Sci., Uni. Of Sheffield, 1999

[DIXO97] An Overview of Document Mining, Dixon M, http://www.geocities.com/ResearchTriangle/Thinktank/1997/mark/writings/main-html

[MUSL98] STALKER: Learning Extraction Rules for Semistructured, Web-based Information Sources - Muslea I, Minton S, Knoblock C, AAAI'98 Workshop "AI and Information Integration"

[RAJM98] Text Mining - Knowledge extraction from unstructured textual data, M Rajman and Besançon - 6th Conf. of International Federation of Classification Societies

[RILO94] Information Extraction as a Basis for High-Precision Text Classification, E Riloff and W Lehnert, ACM Transactions on Information Systems vol. 12 no. 3 1994.

[SODE97] Learning to Extract Test-based Information from the World Wide Web, S Soderland, Proceedings of 3rd Int. Conf. On Knowledge Discovery and Data Mining (KDD-97)

[SOND99] Learning Information Extraction Rules for Semi-Structured and Free Text, S Sonderland, Machine Learning 34, 233-272, 1999

[XIAO01a] Information Extraction from HTML:Combining XML and standard Techniques for IE from the Web, L Xiao, D Wissmann, M Brown, S Jablonski, to be published, IEA/AIE 2001; The Fourteenth International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems 2001 Budapest.

[XIAO01b] Where to Position the Precision in Knowledge Extraction from Text, L Xiao, D Wissmann, M Brown, S Jablonski, to be published, IEA/AIE 2001; The Fourteenth International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems 2001 Budapest.