Journal of Universal Computer Science, vol. 21, no. 11 (2015), 1482-1495
submitted: 10/5/14, accepted: 12/1/15, appeared: 1/11/15 © *J.UCS*

A Framework for Extraction of Relations from Text using Relational Learning and Similarity Measures

Maria Vargas-Vera
(Adolfo Ibanez University, Vinia del Mar, Chile
mvargasvera@gmail.com)

Abstract: Named entity recognition (NER) has been studied largely in the Information Extraction community as it is one step in the construction of an Information Extraction System. However, to extract only names without contextual information is not sufficient if we want to be able to describe facts encountered in documents, in particular, academic documents. Then, there is a need for extracting relations between entities. This task is accomplished using relational learning algorithms embedded in an Information Extraction framework. In particular, we have extended two relational learning frameworks RAPIER and FOIL. Our proposed extended frameworks are equipped with DSSim (short for Dempster-Shafer Similarity) our similarity service. Both extended frameworks were tested using an electronic newsletter consisting of news articles describing activities or events happening in an academic institution as our main application is on education.

Key Words: Semantic Learning, Relational Learning, Similarity Measures, Semantic Web
Category: K.3, K.3.1, K.3.2, H.0, H.3.3

1 Introduction

An important pre-condition for realizing the goal of the semantic web is the ability to extract relations between entities from a large collection of documents. We argue that to recognize entities in isolation in documents is not enough to drawn conclusions (i.e. inferences cannot be drawn safely). In order to reason about entities, a system should be able to extract facts involving them. We envisage two problems which could occur when extracting relations from text.

1. An information extraction system using Machine Learning relies on an annotated set of documents for the training phase. This training phase is performed by presenting positive examples of the concept to be learnt. Sometimes a set of negative examples needs to be provided as well depending of the learning algorithm used. After applying the learning algorithm, a library of patterns is learnt by the system. These patterns are then used for extraction of similar patterns on unseen documents [Lloyd 2000] [Michalski etal, 1986]. The main drawback is that training an information extraction system requires a big effort, particularly by non-experts users. However, there are several research efforts in the direction of reducing the amount of effort required by the training phase, such as Melita [Ciravegna et al. 2002] and

PANKOW [Cimiano et al. 2004] which aim to reduce the effort in the training phase.

2. Currently, there are several information extraction engines available. However, information extraction engines present the problem that if a document contains more than one instance of a concept, then these information extraction engines will not be able to allocate the correct properties to the correct instance because they are simply unable to differentiate among them. A typical example is a home page with several names and phone numbers. These information extraction engines would not be able to assign phone numbers to persons. This can be avoided by ensuring that no document has more than one instance of a concept. Ideally, however, information extraction systems should make use of concept-property structures when suggesting annotations. Since an ontology contains the conceptual structure of a domain and the populated ontology contains domain-specific knowledge, it would seem sensible to make use of this when performing information extraction. This suggests that there should be a two-way exchange with information extraction modules – the results of information extraction should be used to populate knowledge bases and the contents of knowledge bases should be available for the information extraction process. Even, if information extraction systems cannot make use of specific domain knowledge they should still be able to make use of its domain-specific structure. This direction of research has been reported in [Vargas-Vera etal, 2001].

The range of possible applications from a system which extract relations from text is vast. This could range from text summarization, search engines, automatic creation of knowledge bases/ontologies, ontology maintenance (adding instances of a specific class), among others.

Our main contributions are described as follows: we analyze existing systems for information extraction using relational learning algorithms and perform experiments using an electronic newsletter. In this way, we have assessed the main capabilities and restrictions of some relational systems available. Finally, we outline a new framework that performs relational learning and in addition, performs reasoning using the Web as a resource and in addition it uses similarity metrics.

The rest of the paper is organized as follows: in the next section we present a framework using KNOW-RAPIER as a relational learning system. Section 3 presents a framework for relational learning using FOIL[1] . Section 4 presents an evaluation of the similarity algorithm added to our framework. Section 5 presents related work. Finally, section 6 re-states the main results from our research.

[1] FOIL learns First Order Logic formulae from a given specification.

2 Relational Learning using RAPIER

This section presents background on RAPIER and a proposal for an extended architecture called KNOW-RAPIER (Knowledge ON the Web RAPIER). This extension uses the web as a resource in order to prove facts extracted from documents. Of course, this leads to associating confidence values to the facts proved using the Web. In this research, we used Prolog [Clocksind and Mellish 1981] from the POW system [Kushmerick 2003], [Kushmerick etal, 1997] to achieve reasoning capabilities.

RAPIER was inspired by Inductive Logic Programming systems (ILP). It has ideas from GOLEM [Muggleton and Feng 1992] CHILLIN [Zelle etal, 1994] and PROGOL [Muggleton 1995]. As a remainder to the reader, we introduce briefly RAPIER. RAPIER uses patterns that make use of syntactic and semantic information. It uses a part-of-speech tagger called POS Brill tagger [Brill 1994] as well as a lexicon with semantic classes. Each rule in RAPIER is defined as follows: Pre-filler + Filler + Post-filler

Where the symbol + denotes the concatenation operator. The idea behind these rules is that the context of the words in front of and behind the filler is kept. The tags are the tags generated by the Brill tagger [Brill 1994]. RAPIER induces rules from a pair consisting of document and a document template. The RAPIER learning algorithm uses techniques from several Inductive Logic Programming systems and it learns unbound patterns that include constraints at the level of words and POS tags surrounding the filler [Califf and Mooney 1997]. As a first instance, we have performed experiments using the same corpus used in [Califf and Mooney 1997] job postings from newsgroup misc.jobs.offered. From these experiments, we concluded that RAPIER uses semantic constraints in a limited way lexicons associated to a semantic class by means of is-a relation. Therefore, our proposal tries to extend the reasoning capabilities using semantic relations defined in a given ontology.

2.1 Learning Algorithm

The RAPIER learning algorithm is not shown in this paper. However, a thorough description of the algorithm can be found in [Califf 1998]. In the RAPIER algorithm, a best rule, is a rule which produces only valid fillers or if the rule still extracts spurious fillers, then the number of spurious fillers has to be below a given threshold. The best rule is estimated using Laplace estimate of the probabilities.

rule_evaluation_metric = - log2(p+1 / p+n+2) + rule_size / p

Where p is the number of correct fillers extracted by the rule and n is the number of spurious fillers extracted by the rule.

The rule-size is computed by using the following heuristic: each pattern item counts 2; each pattern list counts 3, each disjoint in a word constraint counts 2 and each disjoint in a POS tag constraint or semantic constraint counts 1 [Califf and Mooney 1997] .

2.2 KNOW-RAPIER process model

The process model of KNOW-RAPIER can be summarized in four main processes: annotation, learning, extraction, and validation.

2.2.1 Annotation

The activity of semantic tagging refers to the activity of annotating text documents (written in plain ASCII or HTML). In RAPIER, the annotation process is performed using a filled template from each example in the training set, whilst in KNOW-RAPIER annotation is performed using the slots names of classes in a selected ontology. For example, let us consider ontology of events, for instance, the structure of the conferring an award event is presented below.

Class award description: Class of an event describing an event of presenting an award to someone

Slots:
Has-duration (when or how long for the event took place)
Start-time (time-point)
End-time (time-point)
Has-location (a place where it took place) recipient-agents (the agents who received the award)
Has-awarding-body (an organization, donor)
Has-award-rationale (what the award is for)
Object-acted-on (award, name of the award or amount of money)

Annotations were performed using MnM ([Vargas-Vera etal, 2002]) by selecting slots names such as location, recipient-agent, awarding-body and so forth.

2.2.2 Learning

The learning phase is performed using the RAPIER algorithm. RAPIER begins with a most specific definition and then attempts to compact that definition by replacing rules with more general rules. The RAPIER generalization method operates on the principle that the relevant information for extracting a slot-filler will be close to that filler in the document. A summary of a generalization of a pair of patterns is as follows: RAPIER starts by generalizing the two filler patterns. Then, RAPIER creates rules with the resulting generalized filler patterns and empty pre-filler and post-filler patterns. RAPIER specializes subsequently those rules by adding pattern elements to the pre-filler and post-filler patterns working outward from the filler. The elements to be added to the patterns are created by generalizing the pre-fillers or post-fillers of the pair of rules from which a generalized rule is created. A thorough description of the generalization algorithm can be found in [Califf 1998].

2.2.3 Extraction

RAPIER built a library of induced rules, so they can be used to extract information from unseen texts. When working in extraction mode, RAPIER receives as input a collection of texts with the associated templates. It preprocesses the texts by using templates filled by each document and Brill POS tagged documents. Then it applies its rules and returns as output a filled template for each document in the training set. Then, the information extracted is presented to the user for approval. Finally, the extracted information is sent to the ontology server which populates the selected ontology.

We used RAPIER in the domain of an electronic newsletter. In particular, it was used to generate rules for example for two events: visiting-a-place-or-people and conferring-an-award. The information extraction rules learnt from our experiment are described below.

The extraction rules are indexed by template name and slot name. They consist of the three parts 1) a pre-filler that matches text immediately preceding the filler, 2) a pattern that must match the actual slot filler and 3) a post-filler pattern that must match the text immediately following the filler. Each pattern is a sequence of pattern elements. These pattern elements can be pattern items or pattern lists. A pattern item matches exactly one word or symbol from the document that meets the items constraints. Whilst a pattern list specifies a maximum length N and it matches from 0 to N symbols from the document, each of these must match the list constraints.

The format in which the rules are presented is as follows:

Rule(template_name, slotName, NumPosCovered, NumNegCovered, Pre-filler-pattern, Filler-Pattern, Post-filler-Pattern).

Each of the obtained patterns are presented as Item(WordConstraints, Tag-Constraints) List(Length,WordConstraints,tagConstraints)

Where the constraints are - if empty, a single constant if there is one constraint and a Prolog list if the constraint has more than one disjoint. For a further description of Prolog syntax see [Clocksind and Mellish 1981].

The set of rules obtained by RAPIER (written in Prolog notation) are shown below. The rules below were extracted using our corpus of news consisting of articles from an academic organization.

Rule 1 is a rule for extracting a place and Rule 2 is a rule for extracting money. As a reminder to the reader, place, money and project are slots to be extracted in a conferring-an-award event.

Rule 1: rule(award_template, place, 8, 0,[],[item('ou,'nnp',_,_,_,_)],[]).

Rule 2: rule(award_template, money, 4,0,[],[item('1m','cd',_,_,_,_)],[]).

Rule 1 has an empty pre-filler, filler is a single word (ou) constrained to a POS tag label of proper noun singular (nnp) and the post-filler in the rule is also empty. In a similar fashion, Rule 2 has pre-filler and post-filler empty and the filler is a single word (1m) constrained with a POS tag label of cardinal number (cd).

2.3 Extended RAPIER (KNOW-RAPIER)

We have extended RAPIER with a reasoner. This extensiom is called from now on KNOW-RAPIER. The reasoner validates the extracted information using two sources 1) predefined axioms already defined in a selected ontology and 2) the WEB (as an alternative resource). The inferences are performed using POW [Kushmerick 2003].

The KNOW-RAPIER extension uses a given ontology for the annotation phase and for reasoning using generalization/specialization. The KNOW-RAPIER learning algorithm is shown in algorithm 1.

In our framework, DSSim produces a set of mappings by means of the use of the agent archicterure which uses Depmpter Shafer and Fuzzy Voting Model. Further description can be found in [Nagy and Vargas-Vera 2011].

POW allows retrieval of home pages by finding evidences from specific concepts. For example, POW could find evidences for the query Is Maria Vargas-Vera a researcher? In this example, Pow attempts to prove: researcher(maria_vargas-vera) using the CiteSeer Database and Institutional Databases as a background knowledge.

3 Relational Learning using FOIL

We have also used FOIL [Quinlan and Cameron-Jones 1993], [Mitchell 1997] in our analysis of existing relational systems. FOIL generates First Order Logic Formulas from a given specification. The rules learnt by FOIL are restricted Horn Clauses since it is not allowed to have predicates as arguments and, secondly, clauses in the body of a clause could be negated. This gives more expressivity to the FOIL rules than a logic language without negation such as Datalog [Ullman 1988]. A full description of FOIL can be found in [Mitchell 1997], [Mitchell 1982], [Quinlan and Cameron-Jones 1993].

We used FOIL on an electronic newsletter consisting of news articles from an academic organization. In particular, the FOIL system was used to induce

Input: A set of documents $\{D_1, .., D_k\}$
Output: Extracted Values
θ = TestSet
Extract rules using RAPIER algorithm
Test ListRules on θ
Take(Doc.name.filler) documents obtained from RAPIER
Redefine extracted entities obtained by RAPIER into FOL (First Order Formulas) as SlotName(ExtractedValues)
Call DSSim similarity and create a list of possible mappings (MappingSet)
Call POW using domain-specific rules and test all elements of the MappingSet
Prove facts using Web as repository
if *SlotName(ExtractedValue)= true* **then**
    Offer ExtractedValues as extracted relation
end
else
    Highlight extracted value as they need to be checked by the user AND show information to the user
end

Algorithm 1: KNOW-RAPIER

rules for the event visiting-a-place-or-people. FOIL requires specification defining relations, the type of each argument in the relation and examples of each defined relation. The specification given to FOIL is shown below.

Types:
Person: David, Tony, Maria, Victoria, Martin, Clara.
Organization: Open, Sheffield.
Relations:
people(Person).
people(David), people(Maria), people(Tony),
people(Victoria), people(Martin),people(Clara)
places(Organization)
place(Open), place(Sheffield)
visitor(Person, Organization)
visitor(David, Open),visitor(Maria, Sheffield),
visitor(Tony, Open),visitor(Victoria, Open),
visitor(Martin, Open),visitor(Clara, Sheffield)
works(Person, Organization)
works(David, Sheffield),works(Tony, Sheffield),

works(Victoria, Sheffield), works(Martin, Sheffield),
works(Clara, Open)

The results obtained by FOIL are shown as follows:

visitor(A,B) :- not works(A,B), people(A), place(B).

The visitor rule can be interpreted as A is a visitor of B if A does not work in B and A is person and B is a place.

We outline our vision of an extended FOIL architecture for extracting information from a given corpus. The architecture needs several components. 1) creation of specification by an expert using an interface. 2) generation of rules by using FOIL 3) translation of test documents in First Order Logic formulas, 4) validation of rules using the test set and 5) filling templates and creation of instances. Each of the components are self explicated and for the sake of space, in this paper we only describe "validation of the rule base" in more detail. Validation of rule base implies translating the test set corpus into First Order Logic formulae, then, the system evaluates FOL using the Rule Base. One problem encountered was the difference between vocabularies used in the FOIL- extracted First Order Formulas and the First Order Logic Formulas obtained from the Test Set. Therefore, we concluded that a similarity algorithm which allows the system to find similarity between relation/concepts was needed. Work in this direction has been carried out in the AQUA (Automated Question Answering system) described in [Vargas-Vera and Lytras 2010], [Vargas-Vera and Motta 2004], [Vargas-Vera etal, 2003]. Further details on similarity algorithm are described in turn.

4 Similarity algorithm evaluation

Our approach to assess similarity between a pair of terms is an agent based approach where we have several agents performing mappings and then we combine the evidences found by each of them using Dempster-Shafer Theory. This solution is generic as the system does not need to learn mappings in advance like other approaches which use Machine Learning techniques. Our mapping algorithms get evidences from different sources like WordNet and background knowledge (specific to the domain's ontology). Furthermore, our approach deals with uncertainty in mappings whilst other approaches only handle two valued logic 0, 1 [Nagy and Vargas-Vera 2011]. Also, we included into Dempster-Shafer rule of combination a Fuzzy Voting Model to resolve the problem of "contradictory evidences" which are ignored when using Dempster-Shafer rule of combination. The Dempster-Shafer rule strongly emphasises the agreement between multiple sources and ignores all the conflicting evidence through a normalization factor. Our similarity algorithm is shown in algorithm 2.

We have carried out evaluation using the benchmark ontologies from the Ontology Alignment Evaluation Initiative (OAEI), which is an international ini-

1 The user poses a natural language query to the AQUA system which converts it into FOL (First Order Logic) formula.
2 Broker agent receives FOL formula, decomposes it and distributes the sub queries to the mapping agents.
3 Mapping agents retrieve sub query class and property hypernyms from WordNet.
4 Mapping agents retrieve ontology fragments from the external ontologies which are candidate mappings to the received sub-queries.
5 Mapping agents use Word-Net as background knowledge in order to enhance their beliefs on the possible meaning of the concepts or properties in the particular context.
6 Mapping agents build up coherent beliefs by combining all possible beliefs over the similarities of the sub queries and ontology fragments.
7 Mapping agents utilize both syntactic and semantic similarity algorithms and build their beliefs over the correctness of the mapping.
8 Broker agent passes the possible mappings into the AQUA system for particular sub-query ontology fragment mapping in which the belief function has the highest value.
9 AQUA retrieves the concrete instances from the external ontologies or data sources which will be included into the answer and it creates an answer to the users question.

Algorithm 2: Similarity Algorithm

tiative that has been set up for evaluating ontology matching algorithms. The experiments were carried out to assess how trust management influences results of our mapping algorithm. Our main objective was to evaluate the impact of establishing trust before combining beliefs in similarities between concepts and properties in the ontology. The OAEI benchmark contains tests, which were systematically generated starting from some reference ontology and discarding a number of information in order to evaluate how the algorithm behave when this information is lacking. The bibliographic reference ontology (different classifications of publications) contained 33 named classes, 24 object properties, 40 data properties. Furthermore, each of the generated ontologies was aligned with the reference ontology. The benchmark tests were created and grouped by the following criteria:

– Group 1xx: simple tests such as comparing the reference ontology with itself, with another irrelevant ontology or the same ontology in its restriction to OWL-Lite

– Group 2xx: systematic tests that were obtained by discarding some features from some reference ontology e.g. name of entities replaced by random strings or synonyms

- Group 3xx: four real-life ontologies of bibliographic references that were found on the web e.g. BibTeX/MIT, BibTeX/UMBC.

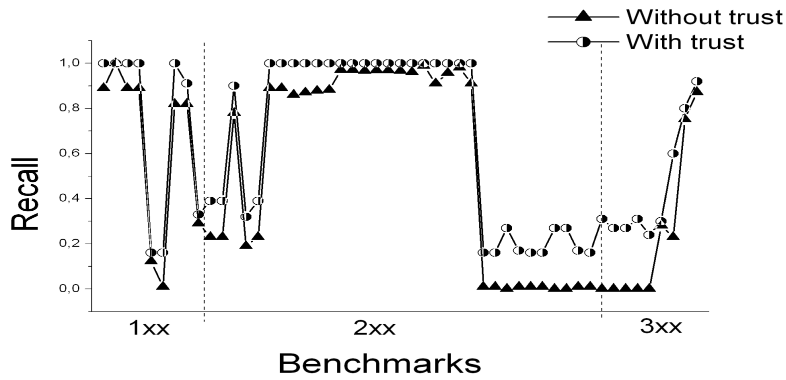

Figure 1: Recall graph with and without applying fuzzy voting

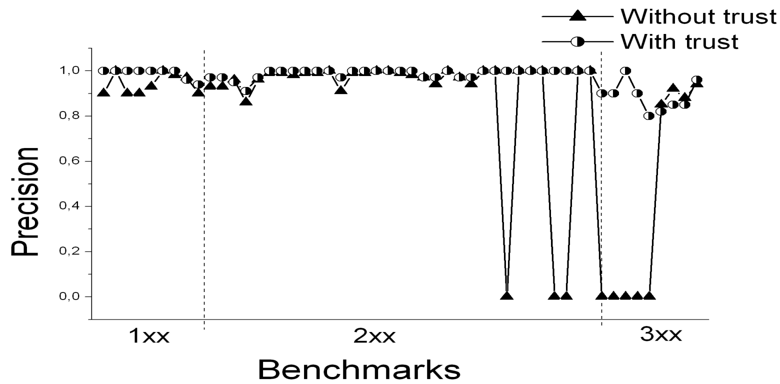

Figure 2: Precision graph with and without applying fuzzy voting

Figure 1 and 2 show the improvement in recall and precision that we have achieved by applying our trust model for combining contradictory evidence. From the precision point of view, the increased recall values have not impacted the results significantly, which is good because the objective is always the improvement of both recall and precision together.

As a basic comparison we have modified our algorithm (without trust), which does not evaluate trust before conflicting belief combination just combine them using Dempster's combination rule. The recall and precision graphs for the algorithm with trust and without trust over the whole benchmarks are depicted on Figure 1 and 2. Experiments have proved that with establishing trust one can reach higher average precision and recall rate. Figure 1 and 2 show the improvement in recall and precision that we have achieved by applying our trust model for combining contradictory evidences. From the precision point of view the increased recall values have not impacted the results significantly, which is good because the objective is always the improvement of both recall and precision together. We have measured the average improvement for the whole benchmark test set that contains 51 ontologies. Based on the experiments the average recall has increased by 12 % and the precision is by 16 %. The relative high increase in precision compared to recall is attributed to the fact that in some cases the precision has been increased by 100 % as a consequence of a small recall increase of 1 %. This is perfectly normal because if the recall increases from 0 to 1 % and the returned mappings are all correct (which is possible since the number of mappings are small) then the precision is increases from 0 to 100 %. Further, the increase in recall and precision greatly varies from test to test. Surprisingly, the precisions have decreased in some cases (5 out of 51). The maximum decrease in precision was 7 % and maximum increase was 100 %. The recalls have never decreased in any of the tests and the minimum increase was 0.02 % whereas the maximum increase was 37 %.

5 Related work

A number of information extraction systems have been described in the literature. However, only the most closely related to our work are shown below.

Crystal creates a concept dictionary of extraction patterns by generalizing patterns annotated by an expert in linguistic [Soderland etal, 1996], [Soderland etal, 1995].

Whisk learns automatically regular expressions from text. It does not requires sentence analysis like Crystal [Soderland etal, 1995].

AutoSlog creates a dictionary of extraction patterns by specializing a set of general syntactic patterns [Riloff 1996]. [Riloff 1993].

PALKA learns extraction patterns relying on a concept hierarchy to guide generalization and specialization [Kim and Moldovan 1995].

MnM [Vargas-Vera etal, 2002] is an annotation system which provides both automated and semi-automated support for marking up web pages with semantic contents. MnM integrates a web browser with an ontology editor and provides open APIs to link up to ontology servers and for integrating information extraction tools. MnM had been equipped for the learning phase. with Almicare from Sheffield University and Badger and Crystal from Amherst University of Massachusetts.

Melita [Ciravegna et al. 2002] adopts an approach similar to MnM in providing information extraction-based semantic annotation. Work on Melita has focused on Human Computer Interaction issues such as limiting intrusivity of the information extraction system and maximizing proactivity and timeliness in suggestions. Melita does not provide sophisticated access to the ontology, as MnM provides. In this sense Melita explores issues that are complementary to those explored in developing MnM and indeed, the two approaches could be integrated.

6 Conclusions and Future Work

This paper has described two different frameworks for Relational Learning KNOW-RAPIER and extended FOIL. The first framework, called KNOW-RAPIER, has an architecture which combines Information extraction technologies and internet as a resource and a repository of facts. It uses RAPIER, POW and DSSim internally. Preliminary experiments have been carried out using news articles (describing events happening in an academic institution) and benchmarks from the OAEI community. Early results are encouraging in terms of information extraction. However, there is clearly a lot more work needed to make this technology easy to use for our target users (people who are neither experts in language technologies nor 'power knowledge engineers'). More work is needed in the direction of usability. The second framework - extended FOIL, has shown the need for similarity algorithms which are embedded in our DSSim. Future work is to carry out more evaluation using a different corpus.

Acknowledgements

The authors would like to thank Miklos Nagy for their invaluable help performing benchmarks evaluation.

References

[Brill 1994] Brill E.: "Some advances in rule-based part of speech tagging". Proc. of the Twelfth National Conference on Artificial Intelligence, 722-727, 1994.

[Califf 1998] Califf M.E.: "Relational Learning Techniques for Natural Language Information Extraction". PhD thesis, Department of Computer Sciences, University of Texas, Austin, TX, August 1998.

[Califf and Mooney 1997] Califf M.E. and Mooney R.J.: "Relational Learning of Pattern-Match Rules for Information Extraction". Proc. of the ACL Workshop on Natural Language Learning, 9-15, Madrid, Spain, July 1997.

[Cimiano et al. 2004] Cimiano P., Handshuh S. and Staab S.: "Towards the self annotated web". Proc. of the 13th international conference on World Wide Web. 462 471, New York USA, 2004.

[Ciravegna et al. 2002] Ciravegna F., Dingli A., Petrelli D., and Wilks Y.: "User-System Cooperation in Document Annotation based on Information Extraction". Proc. of the 13th International Conference on Knowledge Engineering and Knowledge Management, EKAW02, publisher Springer Verlag, 2002.

[Chirico 2003] Chirico U.: "JIProlog 2.0 Reference Manual", 2003. http://www.ugosweb.com/jiprolog/

[Clocksind and Mellish 1981] Clocksin W. F. and Mellish C. S.: Programming in Prolog, Springer-Verlag, 1981.

[Kim and Moldovan 1995] Kim J-T. and Moldovan D. I.: "Acquisition of Linguistic patterns for knowledge-based information extraction". IEEE Transactions on Knowledge and Data Engineering, 7(5):713-724, 1995.

[Kushmerick 2003] Kushmerick N.: "POW Prolog on the Web". Version pow-2.0.Computer Science, Department, University College Dublin, 2003.

[Kushmerick etal, 1997] Kushmerick N. and Weld D. and Doorenbos R.: "Wrapper induction for information extraction". Proc. of 15th International Conference on Artificial Intelligence, IJCAI-97, 1997.

[Lloyd 2000] Lloyd J. W.: "A Logical Setting for the Unification of Attribute-Value and Relational Learning". Proc. of the workshop Attribute-Value and Relational Learning: Crossing the Boundaries in association with the Seventeenth International Conference on Machine Learning (ICML-2000). Stanford University, 2000.

[Michalski etal, 1986] Michalski R. S. and Mozetic I. and Hong J. and Lavrack H.: "The multi purpose incremental learning system AQ15 and its testing application to three medical domains". Proc. of the 5th National Conference on Artificial Intelligence, Philadelphia. Morgan Kaufmann publisher, 1986.

[Mitchell 1997] Mitchell T.: "Machine Learning". McGraw-Hill International Editions, 1997.

[Mitchell 1982] Mitchell. T.: "Generalization as search". Artificial Intelligence, 18 203-226, 1982.

[Miller etal, 1997] Miller G. A., and Charles W. G.: "Contextual correlates of semantic similarity. Language and Cognitive Processes", 6(1), 1-28, 1997.

[Muggleton 1995] Muggleton S.: "Inverse entailment and Progol". New Generation Computing Journal, 13:245-286, 1995.

[Muggleton and Feng 1992] Muggleton S and Feng C.: "Efficient induction of logic programs". In S Muggleton, editor, Inductive Logic Programming. Academic Press, New York, 281-297, 1992.

[Nagy and Vargas-Vera 2011] Miklos Nagy and Maria Vargas-Vera.: "Multiagent Ontology Mapping Framework for the Semantic Web". IEEE Transactions on Systems, Man, and Cybernetics - Part A Journal (TSMCA), 41(4), 693-704; 2011.

[Quinlan and Cameron-Jones 1993] Quinlan J. R and Cameron-Jones R. M.: "FOIL". A Midterm Report. ECML 1993: 3-20, 1993.

[Riloff 1996] Riloff E.: "Automatically generating extracting patterns from untagged text". Proc. of the Thirteenth National Conference on Artificial Intelligence, 1044-1049, 1996.

[Riloff 1993] Riloff E.: "Automatically constructing a dictionary for information extraction tasks". Proc. of the Eleventh National Conference on Artificial Intelligence, 811-816, 1993.

[Soderland 1999] Soderland S.: Learning Information Extraction Rules for Semi-Structured and Free Text. Special issue on natural language learning, Volume 34 , Issue 1-3, 233 272, 1999.

[Soderland etal, 1996] Soderland S., Fisher D., Aseltine J., Lehnert W.: Issues in inductive learning of domain-specific text extraction rules. In S. Wermter, E. Rillof and G. Scheller editors, Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing. Lectures Notes in Artificial Intelligence Springer 290-301, 1996.

[Soderland etal, 1995] Soderland S., Fisher D., Aseltine J., Lehnert W.: "Crystal: Inducing a Conceptual dictionary". Proc. of the Fourteenth International Join Conference on Artificial Intelligence (IJCAI-95), 1314-1319, 1995.

[Ullman 1988] Ullman J.: "Principles of Database and Knowledge-Base Systems", Rockville (Md), 1988.

[Vargas-Vera and Lytras 2010] Vargas-Vera M. and Lytras M.: "AQUA: Hybrid Architecture for Question Answering Services". Special Issue on Semantic Web Support for Software Development; IET Software Journal. Colomo-Palacios R. and Gomez-Bebis J. M. (eds); 4(6), 418-433; December 2010.

[Vargas-Vera and Motta 2004] Vargas-Vera M. and Motta E.: "AQUA - Ontology-based Question Answering System". Third International Mexican Conference on Artificial Intelligence (MICAI-2004), Lecture Notes in Computer Science 2972 Springer Verlag, (eds R. Monroy et al), April 26-30, 2004.

[Vargas-Vera etal, 2003] Vargas-Vera M., Motta E. and Domingue J.: "AQUA: An Ontology-Driven Question Answering System". AAAI Spring Symposium, New Directions in Question Answering, Stanford University, March 24-26, 2003.

[Vargas-Vera etal, 2002] Vargas-Vera M, Motta E, Domingue J, Lanzoni M, Stutt A, Ciravegna F.: MnM: Ontology Driven Semi-Automatic and Automatic Support for Semantic Markup. The 13th International Conference on Knowledge Engineering and Management (EKAW 2002), Lecture Notes in Computer Science 2473, ed Gomez-Perez, A., Springer Verlag, 379-391, 2002.

[Vargas-Vera etal, 2001] Vargas-Vera M. and Domingue J. and Y. Kalfoglou Y. and Motta E. and Buckingham-Shum S.: "Template-driven information extraction for populating ontologies". Proc of the IJCAI'01 Workshop on Ontology Learning, Seattle, WA, USA, 2001.

[Zelle etal, 1994] Zelle J. M and Mooney JR. and Konvisser J.B.: "Combining Top-down and Bottom-up Methods in Inductive Logic Programming". Proc of the 11th International Conference on Machine Learning (ML-94), 343–351, 1994.