Induction and Revision of Terminologies

F. Esposito, N. Fanizzi, L. Iannone, I. Palmisano and G. Semeraro[1]

1 INTRODUCTION

Description Logics (DLs) and derived markup languages are a standard for representing ontological knowledge bases which can be a powerful tool for supporting other services, such as reasoning and retrieval. Such languages are generally endowed with well-founded semantics and reasoning services investigated in the DLs field [1]. In this context, we examine the problem of the induction and refinement of definitions for the concepts and their properties forming the *T-box*, on the ground of basic information (assertions) made on individuals that may be available in a knowledge base, i.e. the *A-box*, representing the world state. The induction and refinement of structural knowledge is not new in *Machine Learning*. Recently, in *Inductive Logic Programming*, attempts have been made to extend the classic relational learning techniques toward hybrid representations [5]. In order to cope with the problem complexity, former methods are based on a heuristic search and generally implement bottom-up algorithms, such as the *least common subsumer* (*lcs*) [3], that tend to induce overly specific definitions which may suffer for poor predictiveness (*overfitting*). Hence, in some approaches maximal generalizations are preferred [4]. Moreover, like the *least general generalizations* for clausal representations, lcs's sizes tend to increase exponentially. Other approaches have shown that also a top-down search is feasible [2], yet the refinement is less operational: it is intended to show the properties of the related search space rather than specifying how to exploit the heuristics based on the assertions in the A-box. Intending to give operational instruments for performing the inference and refinement of conceptual descriptions, we introduce an algorithm based on *multilevel counterfactuals* [7] for operating with an $\mathcal{ALC}$ representation.

2 SYNTAX AND SEMANTICS

Syntax and semantics for the $\mathcal{ALC}$ representation [1] adopted can be sketched as follows. In a DL language, primitive *concepts* $N_C = \{C, D, \ldots\}$ are interpreted as subsets of a certain domain of objects and primitive *roles* $N_R = \{R, S, \ldots\}$ are interpreted as binary relations on such a domain. A *knowledge base* $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ contains two components: a T-box $\mathcal{T}$ and an A-box $\mathcal{A}$. $\mathcal{T}$ is a set of concept definitions $C \equiv D$, meaning[2] $C^{\mathcal{I}} = D^{\mathcal{I}}$, where C is the concept name and D is a description given in terms of the language constructors presented in the following table. $\mathcal{A}$ contains extensional assertions on concepts and roles, e.g. $C(a)$ and $R(a, b)$, meaning, respectively, that $a^{\mathcal{I}} \in C^{\mathcal{I}}$ and $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$.

[1] Dipartimento di Informatica, Università degli Studi di Bari, Campus, Via Orabona 4, 70125 Bari, Italy email: *lastname*@di.uniba.it

[2] For any interpretation $I = (\Delta^I, \cdot^I)$ defined as in [1]

NAME	SYNTAX	SEMANTICS
top concept	$\top$	$\Delta^{\mathcal{I}}$
bottom concept	$\bot$	$\emptyset$
concept negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
concept conjunction	$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
concept disjunction	$C_1 \sqcup C_2$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$
existential restriction	$\exists R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y\,(x, y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
universal restriction	$\forall R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \forall y\,(x, y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$

The notion of *subsumption* between concepts can be given in terms of the interpretations:

Definition 2.1 *Given two concept descriptions C and D in $\mathcal{T}$, C* subsumes *D, denoted by $C \sqsupseteq D$, iff for every interpretation $\mathcal{I}$ of $\mathcal{T}$ it holds that $C^{\mathcal{I}} \supseteq D^{\mathcal{I}}$.*

Many semantically equivalent (yet syntactically different) descriptions can be given for the same concept. However they can be reduced to a canonical form by means of rewriting rules that preserve their equivalence [1]. Preliminarily, some notation is needed to name the different parts of an $\mathcal{ALC}$ description: $\mathsf{prim}(C)$ is the set of all the concepts at the top-level conjunction of C; $\mathsf{val}_R(C)$ represents the conjunction of the concepts in universal restrictions on role R, whereas $\mathsf{ex}_R(C)$ represents the set of all concepts in the existential restrictions on role R (if R does not appear in C both $\mathsf{val}_R(C)$ and $\mathsf{ex}_R(C) \equiv \top$).

Definition 2.2 *A concept description D is in $\mathcal{ALC}$* normal form *iff $D \equiv \bot$ or $D \equiv \top$ or if $D = D_1 \sqcup \cdots \sqcup D_n$ with*

$$D_i = \bigsqcap_{A \in \mathsf{prim}(D_i)} A \;\sqcap\; \bigsqcap_{R \in N_R} \forall R.\mathsf{val}_R(D_i) \;\sqcap\; \bigsqcap_{\substack{R \in N_R \\ E \in \mathsf{ex}_R(D_i)}} \exists R.E$$

where, for all $i = 1, \ldots, n$, $D_i \not\equiv \bot$ and, for any R, every concept description in $\mathsf{ex}_R(D_i)$ and $\mathsf{val}_R(D_i)$ is in normal form.

3 INDUCTION OF CONCEPT DESCRIPTIONS

The methodology for the induction and refinement of T-boxes proposed in this work is based on the notion of counterfactuals built on the ground of residual learning problems [7]. Each assertion is not processed as such: it is supposed that preliminarily a representative at the concept language level is derived in the form of *most specific concept* (*msc*). The msc required by the algorithm is a DL concept description that entails the given assertion. Moreover it is bound to be among the most specific ones (or its approximations [3]). Hence, in the algorithm the positive and negative examples will be very specific conjunctive descriptions obtained by means of the *realization* [1] of the assertions concerning the target concept. For many DLs the msc cannot be easily computed or simply it is not unique. For our purposes it suffices to have good (upper) approximations. The algorithm, reported in the next figure, relies on two interleaving routines

that perform, respectively, generalization and specialization, calling each other up to converging toward a correct definition.

generalization(*Positives*, *Negatives*, *Generalization*)
input *Positives*, *Negatives*: positive and negative instances at concept level;
output *Generalization*: generalized concept definition
ResPositives $\leftarrow$ *Positives*
Generalization $\leftarrow \bot$
while *ResPositives* $\neq \emptyset$ **do**
  $ParGen \leftarrow$ select_seed(*ResPositives*)
  CoveredPos $\leftarrow \{Pos \in$ *ResPositives* $\mid ParGen \sqsupseteq Pos\}$
  CoveredNeg $\leftarrow \{Neg \in$ *Negatives* $\mid ParGen \sqsupseteq Neg\}$
  while *CoveredPos* $\neq$ *ResPositives* **and** *CoveredNeg* $= \emptyset$ **do**
    $ParGen \leftarrow$ select($\delta(ParGen)$, *ResPositives*)
    CoveredPos $\leftarrow \{Pos \in$ *ResPositives* $\mid ParGen \sqsupseteq Pos\}$
    CoveredNeg $\leftarrow \{Neg \in$ *Negatives* $\mid ParGen \sqsupseteq Neg\}$
  if *CoveredNeg* $\neq \emptyset$ **then**
    $K \leftarrow$ **counterfactuals**($ParGen$, *CoveredPos*, *CoveredNeg*)
    $ParGen \leftarrow ParGen \sqcap \neg K$
  Generalization $\leftarrow$ *Generalization* $\sqcup ParGen$
  ResPositives $\leftarrow$ *ResPositives* $\setminus$ *CoveredPos*
return *Generalization*

counterfactuals(*ParGen*, *CoveredPos*, *CoveredNeg*, K)
input *ParGen*: inconsistent concept definition
  CoveredPos, *CoveredNeg*: covered positive and negative descriptions
output K: counterfactual
NewPositives $\leftarrow \emptyset$
NewNegatives $\leftarrow \emptyset$
for each $N_i \in$ *CoveredNeg* **do**
  NewP$_i \leftarrow$ residual(N_i, *ParGen*)
  NewPositives $\leftarrow$ *NewPositives* $\cup \{$*NewP*$_i\}$
for each $P_j \in$ *CoveredPos* **do**
  NewN$_j \leftarrow$ residual(P_j, *ParGen*)
  NewNegatives $\leftarrow$ *NewNegatives* $\cup \{$*NewN*$_j\}$
$K \leftarrow$ **generalization**(*NewPositives*, *NewNegatives*)
return K

The generalization algorithm is a greedy covering one: it constructs a disjunctive definition of the positive examples. At each outer iteration, a very specialized definition (the msc of an example) is selected as a starting seed for a new partial generalization; then, iteratively, the hypothesis is generalized by means of the upward operator δ (with a heuristic that privileges the refinements that cover the most of positives) until all positive instances are covered or some negative instances are explained. In the latter case, the current concept definition *ParGen* has to be specialized by some counterfactuals. The co-routine, which receives the covered examples, finds a sub-description K that can rule out the covered negative examples (once negated). In the routine for building counterfactuals, given a previously computed hypothesis *ParGen*, which is supposed to be complete for the positive assertions, yet inconsistent with respect to some negative assertions, the aim is finding those counterfactuals to be conjuncted to the initial hypothesis for restoring a correct definition, that can rule out the negative instances. The algorithm is based on the construction of residual learning problems based on the sub-descriptions that caused the subsumption of the negative examples, represented by their msc's. In this case, for each model a residual is derived by considering that part of the incorrect definition *ParGen* that did not play a role in the subsumption. The residual will be successively employed as a positive instance of that part of description that should be ruled out of the definition (through negation). Analogously the msc's derived from positive assertions will play the opposite role of negative instances for the residual learning problem under construction. Finally, this problem is solved by calling the co-routine which generalizes these example descriptions and the conjoining the negation of the returned result. For the sake of simplicity, the generalization routine described in the method is non-deterministic: among the possible generalizations computed by δ, those that maximize the number of covered positives should be preferred. The residual operator is essentially a difference function [6]. In the case $\mathcal{ALC}$, the difference can be simply defined as $C - D = C \sqcup \neg D$. The correctness of the method can be proved (proof omitted for the sake of brevity). The algorithm may fail when a call to counterfactuals with null positive and negative descriptions indicates that it is impossible to discriminate between two identical descriptions.The counterfactuals algorithm is linear except for the dependency on the generalization algorithm. Then it suffices here to discuss the complexity of such an algorithm. The generalization proposed here is a generic divide and conquer algorithm which performs a greedy search using the refinement operator δ. Introducing a better refinement operator based on examples, the heuristic information conveyed by the examples can be better exploited so to have a faster convergency. The cycles are linear on the number of instances. The only source of complexity are the subsumption tests which are known to be PSpace-complete in $\mathcal{ALC}$ [1].For many expressive DLs, msc's are difficult to compute (provided they exist). In $\mathcal{ALC}$, although lcs is simply the disjunction of the inputs, there is no algorithm for computing msc's. Our algorithm processes approximations of the msc's, for it is endowed with the specialization mechanism of the counterfactuals, whereas the lcs can only generalize starting from very specialized definitions. The method illustrated in this work has been implemented in a prototype of a refinement system (YINYANG, *Yetanother INduction Yields A New Generalization*) to induce or refine $\mathcal{ALC}$ knowledge bases which can be maintained by means of OWL markup. This will allow to design a learning and refinement service for the Semantic Web. The proposed method could be extended along three directions. First, an investigation on the properties of the refinement operators on DL languages is required. In order to increase the efficiency of learning, redundancy during the search for solutions is to be avoided. This can be done by defining minimal refinement operators [2]. Secondly, the method can be extended to other DLs by changing the residual operator and devising a proper representation for counterfactuals. Another promising direction seems to be investigating hybrid representations, where clausal logic descriptions are mixed with description logics accounting for available ontological knowledge.

4 ACKNOWLEDGEMENT

This research was partially funded by the European Commission under the IST Integrated Project VIKEF - Virtual Information and Knowledge Environment Framework (Contract no. 507173); more information at `http://www.vikef.net`.

REFERENCES

[1] *The Description Logic Handbook*, eds., F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, Cambridge University Press, 2003.

[2] L. Badea and S.-H. Nienhuys-Cheng, 'A refinement operator for description logics', in *Proceedings of the 10th International Conference on Inductive Logic Programming*, eds., J. Cussens and A. Frisch, volume 1866 of *LNAI*, pp. 40–59. Springer, (2000).

[3] W.W. Cohen and H. Hirsh, 'Learning the CLASSIC description logic', in *Proceedings of the 4th International Conference on the Principles of Knowledge Representation and Reasoning*, eds., P. Torasso, J. Doyle, and E. Sandewall, pp. 121–133. Morgan Kaufmann, (1994).

[4] J.-U. Kietz and K. Morik, 'A polynomial approach to the constructive induction of structural knowledge', *Machine Learning*, **14**(2), 193–218, (1994).

[5] C. Rouveirol and V. Ventos, 'Towards learning in CARIN-$\mathcal{ALN}$', in *Proceedings of the 10th International Conference on Inductive Logic Programming*, eds., J. Cussens and A. Frisch, volume 1866 of *LNAI*, pp. 191–208. Springer, (2000).

[6] G. Teege, 'A subtraction operation for description logics', in *Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning*, eds., P. Torasso, J. Doyle, and E. Sandewall, pp. 540–550. Morgan Kaufmann, (1994).

[7] S.A. Vere, 'Multilevel counterfactuals for generalizations of relational concepts and productions', *Artificial Intelligence*, **14**, 139–164, (1980).