Combining Prototypes in Description Logics with Prototype Descriptions: First Steps

Gabriele Sacco[1,2], Loris Bozzato[3] and Oliver Kutz[2]

[1]Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy
[2]Free University of Bozen-Bolzano, Piazza Università 1, 39100, Bolzano, Italy
[3]DiSTA - Università dell'Insubria, Via O. Rossi 9, 21100 Varese, Italy

Abstract

Defeasible reasoning has been studied as a fundamental component of common-sense reasoning: different logic based solutions have been proposed for its formal modelling. We recently proposed a non-monotonic Description Logic (DL) framework, DLs with Prototype Descriptions, which deals with defeasible reasoning by combining aspects from prototype theory, weighted logics, and justifiable exceptions. Prototype descriptions are weighted characterizations of concepts that denote the typical features of prototype members in a quantitative way.

In this paper, we further develop the notion of prototype description by suggesting different directions for combining prototypes and features. We consider a way to reason with conjunction of prototypes in defeasible inclusions. On the other hand, we discuss how complex features can influence the definition of weights in prototype descriptions. While preliminary, the directions we outline are worth exploring in order to increase the flexibility of our formalisms, as shown through examples.

Keywords

Non-monotonic logic, Typicality, Prototype theory, Description Logics, Answer Set Programming

1. Introduction

An important aspect for modelling common-sense reasoning is *defeasible reasoning*: this represents the capability of a system to reason with general statements that may admit *exceptions*. The interest in ways to model defeasible reasoning dates back to the earlier years of Artificial Intelligence [1, 2]. In logical formalisms, these have been studied in non-monotonic logics: in the area Knowledge Representation, many approaches for representing non-monotonicity have been proposed also in the context of Descriptions Logics (DL), like e.g. [3, 4].

In this regard, we recently introduced in [5] a formal approach for defeasible reasoning in DLs, namely *DLs with Prototype Descriptions*, which satisfies desiderata extracted from the studies on Generics [6] and insights from the philosophical and cognitive research on phenomena related to notions of exceptions and defeasibility. The formalization of our approach combines ideas from prototype theory, weighted DLs [7] and defeasible DLs with Justifiable Exceptions [8]. The main component of the framework is the definition of *prototype description*, weighted characterisations of concepts denoting the typical features of its members. Prototype descriptions provide a way to include into DL knowledge bases quantitative similarity-based numerical characterisations of prototype instances. These weights are then used to compute a "typicality score" of instances that is considered in defining a preference over DL models. The framework has then been further developed in [9, 10] to include different solutions to define model preference, propose an ASP encoding [11] and study its formal properties.

In this paper, we outline our current research direction to further develop the framework: we are interested in understanding the possibilities to *combine concepts* in order to offer more complex and refined representations of prototypes' properties. In particular, there are two directions in which such combination can take place: *(i).* TBox axioms asserting properties over *conjunctions of prototypes*; *(ii).* prototype descriptions using *combinations of features* to denote the typical prototype instances. These

CILC 2026: The 41st Italian Conference on Computational Logic, June 23–25, 2026, Ferrara, Italy
gsacco@fbk.eu (G. Sacco); loris.bozzato@uninsubria.it (L. Bozzato); oliver.kutz@unibz.it (O. Kutz)
0000-0001-5613-5068 (G. Sacco); 0000-0003-1757-9859 (L. Bozzato); 0000-0003-1517-7354 (O. Kutz)

combinations ask for an extension of our current definitions of the framework: the first needs a way to compute scores for typical elements of the conjunct prototypes (starting from the scores associated to the "atomic" prototypes), while the second asks for a method to consistently associate weights to features combinations (considering the weights of the base features).

These study directions are worth exploring in order to increase the flexibility of our framework: on the one hand, the current definitions of prototype descriptions and axioms may be too limited to capture some needed relations across prototypes (e.g. emerging from data); on the other hand, such combinations allow us to link our work to studies on concept combinations [12] and show further non-monotonicity properties on our formalization.

In the following, we first provide a brief summary of definitions for DLs with Prototype Descriptions as defined in [10, 13]. We then discuss possible directions to extend this formalisms with combination of prototypes and combinations of features, explaining the intuitions by means of examples.

2. DLs with Prototype Descriptions

We distinguish two parts in knowledge bases: *(i).* a DL knowledge base representing the knowledge of interest: the knowledge base can include defeasible TBox axioms about specific base concepts called *prototypes* and ABox assertions about prototype instances and their features; *(ii).* an additional set containing *prototype descriptions*, weighted characterizations of prototypes, expressing the "degree of typicality" of the features of their instances.

The following definitions are independent from the DL language used for representing the main knowledge base: we consider a fixed concept language $\mathscr{L}_\Sigma$ based on a DL signature Σ with disjoint and non-empty sets NC of *concept names*, NR of *role names*, and NI of *individual names*. We identify a subset of the concept names $\mathrm{NP} \subseteq \mathrm{NC}$ as denoting *prototype names*. For simplicity, we call *general concepts* the concepts composed only of concepts in $\mathrm{NC} \setminus \mathrm{NP}$ (i.e., atomic or complex concepts which do not include prototypes names).

The features associated with prototypes together with the degree of their importance are given in *prototype descriptions*.

Definition 1 (Positive prototype description)**.** *Let $P \in \mathrm{NP}$ be a prototype name, let $C_1, \dots, C_m$ be general concepts of $\mathscr{L}_\Sigma$ and let $\overline{w} = (w_1, \dots, w_m) \in \mathbb{Q}^m$ be a weight vector of rational numbers, where for every $i \in \{1, \dots, m\}$ we have $w_i > 0$. Then, the expression $P(C_1 : w_1, \dots, C_m : w_m)$ is called a* (positive) prototype description *for P.*

Intuitively, the weights associated with features can be combined to compute a score denoting the degree of typicality of an instance w.r.t. the prototype.

In the knowledge part of the KB, we can use prototype names in DL axioms to describe properties of the members of such classes.

Definition 2 (Prototype axiom)**.** *A concept inclusion of the type $P \sqsubseteq D$ is a* prototype axiom *of $\mathscr{L}_\Sigma$ if $P \in \mathrm{NP}$ and D is a general concept of $\mathscr{L}_\Sigma$.*

Intuitively, these axioms are not absolute and can be "overridden" by prototype instances (cf. *defeasible axioms* in [8]), also depending on the "degree of membership" of the individual to the given prototype (i.e., the satisfaction of its features).

Thus, we consider knowledge bases which can contain prototype axioms and which are enriched with an accessory KB, the PBox $\mathscr{P}$, providing prototype descriptions.

Definition 3 (Prototyped Knowledge Base, PKB)**.** *A* prototyped knowledge base (PKB) *in language $\mathscr{L}_\Sigma$ is a triple $\mathfrak{K} = \langle \mathscr{T}, \mathscr{A}, \mathscr{P} \rangle$ where:*

- *$\mathscr{T} = T_P \uplus T_C$ is a DL TBox consisting of concept inclusion axioms of the form $C \sqsubseteq D$, where T_P is the set of prototype axioms and T_C is the set of general concept inclusions based on general concepts;*

– $\mathcal{A} = A_P \uplus A_C$ *is a set of ABox assertions, where* A_P *is the set of prototype assertions (of the form* $P(a)$ *with* $P \in$ NP *and* $a \in$ NI*) and* A_C *is the set of ABox assertions for general concepts and roles;*

– $\mathcal{P}$ *is a set of prototype descriptions, exactly one for each prototype name* $P \in$ NP *appearing in* T_P*.*

We present an example to clarify the notions and syntax introduced thus far.

Example 1. *Consider the following* prototyped knowledge base $K = \langle \mathcal{T}, \mathcal{A}, \mathcal{P} \rangle$ *where* NC = {Dog, Wolf, Trusted, hasLegs, livesInWoods, isTamed, hasCollar, Hunts, livesInPack, livesInHouse} *and* NP = {Dog, Wolf}*:*

$$\mathcal{T} = \{\, \texttt{Dog} \sqsubseteq \texttt{Trusted},\ \texttt{Wolf} \sqsubseteq \neg\texttt{Trusted}, \texttt{Dog} \sqsubseteq \texttt{hasLegs},\ \texttt{Wolf} \sqsubseteq \texttt{hasLegs} \,\},$$

$$\begin{aligned}\mathcal{A} = \{\, &\texttt{Dog(balto)}, \texttt{Wolf(balto)},\ \texttt{Dog(cerberus)},\\ &\texttt{livesInWoods(balto)},\ \texttt{hasLegs(balto)},\ \texttt{isTamed(balto)},\\ &\neg\texttt{Trusted(cerberus)}\},\end{aligned}$$

$$\begin{aligned}\mathcal{P} = \{\, &\texttt{Wolf(livesInWoods : 10, hasLegs : 4, livesInPack : 8, Hunts : 11)},\\ &\texttt{Dog(hasCollar : 33, livesInHouse : 22, hasLegs : 11, isTamed : 44)}\}\end{aligned}$$

The weights in prototype descriptions are meant to represent how relevant the associated features are for a typical instance of the prototype. Instances satisfying features with larger weights are interpreted as "more typical" members of the prototype concept. Below we give a semantics for this kind of PKB which will entail and justify the conclusion that balto *is a trusted dog which is a wolf, without being inconsistent, and that* cerberus *is an exceptional dog with respect to the property of dogs of being trusted.*

Note that the conflict regarding balto *has the same structure as the so called* Nixon diamond *[14].* ◇

Next, we define the semantic mechanism through which we manage conflicts in order to avoid treating exceptions as contradictions, making the system inconsistent. The semantics of PKBs is based on standard interpretations for the underlying DL $\mathcal{L}_\Sigma$. In fact, *interpretations* of a PKB are DL interpretations for its knowledge base part $\langle \mathcal{T}, \mathcal{A} \rangle$.

Note that we are not giving a DL interpretation to the prototype description expressions in $\mathcal{P}$. However, we need to introduce additional semantic structure to manage exceptions to prototype axioms in T_P, exploiting the prototype description expressions in $\mathcal{P}$. We consider the notion of axiom instantiation as defined in [8]: intuitively, for an axiom $\alpha \in L_\Sigma$ the *instantiation* of α with $e \in$ NI, written $\alpha(e)$, is the specialisation of α to e.

Definition 4 (Exception assumptions and clashing sets)**.** *An* exception assumption *is a pair* $\langle \alpha, e \rangle$ *where* $\alpha \in T_P$ *is a prototype axiom,* $e \in$ NI *is an individual name appearing in* $\mathcal{A}$ *and such that* $\alpha(e)$ *is an axiom instantiation of* α*. A* clashing set *for* $\langle \alpha, e \rangle$ *is a satisfiable set* $S_{\langle \alpha, e \rangle}$ *of ABox assertions s.t.* $S_{\langle \alpha, e \rangle} \cup \{\alpha(e)\}$ *is unsatisfiable.*

Intuitively, an exception assumption $\langle P \sqsubseteq D, e \rangle$ states that we assume that e is an exception to the prototype axiom $P \sqsubseteq D$ in a given interpretation. Then, the fact that a clashing set $S_{\langle P \sqsubseteq D, e \rangle}$ for $\langle P \sqsubseteq D, e \rangle$ is verified by such an interpretation gives a "justification" of the validity of the assumption of overriding in terms of ABox assertions. This intuition is reflected in the definition of models: we first extend interpretations with a set of exception assumptions.

Definition 5 (χ-interpretation)**.** *A* χ-interpretation *is a structure* $\mathcal{I}_\chi = \langle \mathcal{I}, \chi \rangle$ *where* $\mathcal{I}$ *is an interpretation and* χ *is a set of exception assumptions.*

Then, χ-models for a PKB $\mathfrak{K}$ are those χ-interpretations that verify "strict" axioms in $T_C \cup \mathcal{A}$ and defeasibly apply prototype axioms in T_P (excluding the exceptional instances in χ).

Definition 6 (χ-model)**.** *Given a PKB* $\mathfrak{K}$*, a* χ*-interpretation* $\mathcal{I}_\chi = \langle \mathcal{I}, \chi \rangle$ *is a* χ-model *for* $\mathfrak{K}$ *(denoted* $\mathcal{I}_\chi \models \mathfrak{K}$*), if the following holds:*

(i) for every $\alpha \in T_C \cup \mathcal{A}$ *of* $\mathcal{L}_\Sigma$*,* $\mathcal{I} \models \alpha$*;*

(ii) for every $\alpha = P \sqsubseteq D \in T_P$, if $\langle \alpha, d \rangle \notin \chi$, then $\mathcal{I} \models \alpha(d)$.

Two DL interpretations $\mathcal{I}_1$ and $\mathcal{I}_2$ are NI-*congruent*, if $c^{\mathcal{I}_1} = c^{\mathcal{I}_2}$ holds for every $c \in$ NI. This extends to χ-interpretations $\mathcal{I}_\chi = \langle \mathcal{I}, \chi \rangle$ by considering interpretations $\mathcal{I}$. Intuitively, we say that a χ-interpretation is justified if all of its exception assumptions have a clashing set that is verified by the interpretation (and all its NI-congruent interpretations).

Definition 7 (Justifications)**.** *We say that $\langle \alpha, e \rangle \in \chi$ is* justified *for a χ-model $\mathcal{I}_\chi$, if some clashing set $S_{\langle \alpha, e \rangle}$ exists such that, for every $\mathcal{I}'_\chi = \langle \mathcal{I}', \chi \rangle$ of $\mathfrak{K}$ that is* NI*-congruent with $\mathcal{I}_\chi$, it holds $\mathcal{I}' \models S_{\langle \alpha, e \rangle}$.*

A χ-model $\mathcal{I}_\chi$ of a PKB $\mathfrak{K}$ is justified*, if every $\langle \alpha, e \rangle \in \chi$ is justified in $\mathfrak{K}$.*

We consider the notion of logical consequence from justified χ-models (i.e. axioms that are valid in all χ-models that are justified): we write $\mathfrak{K} \models_J \alpha$ if $\mathcal{I}_\chi \models \alpha$ for every justified χ-model $\mathcal{I}_\chi$ of $\mathfrak{K}$.

A second part of the semantics takes care of defining a preference over exceptions in case of conflicts between different prototype axioms. The main intuition of prototype descriptions is that each individual which is an instance of a prototype is associated with a score which denotes the "degree of typicality" of the individual with respect to the concept described by the prototype. Ideally, the prototype score of an individual allows us to determine preferences over models: for an individual, axioms on prototypes with higher score are preferred to the ones on lower scoring prototypes; thus the measure needs to be comparable across different prototypes.

Given the set NP($\mathcal{P}$) of prototype names of $P \in$ NP appearing in $\mathcal{P}$, a family of *prototype score functions* $\{f_P\}_{P \in \mathrm{NP}(\mathcal{P})}$ is composed of functions $f_P : \mathrm{NI} \to \mathbb{R}$ for each prototype name P appearing in $\mathcal{P}$ such that every function of the family has range in a fixed interval $[x, ..., y] \subseteq \mathbb{R}$.

In general, a *preference* between exception assumption sets is some relation between sets χ for $\mathfrak{K}$, denoted $\chi_1 \geq \chi_2$. Given two χ-interpretations $\mathcal{I}^1_\chi = \langle \mathcal{I}^1, \chi_1 \rangle$ and $\mathcal{I}^2_\chi = \langle \mathcal{I}^2, \chi_2 \rangle$, we then say that $\mathcal{I}^1_\chi$ is *preferred* to $\mathcal{I}^2_\chi$ (denoted $\mathcal{I}^1_\chi \geq \mathcal{I}^2_\chi$) if $\chi_1 \geq \chi_2$.

Finally, we define the notion of PKB model as a minimal justified model for the PKB.

Definition 8 (PKB model)**.** *An interpretation $\mathcal{I}$ is a* PKB model *of $\mathfrak{K}$ (denoted $\mathcal{I} \models \mathfrak{K}$) if*

- *$\mathfrak{K}$ has some justified χ-model $\mathcal{I}_\chi = \langle \mathcal{I}, \chi \rangle$.*
- *there exists no justified $\mathcal{I}'_\chi = \langle \mathcal{I}', \chi' \rangle$ that is preferred to $\mathcal{I}_\chi$.*

The consequence from PKB models of $\mathfrak{K}$ (denoted $\mathfrak{K} \models \alpha$) characterises the "preferred" consequences of the PKB, on the basis of the degree of typicality of instances.

The above definitions are provided for any score function and preference: a simple score function can be defined by considering the features that are inferable from the KB (in all justified models):

Definition 9 (Model independent prototype score)**.** *Given a prototype description $P(C_1 : w_1, ..., C_m : w_m)$, we define the* (model independent) score function *$score_P : \mathrm{NI} \to \mathbb{R}$ for prototype P as: $score_P(a) = \sum_{\mathfrak{K} \models_J C_i(a)} w_i$*

This measure, however, depends on the value interval over which the prototype weights have been defined: in order to compare the score of an individual with scores relative to other prototypes, this value needs to be normalised. We do so by computing the maximum score max_P and minimum score min_P for all prototypes.

The maximum score max_P denotes the score of the maximum value of $score_P$ obtainable from the weights of consistent subset of features of P. Formally, given a prototype description $P(C_1 : w_1, \ldots, C_m : w_m)$, let $\mathbf{S}_P$ be the set of sets $S_P \subseteq \{C_1, \ldots, C_n\}$ s.t. $S_P \cup \mathfrak{K}$ is consistent. Then: $max_P = \max(\sum_{C_i \in S_p} w_i \mid S_P \in \mathbf{S}_P)$. The minimal score min_P denotes the sum of the weights for "unavoidable" features, namely those that are strictly implied by the membership to the prototype concept. Formally: $min_P = \sum_{\mathfrak{K} \models_J P \sqsubseteq C_i} w_i$. A normalised score function $nscore_P$ can be derived from $score_P$ as: $nscore_P(a) = (score_P(a) - min_P)/(max_P - min_P)$.

Note that with such a normalisation we obtain a family of prototype score functions with range in the same interval $[0, 1]$, allowing for comparison of prototype scores on the same individual.

We can then define a simple preference using the model independent score as defined above: we thus prefer justified χ-models where the *exceptions* appear for elements of the *lower* scoring prototypes.

Definition 10 (Preference MIP)**.** *$\chi_1 \geq \chi_2$ if, for every $\langle P \sqsubseteq D, e\rangle \in \chi_1 \setminus \chi_2$ such that there exists a $\langle Q \sqsubseteq E, e\rangle \in \chi_2 \setminus \chi_1$ with $\mathfrak{K} \cup \{D(e), E(e)\}$ unsatisfiable, it holds that $nscore_P(e) < nscore_Q(e)$.*

The intuition behind the condition "$\mathfrak{K} \cup \{D(e), E(e)\}$ unsatisfiable" is that we want to make the comparison between the exception assumptions that are directly in conflict.

Example 2. *Considering the PKB reported in the example above, assume to have two PKB interpretations $\mathcal{I}^1$ and $\mathcal{I}^2$ associated respectively with the following two sets of exception assumptions:*

$$\chi_1 = \{\langle \texttt{Wolf} \sqsubseteq \neg\texttt{Trusted}, \texttt{balto}\rangle, \langle \texttt{Dog} \sqsubseteq \texttt{Trusted}, \texttt{cerberus}\rangle\}$$
$$\chi_2 = \{\langle \texttt{Dog} \sqsubseteq \texttt{Trusted}, \texttt{balto}\rangle, \langle \texttt{Dog} \sqsubseteq \texttt{Trusted}, \texttt{cerberus}\rangle\}$$

*We have now two χ-interpretations corresponding to $\langle \mathcal{I}^1, \chi_1\rangle$ and $\langle \mathcal{I}^2, \chi_2\rangle$. Assuming that they are also χ-*models*, we can check if the two are also* justified*. Since, the exception assumptions have the following clashing sets, respectively* $\{\texttt{Wolf}(\texttt{balto}), \texttt{Trusted}(\texttt{balto}), \texttt{Dog}(\texttt{cerberus}), \neg\texttt{Trusted}(\texttt{cerberus})\}$ *for the exception assumptions in χ_1 and* $\{\texttt{Dog}(\texttt{balto}), \neg\texttt{Trusted}(\texttt{balto}), \texttt{Dog}(\texttt{cerberus}), \neg\texttt{Trusted}(\texttt{cerberus})\}$ *for those in χ_2, they are both justified.*

In order to decide which model is preferred, we need to compute the prototype scores for balto *and for* cerberus*: we have $score_{\texttt{Wolf}}(\texttt{balto}) = 14$, $score_{\texttt{Dog}}(\texttt{balto}) = 55$, $score_{\texttt{Dog}}(\texttt{cerberus}) = 11$. After normalization, we obtain: $nscore_{\texttt{Dog}}(\texttt{balto}) \approx 0.4$, $nscore_{\texttt{Wolf}}(\texttt{balto}) \approx 0.3$, $nscore_{\texttt{Dog}}(\texttt{cerberus}) = 0$. Then, $score_{\texttt{Wolf}}(\texttt{balto}) < score_{\texttt{Dog}}(\texttt{balto})$ and, since $\langle \texttt{Dog} \sqsubseteq \texttt{Trusted}, \texttt{cerberus}\rangle$ is present in both χ_1 and χ_2 so it does not influence the preference order, then we can conclude that $\chi_1 \geq \chi_2$. Hence, we obtain that the preferred model, i.e. the PKB model, is $\mathcal{I}^1$ where* balto *is an exception to* $\texttt{Wolf} \sqsubseteq \neg\texttt{Trusted}$ *and* cerberus *is an exception to* $\texttt{Dog} \sqsubseteq \texttt{Trusted}$*. Consequently, it holds that $\mathfrak{K} \models \texttt{Trusted}(\texttt{balto})$ and $\mathfrak{K} \models \neg\texttt{Trusted}(\texttt{cerberus})$.* ◇

3. Combining Prototypes and Features

In the following sections, we discuss two possible refinements to the presented framework: the first concerns the use of complex of prototypes in prototype axioms, the second proposes restrictions in the use of dependent features in prototype descriptions.

3.1. Conjunction of Prototypes in Prototype Axioms

In the current formalization of PKBs, we only allow for atomic prototypes to appear on the left of prototype axioms, that is only axioms of the form $P \sqsubseteq C$ with $P \in \text{NP}$ are permitted. Of course, it would be interesting to allow for complex concepts to appear on the left-hand side of such inclusions, so to provide more articulated axioms over prototypes.

While complex antecedents can be allowed by preserving the interpretation of prototype axioms as defeasible axioms (as shown in [8]), the problem that arises is how to manage the model preference in the case of complex prototypes. We note that, in general, this can be solved by treating the case of such combined prototypes in formulating the prototype score function: that is, the score of an instance for the combined prototype can be obtained from the score of the atomic prototypes with some aggregation criteria.

For example, a first combination we can consider is the *conjunction* of prototypes: that is, we can allow prototype axioms of the kind $P_1 \sqcap P_2 \sqsubseteq C$ (and, more in general, $P_1 \sqcap \cdots \sqcap P_n \sqsubseteq C$). The problem then is how to associate a score to the complex prototype given by the conjunction. A simple possibility

for defining such score function is to consider the scores of the conjuncts and equally combine their (normalized) score. Namely, one can define:

$$nscore_{P_1 \sqcap \cdots \sqcap P_n}(a) = \frac{\sum_{i=1}^{n} nscore_{P_i}(a)}{n}$$

Even with this naive score combination function, we can already solve situations like the one shown in the following example.

Example 3. *Consider the following PKB, in which we introduce a conjunction in the previous prototype axioms:*

$$\mathcal{T} = \{\texttt{Dog} \sqcap \texttt{Puppy} \sqsubseteq \texttt{Trusted},\ \texttt{Wolf} \sqcap \texttt{Puppy} \sqsubseteq \neg\texttt{Trusted}, \texttt{Dog} \sqsubseteq \texttt{hasLegs},\ \texttt{Wolf} \sqsubseteq \texttt{hasLegs}\},$$

$$\begin{aligned}\mathcal{A} = \{&\texttt{Dog(balto)}, \texttt{Wolf(balto)}, \texttt{Puppy(balto)},\\ &\texttt{livesInWoods(balto)},\ \texttt{hasLegs(balto)},\ \texttt{isTamed(balto)},\\ &\texttt{isFluffy(balto)}\},\end{aligned}$$

$$\begin{aligned}\mathcal{P} = \{&\texttt{Wolf(livesInWoods : 10, hasLegs : 4, livesInPack : 8, Hunts : 11)},\\ &\texttt{Dog(hasCollar : 33, livesInHouse : 22, hasLegs : 11, isTamed : 44)},\\ &\texttt{Puppy(isFluffy : 5, isSmall : 5)}\}\end{aligned}$$

We obtain the following normalized scores for the atomic prototypes: $nscore_{\texttt{Dog}}(\texttt{balto}) \approx 0.4$, $nscore_{\texttt{Wolf}}(\texttt{balto}) \approx 0.3$, $nscore_{\texttt{Puppy}}(\texttt{balto}) = 0.5$. *Thus, if we consider the combined score defined above for prototypes conjunction we obtain:* $nscore_{\texttt{Dog}\sqcap\texttt{Puppy}}(\texttt{balto}) = 0.4 + 0.5/2 = 0.45$ *and* $nscore_{\texttt{Wolf}\sqcap\texttt{Puppy}}(\texttt{balto}) = 0.3 + 0.5/2 = 0.4$. *We still have that* `balto` *is a more typical* `Dog` *than a* `Wolf`*: thus, also in this case, the first axiom is preferred and we can obtain* $\mathfrak{K} \models \texttt{Trusted(balto)}$. ◇

Note that in this simple combination of scores, all of the prototypes are given the same "importance" in the combination: if we want to define a priority on the prototypes (e.g., in the order in which they appear in the axiom), the arithmetic mean can be substituted with a weighted mean (e.g., with a weight that depends on the order).

The definition of different methods for combining prototypes leads to a direct comparison to the works in *Concept combinations* [12]: we need to further investigate these aspects, both to understand how to relate to them from a formal point of view, but also to understand the meaning of combinations within the field of cognitive science. We note that a definition for the combination of prototypes on the left-hand side of prototype axioms is also needed to prove in our context some of the *KLM properties* [15] (in particular, *Or* and *Cautious Monotonicity*) that, for the moment, can not be formulated in our framework [10].

3.2. Combination of Features in Prototype Descriptions

Another current limitation of our framework is the assumption of independence of features in the definition of a prototype description: for example, if a prototype description has features `hasTail` and `hasWhiteTail`, there is no definite relation across the weights of such related features.

We note that a way to characterize such relations is to impose a restriction on the use of the weights. A first sketch of such restrictions can be the following: given A, B features appearing in a prototype description for P with weights w_1 and w_2, if $T_C \models A \sqsubseteq B$, then $w_1 < w_2$. The idea is that, since being A implies being B, then the weights of the two are added up when computing the score for an instance having the more specific feature. Note that this also directs how complex features obtained by conjunction and disjunction of concepts behave in this context: since $A \sqsubseteq A \sqcup B$ and $B \sqsubseteq A \sqcup B$, then the weights of A and B need to be less that the one of their union; since $A \sqcap B \sqsubseteq A$ and $A \sqcap B \sqsubseteq B$, then the score of the intersection of features needs to be less that the one of both features. For disjoint features (A and $\neg A$), then one can assign independent scores: intuitively, this means that an optional feature can have a score in case it is not present in the instance, but can have another score if it is not.

Example 4. *We can consider to describe typical features of a racing bycicle by using complex features*

$$\texttt{RacingBycicle}(\exists \texttt{hasHandle.CurvedHandle} : 5, \exists \texttt{hasHandle.StraightHandle} : 3, \exists \texttt{hasHandle.}\top : 10, \texttt{hasRearlight} : 5, \neg \texttt{hasRearlight} : 2, \dots)$$

Intuitively, since $\exists \texttt{hasHandle.CurvedHandle} \sqsubseteq \exists \texttt{hasHandle.}\top$ *and* $\exists \texttt{hasHandle.StraigntHandle} \sqsubseteq \exists \texttt{hasHandle.}\top$*, then the features have a lower score than the concept they subsume. If an element is recognized to have an handle, then the shape of the handle adds further information (i.e. adds up to the base score) to the more general feature. On the other hand,* `hasRearlight` *can be seen as an optional feature and we assign different typicality weights in case it holds or not.* ◇

Of course, a more precise definition of such relations is needed, also considering the consistency with the common-sense notion of prototypes and typicality we are adopting. The idea of feature combinations has been discussed in the context of perceptron operators: for example, the topic of computing scores for R-successors in the case of weighted DLs with perceptron operators has been discussed in [16] and some complexity results have been presented in [17]. These works can provide some insights in how to manage roles also for the combination of features in our formalism.

4. Conclusions and Future Work

In this paper we introduced the problem of combining prototypes and features in DLs with prototype descriptions, providing some initial insights on these problems by means of examples and defining a direction for their solution.

Indeed, this paper represents only a first step in providing a well-structured solution to such problems: to further develop the presented ideas, it is necessary to include the suggested solutions in the formal definitions of score and prototype description of our framework and show that the framework properties hold for the new score criteria. Introducing a more complex combination of prototypes will also ask for an extension of the current implementation for PKBs via ASP [11], which is limited to the base language of *DL-Lite*$_R$. More in general, in formulating such solutions we need to consider the interpretation of these also from a cognitive perspective, to better fit the general research line that led to the definition of our framework.

Acknowledgments

We acknowledge the financial support through the 'Abstractron' project funded by the Autonome Provinz Bozen - Südtirol (Autonomous Province of Bolzano-Bozen) through the Research Südtirol/Alto Adige 2022 Call.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] C. Strasser, G. A. Antonelli, Non-monotonic Logic, in: E. N. Zalta (Ed.), The Stanford Encyclopedia of Philosophy, Summer 2019 ed., Metaphysics Research Lab, Stanford University, 2019.

[2] J. McCarthy, P. J. Hayes, Some Philosophical Problems from the Standpoint of Artificial Intelligence, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1987, p. 26–45.

[3] L. Giordano, V. Gliozzi, N. Olivetti, G. Pozzato, Semantic characterization of rational closure: From propositional logic to description logics, Artificial Intelligence 226 (2015) 1–33. URL: https://www.sciencedirect.com/science/article/pii/S0004370215000673. doi:https://doi.org/10.1016/j.artint.2015.05.001.

[4] K. Britz, G. Casini, T. Meyer, K. Moodley, U. Sattler, I. Varzinczak, Principles of klm-style defeasible description logics, ACM Trans. Comput. Logic 22 (2020). URL: https://doi.org/10.1145/3420258. doi:10.1145/3420258.

[5] G. Sacco, L. Bozzato, O. Kutz, Defeasible reasoning with prototype descriptions: First steps, in: Proceedings of the 36th International Workshop on Description Logics (DL 2023), volume 3515 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2023.

[6] G. Sacco, L. Bozzato, O. Kutz, Generics in defeasible reasoning. exceptionality, gradability, and content sensitivity, in: 7th CAOS Workshop 'Cognition and Ontologies', 9th Joint Ontology Workshops (JOWO 2023), co-located with FOIS 2023, 19-20 July, 2023, Sherbrooke, Québec, Canada, volume 3637 of *CEUR-WS*, 2023.

[7] G. Righetti, P. Galliani, O. Kutz, D. Porello, C. Masolo, N. Troquard, Weighted Description Logic for Classification Problems, in: D. Calvanese, L. Iocchi (Eds.), GCAI 2019. Proceedings of the 5th Global Conference on Artificial Intelligence, volume 65 of *EPiC Series in Computing*, EasyChair, 2019, pp. 108–112. URL: https://easychair.org/publications/paper/S5bV. doi:10.29007/vd1q.

[8] L. Bozzato, T. Eiter, L. Serafini, Enhancing context knowledge repositories with justifiable exceptions, Artif. Intell. 257 (2018) 72–126.

[9] G. Sacco, L. Bozzato, O. Kutz, Defeasible reasoning with prototype descriptions: A new preference order, in: L. Giordano, J. C. Jung, A. Ozaki (Eds.), Proceedings of the 37th International Workshop on Description Logics (DL 2024), Bergen, Norway, June 18-21, 2024, volume 3739 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2024. URL: https://ceur-ws.org/Vol-3739/paper-9.pdf.

[10] G. Sacco, L. Bozzato, O. Kutz, Defeasible reasoning in description logics with prototype descriptions, in: C. Dima, A. Ferrando, V. Malvone (Eds.), PRIMA 2025: Principles and Practice of Multi-Agent Systems - 26th International Conference, volume 16366 of *Lecture Notes in Computer Science*, Springer, Modena, Italy, 2025, pp. 506–523. doi:10.1007/978-3-032-13562-9_39.

[11] G. Sacco, L. Bozzato, O. Kutz, An ASP translation for non-monotonic reasoning on dl-lite_r with prototype descriptions, in: D. Guidotti, L. Pandolfo, L. Pulina (Eds.), Proceedings of the 40th Italian Conference on Computational Logic, Alghero, Italy, June 25-27, 2025, CEUR Workshop Proceedings, CEUR-WS.org, 2025. URL: https://ceur-ws.org/Vol-4003/paper32.pdf.

[12] A. Lieto, G. L. Pozzato, A description logic framework for commonsense conceptual combination integrating typicality, probabilities and cognitive heuristics, Journal of Experimental & Theoretical Artificial Intelligence 32 (2020) 769–804. doi:10.1080/0952813X.2019.1672799.

[13] G. Sacco, L. Bozzato, O. Kutz, Introducing weighted prototypes in description logics for defeasible reasoning, in: A. F. Agostino Dovier (Ed.), Proceedings of the 38th Italian Conference on Computational Logic, CEUR Workshop Proceedings, Udine, Italy, 2023.

[14] V. W. Marek, M. Truszczynski, Nonmonotonic logic - context-dependent reasoning, Artificial intelligence, Springer, 1993.

[15] S. Kraus, D. Lehmann, M. Magidor, Nonmonotonic reasoning, preferential models and cumulative logics, Artificial Intelligence 44 (1990) 167–207. URL: https://www.sciencedirect.com/science/article/pii/0004370290901015. doi:https://doi.org/10.1016/0004-3702(90)90101-5.

[16] P. Galliani, O. Kutz, N. Troquard, Perceptron operators that count, in: M. Homola, V. Ryzhikov, R. A. Schmidt (Eds.), Proceedings of the 34th International Workshop on Description Logics (DL 2021) part of Bratislava Knowledge September (BAKS 2021), Bratislava, Slovakia, September 19th to 22nd, 2021, CEUR Workshop Proceedings, CEUR-WS.org, 2021. URL: https://ceur-ws.org/Vol-2954/paper-15.pdf.

[17] P. Galliani, O. Kutz, N. Troquard, Succinctness and complexity of ALC with counting perceptrons, in: P. Marquis, T. C. Son, G. Kern-Isberner (Eds.), Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning, KR 2023, Rhodes, Greece, September 2-8, 2023, 2023, pp. 291–300. URL: https://doi.org/10.24963/kr.2023/29. doi:10.24963/KR.2023/29.