pFOIL-DL: Learning (Fuzzy) EL Concept Descriptions from Crisp OWL Data Using a Probabilistic Ensemble Estimation

Umberto Straccia
ISTI - CNR
Via G. Moruzzi 1
Pisa, Italy

Matteo Mucci
ISTI - CNR
Via G. Moruzzi 1
Pisa, Italy

ABSTRACT

OWL ontologies are nowadays a quite popular way to describe structured knowledge in terms of classes, relations among classes and class instances.

In this paper, given an OWL target class T, we address the problem of inducing $\mathcal{EL}(\mathbf{D})$ concept descriptions that describe sufficient conditions for being an individual instance of T. To do so, we use a FOIL-based method with a probabilistic candidate ensemble estimation. We illustrate its effectiveness by means of an experimentation.

1. INTRODUCTION

OWL 2 ontologies [47] are nowadays a popular means to represent *structured* knowledge and its formal semantics is based on *Description Logics* (DLs) [3]. The basic ingredients of DLs are concept descriptions (in First-Order Logic terminology, unary predicates), inheritance relationships among them and instances of them.

Although an important amount of work has been carried about DLs, the application of machine learning techniques to OWL 2 data, is relatively marginally addressed [6, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 22, 24, 25, 26, 28, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 41, 42, 43, 44, 46, 50, 51, 52, 53, 54]. We refer the reader to [50] for a recent overview.

In this work, we focus on the problem of automatically learn concept descriptions from OWL 2 data. More specifically, given an OWL target class T, we address the problem of inducing $\mathcal{EL}(\mathbf{D})$ [2] concept descriptions that describe sufficient conditions for being an individual instance of T.

EXAMPLE 1.1 ([39]). *Consider an ontology that describes the meaningful entities of a city.* [1] *Now, one may fix a city, say Pisa, extract the properties of the hotels from Web sites, such as location, price, etc., and the hotel judgements of the users,* e.g., *from Trip Advisor.* [2] *Now, using the terminology of the ontology, one may ask about what are sufficient conditions characterising good hotels in Pisa according the user feedback. Then one may learn from the user feedback that, for instance, 'An expensive Bed and Breakfast is a good hotel'. The goal here is to have both a human understandable characterisation as well as a good classifier for new, unclassified hotels.*

To improve the readability of such characterisations, we further allow to use so-called *fuzzy concept descriptions* [40, 58] such as 'an *expensive* Bed and Breakfast is a good hotel'. Here, the concept *expensive* is a so-called fuzzy concept [59], *i.e.* a concept for which the belonging of an individual to the class is not necessarily a binary yes/no question, but rather a matter of degree. For instance, in our example, the degree of expensiveness of a hotel may depend on the price of the hotel: the higher the price the more expensive is the hotel.

More specifically, we describe pFOIL-$\mathcal{DL}$, a method for the automated induction of fuzzy $\mathcal{EL}(\mathbf{D})$ concept descriptions [3] by adapting the popular rule induction method FOIL [49]. But, unlike FOIL, in which the goodness of an induced rule is evaluated independently of previously learned rules, here we define a method that evaluates the goodness of a candidate in terms of the *ensemble* of so far learned rules, in a similar way as it happens in nFOIL [29].

Related Work. Related FOIL-like algorithms for the fuzzy case are reported in the literature [13, 55, 56] but they are not conceived for DL ontologies. In the context of DL ontologies, DL-FOIL adapts FOIL to learn crisp OWL DL equivalence axioms under OWA [15]. DL-Learner supports the same learning problem as in DL-FOIL but implements algorithms that are not based on FOIL [24]. Likewise, Lehmann and Haase [33] propose a refinement operator for concept learning in $\mathcal{EL}$ (implemented in the ELTL algorithm within DL-Learner) whereas Chitsaz *et al.* [9] combine a refinement operator for $\mathcal{EL}^{++}$ with a reinforcement learning algorithm. Both works deal with crisp ontologies. [19, 53] uses crisp terminological decision trees [5], while [25] is tailored to learn crisp $\mathcal{ALC}$ definitions. Very recently, an extension of DL-Learner with some of the most up-to-date fuzzy ontology tools has been proposed [26]. Notably, it can learn fuzzy OWL DL equivalence axioms from FuzzyOWL 2 ontologies [4] by interfacing the *fuzzyDL* reasoner [7]. However, it has been tested only on a toy ontology with crisp training

[1] For instance, `http://donghee.info/research/SHSS/ObjectiveConceptsOntology(OCO).html`
[2] `http://www.tripadvisor.com`

[3] $\mathcal{EL}$ is often used as learning language as illustrated in the related work section, and expressions are easily interpretable by humans.
[4] `http://www.straccia.info/software/FuzzyOWL`

SAC'15 April 13-17, 2015, Salamanca, Spain.

http://dx.doi.org/10.1145/2695664.2695707

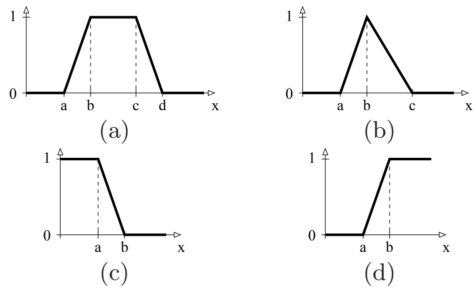

Figure 1: (a) Trapezoidal function $trz(a,b,c,d)$, (b) triangular function $tri(a,b,c)$, (c) left shoulder function $ls(a,b)$, and (d) right shoulder function $rs(a,b)$.

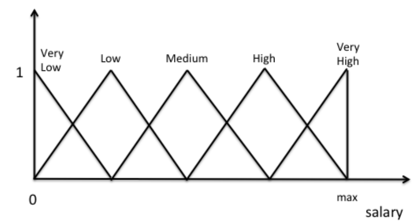

Figure 2: Fuzzy sets over salaries.

Table 1: Combination functions for fuzzy logics.

	Łukasiewicz	Gödel	Product	standard
$\alpha \otimes \beta$	$\max(\alpha+\beta-1,0)$	$\min(\alpha,\beta)$	$\alpha \cdot \beta$	$\min(\alpha,\beta)$
$\alpha \oplus \beta$	$\min(\alpha+\beta,1)$	$\max(\alpha,\beta)$	$\alpha+\beta-\alpha\cdot\beta$	$\max(\alpha,\beta)$
$\alpha \Rightarrow \beta$	$\min(1-\alpha+\beta,1)$	$\begin{cases}1 & \text{if } \alpha \leq \beta\\ \beta & \text{otherwise}\end{cases}$	$\min(1,\beta/\alpha)$	$\max(1-\alpha,\beta)$
$\ominus\, \alpha$	$1-\alpha$	$\begin{cases}1 & \text{if } \alpha = 0\\ 0 & \text{otherwise}\end{cases}$	$\begin{cases}1 & \text{if } \alpha = 0\\ 0 & \text{otherwise}\end{cases}$	$1-\alpha$

examples. The work reported in [28] is based on an ad-hoc translation of fuzzy Łukasiewicz $\mathcal{ALC}$ DL constructs into *Logic Programming* (LP) and then uses a conventional *Inductive Logic Programming* ILP method to lean rules. The method is not sound as it has been recently shown that the mapping from fuzzy DLs to LP is incomplete [45] and entailment in Łukasiewicz $\mathcal{ALC}$ is undecidable [8]. A similar investigation of the problem considered in the present paper can be found in [36, 38, 39]. In [36, 38], the resulting method (named *Soft*FOIL) provides a different solution from pFOIL-$\mathcal{DL}$ as for the weaker knowledge representation language DL-Lite [1], the confidence degree computation, the refinement operator and the heuristic. Moreover, unlike pFOIL-$\mathcal{DL}$, *Soft*FOIL has not been implemented. Finally, [39] is the closest to our work and presents FOIL-$\mathcal{DL}$, a FOIL variant for fuzzy and crisp DLs. Unlike [39], however, we rely here on a novel (fuzzy) probabilistic ensemble evaluation of the fuzzy concept description candidates that seems to provide better effectiveness on our test set.

The paper is structured as follows. Section 2 is devoted to the introduction of basic definitions to make the paper reasonably self-contained. Section 3 describes the learning problem and our algorithm. Section 4 addresses an experimental evaluation and Section 5 concludes the paper.

2. PRELIMINARIES

At first, we address *Fuzzy Logics* and then *Fuzzy Description Logics* basics (see [58] for an introduction to both).

Mathematical Fuzzy Logic. *Fuzzy Logic* is the logic of *fuzzy sets* [59]. A *fuzzy set* A over a countable crisp set X is a function $A\colon X \to [0,1]$, called *fuzzy membership* function of A. A crisp set A is characterised by a membership function $A\colon X \to \{0,1\}$ instead. The standard fuzzy set operations conform to $(A \cap B)(x) = \min(A(x), B(x))$, $(A \cup B)(x) = \max(A(x), B(x))$ and $\bar{A}(x) = 1 - A(x)$ ($\bar{A}$ is the set complement of A), the *inclusion degree* between A and B is defined typically as $deg(A,B) = \frac{\sum_{x \in X}(A \cap B)(x)}{\sum_{x \in X} A(x)}$, while the *cardinality* of a fuzzy set is often defined as $|A| = \sum_{x \in X} A(x)$. The trapezoidal (Fig. 1 (a)), the triangular (Fig. 1 (b)), the L-function (left-shoulder function, Fig. 1 (c)), and the R-function (right-shoulder function, Fig. 1 (d)) are frequently used to specify membership functions of fuzzy sets. Although fuzzy sets have a greater expressive power than classical crisp sets, its usefulness depends critically on the capability to construct appropriate membership functions for various given concepts in different contexts. We refer the interested reader to, *e.g.*, [27]. One easy and typically satisfactory method to define the membership functions is to uniformly partition the range of, *e.g.* salary values (bounded by a minimum and maximum value), into 5 or 7 fuzzy sets using triangular functions (see Figure 2).

In *Mathematical Fuzzy Logic* [23], the convention prescribing that a statement is either true or false is changed and is a matter of degree measured on an ordered scale that is no longer $\{0,1\}$, but typically $[0,1]$. This degree is called *degree of truth* of the logical statement ϕ in the interpretation $\mathcal{I}$. Here, *fuzzy statements* have the form $\langle \phi, \alpha \rangle$, where $\alpha \in (0,1]$ and ϕ is a First-Order Logic (FOL) statement, encoding that the degree of truth of ϕ is *greater or equal* α. So, for instance, $\langle Cheap(HotelVerdi), 0.8\rangle$ states that the statement 'Hotel Verdi is cheap' is true to degree greater or equal 0.8. From a semantics point of view, a *fuzzy interpretation* $\mathcal{I}$ maps each atomic statement p_i into $[0,1]$ and is then extended inductively to all statements as follows:

$$
\begin{aligned}
\mathcal{I}(\phi \wedge \psi) &= \mathcal{I}(\phi) \otimes \mathcal{I}(\psi) \ , \ \mathcal{I}(\phi \vee \psi) = \mathcal{I}(\phi) \oplus \mathcal{I}(\psi) \\
\mathcal{I}(\phi \rightarrow \psi) &= \mathcal{I}(\phi) \Rightarrow \mathcal{I}(\psi) \ , \ \mathcal{I}(\neg \phi) = \ominus \mathcal{I}(\phi) \\
\mathcal{I}(\exists x.\phi(x)) &= \sup_{y \in \Delta^{\mathcal{I}}} \mathcal{I}(\phi(y)) \ , \ \mathcal{I}(\forall x.\phi(x)) = \inf_{y \in \Delta^{\mathcal{I}}} \mathcal{I}(\phi(y)) \ ,
\end{aligned}
$$

where $\Delta^{\mathcal{I}}$ is the domain of $\mathcal{I}$, and $\otimes$, $\oplus$, $\Rightarrow$, and $\ominus$ are so-called *t-norms*, *t-conorms*, *implication functions*, and *negation functions*, respectively, which extend the Boolean conjunction, disjunction, implication, and negation, respectively, to the fuzzy case.

One usually distinguishes three different logics, namely Łukasiewicz, Gödel, and Product logics [23] [5], whose truth combination functions are reported in Table 1.

Note that the operators for standard fuzzy logic, namely $\alpha \otimes \beta = \min(\alpha,\beta)$, $\alpha \oplus \beta = \max(\alpha,\beta)$, $\ominus\,\alpha = 1-\alpha$ and $\alpha \Rightarrow \beta = \max(1-\alpha,\beta)$, can be expressed in Łukasiewicz logic. More precisely, $\min(\alpha,\beta) = \alpha \otimes_l (\alpha \Rightarrow_l \beta), \max(\alpha,\beta) = 1 - \min(1-\alpha, 1-\beta)$. Furthermore, the implication $\alpha \Rightarrow_{kd} \beta = \max(1-\alpha, b)$ is called *Kleene-Dienes implication* (denoted $\Rightarrow_{kd}$), while *Zadeh implication* (denoted $\Rightarrow_z$) is the implication $\alpha \Rightarrow_z \beta = 1$ if $\alpha \leq \beta$; 0 otherwise.

[5] Notably, a theorem states that any other continuous t-norm can be obtained as a combination of them.

An *r-implication* is an implication function obtained as the residuum of a continuous t-norm $\otimes$ [6], *i.e.* $\alpha \Rightarrow \beta = \max\{\gamma \mid \alpha \otimes \gamma \leq \beta\}$. Note also, that given an r-implication $\Rightarrow_r$, we may also define its related negation $\ominus_r \alpha$ by means of $\alpha \Rightarrow_r 0$ for every $\alpha \in [0,1]$.

The notions of satisfiability and logical consequence are defined in the standard way, where a fuzzy interpretation $\mathcal{I}$ *satisfies* a fuzzy statement $\langle \phi, \alpha \rangle$, or $\mathcal{I}$ is a *model* of $\langle \phi, \alpha \rangle$, denoted as $\mathcal{I} \models \langle \phi, \alpha \rangle$, iff $\mathcal{I}(\phi) \geq \alpha$. Notably, from $\langle \phi, \alpha \rangle$ and $\langle \phi \to \psi, \beta \rangle$ one may conclude (if $\to$ is an r-implication) $\langle \psi, \alpha \otimes \beta \rangle$ (called fuzzy modus ponens).

Fuzzy Description Logics basics. We recap here the fuzzy variant of the DL $\mathcal{ALC}(\mathbf{D})$ [57], which is expressive enough to capture the main ingredients of fuzzy DLs.

We start with the notion of *fuzzy concrete domain*, that is a tuple $\mathbf{D} = \langle \Delta^{\mathbf{D}}, \cdot^{\mathbf{D}} \rangle$ with data type domain $\Delta^{\mathbf{D}}$ and a mapping $\cdot^{\mathbf{D}}$ that assigns to each data value an element of $\Delta^{\mathbf{D}}$, and to every 1-ary data type predicate $\mathbf{d}$ a 1-ary fuzzy relation over $\Delta^{\mathbf{D}}$. Therefore, $\cdot^{\mathbf{D}}$ maps indeed each data type predicate into a function from $\Delta^{\mathbf{D}}$ to $[0,1]$. Typical examples of data type predicates $\mathbf{d}$ are the well known membership functions

$$\begin{array}{rcl}\mathbf{d} & \to & ls(a,b) \mid rs(a,b) \mid tri(a,b,c) \mid trz(a,b,c,d) \\ & & \mid \ \geq_v \ \mid \ \leq_v \ \mid \ =_v \ ,\end{array}$$

where *e.g.* $ls(a,b)$ is the left-shoulder membership function and $\geq_v$ corresponds to the crisp set of data values that are greater or equal than the value v.

Now, let $\mathbf{A}$ be a set of *concept names* (also called atomic concepts), $\mathbf{R}$ be a set of *role names*. Each role is either an *object property* or a *data type property*. The set of *concepts* are built from concept names A using connectives and quantification constructs over object properties R and data type properties S, as described by the following syntactic rule:

$$\begin{array}{rcl}C & \to & \top \mid \ \bot \ \mid A \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid \neg C \mid C_1 \to C_2 \\ & & \mid \exists R.C \mid \forall R.C \mid \exists S.\mathbf{d} \mid \forall S.\mathbf{d} \ .\end{array}$$

An *ABox* $\mathcal{A}$ consists of a finite set of assertion axioms. An *assertion* axiom is an expression of the form $\langle a{:}C, \alpha \rangle$ (*concept assertion*, a is an instance of concept C to degree at least α) or of the form $\langle (a_1,a_2){:}R, \alpha \rangle$ (*role assertion*, (a_1,a_2) is an instance of object property R to degree at least α), where a, a_1, a_2 are individual names, C is a concept, R is an object property and $\alpha \in (0,1]$ is a truth value. A *Terminological Box* or *TBox* $\mathcal{T}$ is a finite set of *General Concept Inclusion* (GCI) axioms, where a GCI is of the form $\langle C_1 \sqsubseteq C_2, \alpha \rangle$ (C_1 is a sub-concept of C_2 to degree at least α), where C_i is a concept and $\alpha \in (0,1]$. We may omit the truth degree α of an axiom; in this case $\alpha = 1$ is assumed. A *Knowledge Base* (KB) is a pair $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$. With $\mathsf{I}_{\mathcal{K}}$ we denote the set of individuals occurring in $\mathcal{K}$.

Concerning the semantics, let us fix a fuzzy logic. Unlike classical DLs in which an interpretation $\mathcal{I}$ maps *e.g.* a concept C into a set of individuals $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, *i.e.* $\mathcal{I}$ maps C into a function $C^{\mathcal{I}} : \Delta^{\mathcal{I}} \to \{0,1\}$ (either an individual belongs to the extension of C or does not belong to it), in fuzzy DLs, $\mathcal{I}$ maps C into a function $C^{\mathcal{I}} : \Delta^{\mathcal{I}} \to [0,1]$ and, thus, an individual belongs to the extension of C to some degree in $[0,1]$, *i.e.* $C^{\mathcal{I}}$ is a fuzzy set. Specifically, a *fuzzy interpretation* is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consisting of a nonempty (crisp) set $\Delta^{\mathcal{I}}$ (the *domain*) and of a *fuzzy interpretation function* $\cdot^{\mathcal{I}}$ that assigns: *(i)* to each atomic concept A a function $A^{\mathcal{I}} \colon \Delta^{\mathcal{I}} \to [0,1]$; *(ii)* to each object property R a function $R^{\mathcal{I}} \colon \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \to [0,1]$; *(iii)* to each data type property S a function $S^{\mathcal{I}} \colon \Delta^{\mathcal{I}} \times \Delta^{\mathbf{D}} \to [0,1]$; *(iv)* to each individual a an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ such that $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ if $a \neq b$ (Unique Name Assumption); and *(v)* to each data value v an element $v^{\mathcal{I}} \in \Delta^{\mathbf{D}}$. Now, a fuzzy interpretation function is extended to concepts as specified below (where $x \in \Delta^{\mathcal{I}}$):

$$\begin{array}{rcl}\bot^{\mathcal{I}}(x) & = & 0 \ , \ \ \top^{\mathcal{I}}(x) \ = \ 1 \\ (C \sqcap D)^{\mathcal{I}}(x) & = & C^{\mathcal{I}}(x) \otimes D^{\mathcal{I}}(x) \\ (C \sqcup D)^{\mathcal{I}}(x) & = & C^{\mathcal{I}}(x) \oplus D^{\mathcal{I}}(x) \\ (\neg C)^{\mathcal{I}}(x) & = & \ominus C^{\mathcal{I}}(x) \\ (C \to D)^{\mathcal{I}}(x) & = & C^{\mathcal{I}}(x) \Rightarrow D^{\mathcal{I}}(x) \\ (\forall R.C)^{\mathcal{I}}(x) & = & \inf_{y \in \Delta^{\mathcal{I}}} \{R^{\mathcal{I}}(x,y) \Rightarrow C^{\mathcal{I}}(y)\} \\ (\exists R.C)^{\mathcal{I}}(x) & = & \sup_{y \in \Delta^{\mathcal{I}}} \{R^{\mathcal{I}}(x,y) \otimes C^{\mathcal{I}}(y)\} \\ (\forall S.\mathbf{d})^{\mathcal{I}}(x) & = & \inf_{y \in \Delta^{\mathbf{D}}} \{S^{\mathcal{I}}(x,y) \Rightarrow \mathbf{d}^{\mathbf{D}}(y)\} \\ (\exists S.\mathbf{d})^{\mathcal{I}}(x) & = & \sup_{y \in \Delta^{\mathbf{D}}} \{S^{\mathcal{I}}(x,y) \otimes \mathbf{d}^{\mathbf{D}}(y)\} \ .\end{array}$$

The *satisfiability of axioms* is then defined by the following conditions: *(i)* $\mathcal{I}$ satisfies an axiom $\langle a{:}C, \alpha \rangle$ if $C^{\mathcal{I}}(a^{\mathcal{I}}) \geq \alpha$; *(ii)* $\mathcal{I}$ satisfies an axiom $\langle (a,b){:}R, \alpha \rangle$ if $R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) \geq \alpha$; *(iii)* $\mathcal{I}$ satisfies an axiom $\langle C \sqsubseteq D, \alpha \rangle$ if $(C \sqsubseteq D)^{\mathcal{I}} \geq \alpha$ where [7] $(C \sqsubseteq D)^{\mathcal{I}} = \inf_{x \in \Delta^{\mathcal{I}}} \{C^{\mathcal{I}}(x) \Rightarrow D^{\mathcal{I}}(x)\}$. $\mathcal{I}$ is a *model* of $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$ iff $\mathcal{I}$ satisfies each axiom in $\mathcal{K}$. If $\mathcal{K}$ has a model we say that $\mathcal{K}$ is *satisfiable* (or *consistent*). We say that $\mathcal{K}$ *entails* axiom τ, denoted $\mathcal{K} \models \tau$, if any model of $\mathcal{K}$ satisfies τ. The *best entailment degree* of τ of the form $C \sqsubseteq D$, $a{:}C$ or $(a,b){:}R$, denoted $bed(\mathcal{K}, \tau)$, is defined as

$$bed(\mathcal{K}, \tau) = \sup\{\alpha \mid \mathcal{K} \models \langle \tau, \alpha \rangle\} \ .$$

The *cardinality* of a concept C w.r.t. a KB $\mathcal{K}$ and a set of individuals I, denoted $|C|^{\mathsf{I}}_{\mathcal{K}}$, is defined as

$$|C|^{\mathsf{I}}_{\mathcal{K}} = \sum_{a \in \mathsf{I}} bed(\mathcal{K}, a{:}C) \ .$$

Example 2.1. *Let us consider the following axiom*

$$\langle \exists hasPrice.High \sqsubseteq GoodHotel, 0.569 \rangle \ ,$$

where hasPrice is a data type property whose values are measured in euros and the price concrete domain has been

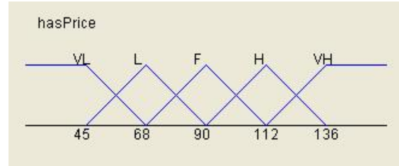

Figure 3: Fuzzy concepts derived from the data type property `hasPrice`.

[6]Note that Łukasiewicz, Gödel and Product implications are r-implications, while Kleene-Dienes implication is not.

[7]However, note that under standard logic $\sqsubseteq$ is interpreted as $\Rightarrow_z$ and not as $\Rightarrow_{kd}$.

automatically fuzzified as illustrated in Figure 3. Now, it can be verified that for hotel verdi, whose room price is 105 *euro,* i.e. *we have the assertion* $verdi{:}\exists hasPrice.=_{105}$ *in the KB, we infer under Product logic that* [8]

$$\mathcal{K} \models \langle verdi{:}GoodHotel, 0.18\rangle \ .$$

3. THE LEARNING ALGORITHM

The problem statement. Given the target concept name T, the *training set* $\mathcal{E}$ consists of crisp concept assertions of the form $a{:}T$ where a is an individual occurring in $\mathcal{K}$. Also, $\mathcal{E}$ is partitioned into $\mathcal{E}^+$ and $\mathcal{E}^-$ (the positive and negative examples, respectively). Now, the learning problem is defined as follows. Given:

- a consistent KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$ (the *background theory*);
- a concept name T (the *target concept*);
- a training set $\mathcal{E} = \mathcal{E}^+ \cup \mathcal{E}^-$ such that $\forall e \in \mathcal{E}, \mathcal{K} \not\models e$;
- a set $\mathcal{L}_\mathcal{H}$ of fuzzy GCIs (the *language of hypotheses*)

the goal is to find a finite set $\mathcal{H} \subset \mathcal{L}_\mathcal{H}$ (a *hypothesis*) of GCIs $\mathcal{H} = \{C_1 \sqsubseteq T, ..., C_n \sqsubseteq T\}$ such that:

Consistency. $\mathcal{K} \cup \mathcal{H}$ is a consistent;

Non-Redundancy. $\forall \tau \in \mathcal{H}, \mathcal{K} \not\models \tau$

Soundness. $\forall e \in \mathcal{E}^-, \mathcal{K} \cup \mathcal{H} \not\models e$;

Completeness. $\forall a{:}T \in \mathcal{E}^+, \mathcal{K} \models a{:}C_i$, for some $i \in \{1, ..., n\}$.

We say that an axiom $\tau \in \mathcal{L}_\mathcal{H}$ *covers* an example $e \in \mathcal{E}$ iff $\mathcal{K} \cup \{\tau\} \models e$. Note that the way completeness is stated guarantees that all positive examples are covered by at least one axiom. Also, in practice one may allow a hypothesis that does not necessarily cover all positive examples, which is what we allow in our experiments.

The language of hypotheses. Given the target concept name T, the hypotheses to be induced are fuzzy $\mathcal{EL}(\mathbf{D})$ GCIs, [9] which are of the form

$$C \sqsubseteq T \ , \tag{1}$$

where the left-hand side is defined according to the following syntax:

$$\begin{array}{lll} C & \longrightarrow & \top \mid A \mid \exists R.C \mid \exists S.\mathbf{d} \mid C_1 \sqcap C_2 \\ \mathbf{d} & \rightarrow & ls(a,b) \mid rs(a,b) \mid tri(a,b,c) \mid trz(a,b,c,d) \ . \end{array}$$

Note that the language $\mathcal{L}_\mathcal{H}$ generated by this $\mathcal{EL}(\mathbf{D})$ syntax is potentially infinite due, *e.g.*, to the nesting of existential restrictions yielding to complex concept expressions such as $\exists R_1.(\exists R_2 \ldots .(\exists R_n.(C)) \ldots)$. The language is made finite by imposing further restrictions on the generation process such as the maximal number of conjuncts and the depth of existential nestings allowed in the left-hand side.

Also, note that the learnable GCIs do not have an explicit truth degree. However, even if $\mathcal{K}$ is a crisp $\mathcal{DL}$ KB, the possible occurrence of fuzzy concrete domains in expressions of the form $\exists S.\mathbf{d}$ in the left-hand side of a fuzzy GCI of the form (1) may imply both that $bed(\mathcal{K}, C \sqsubseteq T) \notin \{0,1\}$ and $bed(\mathcal{K}, a{:}C) \notin \{0,1\}$. Furthermore, as we shall see later on, once we have learned a fuzzy GCI $C \sqsubseteq T$, we may attach to it a degree (see Eq. 2), which may be seen as a kind of fuzzy set inclusion degree (see also Section 2) between the fuzzy concept C and T.

Finally, note that the syntactic restrictions of $\mathcal{L}_\mathcal{H}$ allow for a straightforward translation of the inducible axioms of the form $C_1 \sqcap \ldots \sqcap C_n \sqsubseteq T$ as a rule "if C_1 and ... and C_n then T".

pFoil-$\mathcal{DL}$. Overall, pFOIL-$\mathcal{DL}$ is inspired on FOIL, a popular ILP algorithm for learning sets of rules which performs a greedy search in order to maximise a gain function [49]. More specifically, pFOIL-$\mathcal{DL}$ has many commonalities with FOIL-$\mathcal{DL}$ [39], a FOIL variant for fuzzy and crisp DLs. For reasons of space, we do not present FOIL-$\mathcal{DL}$ here. Let us remark, however, that pFOIL-$\mathcal{DL}$ is a probabilistic variant of FOIL-$\mathcal{DL}$. It has three main differences from FOIL-$\mathcal{DL}$:

- it uses a probabilistic measure to evaluate concept expressions;
- instead of removing positive examples covered from the training set as FOIL-$\mathcal{DL}$ does, once an axiom has been learnt, positive examples are left unchanged after each learned rule;
- concept expressions are no longer evaluated one at a time, but the whole set of learnt expressions is evaluated as an ensemble.

Roughly, our learning algorithm proceeds as follows:

1. start from concept $\top$;
2. apply a refinement operator to find more specific concept description candidates;
3. exploit a scoring function to choose the best candidate;
4. re-apply the refinement operator until a good candidate is found;
5. iterate the whole procedure until a satisfactory coverage of the positive examples is achieved.

We are going now to detail the steps.

Computing fuzzy data types. For numerical data types, we adopt a discretisation method to partition it into a finite number of sets. After such a discretisation has been obtained we can define a fuzzy function on each set and obtain a fuzzy partition. In this work, we chose *equal width triangular partition* (see Figure 2). Specifically, for the triangular case, we look for the minimum and maximum value of a data property S in the ontology $\mathcal{K}$, specifically $min_S = \min\{v \mid \mathcal{K} \models a{:}\exists S.=_v \text{ for some } a \in \mathsf{I}_\mathcal{K}\}$ and $max_S = \max\{v \mid \mathcal{K} \models a{:}\exists S.=_v \text{ for some } a \in \mathsf{I}_\mathcal{K}\}$. Then the interval $[min_S, max_S]$ is split into $n > 1$, intervals of same width. Each of the n intervals, $I_1, ..., I_n$, has a fuzzy membership function associated. Let $\Delta_S = \dfrac{max_S - min_S}{n-1}$, then I_1 has associated a left shoulder $ls(min_S, min_S + \Delta)$, I_n instead has associated a right shoulder $rs(max_S - \Delta, max_S)$ and I_i with $i = 2, ..., n-1$, has associated a triangular function $tri(min_S + (i-2)\cdot\Delta, min_S + (i-1)\cdot\Delta, min_S + i\cdot\Delta)$.

The partition with $n = 5$ is the typical approach and indeed is also what we have chosen in our experiments.

[8] Using fuzzy modus ponens, $0.18 = 0.318 \cdot 0.569$, where $0.318 = tri(90, 112, 136)(105)$.

[9] Note that $\mathcal{EL}$ is a basic ingredient of the OWL profile language OWL EL [48].

Table 2: Downward Refinement Operator.

$$\rho(C) = \begin{cases} \mathcal{A} \cup \{\exists R.\top \mid R \in \mathcal{R}\} \cup \{\exists S.d \mid S \in \mathcal{S}, d \in \mathcal{D}\} & \text{if } C = \top \\ \{A' \mid A' \in \mathcal{A}, \mathcal{K} \models A' \sqsubseteq A\} \cup \{A \sqcap A'' \mid A'' \in \rho(\top)\} & \text{if } C = A \\ \{\exists R.D' \mid D' \in \rho(D)\} \cup \{(\exists R.D) \sqcap D'' \mid D'' \in \rho(\top)\} & \text{if } C = \exists R.D, R \in \mathcal{R} \\ \{(\exists S.d) \sqcap D \mid D \in \rho(\top)\} & \text{if } C = \exists S.d, S \in \mathcal{S}, d \in \mathcal{D} \\ \{C_1 \sqcap \ldots \sqcap C_i' \sqcap \ldots \sqcap C_n \mid i = 1, \ldots, n, C_i' \in \rho(C_i)\} & \text{if } C = C_1 \sqcap \ldots \sqcap C_n \end{cases}$$

The refinement operator. The refinement operator we employ follows [37, 38, 39]. Essentially, this operator takes as input a concept C and generates new, more specific concept description candidates D (*i.e.*, $\mathcal{K} \models D \sqsubseteq C$) and is defined as follows. Let $\mathcal{K}$ be an ontology, $\mathcal{A}$ be the set of all atomic concepts in $\mathcal{K}$, $\mathcal{R}$ the set of all object properties in $\mathcal{K}$, $\mathcal{S}$ the set of all data type properties in $\mathcal{K}$ and $\mathcal{D}$ a set of (fuzzy) datatypes. The refinement operator ρ is defined in Table 2. An example of refinement is the following.

EXAMPLE 3.1. *Consider the KB in Example 1.1. Assume that the target concept is Good_Hotel, that*

$$\mathcal{K} \models Hotel\_4\_Stars \sqsubseteq Accommodation$$

and also assume that the current concept description candidate C is

$$Accommodation \sqcap \exists hasPrice.VeryHigh \ .$$

Then the following concept description belongs to the refinement of C:

$$Hotel\_4\_Stars \sqcap \exists hasPrice.VeryHigh \ .$$

Note that the aim of a refinement is to reduce the number of covered negative examples, while still keeping some covered positive examples.

The scoring function. pFOIL-$\mathcal{DL}$ uses a scoring function to select the best candidate at each refinement step. For ease of presentation, let us assume that the hypothesis contains one axiom only, *i.e.* $\mathcal{H} = \{C \sqsubseteq T\}$. Now, we define

$$\begin{aligned} T^+ &= |T|_{\mathcal{K}}^{\mathcal{E}} \\ C^+ &= |C|_{\mathcal{K}}^{\mathcal{E}} \\ T^+ \sqcap C^+ &= |T \sqcap C|_{\mathcal{K}}^{\mathcal{E}} \ . \end{aligned}$$

So, for instance, C^+ is the cardinality of the fuzzy concept C w.r.t. the KB $\mathcal{K}$ and the set of individuals occurring in the training set $\mathcal{E}$. Now, we define

$$P(C^+ \mid T^+) = \frac{C^+ \sqcap T^+}{T^+} \ , \tag{2}$$

as the probability of correctly classifying a positive example, and

$$P(T^+ \mid C^+) = \frac{C^+ \sqcap T^+}{C^+} \ , \tag{3}$$

as the probability that a sample classified as positive is actually positive. Notice that, Eq. 2 measures the so-called *recall* of C w.r.t. T, while Eq. 3 measures the *precision* of C w.r.t. T. Eventually, we combine these two measures using the well-known $F_\beta-$score function [4]:

$$score_\beta(C) = (1+\beta^2) \cdot \frac{P(T^+ \mid C^+) \cdot P(C^+ \mid T^+)}{\beta^2 \cdot P(T^+ \mid C^+) + P(C^+ \mid T^+)}$$

to compute the confidence degree of concept C.

For the general case in which we have a set $\mathcal{H} = \{C_1 \sqsubseteq T, \ldots, C_n \sqsubseteq T\}$, we define

$$\begin{aligned} \mathcal{H}^+ &= |C_1 \sqcup \ldots \sqcup C_n|_{\mathcal{K}}^{\mathcal{E}} \ , \\ \mathcal{H}^+ \sqcap T^+ &= |(C_1 \sqcup \ldots \sqcup C_n) \sqcap T|_{\mathcal{K}}^{\mathcal{E}} \ , \end{aligned}$$

which derives from viewing the axioms in $\mathcal{H}$ as a GCI $C_1 \sqcup \ldots \sqcup C_n \sqsubseteq T$ and, thus, we have:

$$P(T^+ \mid \mathcal{H}^+) = \frac{\mathcal{H}^+ \sqcap T^+}{\mathcal{H}^+} \tag{4}$$

$$P(\mathcal{H}^+ \mid T^+) = \frac{\mathcal{H}^+ \sqcap T^+}{T^+} \tag{5}$$

$$score_\beta(\mathcal{H}) = (1+\beta^2) \cdot \frac{P(T^+ \mid \mathcal{H}^+) \cdot P(\mathcal{H}^+ \mid T^+)}{\beta^2 \cdot P(T^+ \mid \mathcal{H}^+) + P(\mathcal{H}^+ \mid T^+)} \tag{6}$$

Finally, the confidence degree of a concept expression C w.r.t. axioms $\mathcal{H} = \{C_1 \sqsubseteq T, \ldots, C_n \sqsubseteq T\}$ that have been already learnt is defined as

$$score_\beta(C, \mathcal{H}) = score_\beta(\mathcal{H} \cup \{C \sqsubseteq T\}) \ . \tag{7}$$

Stop Criterion. FOIL-$\mathcal{DL}$ stops when all positive samples are covered or when it becomes impossible to find a concept without negative coverage [39]. In pFOIL-$\mathcal{DL}$, instead, we impose that no axioms are learnt unless they improve the ensemble score. So if adding a new axiom the score of the ensemble does not vary (or decreases) the axiom is not learnt. Moreover, we use a threshold δ and stop a soon as the score improvement is below δ.

Once an axiom has been learnt, the ensemble performance is evaluated using Eq. 6, while a candidate concept description is evaluated according to Eq. 7. In our algorithm, we allow to specify the β used during concept evaluation as β_1, while the β used during ensemble evaluation is specified as β_2.

Additionally, to guarantee termination, we provide two parameters to limit the search space: namely, the maximal number of conjuncts and the maximal depth of existential nestings allowed in a fuzzy GCI (therefore, the computation may end without covering all positive examples).

Backtracking. The implementation of pFOIL-$\mathcal{DL}$ features, as for FOIL-$\mathcal{DL}$ [39], also a backtracking mode. This option allows to overcome one of the main limits of FOIL. Indeed, as it performs a greedy search, formulating a sequence of axioms without backtracking, FOIL does not guarantee to find the smallest or best set of rules that explain the training examples. Also, learning axioms one by one could lead to less and less interesting rules. To reduce the risk of a suboptimal choice at any search step, the greedy search can be replaced in pFOIL-$\mathcal{DL}$ by a *beam search* which maintains a list of k best candidates at each step instead of a single best candidate.

The Algorithm. The pFoil-$\mathcal{DL}$ algorithm is defined in Algorithm 1. The algorithm `learnSetOfAxioms` loops to learn one axiom at time, invoking `learnOneAxiom`. This latter procedure is detailed in Algorithm 2. Note that we do not allow the covering of negative examples due to the condition $|C|_{\mathcal{K}}^{\mathsf{I}_{\mathcal{E}^-}} \neq 0$ at step 3. Its inner loop (steps 7 - 11) just determines the best candidate concept among all refinements. For completeness, in Algorithm 3 we report the variant of `learnOneAxiom` with backtracking using *Top-k backtracking*. Top-k backtracking saves the k best concept expressions obtained from refinements. We use a stack of k concepts ordered in decreasing order of the score. The `ADD` function, called at step 10, adds a concept to the stack if it makes into the top-k ones (and removes the $k+1$ ranked one), while the `POP` function, called in step 16, instead removes from the stack the concept expression that has the highest score. The removed concept is returned as result.

Algorithm 1 pFoil-$\mathcal{DL}$: learnSetOfAxioms

1: **function** `learnSetOfAxioms`($\mathcal{K}$, T, $\mathcal{E}^+$, $\mathcal{E}^-$, θ, β_1, β_2)
2: $\quad \mathcal{H} \leftarrow \emptyset$;
3: $\quad oldScore \leftarrow 0$;
4: $\quad newScore \leftarrow score_{\beta_2}(\{\top \sqsubseteq T\})$;
5: $\quad$ **while** $newScore - oldScore > \theta$ **do**
6: $\quad\quad C \leftarrow$ learnOneAxiom($\mathcal{K}$, T, $\mathcal{H}$, $\mathcal{E}^+$, $\mathcal{E}^-$, β_1);
7: $\quad\quad$ **if**(($C = \top$) **or** ($C =$ `null`))
8: $\quad\quad\quad$ **return** $\mathcal{H}$;
9: $\quad\quad \mathcal{H} \leftarrow \mathcal{H} \cup \{C \sqsubseteq T\}$;
10: $\quad\quad oldScore \leftarrow newscore$
11: $\quad\quad newScore \leftarrow score_{\beta_2}(\mathcal{H})$;
$\quad$ **return** $\mathcal{H}$;
12: **end**

Algorithm 2 pFoil-$\mathcal{DL}$: learnOneAxiom

1: **function** `learnOneAxiom`($\mathcal{K}$, T, $\mathcal{H}$, $\mathcal{E}^+$, $\mathcal{E}^-$, β_1)
2: $\quad C \leftarrow \top$;
3: $\quad$ **while** $|C|_{\mathcal{K}}^{\mathsf{I}_{\mathcal{E}^-}} \neq 0$ **do**
4: $\quad\quad C_{best} \leftarrow C$;
5: $\quad\quad maxscore \leftarrow score_{\beta_1}(C_{best}, \mathcal{H})$;
6: $\quad\quad \mathcal{C} \leftarrow$ refine(C);
7: $\quad\quad$ **for all** $C' \in \mathcal{C}$ **do**
8: $\quad\quad\quad score \leftarrow score_{\beta_1}(C', \mathcal{H})$;
9: $\quad\quad\quad$ **if** $score > maxscore$ **then**
10: $\quad\quad\quad\quad maxscore \leftarrow score$;
11: $\quad\quad\quad\quad C_{best} \leftarrow C'$;
12: $\quad\quad$ **if**($C_{best} = C$)
13: $\quad\quad\quad$ **return** `null`;
14: $\quad\quad C \leftarrow C_{best}$;
$\quad$ **return** C;
15: **end**

Algorithm 3 pFoil-$\mathcal{DL}$: learnOneAxiom (Top-k backtracking)

1: **function** `learnOneAxiom`($\mathcal{K}$, T, $\mathcal{H}$, $\mathcal{E}^+$, $\mathcal{E}^-$, β_1)
2: $\quad C \leftarrow \top$;
3: $\quad stack \leftarrow \emptyset$;
4: $\quad$ **while** $|C|_{\mathcal{K}}^{\mathsf{I}_{\mathcal{E}^-}} \neq 0$ **do**
5: $\quad\quad C_{best} \leftarrow C$;
6: $\quad\quad maxscore \leftarrow score_{\beta_1}(C_{best}, \mathcal{H})$;
7: $\quad\quad \mathcal{C} \leftarrow$ refine(C);
8: $\quad\quad$ **for all** $C' \in \mathcal{C}$ **do**
9: $\quad\quad\quad score \leftarrow score_{\beta_1}(C', \mathcal{H})$;
10: $\quad\quad\quad$ Add($C', score, stack$); ,
11: $\quad\quad\quad$ **if** $score > maxscore$ **then**
12: $\quad\quad\quad\quad maxscore \leftarrow score$;
13: $\quad\quad\quad\quad C_{best} \leftarrow C'$;
14: $\quad\quad$ **if** $C_{best} = C$ **then**
15: $\quad\quad\quad$ **if** $stack \neq \emptyset$ **then**
16: $\quad\quad\quad\quad C_{best} \leftarrow$pop($stack$);
17: $\quad\quad\quad$ **else return** `null`;
18: $\quad\quad C \leftarrow C_{best}$;
$\quad$ **return** C;
19: **end**

4. EVALUATION

Setup. We conducted a preliminary experimentation. A prototype has been implemented (on top of the owl-api and a classical DL reasoner), supporting both Foil-$\mathcal{DL}$ as well as pFoil-$\mathcal{DL}$, and applied them to test a classification case. To this purpose, a number of OWL ontologies have been considered as illustrated in Table 3. For each ontology $\mathcal{K}$ a meaning full target concept T has manually been selected such that the conditions of the learning problem statement (see Section 3) are satisfied. Then we learned inclusion axioms of the form $C \sqsubseteq T$ and measured how good these inclusion axioms are to classify individuals of being instances of T. We report the DL the ontology refers to, the number of concept names, object properties, datatype properties and individuals in the ontology. We also report the maximal nesting depth and maximal number of conjuncts (last column) during the learning phase.

In our tests, the set of negative examples $\mathcal{E}^-$ is the set of the individuals occurring in the KB which cannot be proved to be instance of T (*Closed World Assumption*, CWA). A 5-fold cross validation design was adopted to determine the average performance indices, namely *Precision* (denoted P), see Eq. 4, *Recall* (denoted R), see Eq. 5, $F1$-score (denoted $F1$), see Eq. 6 and the *Mean Squared Error* (MSE), defined as

$$MSE = \sum_{a \in \mathsf{I}_{\mathcal{E}}} (\mathcal{H}(a) - \mathcal{E}(a))^2 \ ,$$

where $\mathcal{E}(a) = 1$ if a is a positive example (*i.e.*, $a \in \mathsf{I}_{\mathcal{E}^+}$), $\mathcal{E}(a) = 0$ if a is a negative example (*i.e.*, $a \in \mathsf{I}_{\mathcal{E}^-}$), and $\mathcal{H}(a)$ is the degree of being a instance of T [10], defined as

$$\mathcal{H}(a) = bed(\mathcal{K} \cup \mathcal{H}, a{:}T) \ .$$

For each measure, the average value over the various folds is reported in the tables.

For both Foil-$\mathcal{DL}$ as well as pFoil-$\mathcal{DL}$ we used top-5 backtracking, as the backtracking mode performed generally better than without it, and standard fuzzy logic has been chosen to compute the scores. For Foil-$\mathcal{DL}$ we have $\theta = 0$, while for pFoil-$\mathcal{DL}$ the parameter setting was as follows: $\theta = 0.05$, $\beta_1 = 1$ and $\beta_2 = 1$. Parameters has been chosen manually and do not necessarily maximise performance . All the data and implementation can be downloaded from here. [11]

Results. For illustrative purposes, example expressions learnt by pFoil-$\mathcal{DL}$ during the experiments are reported in Table 4. Note that both in the Hotel and the UBA ontology case a fuzzy GCI has been induced.

Table 5 summarises the results of the tests for each ontology and measure. The t column reports the average time (in

[10] We recall that concept descriptions may be fuzzy.

[11] `http://nmis.isti.cnr.it/~straccia/ftp/SAC15.Tests.zip`.

Table 3: **Facts about the ontologies of the experiment.**

ontology	DL	classes	obj. prop.	data. prop.	individuals	target	max d./c.
Family-tree	$\mathcal{SROIF}(\mathcal{D})$	22	52	6	368	Uncle	1/3
Hotel	$\mathcal{ALCOF}(\mathcal{D})$	89	3	1	88	Good_Hotel	1/3
Moral	$\mathcal{ALC}$	46	0	0	202	Guilty	1/2
SemanticBible	$\mathcal{SHOIN}(\mathcal{D})$	51	29	9	723	Woman	1/2
UBA	$\mathcal{SHI}(\mathcal{D})$	44	26	8	1268	Good_Researcher	1/1
FuzzyWine	$\mathcal{SHI}(\mathcal{D})$	178	15	7	138	DryWine	1/3

Table 4: Examples of learned axioms by pFoil-$\mathcal{DL}$.

ontology	induced axiom
Family-tree	$(\texttt{Person} \sqcap (\exists \texttt{ancestorOf}.\top)) \sqsubseteq \texttt{Uncle}$
Hotel	$\texttt{Bed\_and\_Breakfast} \sqcap (\exists \texttt{hasPrice.High}) \sqsubseteq \texttt{Good\_Hotel}$
Moral	$\texttt{blameworthy} \sqsubseteq \texttt{Guilty}$
SemanticBible	$\exists \texttt{spouseOf.Man} \sqsubseteq \texttt{Woman}$
UBA	$\exists \texttt{hasNumberOfPublications.VeryHigh} \sqsubseteq \texttt{Good\_Researcher}$
FuzzyWine	$\texttt{OffDryWine} \sqcap \texttt{SweetWine} \sqsubseteq \texttt{DryWine}$

Table 5: Classification results.

ontology	P	R	F1	MSE	t
Family-tree	1.0	1.0	1.0	0.0	18.05
	1.0	1.0	1.0	0.0	161.38
Hotel	0.2855	0.1211	0.1696	0.0626	1.00
	0.4223	0.3211	0.2845	0.0536	3.27
Moral	1.0	1.0	1.0	0.0	0.85
	1.0	1.0	1.0	0.0	27.41
SemanticBible	0.4300	0.1390	0.2038	0.0311	5.20
	0.4408	0.7729	0.5524	0.0203	8.44
UBA	1.0	0.9453	0.9683	0.0005	0.44
	1.0	0.9600	0.9778	0.0004	1.46
FuzzyWine	0.9267	0.9500	0.9310	0.0110	1.25
	0.9267	0.9500	0.9310	0.0110	35.1

seconds) to execute each fold. For each ontology, the first line refers to the performance of FOIL-$\mathcal{DL}$, while the second line refers to the performance of pFOIL-$\mathcal{DL}$, which appears to be more effective than FOIL-$\mathcal{DL}$. However, the latter is definitely faster than the former. More detailed data about the experiment can be found in the download.

5. CONCLUSIONS

In this work we addressed the problem to induce (possibly fuzzy) $\mathcal{EL}(D)$ concept descriptions that describe sufficient conditions of being an individual instance of a target class. To do so, we described pFOIL-$\mathcal{DL}$, which stems from a previous method, called FOIL-$\mathcal{DL}$ [39]. But unlike FOIL-$\mathcal{DL}$, in pFOIL-$\mathcal{DL}$ the goodness of a candidate is evaluated in terms of the *ensemble* of so far learned candidates. Both, FOIL-$\mathcal{DL}$ and pFOIL-$\mathcal{DL}$ have been submitted to a preliminary experimentation to measure their effectiveness. Our experiments seem to indicate that pFOIL-$\mathcal{DL}$ performs better so far, but comes at a price of a higher execution time. We plan to conduct a more extensive experimentation and look at various other learning algorithms as well.

6. REFERENCES

[1] A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. The DL-Lite family and relations. *J. of Artificial Intelligence Research*, 36:1–69, 2009.

[2] F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proc. of IJCAI*, pages 364–369, 2005. Morgan-Kaufmann Publishers.

[3] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.

[4] R. A. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* Addison-Wesley Longman Publishing Co., Inc., 1999.

[5] H. Blockeel and L. D. Raedt. Top-down induction of first-order logical decision trees. *Artif. Intell.*, 101(1-2):285–297, 1998.

[6] S. Bloehdorn and Y. Sure. Kernel methods for mining instance data in ontologies. In *Proc. of ISWC*, LNCS 4825, pages 58–71. Springer Verlag, 2007.

[7] F. Bobillo and U. Straccia. fuzzyDL: An expressive fuzzy description logic reasoner. In *2008 Int. Conf. on Fuzzy Systems (FUZZ-08)*, pages 923–930. IEEE Computer Society, 2008.

[8] M. Cerami and U. Straccia. On the (un)decidability of fuzzy description logics under Łukasiewicz t-norm. *Information Sciences*, 227:1–21, 2013.

[9] M. Chitsaz, K. Wang, M. Blumenstein, and G. Qi. Concept learning for $\mathcal{EL}^{++}$; by refinement and reinforcement. In *Proc. of PRICAI*, pages 15–26, Berlin, Heidelberg, 2012. Springer-Verlag.

[10] C. d'Amato and N. Fanizzi. Lazy learning from terminological knowledge bases. In *Proc. of ISMIS* 2006, LNCS 4203, pages 570–579. Springer Verlag, 2006.

[11] C. d'Amato, N. Fanizzi, and F. Esposito. Query answering and ontology population: An inductive approach. In *Proc. of ESWC*, LNCS 5021, pages 288–302. Springer Verlag, 2008.

[12] C. d'Amato, N. Fanizzi, B. Fazzinga, G. Gottlob, and T. Lukasiewicz. Ontology-based semantic search on the web and its combination with the power of inductive reasoning. *Annals of Mathematics and Artificial Intelligence*, 65(2-3):83–121, 2012.

[13] M. Drobics, U. Bodenhofer, and E.-P. Klement. Fs-Foil: an inductive learning method for extracting interpretable fuzzy descriptions. *Int. J. of Approximate Reasoning*, 32(2-3):131–152, 2003.

[14] N. Fanizzi. Concept induction in description logics using information-theoretic heuristics. *Int. J. Semantic Web Inf. Syst.*, 7(2):23–44, 2011.

[15] N. Fanizzi, C. d'Amato, and F. Esposito. DL-FOIL concept learning in description logics. In *Proc. of ILP*, LNCS 5194, pages 107–121. Springer, 2008.

[16] N. Fanizzi, C. d'Amato, and F. Esposito. Induction of classifiers through non-parametric methods for approximate classification and retrieval with ontologies. *Int. J. Semantic Computing*, 2(3):403–423, 2008.

[17] N. Fanizzi, C. d'Amato, and F. Esposito. Learning with kernels in description logics. In *Proc. of ILP*, LNCS 5194, pages 210–225. Springer Verlag, 2008.

[18] N. Fanizzi, C. d'Amato, and F. Esposito. Reduce: A reduced coulomb energy network method for approximate classification. In *Proc. of ESWC*, LNCS 5554, pages

323–337. Springer, 2009.

[19] N. Fanizzi, C. d'Amato, and F. Esposito. Induction of concepts in web ontologies through terminological decision trees. In *Proc. of ECML* LNCS 6321, Part I, pages 442–457. Springer Verlag, 2010.

[20] N. Fanizzi, C. d'Amato, and F. Esposito. Towards the induction of terminological decision trees. In *Proc. of ACM SAC*, pages 1423–1427, 2010. ACM.

[21] N. Fanizzi, C. d'Amato, and F. Esposito. Induction of robust classifiers for web ontologies through kernel machines. *J. Web Sem.*, 11:1–13, 2012.

[22] N. Fanizzi, C. d'Amato, F. Esposito, and P. Minervini. Numeric prediction on OWL knowledge bases through terminological regression trees. *Int. J. Semantic Computing*, 6(4):429–446, 2012.

[23] P. Hájek. *Metamathematics of Fuzzy Logic*. Kluwer, 1998.

[24] S. Hellmann, J. Lehmann, and S. Auer. Learning of OWL class descriptions on very large knowledge bases. *Int. J. Semantic Web Inf. Syst.*, 5(2):25–48, 2009.

[25] L. Iannone, I. Palmisano, and N. Fanizzi. An algorithm based on counterfactuals for concept learning in the semantic web. *Applied Intelligence*, 26(2):139–159, Apr. 2007.

[26] J. Iglesias and J. Lehmann. Towards integrating fuzzy logic capabilities into an ontology-based inductive logic programming framework. In *Proc. of ISDA*. IEEE Press, 2011.

[27] G. J. Klir and B. Yuan. *Fuzzy sets and fuzzy logic: theory and applications*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1995.

[28] S. Konstantopoulos and A. Charalambidis. Formulating description logic learning as an inductive logic programming task. In *Proc. FUZZ-IEEE*, pages 1–7. IEEE Press, 2010.

[29] N. Landwehr, K. Kersting, and L. D. Raedt. Integrating Naïve Bayes and FOIL. *Journal of Machine Learning Research*, 8:481–507, 2007.

[30] J. Lehmann. Hybrid learning of ontology classes. In *Proc. of MLDM*, LNCS 4571, pages 883–898. Springer Verlag, 2007.

[31] J. Lehmann. DL-learner: Learning concepts in description logics. *Journal of Machine Learning Research*, 10:2639–2642, 2009.

[32] J. Lehmann, S. Auer, L. Bühmann, and S. Tramp. Class expression learning for ontology engineering. *J. Web Sem.*, 9(1):71–81, 2011.

[33] J. Lehmann and C. Haase. Ideal downward refinement in the *EL*\mathcal{EL} description logic. In *Proc. of ILP*. Revised Papers, pages 73–87, 2009.

[34] J. Lehmann and P. Hitzler. Foundations of Refinement Operators for Description Logics. In *Proc. of ILP*, LNAI 4894, pages 161–174. Springer, 2008.

[35] J. Lehmann and P. Hitzler. Concept learning in description logics using refinement operators. *Machine Learning*, 78(1-2):203–250, 2010.

[36] F. A. Lisi and U. Straccia. Towards learning fuzzy dl inclusion axioms. In *9th International Workshop on Fuzzy Logic and Applications (WILF-11)*, volume 6857 of *Lecture Notes in Computer Science*, pages 58–66, Berlin, 2011. Springer Verlag.

[37] F. A. Lisi and U. Straccia. Dealing with incompleteness and vagueness in inductive logic programming. In *Proc. of CILC-13*, CEUR 1068, pages 179–193. CEUR Electronic Workshop Proceedings, 2013.

[38] F. A. Lisi and U. Straccia. A logic-based computational method for the automated induction of fuzzy ontology axioms. *Fundamenta Informaticae*, 124(4):503–519, 2013.

[39] F. A. Lisi and U. Straccia. A system for learning GCI axioms in fuzzy description logics. In *Proc. DL-13*, CEUR 1014, pages 760–778. CEUR Electronic Workshop Proceedings, 2013.

[40] T. Lukasiewicz and U. Straccia. Managing uncertainty and vagueness in description logics for the semantic web. *Journal of Web Semantics*, 6:291–308, 2008.

[41] P. Minervini, C. d'Amato, and N. Fanizzi. Learning probabilistic description logic concepts: Under different assumptions on missing knowledge. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, SAC '12, pages 378–383, New York, NY, USA, 2012. ACM.

[42] P. Minervini, C. d'Amato, and N. Fanizzi. Learning terminological bayesian classifiers - A comparison of alternative approaches to dealing with unknown concept-memberships. In *Proc. of CILC*, CEUR 857, pages 191–205, CEUR Electronic Workshop Proceedings, 2012.

[43] P. Minervini, C. d'Amato, N. Fanizzi, and F. Esposito. Transductive inference for class-membership propagation in web ontologies. In *Proc. of ESWC*, LNCS 7882, pages 457–471. Springer Verlag, 2013.

[44] P. Minervini, N. Fanizzi, C. d'Amato, and F. Esposito. Rank prediction for semantically annotated resources. In *Proc. of ACM SAC*, pages 333–338, 2013. ACM.

[45] B. Motik and R. Rosati. A faithful integration of description logics with logic programming. In *Proc. of IJCAI*, pages 477–482, 2007. Morgan Kaufmann Publishers Inc.

[46] M. Nickles and A. Rettinger. Interactive relational reinforcement learning of concept semantics. *Machine Learning*, 94(2):169–204, 2014.

[47] OWL 2 Web Ontology Language Document Overview. `http://www.w3.org/TR/2009/REC-owl2-overview-20091027/`. W3C, 2009.

[48] OWL 2 Web Ontology Language Profiles: OWL 2 EL. `http://www.w3.org/TR/2009/REC-owl2-profiles-20091027/#OWL_2_EL`. W3C, 2009.

[49] J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.

[50] A. Rettinger, U. Lösch, V. Tresp, C. d'Amato, and N. Fanizzi. Mining the semantic web - statistical learning for next generation knowledge bases. *Data Min. Knowl. Discov.*, 24(3):613–662, 2012.

[51] A. Rettinger, M. Nickles, and V. Tresp. Statistical relational learning with formal ontologies. In *Proc. of ECML*, LNCS 5782, Part II, pages 286–301. Springer Verlag, 2009.

[52] G. Rizzo, C. d'Amato, N. Fanizzi, and F. Esposito. Assertion prediction with ontologies through evidence combination. In *Proc. of URSW*, LNCS 7123, pages 282–299. Springer Verlag, 2013.

[53] G. Rizzo, C. d'Amato, N. Fanizzi, and F. Esposito. Towards evidence-based terminological decision trees. In *Proc. of IPMU*, Part I, CCIS 442, pages 36–45. Springer, 2014.

[54] G. Rizzo, N. Fanizzi, C. d'Amato, and F. Esposito. Prediction of class and property assertions on OWL ontologies through evidence combination. In *Proc. of WIMS*, pages 45:1–45:9, 2011. ACM.

[55] M. Serrurier and H. Prade. Improving expressivity of inductive logic programming by learning different kinds of fuzzy rules. *Soft Computing*, 11(5):459–466, 2007.

[56] D. Shibata, N. Inuzuka, S. Kato, T. Matsui, and H. Itoh. An induction algorithm based on fuzzy logic programming. In *Proc. of PAKDD*, LNCS, pages 268–273. Springer, 1999.

[57] U. Straccia. Description logics with fuzzy concrete domains. In *Proc. of UAI*, pages 559–567, 2005. AUAI Press.

[58] U. Straccia. *Foundations of Fuzzy Logic and Semantic Web Languages*. CRC Studies in Informatics Series. Chapman & Hall, 2013.

[59] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.