Learning Symbolic Task Representation from a Human-Led Demonstration: A Memory to Store, Retrieve, Consolidate, and Forget Experiences

Luca Buoncompagni and Fulvio Mastrogiovanni

Abstract—We present a symbolic learning framework inspired by cognitive-like memory functionalities (*i.e.*, storing, retrieving, consolidating and forgetting) to generate task representations to support high-level task planning and knowledge bootstrapping. We address a scenario involving a non-expert human, who performs a single task demonstration, and a robot, which online learns structured knowledge to re-execute the task based on experiences, *i.e.*, observations. We consider a one-shot learning process based on non-annotated data to store an intelligible representation of the task, which can be refined through interaction, *e.g.*, via verbal or visual communication. Our general-purpose framework relies on fuzzy Description Logic, which has been used to extend the previously developed Scene Identification and Tagging algorithm. In this paper, we exploit such an algorithm to implement cognitive-like memory functionalities employing scores that rank memorised observations over time based on simple heuristics. Our main contribution is the formalisation of a framework that can be used to systematically investigate different heuristics for bootstrapping hierarchical knowledge representations based on robot observations. Through an illustrative assembly task scenario, the paper presents the performance of our framework to discuss its benefits and limitations.

Index Terms—Structural Learning and Knowledge Capture, Fuzzy Description Logic, Learning Categories and Concepts, Logic-based learning algorithms, Learning by Demonstration, Human-Robot Interaction, Knowledge Representation and Cognitive Robotics, Cognitive Memory.

I. INTRODUCTION

THE possibilities offered by human-robot collaboration are expected to significantly increase if robots can represent tasks from human-led demonstrations in an intelligible manner. Since robots for in-home assistance or collaboration in factories should adapt to humans and their idiosyncrasies, they should be able to observe their actions and build a relevant knowledge representation. Such representation must be structured and recalled when the robot needs to make decisions, but it should also be shared with humans, who could refine qualitative demonstrations by interaction. In this scenario, the robot requires a knowledge representation that is transparent to the human and that should be generated from a few observations. Furthermore, the method used for structuring knowledge should not be computed offline, and we should not assume that annotations are always available.

The paper presents a general-purpose framework to address the challenge of *bootstrapping* a knowledge representation by interacting with humans and observing their actions [1].

L. Buoncompagni and F. Mastrogiovanni are affiliated with the Department of Informatics, Bioengineering, Robotics and Systems Engineering, University of Genoa, Via Opera Pia 13, 16145, Genoa, Italy.

Corresponding author: luca.buoncompagni@edu.unige.it.

Our approach is inspired by a simplistic and functional model of memory, which is based on concepts broadly accepted in cognitive studies. Memory processes and maintains information over time through three functionalities: *encoding*, *storing* and *retrieving* [2]. In particular, sensory channels are used to encode information in some representation suitable for storage and retrieval within different types of memory, *e.g.*, short or long-term memory, as well as episodic or semantic memory [3]. Due to limited resources, humans drive their attention to specific types of memory, and this is related to *consolidation* (or *rehearsal*) and *forgetting*.

Rehearsal is a process that maintains information in the short and long-term memory [4], whereas consolidating involves episodic and semantic memory [5]. Forgetting, which might be only temporally, is an essential and challenging behaviour for complex tasks since it allows focusing only on relevant aspects, and it might reduce ambiguities and inconsistencies. Different theories have been proposed to model forgetting, *e.g.*, context-dependent strategies [6], interference [7] and trace decay [8]. In the first two, it is foreseen that some information may not be retrieved since stored items may be either corrupted or inconsistent. According to the third one, retrieval is believed to be affected by time. This paper does not focus on an accurate model of the memory from a cognitive standpoint, but we argue that those functionalities can guide the design of effective robot memories for human-robot interaction scenarios.

The paper presents a framework to bootstrap a hierarchical and symbolic knowledge representation based on the Description Logic (DL) formalism [9]. The paper neither addresses perception issues, *i.e.*, we assume a perception module providing the robot with symbolic information, nor actuation, *i.e.*, the robot is a passive observer. Instead, for the sake of argument, we developed a simple heuristic approach to learn a planning domain of an assembly task from a single demonstration, and we performed preliminary experiments. Such a heuristic approach consolidates and forgets knowledge – that is represented via symbols in an ontology – by the means of scores-based ranking mechanisms. Symbols are acquired through the previously developed Scene Identification and Tagging algorithm (SIT), which is the base component of our framework and provides functions to encode, learn, structure and classify knowledge. This paper exploits SIT to design a framework that implements storing, retrieving, consolidating and forgetting functionalities. Although our heuristics need further validation, our contribution is a formal framework that enables systematic investigations of different consolidating and forgetting strategies in support of storing and retrieving abilities.

Our framework addresses scenarios where the knowledge

that the robot bootstraps needs to be intelligible to a human supervisor, which might modify it through qualitative interaction and demonstrations. Moreover, we consider scenarios where the robot acquires limited data that should be used to deduce a structured representation, *e.g.*, a plan concerning an assembly task. The SIT algorithm can bootstrap hierarchical representations, but it would also include irrelevant steps of the task plan since it tends to over-fit the demonstration. Therefore, we extend SIT to consolidate the more relevant steps of a demonstration and forget the irrelevant ones.

II. BACKGROUND

A. Related Work

In the literature, there are many data-driven approaches for learning probabilistic models of robot behaviour. Typically, those methods do not provide structured and intelligible models, and they require many data samples to be processed in an offline training phase with the help of annotations. Imitation learning has been successfully used to map raw data into primitive motions at a trajectory or symbolic level [10], using Reinforcement Learning [11], Deep Learning [12] and Generative Adversarial Neural Networks [13]. While those approaches could be viable within the Programming by Demonstration paradigm [10], they require a high amount of demonstrations, which are time-consuming for humans. To speed up the training process, Meta-Learning has been used to transfer knowledge across domains, *e.g.*, using one-shot Imitation Learning based on Recurrent Neural Networks to model block-stacking tasks [14], and using a Convolutional Neural Network to learn actions from object locations [15]. Probabilistic learning based on Hidden Markov Models has been used to generate intelligible and structured knowledge representations, *e.g.*, for motion planning [16], where humans could also interact within the controlling loop [17]. However, Markovian approaches do not bootstrap knowledge since they require the design of a prior structure for each application. In addition, graph embedding is a suitable technique since planning and contextualised knowledge are often based on structures, and it also involves learning methods that process unstructured data [18].

Symbolic representations have been used for task planning and knowledge bootstrapping not only because they support hierarchical structures, but also because they produce intelligible knowledge. Symbolic and sub-symbolic models have been designed using the Ontology Web Language (OWL-DL) standard [19] and the related *reasoners*. Given a symbolic representation of a domain, the problem of knowledge bootstrapping is to *ground* observations, learn skills [20], and structure them into a representation, *e.g.*, a task plan. With a similar perspective, Inductive Logic Programming has been used to learn OWL structures based on positive and negative examples [21], and a technique for learning Hierarchical Task Networks from annotated datasets has been proposed in [22]. However, those methods are used in an offline fashion and require annotated data. In contrast, an approach to generate symbolic goals for a task planner through learned predicates and refinements, which are given from a supervisor only during the learning phase, has been proposed in [23].

From an analysis of the literature, it seems that the challenges related to the knowledge bootstrapping scenario we target have not been solved so far. Especially with the requirements of showcasing (*i*) online and (*ii*) intelligible model learning (to support human supervision and refinements through interaction), (*iii*) few-shots learning from (*iv*) non-annotated data (to decrease human effort), and (*v*) structured models learning (since it would effectively represent tasks as a sequence of actions). Our framework addresses these requirements based on SIT functionalities, which are introduced in the next sections.

B. Contribution

We used OWL-DL to develop the general-purpose SIT algorithm [24], which performs one-shot learning and classification of scenes that are symbolically represented in a graph of crisp logic implications encoded in an OWL ontology. Although SIT bootstraps intelligible graphs, it is neither robust to vague representation nor to noisy inputs, and it tends to over-fit the demonstration because it might learn each observed scene. Unfortunately, SIT does not scale efficiently since DL reasoning is known to be a complex task [9]. Therefore SIT should be used in small scenarios, or in scenarios that can be represented across multiple ontologies [25].

In a previous work [26], we deploy SIT in an application where a person talks with the robot to supervise and refine the knowledge it bootstraps during the demonstration of a task. In that application, we exploited the *supervisor* to overcome robustness and over-fitting issues of SIT, but the effort demanded to the human was consistent. In order to reduce such an effort, we briefly introduced the idea of tackling over-fitting issues through scene consolidation and forgetting mechanisms based on a simulated crisp representation of knowledge [27]. Instead, to tackle the lack of robustness of SIT, we preliminary presented a fuzzy implementation of SIT [28], [29], which is an extension of SIT that exploits the *fuzzyDL* reasoner [30] to represent a structure of scenes through fuzzy-logic. With respect to our previous work, this paper focuses on a framework that exploits the fuzzy implementation of the SIT algorithm, while implementing a scene consolidation and forgetting policy to overcome both robustness and over-fitting issues in a single implementation.

To present our framework, the next section briefly introduces the SIT algorithm, and Section IV presents the fuzzy ontology formalism used to implement the fuzzy extension of SIT[1], which is detailed in Section V as far as the objectives of this paper are concerned. Our main contribution is presented in Section VI, which describes the knowledge storing and retrieving functionalities, as well as the assumptions and limitations of the consolidation and forgetting strategies that our framework can accommodate based on scores. Section VII discusses the preliminary results obtained from a toy – but not simulated – experiment based on simple consolidation and forgetting heuristics, which lead to the discussion of the benefits and limitations of our framework in Section VIII.

---

[1]The implementation is SIT based on a fuzzy ontology is available at https://github.com/buoncubi/fuzzy_sit.

III. OVERVIEW

SIT processes statements representing the environment, *i.e.*, *facts*, based on a general-purpose interface (presented in Section V-A), which formalises prior knowledge required to represent an observed *scene*. This interface involves symbols semantically representing the *elements* in the scene (*e.g.*, objects of a certain *type*) and the relationships (or *roles*) between pairs of elements (*e.g.*, spatial relations). Given this information, SIT performs *encoding*, *learning*, *structuring* and *classifying* functions that are summarised in Table I, which also introduces the main symbols defined in this paper.

Section V-B presents the encoding functionality, which manipulates input facts to create *beliefs* in the ontology, and computes their *cardinality* to represent the scene. For instance, a scene observed at a certain time can be represented with a belief as "two cups are in front of a glass" (Figure 1).

As detailed in Section V-C, the learning function generates scene *categories* in the ontology, which are defined through *restrictions* based on the cardinalities of the perceived beliefs. For instance, a category can restrict a belief to have "at least 2 cups in front of a glass". In this paper, we extend SIT to associate each category with a time-dependent *score*, which is used to consolidate or forget such a category.

The structuring function (addressed in Section V-D) uses DL-based reasoning to generate a graph of categories based on their restrictions. In particular, SIT creates a *memory graph* (*e.g.*, Figure 3d) having scene categories as nodes, and edges as their logic implications. At the beginning of the interaction, we assume a Lockean *tabula rasa* situation, whereby the memory graph contains only a root node representing the empty scene, *i.e.*, it is restricted to "at least" zero beliefs. Then, during the observation of a task, scene categories would be added to the memory graph for bootstrapping a knowledge representation. Generally, the structuring function consistently arranges the structure of the graph when its categories are added or removed and, in this paper, we also assume that it preserves the score assigned to each category.

Given a memory, *i.e.*, a graph of categories, we can encode the input facts of the current scene for classifying its beliefs with respect to the categories previously learned and structured in the memory. The classification function (presented in Section V-E) provides a sub-graph of the memory, *i.e.*, the *classification graph*, which contains all the categories defined with restrictions that satisfy the cardinality of the beliefs of a perceived scene, *e.g.*, Figure 3d. Each node of the classification graph is related to a *similarity value*, which describes how many restrictions are satisfied and, in the fuzzy implementation, it is also associated with a *classification degree*, which represents how well the scene beliefs satisfy category restrictions.

Section VI presents a combination of the above SIT functionalities used to implement a cognitive-like memory. Section VI-A and Algorithm 1 describe the design of *storing* or *retrieving* functions, while Section VI-B and Algorithm 2 presents a strategy for *consolidating* and *forgetting* categories. Although our algorithms are illustrative and implement simple heuristics, they support our contribution, which focuses on their input and output interfaces based on SIT. Nonetheless,

Function	In → Out	Description
Encoding $\mathcal{E}$	$F \rightarrow \varepsilon$	Given *facts* F^2, $\mathcal{E}$ reifies[3] *beliefs* $\mathbf{r}_{zh}$ and computes their *cardinality* c_{zh} to represent the perceived Scene ε.
Learning $\mathcal{L}$	$\begin{matrix} \varepsilon \\ q^0 \\ a \end{matrix} \rightarrow \begin{matrix} \Phi_\varepsilon \\ q_\varepsilon \end{matrix}$	Given ε, an *initial score* q^0, and a *fuzziness* value a, $\mathcal{L}$ computes cardinality *restrictions* $^a\Omega$ defining a new scene CATEGORY Φ_ε with a *score* q_ε.
Structuring $\mathcal{S}$	$\begin{matrix} M_{t-1} \\ \Phi_\varepsilon \\ q_\varepsilon \end{matrix} \rightarrow M_t$	Given a *memory graph* M_{t-1} in the ontology, and a new category Φ_ε with a score q_ε, $\mathcal{S}$ adds Φ_ε and q_ε as a node of a new graph M_t, which consistently updates M_{t-1} through logic reasoning.
Classifying $\mathcal{C}$	$\begin{matrix} M_t \\ \varepsilon \end{matrix} \rightarrow M^\star$	Given ε and M_t, $\mathcal{C}$ exploits logic reasoning to return the *classification graph* $M^\star \subseteq M_t$, which contains the nodes Φ_j having ε as a realisation in the ontology (*i.e.*, $\varepsilon{:}\Phi_j$). Each node of $M^\star$ is also associated with the fuzzy *classification degree* $p^{\varepsilon:\Phi_j}$ and the *similarity value* $d_\varepsilon^{\Phi_j}$.

Table I: The glossary of symbols and functionalities provided by the SIT algorithm. The latter is based on a fuzzy ontology and extended with *scores* to implement consolidating and forgetting capabilities.

with such heuristics, we learn an effective representation of a table assembling task, as shown in Figure 4.

To enable a discussion about storing, retrieving, consolidating and forgetting functionalities, Section V formalises the SIT functionalities introduced in this Section within an OWL ontology processed by a fuzzy DL reasoner. The semantic associated with OWL and the logic-based reasoning for a fuzzy representation is briefly introduced in the next Section, and additional background material can be found in [9] and [30].

IV. A FUZZY DESCRIPTION LOGIC PRIMER

A DL ontology is a collection of logic statements (*i.e.*, *axioms*, represented as tuples) describing knowledge through symbols with varied semantics. In this work, we rely on the fuzzyDL reasoner, which logically deduces knowledge consistency based on a fuzzy degree of truth $p \in [0, 1]$ assigned to each axiom. Axioms are made of symbols related to two distinct ontological structures, *i.e.*, the Terminological Box (TBox), containing a hierarchical definition of *concepts* and *roles*, and the Assertion Box (ABox), including a set of *instances*. Furthermore, concepts are defined (*i*) as logical *implications* with respect to other concepts, and (*ii*) through roles relating their *definitions* to existing concepts. Roles are defined with mutual *implications*, whereas instances can be described (*i*) as being a specific *realisation* of a concept, and (*ii*) with *assertion* roles in the definition of other instances.

Throughout the paper, we denote *sets* with an upper case Roman letter, their *items* or *numbers* with lower case Roman letters, and *functions* with Italic symbols. Also, Table II summarises the notation we adopt for the DL formalism. The first column shows generic symbols for DL entities and the related fonts that we use to present examples, while the second

---

[2]Facts are element γ of TYPE Γ and related with a *role* $\mathbf{r}$.

[3]The reification operator is defined in Section V-B.

Entity	Axiom	Operator	
Instance α	Assertion (α, β):**r**	Disjunction	$\Lambda \sqcup \Delta$
CONCEPT Λ	Realisation α:Λ	Conjunction	$\Lambda \sqcap \Delta$
role **r**	Definition $\Lambda = \exists \mathbf{r}.\Upsilon$	Subsumption	$\Lambda \sqsubseteq \Delta$, $\mathbf{r} \sqsubseteq \mathbf{p}$

Table II: The glossary of the DL-based symbols used to implement SIT. The first column shows the notation we adopted to identify the different types of *entities* in the ontology. The second and third columns regard the used logic *axioms* and *operators* among entities.

$\langle \gamma_1 \text{:GLASS}, 0.9 \rangle$, $\quad \langle (\gamma_1, \gamma_2) \text{:front}, 1 \rangle$,
$\langle \gamma_2 \text{:CUP}, 0.7 \rangle$, $\quad \langle (\gamma_1, \gamma_3) \text{:front}, 0.6 \rangle$,
$\langle \gamma_3 \text{:CUP}, 0.8 \rangle$, $\quad \langle (\gamma_2, \gamma_3) \text{:front}, 0.2 \rangle$,
$\langle \gamma_3 \text{:GLASS}, 0.1 \rangle$, $\quad \langle (\gamma_2, \gamma_1) \text{:behind}, 1 \rangle$,
$\quad \langle (\gamma_3, \gamma_1) \text{:behind}, 0.6 \rangle$,
$\quad \langle (\gamma_3, \gamma_2) \text{:behind}, 0.2 \rangle$.

Figure 1: An example of input facts (on the right-hand side) required to process the spatial scene ε_1 (on the left-hand side). Facts can be encoded by SIT to perform the functionalities shown in Table I.

and third columns summarise DL axioms and operators having the following definitions.

Let $\langle (\alpha, \beta)\text{:}\mathbf{r}, p^{(\alpha,\beta)\text{:}\mathbf{r}} \rangle$ be a tuple, where α and β are instances, **r** is a specific role, and p denotes the vagueness of this relation to be true for α and β. For instance, the axiom $\langle (\text{Basin}, \text{Jar})\text{:bigger}, 0.9 \rangle$ *asserts* that a Basin is bigger than a Jar with a fuzzy degree of 0.9.

Let $\langle \alpha\text{:}\Lambda, p^{\alpha\text{:}\Lambda} \rangle$ be a tuple, where the instance α is defined as a *realisation* of the concept Λ with a certain degree of truth p, e.g., $\langle \text{Basin:CONTAINER}, 0.7 \rangle$ or $\langle \text{Jar:CONTAINER}, 1 \rangle$.
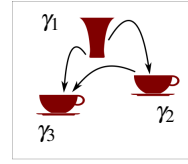
Also, let $\Lambda = \langle \exists \mathbf{r}.\Upsilon, p^{\mathbf{r}.\Upsilon} \rangle$ be a tuple that *defines* a concept Λ with a degree p. Then, each realisation of Λ is an instance α related with some (i.e., $\exists$) role **r** to an individual β that is a realisation of the Υ concept. For instance, CONTAINER $= \langle \exists \text{shape.CONVEX}, 1 \rangle$ states that the instance Basin is a consistent realisation of CONTAINER only if in the ABox exist, with some degrees, (i) the assertion (Basin, Parabolic):shape and (ii) the realisation Parabolic:CONVEX.

In the TBox, axioms can be related to each other through set operators. In particular, we recall the *disjunction* $\sqcup$, based on the fuzzy *t-conorm*, the *conjunction* $\sqcap$, identified with the *t-norm*, and the *subsumption* $\sqsubseteq$. Subsumption (or *supersumption* $\sqsupseteq$) can be considered as an *implication* (i.e., $\Lambda \Rightarrow \Upsilon$), and we can write $\langle \Lambda \sqsubseteq \Upsilon, p^{\Lambda \sqsubseteq \Upsilon} \rangle$ if all the α realisations of Λ are also instances of Υ with a degree p, e.g., $\langle \text{CONTAINER} \sqsubseteq \text{OBJECT}, 1 \rangle$. In DL, the subsumption operator is also defined among two roles **r** and **p** but, in this paper, we consider this operation only to collect different roles with a unique semantic.

By checking the consistency of all the axioms and operators in the Tbox and ABox, the reasoner can (i) classify instances as realisations of concepts and (ii) generate a graph where nodes are concepts and edges are logic implications (i.e., subsumptions). Remarkably, the reasoner complies with the *open-world* assumption [9], which is appropriate for robot scenarios, but more complex to handle.

V. SCENE IDENTIFICATION AND TAGGING

For implementing the SIT functionalities summarised in Table I, we assume in the TBox the availability of two *disjoint* concepts describing scenes CATEGORY (Φ, i.e., the root of the memory graph), and the TYPE (Γ) of elements in the scene. The TBox also involves a role (**r**), while the ABox contains instances representing each *element* of the environment (at least a pair γ_x, γ_y) and the current Scene (ε). SIT represents observations that hold only for a discrete instance of time in the ABox, whereas it memorises a graph of scene categorisation over time in the TBox.

A. The Input Interface

SIT requires a set F of *facts* $\{f_1, \ldots, f_i, \ldots, f_n\}$ describing the state of the robot workspace at a certain instant of time. A generic fact f_i is a set of DL axioms spanning in a combination of a DL entity subsuming (i) the role $\mathbf{r} \sqsupseteq \{\mathbf{r}_1, \ldots, \mathbf{r}_z, \ldots, \mathbf{r}_w\}$, which is applied between an element pair (γ_x, γ_y), where γ_x and γ_y are realisations of (ii) the TYPE concept $\Gamma \sqsupseteq \{\Gamma_1, \ldots, \Gamma_v\}$. Therefore, the sets of w roles and v concepts generally represent the prior knowledge used to define the input interface. Figure 1 shows possible inputs where we consider an input interface with $\{\text{front}, \text{behind}\} \sqsubseteq \mathbf{r}$ and $\{\text{CUP}, \text{GLASS}\} \sqsubseteq \Gamma$, i.e., $w = v = 2$.

In general, given the concepts subsuming Γ, a role $\mathbf{r}_z$, and fuzzy membership degrees $p_{iz}, p_{is}, p_{ih} \in [0, 1]$, we can formalise the input interface as a set of facts F with items

$$f_i = \left\{ \langle (\gamma_x, \gamma_y)\text{:}\mathbf{r}_\mathbf{z}, p_{iz} \rangle; \left\{ \langle \gamma_x\text{:}\Gamma_s, p_{is} \rangle, \forall \Gamma_s \sqsubseteq \Gamma \right\}; \right.$$
$$\left. \left\{ \langle \gamma_y\text{:}\Gamma_h, p_{ih} \rangle, \forall \Gamma_h \sqsubseteq \Gamma \right\} \right\}. \quad (1)$$

A fact f_i describes (i) the relation $\mathbf{r}_z$ between the entities γ_x and γ_y with a degree $p_{iz} \triangleq p^{(\gamma_x, \gamma_y)\text{:}\mathbf{r}_z}$ and (ii) the type of the x-th and y-th entities as realisations of the s-th and h-th concepts subsuming Γ with degrees $p_{is} \triangleq p^{\gamma_x\text{:}\Gamma_s}$ and $p_{ih} \triangleq p^{\gamma_y\text{:}\Gamma_h}$. It is noteworthy that, due to the open-world assumption, an input fact f_i might not specify p_{is} and p_{ih} for some types Γ of the x-th and y-th elements.

B. Encoding

SIT encodes *beliefs* into an instance ε that represents the current *scene* based on F. This phase exploits the *reification* operator $\mathcal{R}(\mathbf{r}_z, \Gamma_h) \triangleq \mathbf{r}_{zh}$ that defines a role through a concatenation of constituent symbols. For instance, $\mathcal{R}(\mathbf{r}_1, \Gamma_2) \equiv \text{front\_GLASS}$ represents the fact that a GLASS $\gamma_y\text{:}\Gamma_2$ has some element $\gamma_x\text{:}\Gamma$ in front, i.e., $\mathbf{r}_1$. In this paper, we simplify the reification $\mathcal{R}$ by disregarding the realisation of γ_x in the Γ_s concept. Instead, we reduce γ_x as an element of a generic TYPE Γ.

This simplification of $\mathcal{R}$ reduces the computation complexity since the number of axioms in the ontology decreases, and this is significant because the reasoning task has an exponential complexity with the number of axioms [9]. In addition, if we assume an input interface where each $\mathbf{r}_z \sqsubseteq \mathbf{r}$ have an *inverse* role $\mathbf{r}_z^! \sqsubseteq \mathbf{r}$, we do not completely lose information about $\gamma_x\text{:}\Gamma_s$ since there will always be a $f_j \in F$ representing the inverse fact of f_i. In other words, f_j involves $\mathbf{r_z}^{-1}$ where the s-th and h-th concepts are swapped with respect to f_i. For this reason, in the example in Figure 1, we consider $\mathbf{r}_1 = \mathbf{r}_2^{-1}$, i.e., (γ_x, γ_y):front is consistent only if the axiom (γ_y, γ_x):behind exists with the
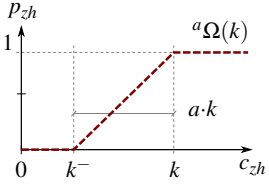
Figure 2: The fuzzy cardinality restriction $^a\Omega(k)$. If the cardinality $c_{zh} \geqslant k$, the restriction is satisfied with fuzzy degree $p_{zh} = 1$. If $c_{zh} \leqslant k^-$, it is not satisfied. Otherwise, it is *fuzzily* satisfied with $p_{zh} \in (0,1)$. We define $k^- = k(1-a), a \in [0,1]$.
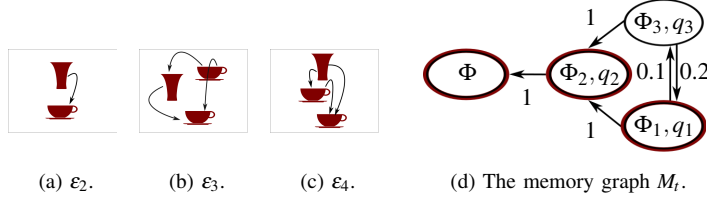

Figure 3: Examples of scenes that are observed, learned and classified in the memory over time. When the ε_1 (Figure 1), ε_2 (3a) and ε_3 (3b) scenes have been perceived, SIT learns and structures in the memory M_t (3d) the categories Φ_1, Φ_2 and Φ_3, respectively. Each category is computed with $a = 0.5$, and it has a score (i.e., q_1, q_2 and q_3). Then, when the ε_4 scene (3c) is perceived, it will be classified in a sub-graph of M_t, i.e., the classification graph $M^\star$, which nodes are highlighted in (3d). Table (3e) shows the classification degree $p^{\varepsilon_4:\Phi_j}$ and similarity value $d^{\Phi_j}_{\varepsilon_4}$ associates for each j-th node in M_t except for Φ, which is the root of the graph and it categorises an empty scene. Since $p^{\varepsilon_4:\Phi_2} = 0$, Φ_3 is not a node of $M^\star$ and, therefore, $d^{\Phi_3}_{\varepsilon_4}$ is undefined.

same degree. However, this simplification also reduces the intelligibility of the bootstrapped knowledge representation.

We define the encoding function $\mathcal{E}$ for a fact f_i and for a set of facts F, respectively, as

$$\mathcal{E}(f_i) = \left\{ \langle (\varepsilon, c_{izh}){:}\mathbf{r}_{zh}, 1 \rangle, \forall z, h : f_i \right\}, \text{ therefore,}$$

$$\mathcal{E}(F) = \left\{ \langle (\varepsilon, c_{zh}){:}\mathbf{r}_{zh}, 1 \rangle, \forall zh \in [1, w{\cdot}v] \right\}. \quad (2)$$

Since in (2) we always define $p^{(\varepsilon, c_{izh}){:}\mathbf{r}_{zh}}$ equal to 1, a belief $\mathcal{E}(f_i)$ is a non-fuzzy (i.e., crisp) role based on a combination of each z-th role and h-th TYPE occurring in f_i. The number of zh-th beliefs generated from an input interface is equal to all the combinations of the symbols subsuming $\mathbf{r}$ and Γ, i.e., $w{\cdot}v$; however, this amount can be lower due to the open-world assumption. Since different facts can be mapped to the same zh-th belief, we represent with $c_{zh} \in \mathbb{R}^+$ the fuzzy *cardinality* of the reified role $\mathbf{r}_{zh}$.

In the crisp formalisation of SIT, the cardinality is the number of equivalent roles applied to ε, but this definition becomes ill-posed and challenging to define in a fuzzy domain [31]. Among several definitions proposed for fuzzy counting, the σ-count is a popular approach that defines the cardinality with the sum of the fuzzy degrees of all the items in a set [32]. This approach treats cardinality as a real positive number, but it has a drawback since it computes the *energy* of a set, which is considered to be a measure of the cardinality assuming that the higher the energy value, the more the number of elements such a set should have. This might not be true, and the σ-count cannot discriminate differences between a set having 10 elements with fuzzy degree 0.1, or one element with degree equal to 1.

The cardinality of a belief $\mathbf{r}_{zh}$ is defined as the conjunction among all the realisation of γ_x in Γ, which is in disjunction with the fact that γ_y is a realisation of Γ_h and it is paired with γ_x through $\mathbf{r}_z$. Thus, c_{izh} is the fuzzy degree of the axiom

$$\bigsqcup_s^{[1,v]} \left\{ \langle (\gamma_x, \gamma_y){:}\mathbf{r}_z, p_{iz} \rangle \sqcap \langle \gamma_x{:}\Gamma_s, p_{is} \rangle \sqcap \langle \gamma_y{:}\Gamma_h, p_{ih} \rangle \right\}. \quad (3)$$

This is computed with the DL Zadeh logic [30] as

$c_{izh} = \max_s^{[1,v]} \{ \min\{p_{iz}, p_{is}, p_{ih}\} \}$, and with the σ-count approach we obtain $c_{zh} = \sum_i^{[1,n]} c_{izh}$.

In the example in Figure 1, the cardinality c_{12} of the front GLASS belief, and c_{21} regarding behind CUP are

$$c_{12} = \max\left\{ \min\{1, 0.9, 0.7\} \right\} + \max\left\{ \min\{0.6, 0.9, 0.8\}, \right.$$
$$\left. \min\{0.6, 0.9, 0.1\} \right\} = 1.3. \quad (4)$$
$$c_{21} = \max\left\{ \min\{1, 0.7, 0.9\} \right\} + \max\left\{ \min\{0.6, , 0.8, 0.9\} \right\}$$
$$+ \max\left\{ \min\{0.2, 0.8, 0.7\} \right\} = 1.5. \quad (5)$$

In other words, in ε_1 "1.3 objects are in front of the glasses", and "1.5 objects are behind the cups". It is worth noticing that the example in Figure 1 involves the γ_3 element, which is a realisation of both Γ_1 and Γ_2 types, i.e., $\langle \gamma_3{:}\Gamma_2, 0.1 \rangle$ due to perception error or vagueness.

C. Learning

Let an initial consolidating *score* q^0 (described more in detail in Section VI-A) and the cardinality c_{zh} of each belief encoded from F, the learning function $\mathcal{L}$ creates in the TBox a new *category* Φ_ε subsuming the root of the memory graph Φ (i.e., the empty scene) by definition.

In particular, for each zh-th belief, Φ_ε is defined through a fuzzy cardinality *restriction* that is based on a left-shoulder membership function $^a\Omega(k)$, which has $k := c_{zh}$, and it does not change over time. Such a membership function is shown in Figure 2, and it is based on the *fuzziness* parameter $a \in [0,1]$, from which we derive $k^- = k(1-a)$. If $a = 0$, Ω would be a crisp restriction, while if $a = 1$ the restriction would always be satisfied, even for small cardinality c_{zh}, but with a low fuzzy degree p_{zh}. Let the cardinality $\bar{c}_{zh}$ of a Scene $\bar{\varepsilon}$ be encoded at a different instant of time, then if $\bar{c}_{zh} \geqslant k$ we consider the restriction to be fully satisfied, i.e., $p_{zh} = 1$. Instead, if $\bar{c}_{zh} \leqslant k^-$ the restriction would not be satisfied since $p_{zh} = 0$, while if $k^- < \bar{c}_{zh} < k$ we consider the cardinality restriction to be *fuzzily* satisfied because $p_{zh} \in (0,1)$.

Based on this evaluation, the learning function creates a concept Φ_ε (with an initial score) where *at least* some zh-th

beliefs must be observed to represent ε. We define the learning function as $\mathcal{L}\left(\varepsilon, q^0, a\right) = \langle \Phi_\varepsilon, q_\varepsilon := q^0 \rangle$ where

$$\Phi_\varepsilon = \langle \Phi, 1 \rangle \prod_{zh}^{\mathcal{E}(F)} \langle \exists \mathbf{r}_{zh}.^a\Omega(k), 1 \rangle. \tag{6}$$

In accordance with Section IV, in (6) it is certain (*i.e.*, $p^{\mathbf{r}_{zh}.^a\Omega(k)} = 1$) that, in order to assert a future scene $\bar{\varepsilon}$ as a realisation of Φ_ε, its cardinality $\bar{c}_{zh}$ needs to satisfy all the *zh*-th restrictions $^a\Omega(k)$ of Φ_ε. As seen above, such a $\bar{\varepsilon}:\Phi_\varepsilon$ realisation can have a fuzzy degree p_{zh}, which is based on $\bar{c}_{zh}$, a, and k; which is equivalent to the c_{zh} cardinality related to ε. When $\bar{\varepsilon}:\Phi_\varepsilon$ occurs with a degree $p_{zh} > 0$, we consider the scene $\bar{\varepsilon}$ be classified in the Φ_ε category, as better discussed in Section V-E. It is noteworthy that, by the definition of (6), Φ_ε always subsumes Φ with a crisp degree.

For instance, the category Φ_2 learned from the scene ε_2 in Figure 3a can be

$$\Phi_2 = \langle \Phi, 1 \rangle \sqcap \langle \exists\, \mathsf{front}\, \mathsf{GLASS}.^a\Omega(0.9), 1 \rangle$$
$$\sqcap \langle \exists\, \mathsf{behind}\, \mathsf{CUP}.^a\Omega(1), 1 \rangle. \tag{7}$$

In other words, a future Scene will be classified into the Φ_2 category if it has "at least 0.9 objects are in front of some glasses" and "at least 1 object is behind some cups".

D. Structuring

After each learning phase, SIT relies on the structuring function $\mathcal{S}$ to add a new category Φ_ε and its score q_ε into the *memory graph* M_{t-1}, which gets updated consistently with respect to the previously learned categories in a new memory graph M_t. The new scene category is added as a node in the graph, and this operation is delegated to the fuzzyDL reasoner, which can also update the memory when a category is removed from the ontology, *i.e.*, forgotten. In particular, the reasoner generates a graph M_t with weighted edges based on the fuzzy degree of subsumption axioms involving each pair of categories in the memory, *i.e.*, $p^{\Phi_i \sqsubseteq \Phi_j} \,\forall \Phi_i, \Phi_j \sqsubseteq \Phi$, and we do not show the edge if $p^{\Phi_i \sqsubseteq \Phi_j} = 0$.

Thus, to compute $\mathcal{S}$, it is sufficient to introduce in the TBox the newly created Φ_ε concept and invoke fuzzyDL to perform instance checking, *i.e.*, it reasons on the comparison of the restrictions $^a\Omega(k_i)$ and $^a\Omega(k_j)$ for each *zh*-th belief of Φ_i and Φ_j, respectively. Let a fixed fuzziness $a = 0.5$ used to learn categories, such a reasoning process provides a degree

(i) $p_{zh}^{\Phi_i \sqsubseteq \Phi_j} = 1$, if $k_i^- \geqslant k_j^-$, *e.g.*, "at least $k_i = 1.4$" always implies "at least $k_j = 0.8$"; or

(ii) $p_{zh}^{\Phi_i \sqsubseteq \Phi_j} = 0$, if $k_i \leqslant k_j^-$, *e.g.*, "at least $k_i = 0.6$" does not imply "at least $k_j = 1.8$"; or

(iii) $p_{zh}^{\Phi_i \sqsubseteq \Phi_j} \in (0, 1)$ linearly to $k_i^- < k_j^- < k_i$, *e.g.*, "at least $k_i = 0.75$" implies "at least $k_j = 1$" with $p_{zh}^{\Phi_i \sqsubseteq \Phi_j} = 0.5$.

Since (6) is based on conjunctions, each edge of M_t has a weight $p^{\Phi_i \sqsubseteq \Phi_j}$ computed by the reasoner as the minimum $p_{zh}^{\Phi_i \sqsubseteq \Phi_j}$ degree among all the *zh*-th restrictions of Φ_j.

As an example, Figure 3d shows how the categories learned from ε_1 (in Figure 1), ε_2 and ε_3 (in Figure 3a and 3b) are structured in the memory graph as Φ_1, Φ_2 and Φ_3, respectively. In particular, Φ_2 is related to the scene that restricts the smaller number of beliefs, thus it is a child of the root Φ. Since ε_2 is a sub-scene of ε_1 and ε_3, Φ_1 and Φ_3 are children of Φ_2, *i.e.*, $\Phi_1 \sqsubseteq \Phi_2$ and $\Phi_3 \sqsubseteq \Phi_2$ with degree 1. Hence, if the restrictions of Φ_1 and Φ_3 are satisfied, then the restrictions of Φ_2 are necessarily satisfied, but the opposite is not true. Since ε_1 and ε_3 share some facts with different degrees (both have a cup in front of a glass, and a cup in front of another), the graph M_t involves two edges with low weight. In general, if two nodes Φ_i and Φ_j are connected by two opposite edges with weight 1, it implies that $\Phi_i \equiv \Phi_j$.

E. Classification

The classifying function $\mathcal{C}$ is based on a query to the fuzzyDL reasoner, through which we obtain all the concepts having ε as a realisation, *i.e.*, all $\Phi_j \sqsubseteq \Phi$ such that $p^{\varepsilon:\Phi_j} \neq 0$. Therefore, the classification is a sub-graph of M_t, which we call *classification graph* $M^\star$. Based on (6), the solution to such a query is the conjunction among the p_{zh} degrees obtained by computing the membership of c_{zh} in the restriction $^a\Omega(k_j)$ of Φ_j; as deducible from Figure 2. Thus, the *classification degree* $p^{\varepsilon:\Phi_j}$ is the minimum p_{zh} value among all the *zh*-th restrictions of Φ_j. Such a degree is associated with each node of $M^\star$ to represent whether the beliefs of ε satisfy the restrictions of Φ_j. If the result to the query is only the root node, *i.e.*, $M^\star \equiv \Phi$, then ε is considered as not classified.

For instance, Figure 3d highlights the nodes of M_t that are part of $M^\star$ when we classify ε_4 (Figure 3c) after having learned and structured ε_1 (Figure 1), ε_2 and ε_3 (Figure 3a and 3b) as Φ_1, Φ_2 and Φ_3, respectively. In accordance with Figure 3e, ε_4 is not classified as Φ_3 since the beliefs do not satisfy all the restrictions. Instead, ε_4 is classified in Φ_1 with a degree $p^{\varepsilon_4:\Phi_1} = 0.8$. Since $\Phi_1 \sqsubseteq \Phi_2$, ε_4 is also classified in Φ_2 with a higher degree (*i.e.*, $p^{\varepsilon_4:\Phi_2} = 1$) because Φ_2 restricts fewer beliefs, *i.e.*, its a parent of Φ_1.

From the classification degree, we cannot assess whether a child node in $M^\star$ better represents ε with respect to its parent. Thus, we introduced the *similarity value* $d_\varepsilon^{\Phi_j}$ as the ratio between the restrictions of Φ_j and the c_{zh} cardinalities of ε. More formally, we associate each node of $M^\star$ with

$$d_\varepsilon^{\Phi_j} \triangleq \frac{d^{\Phi_j}}{d_\varepsilon} \text{ where } d^{\Phi_j} = \sum_{\mathbf{r}_{zh}}^{\Phi_j} k_j, \text{ and } d_\varepsilon = \sum_{\mathbf{r}_{zh}}^{\varepsilon} c_{zh}. \tag{8}$$

In the crisp case, it would hold that $d_\varepsilon^{\Phi_j} \in [0, 1]$ since $d_\varepsilon \geqslant d^{\Phi_j}$ always, otherwise, Φ_j would not be a node of $M^\star$ because its restrictions are not be satisfied, and we would not compute $d_\varepsilon^{\Phi_j}$. In the fuzzy domain, the similarity value might be *slightly* bigger than 1 when c_{zh} is within the range defined by the fuzziness parameter a. However, the more the similarity exceeds 1, the lower $p^{\varepsilon:\Phi_j}$ becomes.

As an example, Figure 3e shows that the similarity between ε_4 (Figure 3c) and Φ_1 (Figure 1) is high (*i.e.*, $d_{\varepsilon_4}^{\Phi_1} = 0.9$) since the category restricts most of the cardinality of the scene. Instead, the similarity with Φ_2 (Figure 3a) is lower (*i.e.*, $d_{\varepsilon_4}^{\Phi_2} = 0.4$) because the category restricts approximately

---

Algorithm 1: The store or retrieve implementation.

State : An ontology representing the memory M_{t-1}.
Input : The input facts F based on $\mathbf{r}$ and Γ (Section V-A).
Output : It updates the ontology for representing M_t.
Param : The initial score $q^0 \in \mathbb{R}^+$. The fuzziness value $a \in [0,1]$. The learning $u, v \in [0,1]$ and the consolidating $e, f \in [0,1]$ thresholds for $p^{\varepsilon:\Phi_j}$ and $d_\varepsilon^{\Phi_j}$ respectively.

   // Encode and classify the input Scene.
1.1 $\varepsilon \leftarrow \mathcal{E}(F)$
1.2 $M^\star \leftarrow \mathcal{C}(M_{t-1}, \varepsilon)$
   // Add a new CATEGORY into the memory.
1.3   **if** $M^\star \equiv \{\Phi\}$ **then**   // Not classified.
1.4 $\big\lfloor\; M_t \leftarrow \mathcal{S}\big(M_{t-1}, \mathcal{L}(\varepsilon, q^0, a)\big)$

1.5   **else if** $\big(p^{\varepsilon:\Phi_j} < u$ & $d_\varepsilon^{\Phi_j} < o\big) \; \forall \langle \Phi_j, q_j, p^{\varepsilon:\Phi_j}, d_\varepsilon^{\Phi_j}\rangle \in M^\star$ **then**
      // Classified with low degree.
1.6 $\big\lfloor\; M_t \leftarrow \mathcal{S}\big(M_{t-1}, \mathcal{L}(\varepsilon, q^0 \cdot \max_{\Phi_j}^{M^\star}\{q_j\}, a)\big)$

   // Consolidate if the Scene persists.
1.7   **foreach** $\langle \Phi_j, q_j, p^{\varepsilon:\Phi_j}, d_\varepsilon^{\Phi_j}\rangle \in M^\star$ **do**
1.8   **if** $\big(p^{\varepsilon:\Phi_j} > e$ & $d_\varepsilon^{\Phi_j} > f\big)$ **then**
1.9 $\big\lfloor\; q_j \leftarrow q_j + p^{\varepsilon:\Phi_j}$ // Increase M_t scores.

Algorithm 2: The consolidate and forget implementation.

State : An ontology representing the memory M_{t-1}.
Output : It updates the ontology for representing M_t.
Param : The score weight $l \in \mathbb{R}^+$. The threshold for weak score $g \in [0,1]$ to forget. A *forgotten* flag initialised to false.

   // Weight the scores.
2.1   **foreach** $\langle \Phi_j, q_j \rangle \in M_{t-1}$ **do**
2.2 $\big\lfloor\; q_j \leftarrow l \cdot q_j$

2.3 $q_{\max} \leftarrow \max_{\Phi_j}^{M_{t-1}}\{q_j\}$
2.4   **foreach** $\langle \Phi_j, q_j \rangle \in M_{t-1}$ **do**
2.5 $q_j \leftarrow q_j / q_{\max}$ // Normalize the scores.
2.6   **if** $q_j < g$ **then**   // Eventually forget.
2.7 $M_{t-1} \leftarrow M_{t-1} \setminus \langle \Phi_j, q_j \rangle$
2.8   *forgotten* $\leftarrow$ *true*

   // Eventually restructure the memory.
2.9   **if** *forgotten* **then**
2.10 $\big\lfloor\; M_t \leftarrow \mathcal{S}(M_{t-1}, \emptyset)$

2.11 **else**
2.12 $\big\lfloor\; M_t \leftarrow M_{t-1}$

---

half of the cardinality of the scene. The similarity with Φ_3 is undefined since it does not belong to $M^\star$, *i.e.*, $p^{\varepsilon_4:\Phi_3} = 0$.

VI. MEMORY CAPABILITIES IMPLEMENTATION

This section presents our main contribution, which concern the implementation of a cognitive-like memory based on the SIT algorithm. In particular, the storing and retrieving functionalities of the memory are based on all the functionalities of SIT, while consolidating and forgetting only exploit structuring. These memory functionalities are also used to manage the scores assigned to each categories in the memory.

A. Storing and Retrieving

Algorithm 1 implements the storing and retrieving functionalities, which are based on the same procedure with the differences that (*i*) input facts F are perceived or requested, respectively, and for retrieving (*ii*) $M^\star$ is *returned* and (*iii*) $\mathcal{L}$ might not be used. Algorithm 1 uses all SIT functionalities to classify the input facts and, if necessary, to learn a new scene category. Then, Algorithm 1 increases the scores of some classified categories since we assume that scenes persistently observed deserve more attention than sporadic ones. In our experiments (detailed in Section VII), we considered only the storing functionality, but the results related to retrieving are compliant with the capability of SIT [26]. It is noteworthy that with this implementation we would also consolidate scenes that are often requested to be retrieved.

In particular, Line 1.2 encodes facts to obtain the beliefs of ε, which are used for classification. If SIT cannot classify the scene, Line 1.4 learns and structures a new category in the memory M_t, which then contains a new node with an initial score q^0. At Line 1.5 the classification is considered not accurate enough if for all the nodes in $M^\star$ the classification degree is less than a threshold u, and the similarity value is less than o. In this case, Line 1.6 adds a new category to M_t, and it exploits a heuristic to set its initial score as the maximum score

of the classified categories weighted by q^0. Then, Line 1.7 iterates over each node of $\Phi_j \in M^\star$ to increase the score q_j if the classification is not negligible, *i.e.*, the classification degree is greater than a threshold e, and the similarity value is greater than f (Line 1.8). If those conditions are true, Line 1.9 implements a heuristic that increases the score of Φ_j by adding its classification degree. Hence, Line 1.6 learns categories based on previous scores, while Line 1.9 consolidates recurrent scenes and sub-scenes.

B. Consolidation and Forgetting

Algorithm 2 shows the implementation of the consolidating and forgetting strategies, and it is sporadically used to update the score q_j of each Φ_j in M_{t-1}. If a score is too low, Algorithm 2 removes the related scene category from the memory, and it structures a new graph M_t. Otherwise, it generates a graph M_t that has the same structure of M_{t-1}, but the nodes would generally have different scores.

At Line 2.1, we iterate for each node in the memory, and we implement a heuristic to update all the scores. In particular, Line 2.2 weights each score q_j with a constant value l and, Line 2.5 divides them by the maximum score in M_t to normalise each q_j in $[0,1]$. Then, if the normalised value is lower than a threshold g (Line 2.6), we consider Φ_j to have a weak consolidation, and Line 2.7 removes Φ_j from M_t. Then, if some categories have been forgotten, Line 2.10 invokes the structuring function without specifying the new category to add, *i.e.*, it invokes the fuzzyDL reasoner to remove the edges related to forgotten nodes in the graph. Otherwise, Line 2.12 only assigns to the memory M_t the graph M_{t-1} with the updated scores.

It is noteworthy that we do not implicitly decrease the scores q_j, but this happens as a consequence of the normalisation, which is needed to compare q_j with a forgetting threshold g to be set *a priori*. However, such a normalisation generates scores that are discontinuous over time, which lead to issues, as discussed in Section VIII.

Also, Algorithm 2 minimizes the usage of $\mathcal{S}$ (by the means of the *forgotten* flag) since it is a time-consuming task that
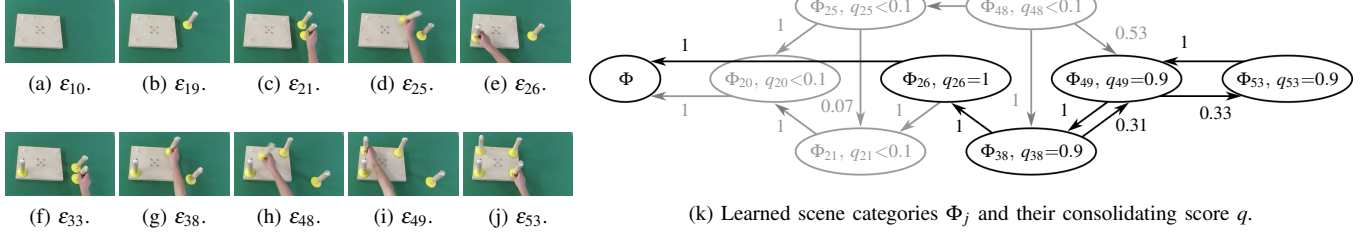
(a) ε_{10}.   (b) ε_{19}.   (c) ε_{21}.   (d) ε_{25}.   (e) ε_{26}.

(f) ε_{33}.   (g) ε_{38}.   (h) ε_{48}.   (i) ε_{49}.   (j) ε_{53}.

(k) Learned scene categories Φ_j and their consolidating score q.

Figure 4: The demonstration of a table assembly task and the bootstrapped representation. Figures (4a-4j) show some of the scenes perceived during the demonstration, and Figure (4k) shows the representation bootstrapped in the robot's memory when the demonstration ended. For clarity, the figures denote scenes ε and categories Φ with an index related to the time instant of the demonstration. Also, each scene category encodes a symbolic representation of the related scene (*i.e.*, how many legs have been connected to the table), which is not shown in the figures. Note that the grey categories in the memory graph have been forgotten, while the other nodes show the scenes that have been consolidated, *i.e.*, the persistent scenes and sub-scenes that are relevant for bootstrapping a representation of the assembly task.

scales exponentially with the number of nodes in M_{t-1}. This also holds for C because both exploit the fuzzyDL reasoner, which makes Algorithm 1 more complex than Algorithm 2.

VII. MEMORY EVOLUTION DURING A DEMONSTRATION

A. Experimental Setup

SIT was configured with an input interface involving one symmetric relation $\mathbf{r} \sqsupseteq \{\text{connected}\}$, and two types of element $\Gamma \sqsupseteq \{\text{CONNECTOR}, \text{LEG}\}$. From point clouds, we perceived the centre of mass of the objects projected in the workbench. Given the 2D position of two objects associated with γ_x and γ_y, we considered them connected with a degree that is linearly dependent on their distance d_{xy}, *i.e.*, $\langle(\gamma_x, \gamma_y):\mathbf{r}_1, 1\rangle$ if $d_{xy}=0$, whereas $p^{(\gamma_x, \gamma_y):\mathbf{r}_1}=0$ if $d_{xy}>0.15$m.

The robot observed a human performing an assembly task involving a *table* having four connecting points, *i.e.*, each one is a CONNECTOR where a LEG should be fixed. The objects were placed in unknown locations and order. The task was completed when all the legs were properly placed. From a demonstration, we collected 68 scenes[4] with a rate of 2.5Hz (some of which are shown in Figure 4) that were encoded in $\varepsilon_1 \dots \varepsilon_{68}$ instances (through Line 1.2).

With a fuzziness of $a = 0.4$, and an initial score of $q^0 = 0.5$, we set Algorithm 1 to store new categories even if the classification had a high degree (*i.e.*, $u = 0.9$) and similarity (*i.e.*, $o = 0.8$). Hence, we tended to store many categories in the memory, and we relied on consolidating and forgetting functionalities to avoid over-fitting, *i.e.*, maintaining only the relevant scenes for the task. Also, we consolidated scenes that are well classified (*i.e.*, $e = 0.9$), even if partially (*i.e.*, $f = 0.2$) to consolidate sub-scenes. We invoked Algorithm 2 with $l = 10$ and $g = 0.1$ to evaluate scores and forget weak categories every five encoded scenes.

B. Store, Consolidate, and Forget Memory Items

Figure 4k shows the task, as well as the consolidated and forgotten categories, as stored in the memory of the robot after the demonstration. In particular, it is possible to derive an undirected graph having only the edges with a weight equal

[4]We evaluate a dataset available at https://github.com/buoncubi/HRI_tasks_dataset, where other executions of similar tasks are also concerned.

to 1, and we can deduce the assembly task as the sequence Φ_{26}, Φ_{38}, Φ_{49} and Φ_{53}, where in (*i*) Φ_{26} a leg is connected, *i.e.*, connected LEG.$^a\Omega(.99)$, (*ii*) Φ_{38} *almost two* legs are connected, *i.e.*, connected LEG.$^a\Omega(1.85)$, (*iii*) Φ_{49} *three* legs are connected, *i.e.*, connected LEG.$^a\Omega(2.55)$, (*iv*) Φ_{53} *four* legs are connected, *i.e.*, connected LEG.$^a\Omega(3.47)$. The restriction was lower than 3 and 4 due to perception vagueness, *i.e.*, on average $p^{(\gamma_x, \gamma_y):\mathbf{r}_1} = 0.75$ instead of 1.

Until the 20-th scene, the distances among objects was too high, and SIT was not provided with any inputs, *i.e.*, $p^{(\gamma_x, \gamma_y):\mathbf{r}_1} = 0$. The 20-th scene represented a configuration having two legs connect between each other with a restriction of $\Omega(0.09)$, which increases to $\Omega(0.73)$ in Φ_{21} since the human moved the second leg closer to the first, as shown in Figure 4h. Similarly, we stored categories Φ_{25} (Figure 4d) and Φ_{48} (Figure 4h). However, those categories were forgotten since the related scenes were not persistent during the demonstration, *i.e.*, $\{\Phi_{20}, \Phi_{21}, \Phi_{25}, \Phi_{48}\}$ did not occur in $M^\star$ often, and Line 1.6 would not be used to consolidate their score. Instead, SIT persistently classified $\{\Phi_{26}, \Phi_{38}, \Phi_{49}, \Phi_{53}\}$ and, through Line 1.9, we increased their score. Due to normalisation (Line 2.5), the not consolidated categories get a lower score with respect to the consolidated ones, until they become weak, and then forgotten through Lines 2.7 and 2.10.

C. Computation Performance

According to Section V-E, we observed a similarity value $d_\varepsilon^{\Phi_j} > 1$ depending on the fuzziness parameter a, the state of the memory M_t, and the current scene ε. We observed this event 37 times over 134 measurements with $d_\varepsilon^{\Phi_j} < 1.036$.

The computation required by an Intel i5 (2.53GHz) to compute Algorithm 1 was 0.55 ± 4.5s on average. The variance was due to OWL reasoning complexity demanded by the structuring and classification functions. The worst-case reasoning time strongly limited SIT, but the average computation time was significantly lower. Noteworthy, the characterisation of incremental OWL reasoning time is far from trivial also because it dependents on previous states. Therefore, to properly characterise the average complexity, a systematic evaluation of different memory graphs is required.

We did not observe any inconsistency when the scenes were perceived up to 0.31Hz (*i.e.*, 8 times slower) since we

were able to sample and process all the relevant scenes of the demonstration, which held for 27.2s. Thus, the ability to compute our framework online is also related to synchronisation issues concerning human speed during the demonstration.

VIII. DISCUSSION

We previously pointed out that our framework can address all the requirements presented in Section II. Indeed, it can learn structured models (*i.e.*, it bootstraps graphs of categories in the ontology) from non-annotated data, which are generated through a single demonstration. Since we used a symbolic formalism, the learned models are intelligible, and they can be refined during the demonstration. However, these models should be presented in a simplified form to result more intuitive to a human. Also, we showed that our framework can bootstrap knowledge while the robot observes the demonstration, *i.e.*, it can be used online. However, our framework is limited by the OWL reasoning complexity since it increases exponentially with the size of the memory graph and the number of symbols in the input interface, *i.e.*, w and v.

Thanks to the OWL symbolic formalisation, we could base the framework on a general-purpose input interface, which should be carefully designed *a priori*. In order to support a general-purpose design, we have shown how the learned model changes based on two different contexts addressed in Figures 3 and 4. The main limitation of the input interface is that it requires elements of the environment that are related to each other. Nonetheless, the symbolic formalisation supports the extension to more complex semantics (*e.g.*, reify colour and affordance: in BLUE CONTAINER) with the drawback of increasing the computation complexity. The framework can be used with specific input interfaces to model different contexts from the same observations. This allows implementing different types of memory, such as short-term or long-term memory, which also involve declarative memories, *e.g.*, semantic and episodic [33].

To use our framework in a real application, we need other components to map the numerical data of sensors and actuators into symbols. With a fuzzy formalisation of symbols, we have shown that the framework is robust to vague perception, but a drawback is that a cardinality restriction based on the σ-count approach might be ambiguous. Since we based our system on OWL, we can rely on a flexible formalism and exploit other reasoning mechanisms as well; but this would further limit scalability due to computation complexity.

In Section VI, we implemented a rehearsal-like heuristic approach to consolidate persistent scenes and forget sporadic configurations through a scoring strategy. We have shown that this approach avoids over-fitting an assembly task demonstration. However, our heuristics need to be further validated and integrated with other strategies to identify the knowledge to be forgotten, *e.g.*, based on the trace decay theory. For this purpose, Algorithms 1 and Algorithm 2 can be extended based on the SIT functionalities, and a systematic evaluation of different heuristics should be undertaken. In this paper, we used a simple consolidation function based on hand-tuned thresholds, which is hard to generalise. However, we are confident that a deeper evaluation of different demonstrations for specific scenarios can lead to the design of a more general consolidating strategy.

Last but not the least, our heuristics need to normalise the consolidating scores for each scene since it relies on thresholds that have to be configured for some range. This is a drawback because it leads to discontinuous scores over time. Depending on the memory state, the robot might forget a scene when it has just been stored, or it might not forget it if other scenes are not frequently observed. To overcome this issue, it is possible to rely on a scoring strategy that is bounded and does not require normalisation (*e.g.*, that relies on Sigmoid-based heuristics), but this would also require to decrease the scores during the storing and retrieving functionalities explicitly.

IX. CONCLUSIONS

We present a general-purpose framework inspired by cognitive-like memory that encodes, stores, retrieves, consolidates, and forgets scenes on the basis of a learning algorithm that exploits fuzzy symbols into an OWL ontology. The framework classifies scenes from previous observations, and it bootstraps an intelligible structure of qualitative sub-scenes. Also, it can online bootstrap knowledge from non-annotated data acquired during a single human-led demonstration.

With a heuristic-based score mechanism, we could learn a symbolic planning domain for an assembly task, and we showed that consolidation and forgetting could be a strategy to avoid over-fitting the demonstration. We also discussed that our framework can be used to systematically investigate different heuristics (*e.g.*, Sigmoid-based) and, as further work, we want to compare them for demonstrations that involve more than one task.

REFERENCES

[1] F. Wörgötter, C. Geib, M. Tamosiunaite, E. E. Aksoy, J. Piater, H. Xiong, A. Ude, B. Nemec, D. Kraft, N. Krüger, M. Wächter, and T. Asfour, "Structural Bootstrapping—A Novel, Generative Mechanism for Faster and More Efficient Acquisition of Action-Knowledge," *IEEE Transactions on Autonomous Mental Development*, vol. 7, no. 2, pp. 140–154, Jun. 2015.

[2] R. Wood, P. Baxter, and T. Belpaeme, "A Review of Long–Term Memory in Natural and Synthetic Systems," *Adaptive Behavior*, vol. 20, no. 2, pp. 81–103, Apr. 2012.

[3] D. Vernon, *Artificial Cognitive Systems: A Primer*. The MIT Press, 2014.

[4] R. S. Lockhart and F. I. Craik, "Levels of processing: A retrospective commentary on a framework for memory research." *Canadian Journal of Psychology/Revue canadienne de psychologie*, vol. 44, no. 1, p. 87, 1990.

[5] B. Subagdja, W. Wang, A.-H. Tan, Y.-S. Tan, and L.-N. Teow, "Memory Formation, Consolidation, and Forgetting in Learning Agents," in *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, ser. AAMAS '12. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2012, p. 1007–1014.

[6] S. M. Smith, "Environmental Context–Dependent Memory," in *Memory in Context: Context in Memory*. Oxford, England: John Wiley & Sons, 1988, pp. 13–34.

[7] B. J. Underwood, "Interference and Forgetting," *Psychological Review*, vol. 64, no. 1, pp. 49–60, Jan. 1957.

[8] J. Jonides, R. L. Lewis, D. E. Nee, C. A. Lustig, M. G. Berman, and K. S. Moore, "The Mind and Brain of Short-Term Memory," *Annual review of psychology*, vol. 59, pp. 193–224, 2008.

[9] F. Baader, D. Calvanese, D. McGuinness, P. Patel-Schneider, D. Nardi *et al.*, *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge university press, 2003.

[10] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot Programming by Demonstration," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 1371–1394.

[11] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep Reinforcement Learning for Robotic Manipulation with Asynchronous Off-Policy Updates," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 3389–3396.

[12] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep Imitation Learning for Complex Manipulation Tasks from Virtual Reality Teleoperation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 1–8.

[13] Y. Liu, Y. Gupta, P. Abbeel, and S. Levine, "Imitation from Observation: Learning to Imitate Behaviors from Raw Video via Context Translation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 1118–1125.

[14] Y. Duan, M. Andrychowicz, B. Stadie, J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba, "One-shot Imitation Learning," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. USA: Curran Associates Inc., 2017, pp. 1087–1098.

[15] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine, "One-Shot Visual Imitation Learning via Meta-Learning," in *Conference on Robot Learning*, Oct. 2017, pp. 357–368.

[16] G. Theocharous, K. Rohanimanesh, and S. Maharlevan, "Learning Hierarchical Observable Markov Decision Process Models for Robot Navigation," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, vol. 1. IEEE, 2001, pp. 511–516.

[17] C.-P. Lam and S. S. Sastry, "A POMDP Framework for Human-In-The-Loop System," in *53rd IEEE Conference on Decision and Control*. IEEE, 2014, pp. 6031–6036.

[18] P. Goyal and E. Ferrara, "Graph Embedding Techniques, Applications, and Performance: A Survey," *Knowledge-Based Systems*, vol. 151, pp. 78–94, 2018.

[19] M. Tenorth and M. Beetz, "KnowRob: A Knowledge Processing Infrastructure for Ccognition–Enabled Robots," *The International Journal of Robotics Research*, vol. 32, no. 5, pp. 566–590, 2013.

[20] G. Konidaris, L. P. Kaelbling, and T. Lozano-Perez, "From Skills to Symbols: Learning Symbolic Representations for Abstract High–Level Planning," *Journal of Artificial Intelligence Research*, vol. 61, pp. 215–289, 2018.

[21] F. A. Lisi and U. Straccia, "A Logic-Based Computational Method for the Automated Induction of Fuzzy Ontology Axioms," *Fundamenta Informaticae*, vol. 124, no. 4, pp. 503–519, 2013.

[22] C. Hogg, H. Muñoz-Avila, and U. Kuter, "Learning Hierarchical Task Models from Input Traces," *Computational Intelligence*, vol. 32, no. 1, pp. 3–48, 2016.

[23] J. Kirk, A. Mininger, and J. Laird, "Learning Task Goals Interactively with Visual Demonstrations," *Biologically Inspired Cognitive Architectures*, vol. 18, pp. 1–8, 2016.

[24] L. Buoncompagni and F. Mastrogiovanni, "Teaching a Robot How to Spatially Arrange Objects: Representation and Recognition Issues," in *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, New Delhi, India, Oct 2019, pp. 1–8.

[25] L. Buoncompagni, S. Y. Kareem, and F. Mastrogiovanni, "Human activity recognition models in ontology networks," *IEEE Transactions on Cybernetics*, pp. 1–20, 2021.

[26] L. Buoncompagni and F. Mastrogiovanni, "Dialogue-Based Supervision and Explanation of Robot Spatial Beliefs: A Software Architecture Perspective," in *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. Nanjing, China: IEEE, Aug. 2018, pp. 977–984.

[27] ——, "A Framework Inspired by Cognitive Memory to Learn Planning Domains from Demonstrations," in *Proceedings of the 6th Italian Workshop on Artificial Intelligence and Robotics*, vol. 2594. Rende, Italy: CEUR-WS, nov 2019, pp. 13–18.

[28] L. Buoncompagni, F. Mastrogiovanni, and A. Saffiotti, "Scene Learning, Recognition and Similarity Detection in a Fuzzy Ontology via Human Examples," in *Proceedings of the 4th Italian Workshop on Artificial Intelligence and Robotics*, vol. 2054. Bari, Italy: CEUR-WS, nov 2017, pp. 10–15.

[29] L. Buoncompagni and F. Mastrogiovanni, "Incremental bootstrapping and classification of structured scenes in a fuzzy ontology," *arXiv*, vol. 2404.11744, 4 2024. [Online]. Available: https://arxiv.org/abs/2404.11744

[30] F. Bobillo and U. Straccia, "The Fuzzy Ontology Reasoner fuzzyDL," *Knowledge-Based Systems*, vol. 95, pp. 12–34, Mar. 2016.

[31] ——, "On Qualified Cardinality Restrictions in Fuzzy Description Logics under Łukasiewicz semantics," in *Proceedings of the 12th International Conference of Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU 2008)*, vol. 8, Jun. 2008, pp. 1008–1015.

[32] J. Chamorro-Martínez, D. Sánchez, J. M. Soto-Hidalgo, and P. M. Martínez-Jiménez, "A discussion on fuzzy cardinality and quantification. Some applications in image processing," *Fuzzy Sets and Systems*, vol. 257, pp. 85–101, Dec. 2014.

[33] L. R. Squire, "Memory systems of the brain: A brief history and current perspective," *Neurobiology of Learning and Memory*, vol. 82, no. 3, pp. 171–177, 2004, multiple Memory Systems. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1074742704000735