A General Framework for Theory Learning. Perpectives for Natural Language Processing

Jordi Alvarez[1]

Abstract. This position paper introduces a general framework for theory learning based on probabilistic Description Logics (DL). The formal base of the framework is briefly described. The paper also argues that the general theory learning approach is suitable to be used for NLP tasks; and it can be specially useful to investigate interrelations among different types of natural language information in order to produce better NLP systems.

1 Introduction

In this paper, I advocate for a general theory learning framework. The representation formalism used in our framework is an extension of Description Logics (DL) that allows to express probabilistic knowledge. A general learning procedure based on the previous framework has been implemented as part of a system called YAYA [2]. This learning procedure is a relational learning procedure that performs computations very similar to those performed by Inductive Logic Programming (ILP) systems.

The learning procedure allows the usage of large taxonomies as background knowledge. The main goal of the learning system can then be seen as that of ontology acquisition or completion (if an initial ontology is available as background knowledge).

2 Knowledge Representation

A probabilistic extension of Description Logics (DL) is used as the underlying knowledge representation and reasoning formalism. DL distinguishes between intensional and extensional knowledge. It provides a *concept language* $\mathcal{L}$ that is used to build concept expressions. A knowledge base Σ is defined as a pair $< \mathcal{T}, \mathcal{A} >$ where $\mathcal{T}$ is called TBox and consists of a set of axioms that provide the intensional knowledge; and $\mathcal{A}$ is called ABox and consists of a set of assertions about individual objects that provide the extensional knowledge. DL semantics is defined in a model-theoretic way. The reader can check [7] for details on DL formalism and concept languages.

A number of concept languages appear in DL literature. YAYA concept language (YCL) is an extension of $\mathcal{ALCN}$[2] that allows inverse roles, role composition, and a construction equivalent to the CLASSIC SAME-AS construct [3].

Why this language? Because in the first experiments done for NLP problems, it seems to be the less expressive language that is expressive enough to solve the problem only by defining a set of axioms and using the reasoning capabilities provided by DL[3].

The axioms allowed by YAYA are of the form $C \leq D$, where C and D are concept expressions in $\mathcal{L}$; that is, they are general concept inclusion axioms. So, YAYA TBoxes are free TBoxes. This, in combination with language complexity would make the reasoning procedures intractable. But in problems in which taxonomic knowledge is important, an accurate representation of it in combination with the implementation of some optimization techniques for DL reasoning procedures [10] can make the overall reasoning process tractable for practical situations.

2.1 Probabilistic Knowledge

In order to represent probabilistic knowledge, axioms are extended to a more general form: $C \leq_{\alpha,\sigma} D$; where α is a probability and σ is a positive real number. Axiom $C \leq_{\alpha,\sigma} D$ states that the conditional probability $P(D|C)$ follows a normal distribution centered in α and with a standard deviation of σ in the set of models of the axiom. This is somewhat similar to the *p-conditioning* notion introduced in [9], but intuitively more precise by the use of the normal distribution. Unlike [12], YAYA does not allow probabilistic assertions.

Model theoretic semantics on which DL is usually defined has to be adapted for probabilistic axioms. For non-probabilistic DL we can divide the set of interpretations of a knowledge base $\Sigma =< \mathcal{T}, \mathcal{A} >$ in two different sets: the set of models of Σ, and the rest of interpretations. Intuitively, for probabilistic DL we forget about this clear distinction. For probabilistic DL we cannot make this clear distinction because given an interpretation we cannot decide whether it is a model of Σ or not. Instead, the set of axioms in $\mathcal{T}$ determines a probability distribution over Σ interpretations. Every Σ interpretation $\mathcal{I}$ is then assigned a probability. This probability may be 0 if $\mathcal{I}$ has no chance to be a model of Σ. Details are out of the scope of this position paper and can be found at [1].

[1] TALP Research Center, Universitat Politècnica de Catalunya, Mòdul C6, Campus Nord, 08034 Barcelona, email: jalvarez@talp.upc.es

[2] $\mathcal{ALCN}$ contains top, bottom, conjunction, disjunction, existential quantification, universal quantification, negation, and unqualified number restrictions (for example, number restrictions allow to define concepts for women having more than 5 children; or roads with less than 3 lanes).

[3] Role axioms also seem interesting, and can easily be introduced following [11].

Probabilistic axioms have two important properties: (a) often a given behaviour can be stated with less and more "natural" axioms if probabilities can take part in axioms[4]; and (b) Given a behaviour to be defined by a set of axioms and two concepts C and D, there always exist some α and σ such that axiom $C \leq_{\alpha,\sigma} D$ is consistent with the behaviour to be learned (and α and σ can be estimated from the set of examples given to the learning procedure).

These two properties make probabilistic axioms specially interesting for learning purposes. The first property states that the theory to be learned will probably be easier to learn if we allow probabilistic axioms. The second one may provide a path of "intermediate" axioms that are completely consistent with the theory to be learned and may help to find out real theory axioms (those "definitive" axioms may convert intermediate axioms into unnecessary ones).

3 Theory Learning

Theory learning is formalized as the process of determining a theory/TBox $\mathcal{T}^*$ from a set of models of $\mathcal{T}^*$. So, the examples used in the learning process are complete models[5]. This differs from usual machine learning approaches in which examples are individual entities.

In order to define what theory learning is, we first adapt Shannon information theory notions to DL domain. An interpretation can be directly represented as a set of boolean variables: for every concept name A and every element $d \in \Delta^{\mathcal{I}}$ we have a boolean variable (stating whether $d \in A^{\mathcal{I}}$ or $d \notin A^{\mathcal{I}}$); and for every role name P and every pair of elements $d_1, d_2 \in \Delta^{\mathcal{I}}$ we have another boolean variable (stating whether $(d_1, d_2) \in P^{\mathcal{I}}$). The *entropy* of $\mathcal{I}$ is defined as the number of boolean variables necessary to represent $\mathcal{I}$ when no additional knowledge is available. This is, in fact, a direct interpretation of Shannon information theory notions.

The amount of information provided by a TBox $\mathcal{T}$ with respect to $\mathcal{I}$ can be intuitively defined as the difference between the necessary number of boolean variables to represent $\mathcal{I}$ when no additional knowledge is available, and the the minimum number of boolean variables necessary to represent $\mathcal{I}$ when $\mathcal{T}$ is available[6]. Then, the learning procedure has to maximize the amount of information provided by $\mathcal{T}$ with respect to a training set. An axiom Φ will be *interesting* for $\mathcal{T}$ if $\mathcal{T} \cup \{\Phi\}$ provides more information than $\mathcal{T}$.

Computing the amount of information provided by a TBox is intractable. Instead, some measures that provide an idea of the information added by an axiom Φ_1 given another axiom Φ_2 can be efficiently computed [2]. These measures have shown to be very useful to reduce the search space explored by a learning procedure.

YAYA learning procedure is not involved with PAC learnability nor Least Common Subsumer (LCS) computation [4], as most work in learning DLs does (see [5, 8] for example). Instead, our learning procedure is a general axiom exploration procedure that is heuristically guided by the information theory measures mentioned above. An initial theory $\mathcal{T}_0$ is evolved[7] through the addition of new axioms. This is done by the application of a set of *induction rules* that produce new axioms from axioms already existing in the theory.

There are 19 different induction rules that can be classified mainly into generalization, specialization, and general exploration rules[8]. In addition, some of these rules are specially designed to work with taxonomic knowledge.

Although there is no space to explain the whole 19 rules, we can sketch some of them to see how they work:

expand antecedent rule It adds an existential construction to the "end" of the antecedent of an axiom.
For example, suppose we are acquiring a theory about family relationships from a a group of families[9]. Suppose $\Phi_1 = man \leq_{0.5,0.3} \exists has\_wife.woman \in \mathcal{T}_t$[10].
Then, this rule would explore axioms by adding simple existential constructions to the end of Φ antecedent. Among others it will produce the axiom $Phi_2 = man \sqcap \exists has\_child.person \leq_{0.87,0.1} \exists has\_wife.woman$, where the parameters for the distribution are supposed to have been inferred from the training set.
As Φ_2 provides more information in what refers to man having childs than Φ_1, it will be added to $\mathcal{T}_t$.

same-as axiom rule It builds new axioms with SAME-AS constructions in the consequent. Going on with the previous example, suppose we have $\Phi_3 = person \sqcap \exists has\_grandparent \leq_{1,0} \exists has\_parent.\exists has\_parent.person$; and we have that $\Phi_3 \in \mathcal{T}_t$.
The application of the same-as axiom rule to Φ_3 would build an axiom Φ_4 with the same antecedent and a consequent constituted by a SAME-AS construct built from the antecedent and the consequent of Φ_3. That is, $\Phi_4 = person \sqcap \exists has\_grandparent \leq_{1,0} [SAME\_AS\ has\_parent \circ has\_parent, has\_granparent]$.
Φ_4 increases the amount of information of $\mathcal{T}_t$, as Φ_4 consequent is more specific than Φ_3 one. So, YAYA learning procedure would add Φ_4 to $\mathcal{T}_t$.

The iterative rule application process is repeated until all the axioms "reachable" through the application of an induction rule are not interesting for $\mathcal{T}$. YAYA learning procedure has been designed from practical principles. Unfortunately there is no theoretical bound on the amount of information captured by the TBox resulting from the learning process with respect to $\mathcal{T}^*$.

---

[4] A very simple and illustrative example is the penguin exception so widely used: the following probabilistic axiomatization $\{bird \leq_{0.95,0.05} can\_fly, penguin \leq bird, penguin \leq \neg\ can\_fly\}$ (supposing probabilities are correct) is simpler than its corresponding non-probabilistic axiomatization: $\{bird \sqcap \neg\ penguin \leq can\_fly, penguin \leq bird, penguin \leq \neg\ can\_fly\}$. Yes, only an antecedent is a bit larger. But this is a quite simple theory; for larger theories differences will sure be bigger.

[5] For example, if we are working in NLP, an example could correspond to a whole sentence, with all the information we want to learn and from which we want to learn (syntactic groups, syntactic relations between them, word senses ...).

[6] This captures the fact that some of the boolean variables necessary when no knowledge is available can be deduced from other boolean variables when $\mathcal{T}$ is available.

[7] This initial theory may be empty, it may contain a subset of the theory to be learned, and it may also take into account ontological knowledge.

[8] The first two types are similar to ILP generalization and specialization operators.

[9] Note that here the examples correspond to families and not to individuals, as it would correspond in most ILP systems.

[10] Throughout this example, it is supposed that *man*, *woman*, and *person* are concept names that appear in the training set, and that *has_wife* and *has_child* are role names.

The big number of YAYA induction rules makes possible that an *interesting* axiom may be reached from a lot of different search states. This is important because in some sense it reduces the amount of local minima in the search space. And as the procedure sketched above is in fact a greedy search, it may be affected by local minima.

By other hand, the YAYA learning procedure is not complete; that is, it does not guarantee it will find a given axiom or a given theory. Nevertheless, it has been tested successfully on some typical ILP benchmarks such as the family relationships, and the chess king-rook-king illegal position ones [16].

4 YAYA and Natural Language Processing

The main goal of YAYA is to be applied to NLP problems. Relational learning systems have been shown to be well suited for Natural Language Learning (NLL). ILP has been successfully applied in a number of NLP problems [15, 14, 6].

With this purpose, the widely used WordNet lexical ontology [13] has been integrated into YAYA. WordNet is automatically converted into a set of axioms that can be used: (1) To provide a background theory to the learning procedure; and (2) For reasoning purposes (once the learning procedure has done its job).

Despite the big amount of concepts (synsets in WordNet terminology) in the mentioned ontology, a careful selection of the axioms into which the taxonomy is mapped, and the implementation of some optimization techniques from [10] maintain the reasoning tasks tractable.

YAYA provides a powerful representation, reasoning, and learning framework for NLP tasks. This has the immediate advantage to make possible to board different NLP tasks using the same environment. But most important, it provides a way to experiment with the integration of several of these tasks; and to study the interaction among them. Additionally, and different to ILP systems, YAYA has been designed specially to take benefit from available taxonomic knowledge.

Traditionally, several phases have been defined in NLP: morfological analysis, syntactical analysis, word sense disambiguation, contextual resolution, conceptual representation, etc. The complete analysis of a natural language text consists of solving each one of the phases in an ordered way. Then, one phase can use information produced by previous phases; but, of course, it cannot use information from later phases as it has not already been produced.

The goal of the *phases* analysis is to maintain the complexity of the problem below certain limits; but it may miss some important interactions between different phases. For NLL systems this may result in the substitution of a missed interaction by a set of special situations that approximate it taking into account only information in previous phases.

Reducing NLP to an only *big phase* is not realistic at this moment. The number of interactions that should be taken into account by a learning procedure is probably too large to obtain interesting results in a reasonable amount of time. Instead, a restricted interaction model can be used to explore interesting interactions. YAYA provides the necessary elements to experiment with different interaction models. Nevertheless, this work is at its initial stage at this moment.

Some experiments have been performed for NLP. More concretely, over 300 sentences from the training set in [17] have been used to learn restrictions between senses and semantic roles. The whole set of restrictions implicitly present in the training set has successfully been learned.

REFERENCES

[1] Jordi Alvarez, 'Extending the tableaux calculus for probabilistic description logics', Technical Report LSI-00-R, Universitat Politcnica de Catalunya, (2000).

[2] Jordi Alvarez, 'A formal framework for theory learning using description logics', Technical Report LSI-00-R, Universitat Politcnica de Catalunya, (2000).

[3] Alex Borgida and Peter F. Patel-Schneider, 'A semantics and complete algorithm for subsumption in the CLASSIC description logic', *Journal of Artificial Intelligence Research*, **1**, 277–308, (1994).

[4] W. Cohen, A. Borgida, and H. Hirsh, 'Computing least common subsumers in description logics', in *Proceedings of AAAI-92*, pp. 754–760, (1992).

[5] W. Cohen and H. Hirsh, 'The learnability of description logics with equality constraints', *Machine Learning*, **17**(2–3), 169–199, (1994).

[6] James Cussens, David Page, Stephen Muggleton, and Ashwin Srinivasan, 'Using Inductive Logic Programming for Natural Language Processing', in *ECML'97 – Workshop Notes on Empirical Learning of Natural Language Tasks*, eds., W. Daelemans, T. Weijters, and A. van der Bosch, pp. 25–34, Prague, (1997).

[7] F.M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf, 'Reasoning in description logics', in *Studies in Logic, Language and Information*, ed., G. Brewka, 193–238, CLSI Publications, (1996).

[8] M. Frazier and L. Pitt, 'CLASSIC learning', *Machine Learning*, **25**, 151–193, (1996).

[9] Jochen Heinsohn, 'Probabilistic description logics', in *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, Seattle, Washington, (1994).

[10] I. Horrocks and P. PatelSchneider, 'Optimising description logic subsumption', *Journal of Logic and Computation*, **9**(3), 267–293, (1999).

[11] Ian Horrocks and Ulrike Sattler, 'A description logics with transitive and inverse roles and role hierarchies', *Journal of Logic and Computation*, **9**(3), 385–410, (1999).

[12] Manfred Jaeger, 'Probabilistic reasoning in terminological logics', in *Proceedings of the 4th International Conference on the Principles of Knowledge Representation and Reasoning*, (1994).

[13] G.A. Miller, 'Wordnet: An online lexical database', *International Journal of Lexicography*, **3**(4), 235–312, (1990).

[14] Raymond J. Mooney, 'Inductive logic programming for natural language processing', in *Proceedings of the Sixth International Inductive Logic Programming Workshop*, (1996).

[15] Stephen Muggleton, 'Inductive logic programming: Issues, results, and the challenge of learning language in logic', *Artificial Intelligence*, **114**(1–2), 283–296, (1999).

[16] J.R. Quinlan, 'Learning logical definitions from relations', *Machine Learning*, **5**, 239–266, (1990).

[17] J. Zelle and R. Mooney, 'Learning semantic grammars with constructive inductive logic programming', in *Proceedings of 11th AAAI*, pp. 817–822, (1993).