Ontosophie: A Semi-Automatic System for Ontology Population from Text

Tech Report kmi-04-19

David Celjuska and Dr. Maria Vargas-Vera

Ontosophie: A Semi-Automatic System for Ontology Population from Text

David Celjuska[*] and **Dr. Maria Vargas-Vera**
KMi - Knowledge Media Institute,
The Open University,
Walton Hall, Milton Keynes, MK7 6AA,
United Kingdom
E-mail: *davidmdsk@centrum.sk* and *m.vargas-vera@open.ac.uk*

Abstract

This paper describes a system for semi-automatic population of ontologies with instances from unstructured text. It is based on supervised learning, learns extraction rules from annotated text and then applies those rules on new articles to populate the ontology. Hence, the system classifies stories and populates a hand-crafted ontology with new instances. It is based on three components: Marmot – a natural language processor; Crystal – a dictionary induction tool; and Badger – an information extraction tool. A part of the entire cycle is a user who accepts, rejects or modifies extracted and suggested instances to be populated. A description of experiments performed with a text corpus consisting of 91 articles is given. The results complete the paper and support the hypothesis that assigning a rule confidence value to each extraction rule improves the performance.

1 Introduction

Ontologies are popular in a number of fields such as knowledge engineering and representation, information retrieval and extraction, knowledge management, agent systems, and more (Guarino, 1998). The problem of their construction and engineering still remains only partially solved and their development today is more a craft than a science. Automated ontology construction tools provide little support for knowledge acquisition. Therefore, the ontology construction process is time consuming and their wide usage is thus limited.

A number of proposals have been published to facilitate ontology engineering (Vargas-Vera et al., 2001; Craven and Kumilien, 1999; Faure and N'edellec, 1998).

Information Extraction – IE is as a technology that can help an ontology expert during the ontology population and maintenance process. It can be viewed as the task of pulling predefined entities –

[*]Department of Artificial Intelligence and Cybernetics, Technical University Kosice, Slovakia

such as name of visitor, location, date, etc. from texts to fill predefined slots in classes. Therefore, our system is based on IE, Machine Learning – ML and Natural Language Processing – NLP. The system, as with most IE systems, uses partial parsing to recognize syntactic constructions. This has the advantage of high speed and robustness, which is necessary when applying IE to a large set of documents.

The main contributions of the paper can be summarized as 1) identification of key entities in text articles that could participate in ontology population with instances, 2) identification of the most probable classes for the population based on newly introduced confidence values, and 3) semi-automatic population of an ontology with those instances.

The paper is organized as follows: Section 2 defines the goal we set to achieve. Section 3 describes phases the systems goes through, including an algorithm for computing and assigning confidence values to the extracted rules. Section 4 presents the validation of the system carried out using a set of news archives. Finally, Section 5 provides conclusions and directions for future work.

2 Problem definition

The design of Ontosophie was motivated by Info-Extractor (Vargas-Vera et al., 2001) and MnM (Vargas-Vera et al., 2002). It is capable of semi-automatic population of a given ontology O with instances. The instances are extracted automatically from text. The systems first task is to identify important entities – slot values $v_1, v_2, \ldots, v_{Ni}$ such as (visitor, date, location, and so on – depending on a class C_i) in a document and thus to construct an instance $I_i = (v_1, \ldots, v_{Ni})$ for class $C_i \in \{C_1, C_2, \ldots, C_M\}$ in the given ontology O. In the next step, it is necessary to determine whether the constructed instance I_i is correct and whether it should be fed into the class C_i or not. This determination is based on the extracted entities and their confidences which will be discussed later.

Experiments were performed by using KMi's event ontology. This consists of events or activities that are defined formally in the ontology O as classes $C_1, \ldots, C_M$. Each class C_i is defined with slots $s_1, s_2, \ldots, s_{Ni}$, which might be instantiated into $v_1, v_2, \ldots, v_{Ni}$.

The following part shows one class definition from the event topology:

Class Event: Conferring-an-Award
Description: Class of an event describing an event of awarding someone
Slots:
has-duration (when or how long the event took)
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 (for what the award is)
object-acted-on (award, name of the award or amount of money)

The class *Conferring-an-Award* describes any event which talks about awarding someone an award for some reason. The slots such as *has-duration*, *has-location*, etc. are attributes of the class and give detailed information about one specific event/instance which the system is intending to construct. The task of the system is therefore to identify those attributes $v_1, \ldots, v_{Ni}$ in a document and then to construct an instance and feed it into an appropriate class C_i within the ontology O.

3 Phases

The system goes through the following phases during its life cycle.

3.1 Annotation

In order to let the system learn extraction rules it has to be provided with a set of examples since it is based on supervised learning. In our case this is a set of documents (plain text or HTML) annotated with XML tags and assigned to one of the predefined classes within the ontology O.

Each slot within the ontology is assigned a unique XML tag – the mark-up step is ontology driven. Once the user identifies a desired class for a displayed document from the ontology he is only offered relevant tags for the annotation. An annotated article might then look as follows:

> **<EV>KMi</EV>** has been awarded **<EZ>L1.2M</EZ>** by the **<EX>UK's Engineering and Physical Sciences Research Council</EX>** to carry out **<EY>research in the application of knowledge technologies</EY>**.

The tag EV refers to *recipient-agent* – an agent who was given an award, EX to *has-awarding-body* – a name of the organization which gave the award, EZ to *object-acted-on* – the award itself and EY to *has-award-rationale* – the reason of the award or for what the award was.

Once a set of documents is annotated with XML tags the learning phase may begin.

3.2 Learning

The learning phase consists of two steps that will be describe in the following.

3.2.1 Natural language processing – NLP

Ontosophie uses shallow parsing to recognize syntactic constructs without generating a complete parse tree for each sentence. The shallow parsing has the advantages of higher speed and robustness. High speed is necessary to apply the IE to a large volume of documents. The robustness achieved by using a shallow parsing is essential to deal with unstructured texts. In particular, Ontosophie uses the Marmot[1], NLP system.

[1]Marmot was developed at University of Massachusetts, MA, USA

Marmot accepts ASCII files and produces an intermediate level of text analysis that is useful for IE applications. Sentences are separated and segmented into noun phrases, verb phrases and other high-level constituents.

After each document has been annotated and pre-processed with the NLP tool, the set of documents enters the Learning phase.

3.2.2 Generating extraction rules

This phase makes use of Crystal[2] (Soderland at al., 1995), a conceptual dictionary induction system. Crystal derives a dictionary of concept nodes – extraction rules, from a training corpus. It is based on the specific-to-general algorithm. For illustration, an extraction rule might be understood as following: *conferring-an-award: <VB-P "been awarded"> <OBJ1 ANY> <PP "by" has-awarding-body> Coverage: 5 Error: 1* The rule's purpose is to extract *conferring-an-award*, which refers to the name of a class from the ontology O. This extraction rule is aimed at extracting *has-awarding-body* – name of a donor. The rule fires only if all the constraints are satisfied. This means, that the entity *conferring-an-award* is extracted from any sentence or its part only in the case where it consists of *"has been awarded"* as passive verb (VB-P), an object (OBJ1) that might be anything and it contains a prepositional phrase (PP), which starts with preposition *"by"*. When this rule fires then the prepositional phase (PP) is extracted as *has-awarding-body*. For example, from the sentence: *"KMi has been awarded L1.2M by the UK's EPSRC to carry out research in . . . "* it will extract *"by the UK's EPSRC. . . "*.

In addition, Crystal provides two values – *Coverage* and *Error*. In this particular example they state that the rule covers five instances (one incorrectly) in the corpus in which the rule was generated from. This helps provide a feel of the rule's precision or confidence (section 3.2.3).

3.2.3 Assigning rule confidence values to extracted rules

We believe that the best approach for semi-automatic tools is to give the user all the options available while pre-selecting/recommending only those that are strongly believed to be correct. Therefore we intend to keep high recall while we also try to achieve a high precision for the automatic pre-selection.

Experimentation showed, that some extraction rules that were learnt by Crystal are very weak and therefore firing too often, while other rules might be overly specific. In addition, previous experiments (Riloff, 1996) showed that precision improves if those rules are manually removed. However, our goal is to take an automatic control over this and to eliminate rules with low rule confidence. In order to achieve this task, Ontosophie attaches a rule confidence value to each rule[3]. The rule confidence expresses how sure the system is about the rule itself.

Ontosophie is equipped with two ways of computing the rule confidence value. The first method uses the *Coverage* and *Error* values provided by Crystal (section 3.2.2). The rule confidence for the rule r_i is computed as $C_{r_i} = \frac{c_{r_i}}{n_{r_i}} = \frac{Coverage_{r_i} - Error_{r_i}}{Coverage_{r_i}}$. Where c_{r_i} is the number of times the rule r_i has fired

[2]Crystal was developed at University of Massachusetts, MA, USA

[3]None of the mentioned systems including MnM and Info-Extractor has this feature.

correctly and n_{r_i} is the number of times the rule is fired in total. However, this does not distinguish between, for example $C_2 = (2-0)/2$ and $C_{10} = (10-0)/10$, because $C_2 = C_{10} = 1.0$. One might argue that C_{10} is more accurate and has higher support, because in this case the rule fired ten times out of ten correctly, while the other fired only two times correctly out of two. This is why Ontosophie uses Laplace Expected Error Estimate (Clark and Boswell, 1991) defined as $1 - LaplaceAccuracy$, where $LaplaceAccuracy = \frac{n_c+1}{n_{tot}+k}$ and[4] n_c is the number of examples in the predicted class covered by the rule, n_{tot} is the total number of examples covered by the rule and k is the number of classes in the domain. Implementing the Laplace accuracy the valuation of confidence is then $C_{r_i} = \frac{c_{r_i}+1}{n_{r_i}+2}$ and $k = 2$ because it deals with two classes for each rule. One, the rule fires and two, the rule does not fire. One might note, that if $C_{r_i} = 0.5$ the rule fires correctly as often as it does incorrectly. At this state, nothing serious can be said about the rule and thus all rules with $C_{r_i} \leq 0.5$ should be eliminated.

The second method computes confidence for each rule by the k-Fold Cross validation methodology (Mitchell, 1997) on the training set. At each run a new set of extraction rules is generated by Crystal. Our algorithm (Celjuska, 2004) computes for each rule r_i how many times it fired correctly c_{r_i}, how many times it fired in total n_{r_i}, performs merging of identical rules and assigns x_{r_i} to each rule that tells how many times the rule was merged. If two rules r_i and r_j generated from two different runs are identical, regarding their constraints, they are merged to form one new rule r_{new} which is identical to the r_i and r_j while the number of times the rule r_{new} is fired correctly $c_{r_{new}} = c_{r_i} + c_{r_j}$ and number of times it is fired in total $n_{r_{new}} = n_{r_i} + n_{r_j}$.

The evaluation of a rule is always performed on the validation set. At each run, after all the rules have been generated by Crystal, Ontosophie enters evaluation state which is based on the extraction. All rules that were responsible for correctly extracted entities are then awarded $c_{r_i} \leftarrow c_{r_i} + 1$. Certainly, n_{r_i} is incremented $n_{r_i} \leftarrow n_{r_i} + 1$ for all rules that were active during the extraction. The confidence C_{r_i} is then computed as: $C_{r_i} = \frac{c_{r_i}}{n_{r_i}}$. The tough phase is to recognize whether an extracted entity is correct or not – see (Celjuska, 2004) for more details.

3.3 Extraction and ontology population

The task of this phase is to extract appropriate entities from an article[5] and feed a newly constructed instances into a given ontology O. The document is pre-processed with Marmot (similarly as described in section 3.2.1) prior to the extraction itself.

The extraction is run class by class. Firstly, a set of extraction rules for only one specific class from the ontology is taken and only those rules are used for the extraction. The step is then repeated for all the classes within the ontology and thus for each class the system gets a couple of entities.

It might happen that extraction component[6] extracts more than one value for a given slot name. This is the collision that has to be solved. Therefore, extraction phase might lead to the following:

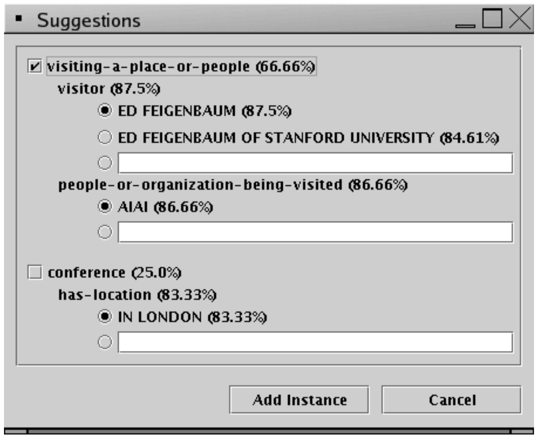
[4]The presented Laplace Error Estimate is borrowed from Classification, that is why the particular variables are defined as they are.

[5]not yet processed nor annotated

[6]in our case Badger from University of Massachusetts, USA

Ed Feigenbaum Visits AIAI
Wednesday, 18th July 2001

Ed Feigenbaum of Stanford University visited AIAI on 2nd July 2001 to hear about the knowledge-based systems and applied AI work of the Institute. He heard about the plans to form CISA on 1st August 2001... He is currently working with the European Office of Aerospace Research and. Development in London, part of the US Air Force Office of Scientific.


Suggestions

[✓] visiting-a-place-or-people (66.66%)
visitor (87.5%)
[●] ED FEIGENBAUM (87.5%)
○ ED FEIGENBAUM OF STANFORD UNIVERSITY (84.61%)
○
people-or-organization-being-visited (86.66%)
[●] AIAI (86.66%)
○

[ ] conference (25.0%)
has-location (83.33%)
[●] IN LONDON (83.33%)
○

Figure 1: A part of original the text and a dialog with extracted entities from the text

1. The same piece of information, an entity was extracted with more than one rule – value collision
2. More than one value was extracted for a given slot – slot collision
3. Entities from different classes where extracted – class collision.
4. Nothing was extracted – the document remains unclassified

Figure 1 shows 2^{nd} and 3^{rd} case. On the left side is a segment of the original text and on the right side is a window with suggestions for ontology population.

The following section describes in detail solutions to resolve those issues.

3.3.1 Solving collisions

In the system, not only are rules assigned confidence values, but also extracted entities, potential slots and instances. If one piece – an entity is extracted by only one rule, then this piece is assigned a value confidence $C_{value} = C$, where rule confidence C is computed as discussed earlier[7]. However, if more than one rule extracts the same entity, then C_{value} is computed as the maximum overall confidences of rules that fired it $C_{value} = \max_{\forall r_i : r_i \textit{ is in collision}} C_{r_i}$. The same applies for the slot confidence C_{slot}. If one value was extracted for a given slot (i.e. *visitor = "Ed Feigenbaum"*) then $C_{slot} = C_{value}$. However, if more then one value was extracted for a slot (*visitor = "Ed Feigenbaum" and visitor = "Ed Feigenbaum of Stanford University"*), then only the value with its highest confidence is considered and also pre-selected (Figure 1). Thus, $C_{slot} = \max_{\forall i : C_{value_i} \textit{ is in collision}} C_{value_i}$.

It might happen, that the system extracts some entities from one class and some entities from another class (*visiting-a-place-or-people* and *conference* – Figure 1). It is important to determine which classes the new instances should be fed into. For this purpose, the instance confidence value $C_{instance}$ is assigned to each potential instance of some class. Although there is more than one

[7]Please note that the index r_i has been omitted

way of computing instance confidence $C_{instance}$ we introduce the following equation: $C_{instance} = \frac{\sum_{\forall\ extracted\ slots\ for\ the\ given\ instance} C_{slot_i}}{m_{instance}}$. Where $m_{instance}$ is the maximum number of different slots that any potential instance of this particular class could possibly have instantiated.

3.3.2 Pruning

There are two different threshold values implemented in the system. One is for pruning slots and one for potential instances. When in the extraction phase some slot is assigned a slot confidence value $C_{slot} < Threshold_{slot}$ then this slot is not pre-selected and also do not play any role in the phase of computing instance confidence value (section 3.3.1). Otherwise it is pre-selected.

The second threshold value $Threshold_{instance}$ is used in case of classification. Classes that have confidence $C_{instance} < Threshold_{instance}$ are not pre-selected. Please note, that instances of more than one class can be extracted from one article.

Threshold values are very helpful to speed up the process of rejecting/accepting potential instances. In case, a user is offered only with trusted and confident pre-selections, the high volume of interaction is avoided and the goal of Ontosophie is achieved.

After the extraction process is finished, a users interaction is required to take the final decision about the extracted instances. The user has the ability to re-select pre-selected options or completely reject to populate the ontology with any instance.

4 Validation of the system and experiments

4.1 Description of the set

The ontology that we used contains 41 classes, but only three of them were used for the experimentation. This was due to low variety of articles talking about different events. In particular, short text articles, similar to the example given in section 3.1, were gathered from five different archives. They were manually annotated, as described in section 3.1, and every document was classified into exactly one of the mentioned classes.

4.2 Validation of the system

For the validation of the system we aim to:

- determine how using the rule confidence effects precision and recall.
- compare two mentioned methods for evaluation of rule confidence (section 3.2.3).
- determine how elimination of rules effects the precision and recall.

When the rule confidence is being computed by k-Fold-Cross methodology, as it was stated in section 3.2.3, at each run a new set of rules is generated by Crystal. Then the rules that are identical are merged and x_i which tells how many time a rule r_i was merged, is computed. It is believed, that a rule which was generated from more than one run is more likely to do good in the entire set and not just within the part it was generated from. It is more probable that it will not be too specific and thus removing all rules $r_i : \forall i; x_i < Merge$ might result in better quality rule dictionary – set. The parameter $Merge$ controls which rules will be kept in the dictionary and which will be removed.

Four different experiments were run. For the validation of each of the experiment the 5-fold-cross[8] validation methodology was used. More over, each experiment was repeated five times to get better statistical results and in the case of experiments where k-Fold-Cross was used to compute rule confidence the dataset was randomly split each time into k folds.

The following experiments were performed:

1. *"No confidence"* – without using any rule confidence value. Thus no pruning was used. The rules were treated as equal without any preference and all generated rules were used.
2. *"Simple"* – the first, simple method for computation of rule confidence was used with Laplace error estimate: $C = \frac{c+1}{n+2} = \frac{Coverage-Error+1}{Coverage+2}$
3. *"k-Fold"* – the second, 5-Fold-Cross validation method was used for computing the rule confidence. No rule was eliminated thus $Merge = 0$ and all the rules generated from each run/fold were used.
4. *"Elimination"* – similarly to *"k-Fold"*, 5-fold-cross was used to compute rule confidence. However, this time the rules that were not merged at least 2 times were removed: $Merge = 2$.

Threshold values in case of experiments 2 – 4 were set as follows: $Threshold_{instance} = 0.3$ and $Threshold_{slot} = 0.7$.

In Figure 2 it shows precision (P) for every experiment and every class separately. The minimum, maximum and the average values observed throughout the five trials are presented. It can be observed, that there is significant change in precision between cases where the rule confidence is taken into consideration and when it is not. The average recall however goes rapidly down from 16.1% for the *"No-confidence"*, 10.2% for the *"Simple"*, 6.5% for the *"k-Fold"* and 4.1% for the *"Elimination"*. Variability also increases. Only looking at total precision it might seem that *"Elimination"* is the best choice with its 89.22% average. However, at this stage the recall was extremely low – for the class *conference* no entity was extracted.

"k-Fold-Cross" obtains a lower average of total precision 85.62% and recall 6.5%, a little higher than in case of *"Elimination"*. At each class it went well, besides *conference*. It might be considered as one of the best options from all the experiments.

It can be concluded that using a rule confidence is a big plus. It also seems that k-Fold-Cross methodology is a better choice to the simple method if in search for high precision. Elimination of rules in case of *"Elimination"* needs to be taken with care. In the experiment the rules that were not generated from at least three different runs out of five were strictly removed. It was observed from examples that from 79 rules it resulted into 24 after the elimination. This is quite a drastic pruning.

[8]5-fold-cross validation is not a standard, but it was used to save processing time

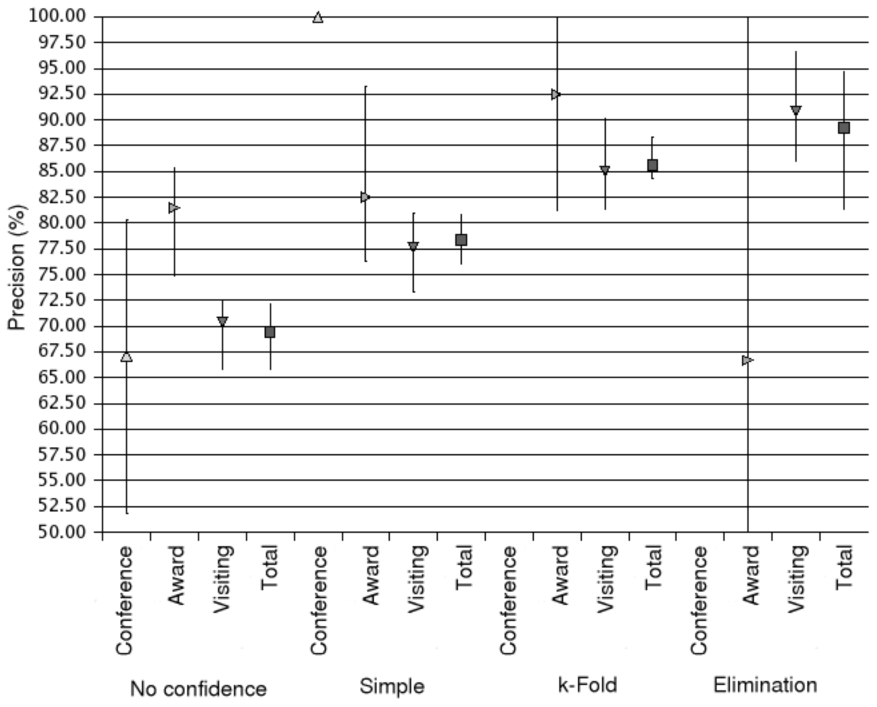


Figure 2: Precision for each method - per class and in total

It is also important to note that the dataset was not very large. Thus the absolute number of positive instances is not very high (from 3 to 132 depending on slot). Learning on such a small set is always leading to more specific rules causing variability to go high.

5 Conclusion and future work

This paper described the Ontosophie, system which is based on three components: a NLP component called Marmot, a dictionary induction tool named Crystal and an IE component called Badger.

In the area of semi-automatic population it is important to have a system that gives a user the control over the process while automatically offering only the most trusted suggestions for the ontology population. Ontosophie is equipped with this feature in that, at the extraction phase, it always performs the full extraction while pre-selecting only those suggestions that are considered to be correct. This is done by applying a pruning technique on a set of extraction rules; the technique is based on threshold parameters set by a user. In order to evaluate an extracted entity, two different designed methods for computing rule confidence were introduced. The experiments conducted using those methods showed that using the the precision can be increased by around 15% depending on different models and parameters. In addition, it was observed that using the k-Fold-Cross methodology for its computation seems to be a better choice to the simple method of taking *Coverage* and *Error* values computed by the learning component Crystal.

It was observed that it is still relatively hard for the user to rapidly determine whether the recommended pre-selection is correct or not. Therefore, for the next generation of the system we suggest performing text highlighting of the extracted entities in the original document. The user could then, by clicking on each of the suggestions, see the extracted highlighted entity within a context and more easily determine its correctness. In addition, the next generation should provide a superior post-processing stage that could also include the entity type validation. This could be done by comparing the type of the slot and the type of the extracted information. It would be valuable also to not just use extracted text segments to fill predefined slot values for a given instance, but also to be able to use other instances or classes as slot values.

References

CELJUSKA, D. 2004. *Semi-automatic Construction of Ontologies from Text.* Master's Thesis, Department of Artificial Intelligence and Cybernetics, Technical University Kosice.

CLARK, P. – BOSWELL, R. 1991. *Rule induction with CN2: Some recent improvements.* Proceeding Fifth European Working Session on Learning, pages 151 – 163, Springer, Berlin.

CRAVEN, M. – KUMILIEN, J. 1999. *Constructing Biological Knowledge Bases by Extracting Information from Text Sources.* Proceedings of The 7th International Conference on Intelligent Systems for Molecular Biology (ISMB-99).

FAURE, D. – N'EDELLEC C. 1998. *ASIUM: Learning sub-categorization frames and restrictions of selection.* 10th Conference on Machine Learning (ECML 98) – Workshop on Text Mining, Chemnitz, Germany.

GUARINO, N. 1998. *Formal Ontology and Information Systems.* Proceedings of the 1st International Conference on Formal Ontologies in Information Systems, FOIS'98, Trento, Italy, pages 3 – 15. IOS Press.

MITCHELL, T. 1997. *Machine Learning.* McGraw-Hill, ISBN 0070428077.

RILOFF, E. 1996b. *Automatically Generating Extracting Patterns from Untagged Text.* Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96), pp 1044-1049.

SODERLAND, S. – FISHER, D. – ASELTINE, J. – LEHNERT, W. 1995. *CRYSTAL: Inducing a Conceptual Dictionary.* Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, pages 1314 – 1319.

VARGAS-VERA, M. – DOMINGUE, J. – KALFOGLOU, Y. – MOTTA, E. – BUCKINGHAM SHUM, S. 2001. *Template-Driven Information Extraction for Populating Ontologies.* In proceedings of the Workshop Ontology Learning IJCAI-2001, Seattle, USA.

VARGAS-VERA, M. – MOTTA, E. – DOMINGUE, J. – LANZONI, M. – STUTT, A. – CIRAVEGNA, F. 2002. *MnM: Ontology Driven Semi-Automatic and Automatic Support for Semantic Mark-up.* The 13th International Conference on Knowledge Engineering and Management (EKAW 2002), Lecture Notes in Computer Science 2473, ed Gomez-Perez, A., Springer Verlag, 2002, 379-391, ISBN 3-540-44268-5.