ANN-based Representation Learning in a Lifelong Open-ended Learning Cognitive Architecture

Alejandro Romero
GII, CITIC research center
University of A Coruña
A Coruña, Spain
alejandro.romero.montero@udc.es

Justus Piater
Dept. of Computer Science and Digital Science Center
University of Innsbruck
Innsbruck, Austria
justus.piater@uibk.ac.at

Francisco Bellas
GII, CITIC research center
University of A Coruña
A Coruña, Spain
francisco.bellas@udc.es

Richard J. Duro
GII, CITIC research center
University of A Coruña
A Coruña, Spain
richard@udc.es

***Abstract*— The frontier in robot autonomy is currently Lifelong Open-Ended Autonomy (LOLA). Within these settings, a robot must be able to operate and learn in domains that are unknown at design time as well as reuse knowledge learnt in one domain to facilitate learning in others throughout its lifetime. Achieving LOLA goes beyond learning specific algorithms and puts us squarely in the realm of cognitive architectures; however, most cognitive architectures were not built to address the LOLA problem, and thus, lack components and capabilities that would be required for it. In fact, even though there is a growing literature on learning representations, especially in the framework of reinforcement and deep learning, hardly any cognitive architecture considers the issue of autonomously learning representations, which is a crucial problem to be able to efficiently learn and abstract information when seeking LOLA. This paper provides a vision of the general requirements in terms of learning knowledge representations within cognitive architectures geared towards LOLA and addresses a specific problem in this context: the problem of learning representations that facilitate obtaining world and utility models and deciding on actions in situations where multiple goals can be activated. The work is carried out in the framework of the development of the e-MDB cognitive architecture for real robots operating autonomously in real domains.**

Keywords — cognitive robotics, cognitive architecture, open-ended learning, lifelong learning, state representation learning

I. Introduction

A really autonomous robot should be able to figure out by itself how the domains it finds itself in work and how things can be achieved in them. That is, throughout its lifetime, the robot must be able to find goals compatible with the function the designer wants from it and learn skills leading to those goals in whatever domains it operates, which are often unknown at design time: we thus require open-ended learning autonomy (OLA). Additionally, to make this learning more efficient, one would like these robots to transfer and reuse knowledge learnt in previous domains to facilitate learning and adaptation in new domains throughout their lifetimes. In other words, the objective would be to endow robots with lifelong open-ended learning autonomy (LOLA) so that they can operate robustly in an unsupervised manner.

Achieving LOLA goes beyond specific learning algorithms, such as Reinforcement Learning [1], that allow the robot to learn particular skills or models related to goals provided by a designer. It requires the capability of managing all the knowledge that is learnt so that it can be contextually related and reused, thus facilitating posterior learning and operation. Also, to get the robots to do something in complex domains, there is also a need to manage motivations, contexts, and attention. In other words, for robots to autonomously learn to operate in domains that were not considered at design time and construct on this knowledge to address new domains as they come up during their lifetimes, all the capabilities mentioned above (and probably a few more) must be integrated and regulated. This is the job of cognitive architectures. However, most cognitive architectures were not built to address the LOLA problem, and thus, lack components and capabilities that would be required.

Following the classification of [2], cognitive architectures come in three basic flavors: symbolic, hybrid, and emergent. It is important to note here that possessing the capability of learning is fundamental in order to be able to address LOLA, and not all of the architectures presented in the literature display this capability. In fact, many symbolic architectures have no learning mechanisms and thus their knowledge must be introduced by the designer when it is built, implying that the domains the robot is going to operate in must be known at that point. Examples of these are EPIC [3] or 4CAPS [4]. Others within the symbolic or hybrid group, such as DUAL [5], ADAPT [6], ICARUS [7], incorporate learning capabilities, but mostly through the modification of rules at the higher level, without a versatile and unconstrained capability of creating new rules for new domains. It is only within the emergent cognitive architectures group, such as iCub [8] or MDB [9], and in a small group of hybrid architectures, such as MicroPSI [10], that versatile low-level learning mechanisms can be found. Thus, only these types of architectures could possibly achieve LOLA in robots.

Notwithstanding the previous comments, all the learning systems within these architectures have implicitly or explicitly assumed that the robot cognitive systems were provided with specific and appropriate state space representations by their designers. That is, the designers decided what is relevant from the sensorial stream of the robot and how these relevant features would be represented. Otherwise, given the large number of sensors required by robots to address complex domains and the dimensionality of the data they provide, the spaces in which learning must take place would become huge, making learning very difficult. Consequently, the learning mechanisms of the architectures have focused on how to learn whatever knowledge components the architectures required (forward or inverse models, utility models, policies, etc.) using these predefined state space representations.

Research supported by the Xunta de Galicia and the European Regional Development Funds under grant ED431C 2017/12 and the Spanish Science and Education Ministry through grant RTI2018-101114-B-I00 and the FPU grant of Alejandro Romero. We wish to acknowledge the support received from the Centro de Investigación de Galicia "CITIC", funded by Xunta de Galicia and the European Union (European Regional Development Fund-Galicia 2014-2020 Program), by grant ED431G 2019/01.

In the last few years it has become quite evident that to be able to produce robots with the capability of progressively and efficiently learning from their interactions with the world when in LOLA settings, state spaces [11] as well as the corresponding action spaces [12] need to be learned. There is no way to predesign what the most appropriate state spaces will be given that the domains and particular goals the robot will have to achieve in each domain are not known when designing it. In fact, within a domain and for the sake of learning efficacy, different goals and tasks will probably involve different state space representations (when trying to block an object thrown at a robot, a 2D cartesian representation of its position in time may be an ideal representation, but for throwing objects, a representation in terms of joint angular speeds may be better). Additionally, there is the problem of the usually very high dimensionality of the sensor data (such as image data). To make learning feasible and efficient lower dimensional and more task specific state spaces must be created.

To address these issues in the case of models, a promising approach found in the Reinforcement Learning (RL) literature is to learn the dynamics in a compact latent space. Thus, State Representation Learning (SRL) algorithms are designed to find a way to automatically compress high-dimensional observation data into a meaningful low dimensional latent space [12], the state space in robotic terms. Within this scope, SRL algorithms have been applied to different problems like reconstructing the observations [13], learning a forward model [14], learning an inverse model [15], using feature adversarial learning [16], or exploiting rewards [17]. Achieving this goal directly from raw perceptions is a challenging quest that raises a lot of interest in the field of Deep Reinforcement Learning (DRL) [18], where deep ANNs are used with the aim of obtaining forward and reward models over compressed latent representations that allow performing predictions. This way optimal policies can be obtained fully autonomously.

A very relevant work in the scope of the current paper, albeit not in robotics, is [19], which gained a lot of attention in the field of DRL. In it, the authors show the possibility of training an agent to perform tasks entirely within its simulated latent space. They learn a Variational Autoencoder model to compress the visual input into a small representative code. In addition, they use a model (called memory) to predict the next values of the latent space based on an RNN. Finally, they also learn a policy (controller) using a simple single layer linear model. The automation degree of the whole process was very high, showing that such approach is possible. The system was applied with high success in two video games.

Later, [20] presented PlaNet, a model-based agent that learns a latent dynamics model from image observations and chooses actions by fast planning in latent space. The main novelty of this approach is that of using a model with both stochastic and deterministic paths. They test the algorithm on 20 visual control tasks of the DeepMind Control Suite [21], using $64 \times 64 \times 3$ images, outperforming the best model-free algorithms, mainly because they require a smaller number of episodes, showing the potential of planning in latent space. An improvement of this work is carried out in [22], where they present an algorithm called Dreamer, that learns long-term policies through models operating in latent space. Their method optimizes a parametric policy by propagating analytic gradients of multi-step values back through learned latent dynamics, with successful results in the same test-bed.

It is important to point out that most of these models have been tested on simulated or game like environments, considering a single model or policy, and that the success of the approaches hinges on the assumption of the availability of dense reward functions [23]. This assumption does not hold in LOLA settings where rewards are defined on-the-fly and are usually very sparse. This implies that state space representation learning strategies would need to be coupled with more elaborate approaches at the cognitive architecture level that, for instance, allow leveraging on measures of competence in the form of intrinsic motivations [24][25], to choose the most adaptive task-directed representations of skills. Moreover, LOLA implies online learning of models that are defined during the lifetime of the system, so the typical off-line parameter tuning in SRL is not feasible in this scope.

We have been addressing the LOLA problem during the last few years from a developmental emergent cognitive architecture perspective by means of the development of the e-MDB (epistemic Multilevel Darwinist Brain) cognitive architecture [9][26]. We contend that these processes for autonomously obtaining the most appropriate representations and producing mechanisms to be able to express data that is provided in one representation in another representation (representational redescription), are fundamental and need to be integrated in cognitive architectures for the progressive creation of really intelligent robots. In this paper, we will address the first steps in the incorporation of SRL strategies within a complete cognitive architecture for robots, in this case the e-MDB cognitive architecture. In addition, we will evaluate the different opportunities this provides, especially in terms of generalization and abstraction, allowing the production of general rules over general representations that can be reused and adapted to different domains and situations.

The remainder of the paper is structured as follows: section II is focused on presenting the basic components of the e-MDB, together with the main knowledge elements involved on its operation. This formalization is required to understand the scope of LOLA. In section III, the global implications of SRL in LOLA are established, and the specific ones faced in this paper are described in detail. Section IV describes the experimental setup and the experiment dynamics, to clarify how the learning processes occur in time. Section V is devoted to the presentation and discussion of the experimental results obtained in a simple real robot setup. Finally, in section VI, the main conclusions of this new research line are commented.

II. E-MDB COGNITIVE ARCHITECTURE

A. Architecture overview

The epistemic-Multilevel Darwinist Brain (e-MDB) is a cognitive architecture for lifelong open-ended learning in real robotic systems. It has been under development since the early 2000s and it allows artificial agents to learn from their experience in dynamic and unknown domains to fulfill their objectives. The architecture is made up of three main components:

- A **motivational system** that establishes the robot's motivations and allows it to find new goals and select which ones are active. Its operation is based on domain independent needs and drives [27], since they allow assigning purpose to the robot regardless of the domain in which it works. A detailed explanation of its operation can be found in [28].

- A **learning system** that allows the creation and learning of utility models [29], [30] to re-achieve goals and learn skills associated with them. It is also devoted to learning world models that represent the domains the robot is in. This system works online supporting learning of emergent models.

- A **memory system** that allows storing and relating goals, models and all the knowledge that is generated. As this knowledge is domain dependent, the memory is based on a contextual representation [31] to be able to reuse knowledge in the appropriate conditions.

All these elements were initially tested using state space representations chosen by hand by the designer. However, to provide the architecture with real autonomy, it should be able not only to learn the necessary knowledge elements, but also to autonomously extract a representation with the most important features for the task to be performed. Therefore, the fourth main element of the architecture should be:

- A **representational system** that allows learning appropriate state space representations. It will learn representational redescription functions that automatically compress high-dimensional observation data into low dimensional latent spaces determining the state spaces with which the different components of the robot cognitive architecture will work.

B. *Knowledge representation*

The e-MDB learns and stores the following types of knowledge nodes:

- **Drives:** A drive, D_j, reflects how far the system is from satisfying an internal need (state the robot seeks to achieve or maintain) provided by the designer as an indication of purpose. Although they do not represent knowledge acquired by the robot, they are included for clarity in the explanation. There are two types: Cognitive and Operational. The former, also called intrinsic drives, lead the robot to learn by improving exploration (novelty drives, competence drives [28]). The latter lead the robot towards achieving its domain independent purpose as defined by the designer.

- **Goals**: They represent perceptual situations (points or areas G_r) that when reached within a domain reduce the value of, at least, one drive D_j. These points provide utility.

- **Utility Models** (*UMs*): They are functions associated with a goal G_r, which indicate the probability of reaching it (expected utility, $\hat{u}$) starting from a certain perceptual point S_t modulated by the utility achieved.

$$\hat{u} = UM(S_t) \quad (1)$$

- **World Models** (*WMs*): They are functions that represent the behavior of the domain in which the robot is. They allow predicting the perceptual state S_{t+1} that will result when an action a_i is applied when in state S_t:

$$S_{t+1} = WM(S_t, a_i) \quad (2)$$

- **Policies** (π): A policy π_i is a decision structure associated with a goal G_r that provides the optimal action to be applied when in a certain perceptual point S_t:

$$a_i = \pi_i(S_t) \quad (3)$$

- **P-nodes**: They are discrete entities that group sets of (continuous) perceptions. These perceptions have in common that by applying the same action in the same domain, the same perceptual state S_{t+1} is reached.

- **C-nodes**: The C-nodes or contextual classes are entities that link P-nodes (initial perceptions), with goals (final perceptions) and the Utility Models or policies necessary to go from the former to the latter. They also relate the World Models (different domains) to these nodes. The activation of a given C-node depends on the activation values of the P-node, goal and *WM* connected to it.

The different knowledge nodes are related to each other through the operational structure shown in Fig. 1. This structure is based on C-nodes, represented in red in the figure. The activation of goals will depend on the drives (that are domain independent). The activation of the P-nodes will depend on the perception of the robot, whereas the activation of the *WMs* will depend on the identification of the domain by the robot. In this way, the most active context (P-node, goal, and *WM*) will determine the action or policy to be executed.

All the knowledge nodes depend on the robot's perceptions. Hence the importance of the robot being able to automatically obtain appropriate representations for each domain and task. For a more detailed explanation of the e-MDB operation, please refer to [28].

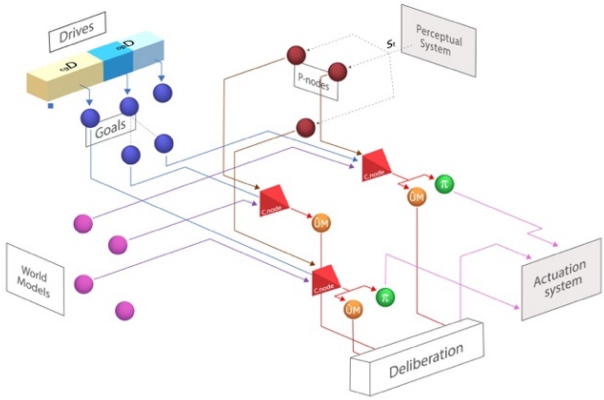

Fig. 1. Simplified schematic of the e-MDB operational structure.

III. Representation and Model Learning

Producing appropriate state space representations within cognitive architectures involves transforming, the often very high dimensional observation flow, from sensor space of the robot to adequate latent representations for the different knowledge components required by the architecture (World Models, policies, Utility Models, etc.). These transformations are carried out by redescription functions ($\mathcal{R}$). In this line, a series of points must be addressed to provide for useful representational learning within cognitive architectures. Some of them are:

- For the sake of learning efficiency, these latent spaces should present the *smallest possible dimensionality that provides the features required* by the particular knowledge component to be able to perform its function, avoiding any kind of distractors. Consequently, the knowledge components of the architecture learn and operate in their latent spaces and are associated to them and, therefore, to their redescription functions.

- Goals and perceptual classes are defined in a given representation (latent space) and this *representation heterogeneity must be handled by the motivational system*.

- Redescription functions ($\mathcal{R}$) are usually implemented as neural networks that transform the information from an, often very high dimensional, observational space (such as that of an imaging system) to a lower dimensional state space. Therefore, they are often large, requiring *many training samples* to be able to train them successfully.

- In a cognitive architecture, different World Models, Utility Models or policies could be obtained for the same

domain/task but operating over different latent spaces produced by *distinct redescription functions*.

- Learning emergent knowledge components in the form of ANNs in latent spaces of unknown dimensionality and complexity implies *learning methods that manage the network structure as well as the weights*.
- When operating in LOLA settings, the assumption of the availability of dense reward functions does not hold. *Knowledge representation learning needs to be coupled with a motivational system*, and to competence based intrinsic motivations to choose the most adaptive task-directed representations of skills.
- Finally, LOLA implies changing domains/tasks throughout the lifetime of the robot. Additionally, the time spent in a domain at one point may not be enough to completely produce the most adequate representation and further future incursions in that domain will be necessary to complete these learning processes. This entails the *need for the contextualization of knowledge representations*.

Many of these points involve architectural aspects for managing multiple redescription functions, densifying rewards, allowing for staged learning, balancing the contributions of components addressing the same issue but using different state representations, etc.

In this paper, we are going to describe a first implementation of state representation learning within the e-MDB by *exploring the learning of world and utility models in latent space based on some of the most extended techniques in SRL* and contemplate the implications this has at the architectural level.

Fig. 2 summarizes the overall learning process that will be analyzed here. It is composed by 3 main stages. The first one is Representation Learning. In this work, we will only consider image sensors obtained from the robot's camera as the raw observational system. In this stage, a SRL process is accomplished that learns a redescription function and provides the latent space. The second stage is devoted to Model Learning. The latent space obtained in stage 1 is stored in a Working Memory and used here for WM and UM learning in online settings. Execution is the third stage. In it, the WM and UM are used in a deliberative process to select the action to be applied by the robot in the environment. Finally, this action is applied through the robot actuation system, providing a new sensorial state that completes this loop. It must be pointed out that stage 3 takes place after stage 1, because a latent space is required, but stage 2 runs in parallel with the other two. The learning processes involved in stage 2 are, typically, highly time consuming, so stage 3 uses the latest available models, without waiting for these learning processes to complete. Section IV contains a description of the learning dynamics.

As it can be observed in Fig. 2, in this approach there is no predefinition of the robot state space, and deliberation is performed in the autonomously obtained latent space. In fact, both WMs and UMs operate over the latent space. Due to the reduction in dimensionality, learning on the latent representation is much simpler than learning on the full observation space. Additionally, learning using the latent representation is cheaper in that it does not require interaction. This setup is completely novel in LOLA setting, being this proposal the main scientific contribution of the current work. The following subsections contain a description of these 3 stages in the scope of the e-MDB.

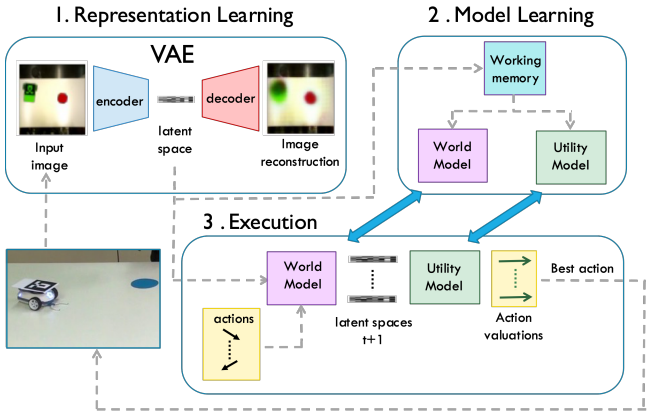

Fig. 2. Overview of the learning process. It consists of a VAE that extracts image features, a WM that models the domain behavior and a Utility Model to evaluate candidate actions.

A. State Representation Learning

To be able to define an appropriate representation from raw observations in a cognitive architecture, one typical option is to use a variational autoencoder (VAE). A VAE has an hourglass structure that learns to make the output the same as the input. The loss function is composed of a reconstruction term (which makes the encoding-decoding efficient) and a regularization term (which makes latent space regular).The VAE allows the compression of raw images, with image features extracted from an intermediate layer (the latent space). As shown in Fig. 2, the Encoder (E) part is the one that provides the desired redescription function. Therefore, the perceptions or states (in the appropriate representation) derived from the robot's observation will be:

$$S_t = E(o_t) \quad (4)$$

The compression level of an autoencoder is given by the dimensionality of the intermediate layer, which determines the latent space. This level is usually set to allow for a quasi-perfect reconstruction of the observations. However, when the purpose of the representation is not reconstruction, but rather, the generation of WMs or UMs, this perfect reconstruction of the observations is not necessary. In a cognitive architecture we seek to produce operational, or goal directed WMs and UMs. That is, models that only contemplate information that is required for performing the tasks the robot is assigned with an accuracy good enough to achieve them. This implies higher levels of compression (lower latent space dimensionalities can be sought) and thus abstraction of the observations.

B. World Model learning

World models are usually seen as predictors of the next observation of the robot. However, within a cognitive architecture they are predictors of the next *state* of the robot at whatever abstraction level it is working. Thus, within a cognitive architecture, a WM is a structure that takes its inputs from a latent space and produces predictions on the next state in that latent space. As such, it is obviously dependent on the latent space in which it learns and thus must be related to the redescription function that leads to it. It will not be valid in a different latent space.

In this work the WM is represented as a simple densely connected ANN whose inputs are directly the learned representation of the state and the action applied, and whose output is the predicted state. To balance network inputs, the low dimensional input actions are repeated into a tensor of the same size as the latent state space. We train the WM to predict

only the difference between (latent) states in time *t* and *t+1*. So that (2) becomes:

$$S_{t+1} = S_t + WM(S_t, a_i) \quad (5)$$

This learning process is performed online here, as required in LOLA settings. Specifically, it consists of introducing the WM between the encoder part and the decoder part of the pretrained VAE and using temporally ordered series of images as a training set (along with the actions applied between images). In this way, the objective will be for the VAE, with the WM modifying the latent space, to be able to reconstruct the image at *t+1*. During the training process the same error function as in the VAE training is used and the VAE weights remain intact. Only those corresponding to WM are varied.

C. Utility Model learning

In the e-MDB, Utility Models are also represented as ANNs, and they can be learned online from the traces experienced by the robot when it reaches a goal. Thus, we can use the information on the trace of states in latent space (perceptions) that were followed to reach the goal as an initial training set to train an ANN as a UM for that goal. Fig. 3 represents this process in a simple setup. Obviously, a single trace will provide a poor model. Therefore, the robot will need to reach the goal from different areas of state space to produce more traces for more complete training. This trace acquisition process must be performed with some exploration method. Once a large enough set of traces with well evaluated states is gathered, the training of the VF should be optimal.

Fig. 3. Schematic representation of the VF learning process. Points in the state space (perceptions) are denoted by p_i. The assigned utility goes from 1 at the goal point and decreases to 0 along the trace.

More details on how the general learning of UMs in open-ended settings that is performed on the e-MDB can be found in [32]. In this paper, the main difference comes from the latent space that is used for state representation, and this specific process will be detailed in section IV.

IV. Real robot experiment

A. Experimental setup

A very simple robotic setup created for the sake of clarity to analyze SRL and its influence on the online learning of the models is shown in Fig. 2. It includes a Robobo robot [33] placed on a table, where there is a red area it must reach.

The robot moves freely on the table at a constant speed, and the actions control its direction of movement. They will modify its orientation by an angle between -180º and 180º. Actions are normalized to [-1, 1] before input to the WM. Regarding perception, we use an RGB camera placed on the ceiling of the room, which produces images of 12288 pixels (height 64 x width 64 x 3 channels). Image data is input to the VAE in its original range [0, 255].

Table 1 shows the construction of the VAE that provided the best results. We use ReLU as the activation functions. Network weights are updated using stochastic gradient descent (Adam optimizer). Raw camera images are compressed into 64-dimensional image features (latent space size). The VAE is trained over 50 epochs with 500 images per epoch (batch size of 32). The parametrization of this VAE has been studied in [34].

The WM has 128 inputs (64 from the latent space and 64 for the action representation), and 64 outputs (predicted latent space). It consists of 5 convolutional layers with 64 neurons each (the representation dimensionality). The activation function is ReLU in the first 4 layers and Sigmoid in the last one. The ANN-based UM parameterization is shown in Table 2. Again, network weights are updated using an Adam optimizer with a mean square error loss function. In this case, the UM has one output (expected utility) and 64 inputs (state representation in latent space). The value of this expected utility is in the range [0,1].

B. Experiment dynamics

The purpose for which the robot has been designed is to reach the target area (red circle), whatever the initial positions of the robot and the red circle on the table. Every time the robot reaches the target area, its operational drive is satisfied, and a new trial is initiated. Robobo starts each trial from a random position, and the target area is also initiated to a random position. In each time step, 10 random candidate actions are generated within the continuous range of actions, from which the robot must choose the best one for its purpose.

The experiment dynamics, based on the stages shown in Fig. 2, is summarized in the following steps:

1) ***Initial steps****:* Initially, the robot explores the scenario driven by an exploratory intrinsic drive that chooses random actions. Hence, the deliberative process shown in Fig. 2 is not used at this moment because no model is available. The robot acquires information related to the domain and generates image sequences in memory that can be used to learn a suitable redescription function to provide an adequate representation. This VAE learning is carried out in parallel with this exploratory operation, using the images obtained in on-line operation of the robot.

2) ***Representation function learned****:* Once the representation function (VAE) has been learned, the robot continues to explore randomly in online fashion (no deliberation yet), obtaining new data in the latent space that is used to learn the WM in parallel with the robot operation. This process finishes when the WM prediction error is below a predefined threshold.

3) ***WM learned****:* Once a reliable version of the WM has been learned, the objective now is to generate traces that permit reaching the operational goal and use this information to produce a UM associated to it. In this case, the robot behavior is guided towards the discovery of unvisited states in its latent space through its novelty cognitive drive. Robot actions (in fact the candidate states that are prospectively generated with them) are evaluated using Novelty. Formally, the novelty of the *k-th* candidate state $S_{c,k}$ is:

$$Nov_k = \frac{1}{M}\sum_{i=1}^{M} dist\left(S_{c,k} - S_i\right)^n \; with k = 1 \; to \; N \quad (6)$$

where *n* is a coefficient that regulates the balance between the relevance of distant and near states, S_i is the *i-th* state in the trajectory buffer and *N* is the number of candidate states considered. At this point the e-MDB is already using the deliberation process for action selection, but the utilities of the final states achieved applying the actions are given by (6), because there is no goal-specific UM available.

TABLE I. STRUCTURE OF THE VAE

Layer	Input	Output	Processing	Kernel Size	Stride	Padding
1	(64,64,3)	(31,31,32)	convolution	(4,4)	(2,2)	no
2	(31,31,32)	(14,14,64)	convolution	(4,4)	(2,2)	no
3	(14,14,64)	(6,6,64)	convolution	(4,4)	(2,2)	no
4	(6,6,64)	(2,2,128)	convolution	(4,4)	(2,2)	no
5	512	64	fully connected	-	-	-
6	64	1024	fully connected	-	-	-
7	(1,1,1024)	(5,5,64)	deconvolution	(5,5)	(2,2)	no
8	(5,5,64)	(13,13,64)	deconvolution	(5,5)	(2,2)	no
9	(13,13,64)	(30,30,32)	deconvolution	(6,6)	(2,2)	no
10	(30,30,32)	(64,64,3)	deconvolution	(6,6)	(2,2)	no

TABLE II. PARAMETERIZATIONS OF THE ANN-BASED UM

ANN Parameter	Value
Input neurons	64
Output neurons	1
Hidden layers	[64, 16, 6, 2, 1]
Activation function	*tanh*
Batch size	Trace length
Training epochs	10

4) ***Utility Model learned****:* From this moment on, the system has learned all the required knowledge nodes. Candidate states (derived from candidate actions) are evaluated using the deliberative process described in Fig. 2, and the one that provides the highest utility is chosen. This allows the robot to consistently achieve the goal, whatever its position on the table. WM and UM are continuously updated and improved as new data is obtained from robot interactions.

Note that in this example there is only one domain and one possible type of goal. However, the functions are trained to generalize over the goal position. For multiple domains and goal types, there would be multiple concurrent learning processes of redescription functions, WMs, and UMs.

V. RESULTS AND DISCUSSION

A. Experimental results

Fig. 4 shows the performance of the robot (number of steps needed to find the goal) in a single run and the different learning stages it has gone through for 5000 iterations. Each iteration corresponds to the execution of an action. The different vertical lines indicate the learning periods of the representation function, the WM, and the UM. These learning processes make the robot improve its efficiency in reaching the goal. It is possible to see how during the learning of the redescription function and the modeling of the domain (section from the beginning to the second vertical line), the behavior of the robot is guided by randomly chosen actions. These movements allow it to create an image dataset and make it reach the goal from time to time. However, when the WM is learned the robot starts to predict the results of its actions and using novelty reaches the goal more consistently. This allows it to create a better set of traces to use for UM training. Similarly, when the robot learns the UM, the average number of steps needed to reach the goal converges (around 5 steps), as now the robot becomes very efficient in its deliberative selection of actions. Thus, as the robot acquires knowledge, there is a clear decrease in the steps necessary to reach the goal.

Fig. 5 shows the results obtained for 30 independent runs of the experiment using the proposed methodology. It includes the values of the median and quartiles 1 and 3 resulting from the 30 runs. For reasons of clarity, the figure shows intervals with the steps necessary to achieve 5 goals. This graph verifies in a more statistical manner that the trend shown in Fig. 4 is maintained. It is important to note here that learning the WM directly from the observations was unsuccessful, obtaining very poor results. However, obtaining a WM from the latent space representation was consistently successful (see Fig. 5).

Finally, Fig. 6 displays the results of the 30 executions of the experiment in terms of accumulated rewards. It shows the values of the median and quartiles 1 and 3 of the accumulated reward (goal achievements) obtained through the iterations for the 30 executions of the experiment. Therefore, the greater the slope of the line, the less time it takes to reach the goal. In the figure, the different learning periods are also marked with vertical lines. Again, it can be seen how as the robot learns the redescription function together with the WM and then the UM, the time needed to correctly solve the task decreases.

To illustrate the learning of the redescription function and the World Model., Fig. 7 shows a series of robot traces to the goal, along with the states predicted by the WM, which were reconstructed into observations using the decoding part of the autoencoder. The predictions are quite accurate even though the reconstruction in terms of image quality is not very good (reconstructing the image was not the objective at the task level). It can also be noticed that the WM fails to predict the states at the end of the image sequences. Since, at that moment, the robot and the red object are placed in a new, unpredictable,.

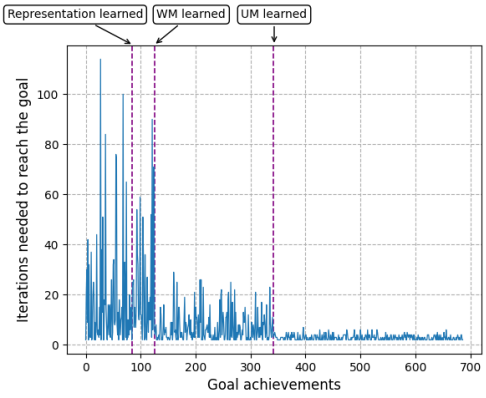

Fig. 4. Goal achievements during a specific open-ended learning process.

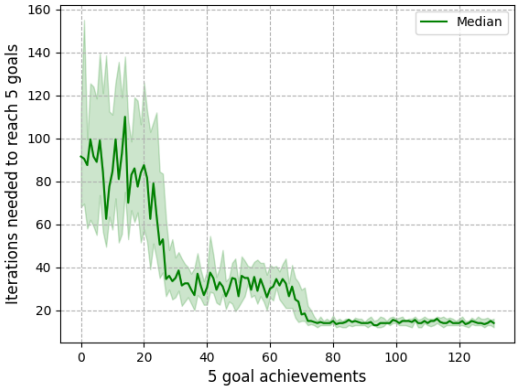

Fig. 5. Iterations needed to achieve 5 goals for 30 runs of the open-ended learning process (median and 25 and 75 percentiles).

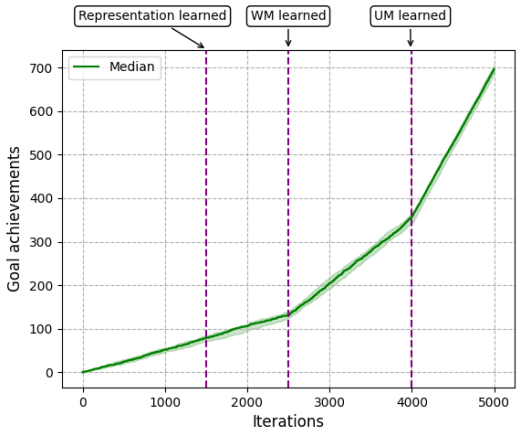

Fig. 6. Performance analysis for 30 runs of the experiment (median and 25 and 75 percentiles).

B. *Discussion*

The results presented in the previous section indicate that it is possible to incorporate SRL strategies into cognitive architectures for robots operating in LOLA settings, however, new issues that are not contemplated by the SRL community arise. In this simple scenario, a robot guided by the e-MDB cognitive architecture has been able to automatically obtain a representation function that has allowed the learning of the rest of the knowledge nodes of the architecture required for this task. This in turn has gradually improved the efficiency of the robot in terms of solving the task.

During the performance of the different experiments, it has been possible to verify that, on occasions, the representation predicted by the WM may be distorted. This may lead the learned UM to misvalue certain actions, since the predicted latent spaces may contain errors. A possible solution would be to study the feasibility of learning a representation and a WM in parallel, which implies being able to stop the learning of the autoencoder (although it does not reconstruct the image) as soon as the WM is adequate. In fact, it would be very interesting to completely do away with the full autoencoder an just concentrate on obtaining the encoding part (redescription function) by designing appropriate loss functions. Another option may be to use the learned deliberative models (UM and WM) to learn a policy that allows obtaining the action to be applied from the current latent state, without the need to make predictions that may result in erroneous evaluations. Furthermore, it would also be interesting to study how the robot can obtain the size of the representation autonomously.

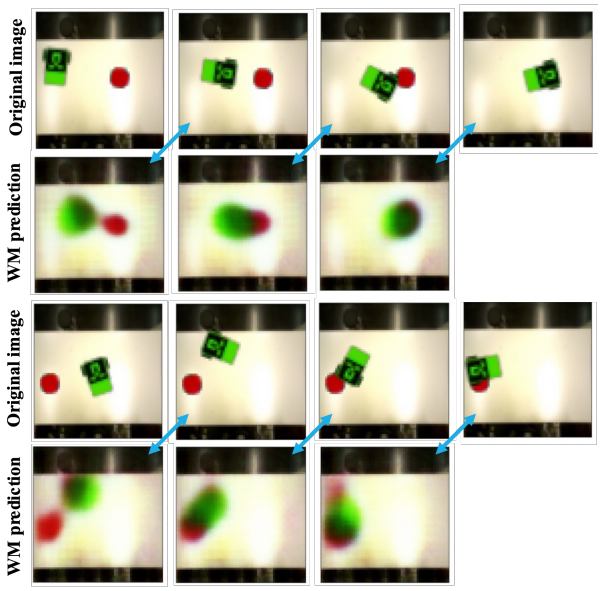

Fig. 7. Two WM prediction examples. Top rows: original images. Bottom rows: reconstructed images from the next state predicted by the WM.

On the other hand, even though in this example there was only one domain and task, LOLA implies facing different worlds and with different goals, which also makes it necessary to study how to manage multiple representations within the cognitive architecture. As it is a developmental process, the change of representation would probably not be done with a single redescription function, but by progressively chaining redescription functions that gradually reduce the dimensionality of the state space. All these functions would be valid at different levels of abstraction (and therefore would give rise to different WMs and UMs that would also work at these levels) and allowing the robot to face increasingly complex tasks and domains. However this brings up issues of consistency and the need for selecting the appropriate level of abstraction for a particular task. All of these issues need to be addressed in future work.

VI. Conclusions

This work represents a first step towards the introduction of representation learning within a cognitive architecture. In this line we have tested a direct VAE type of path towards obtaining the representation, much in the same way as many SRL papers, but within the e-MDB cognitive architecture to verify the effect this would have over the learning of different knowledge components of the architecture. We have concentrated on obtaining world models and utility models in order to construct deliberative processes. The results are very promising, but this experiment has brought up a series of issues on the incorporation of representation learning within cognitive architectures that have not been contemplated by the SRL community, in particular, all the problems of managing multiple representations and the need for their contextualization so that they can be reused.

References

[1] R. S. Sutton and A. G. Barto, “Introduction to Reinforcement Learning,” *Learning*, vol. 4, no. 1996, pp. 1–5, 1998, [Online]. Available: http://dl.acm.org/citation.cfm?id=551283.

[2] I. Kotseruba and J. K. Tsotsos, “A Review of 40 Years of Cognitive Architecture Research: Core Cognitive Abilities and Practical Applications,” *Artif. Intell. Rev.*, vol. 53, no. 1, pp. 17–94, 2020.

[3] D. E. Kieras, G. H. Wakefield, E. R. Thompson, N. Iyer, and B. D. Simpson, “Modeling Two-Channel Speech Processing With the EPIC Cognitive Architecture,” *Top. Cogn. Sci.*, vol. 8, no. 1, 2016, doi: 10.1111/tops.12180.

[4] S. Varma and M. Just, "4CAPS: An adaptive architecture for human information processing," in *AAAI Spring Symposium - Technical Report*, 2006, vol. SS-06-02.

[5] A. Nestor and B. Kokinov, "Towards Active Vision in the DUAL Cognitive Architecture," *BulDML Inst. Math. Informatics*, vol. 11, no. 1, 2004.

[6] D. P. Benjamin, D. Lyons, and D. Lonsdale, "ADAPT : A cognitive architecture for robotics," in *Proc. 6th International Conference on Cognitive Modelling*, 2004.

[7] P. Langley and D. Choi, "A unified cognitive architecture for physical agents," in *Proceedings of the National Conference on Artificial Intelligence*, 2006, vol. 2.

[8] N. G. Tsagarakis *et al.*, "iCub: the design and realization of an open humanoid platform for cognitive and neuroscience research," *Adv. Robot.*, vol. 1, pp. 1–25, 2007.

[9] F. Bellas, R. J. Duro, A. Faiña, and D. Souto, "Multilevel darwinist brain (MDB): Artificial evolution in a cognitive architecture for real robots," *IEEE Trans. Auton. Ment. Dev.*, vol. 2, no. 4, pp. 340–354, 2010, doi: 10.1109/TAMD.2010.2086453.

[10] J. Bach, "MicroPsi 2: The next generation of the MicroPsi framework," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2012, vol. 7716 LNAI, doi: 10.1007/978-3-642-35506-6_2.

[11] S. Kim, A. Coninx, and S. Doncieux, "Kim, S., Coninx, A., & Doncieux, S. (2021). From exploration to control: learning object manipulation skills through novelty search and local adaptation.," *Rob. Auton. Syst.*, vol. 136, p. 103710, 2021.

[12] T. Lesort, N. Díaz-Rodríguez, J.-F. Goudou, and D. Filliat, "State Representation Learning for Control: An Overview," *Neural Networks*, vol. 108, pp. 379–392, 2018.

[13] S. Alvernaz and J. Togelius, "Autoencoder-augmented Neuroevolution for Visual Doom Playing," *ArXiv Prepr.*, p. 1707.03902, 2017.

[14] M. Karl, M. Soelch, J. Bayer, and P. van der Smagt, "Deep variational bayes filters: unsupervised learning of state space models from raw data," *ArXiv Prepr.*, p. 1605.06432., 2016.

[15] A. Zhang, H. Satija, and J. Pineau, "Decoupling dynamics and reward for transfer learning," *ArXiv Prepr.*, p. 1804.10689, 2018.

[16] E. Shelhamer, P. Mahmoudieh, M. Argus, and T. Darrell, "Loss is its own reward: Self-supervision for reinforcement learning," 2019.

[17] S. Parisi, S. Ramstedt, and J. Peters, "Goal-driven dimensionality reduction for reinforcement learning," in *IEEE International Conference on Intelligent Robots and Systems*, 2017, vol. 2017-September, doi: 10.1109/IROS.2017.8206334.

[18] A. Plaat, W. A. Kosters, and M. Presuss, "Model-Based Deep Reinforcement Learning for High-Dimensional Problems, a Survey," *ArXiv Prepr.*, p. abs/2008.05598, 2020.

[19] D. R. Ha and J. Schmidhuber, "World Models," *ArXiv Prepr.*, p. abs/1803.10122, 2018.

[20] D. Hafner *et al.*, "Learning Latent Dynamics for Planning from Pixels," *ArXiv Prepr.*, p. abs/1811.04551, 2019.

[21] Y. Tassa *et al.*, "Deepmind control suite," *ArXiv Prepr.*, p. 1801.00690, 2018.

[22] D. Hafner, T. P. Lillicrap, J. Ba, and M. Norouzi, "Dream to Control: Learning Behaviors by Latent Imagination.," *ArXiv Prepr.*, p. abs/1912.01603, 2020.

[23] E. Banijamali, R. Shu, M. Ghavamzadeh, H. Bui, and A. Ghodsi, "Robust locally-linear controllable embedding," *ArXiv Prepr.*, pp. 1710.05373, 2017, 2017.

[24] G. Baldassarre and M. Mirolli, "Intrinsically motivated learning in natural and artificial systems," in *Intrinsically Motivated Learning Systems: an Overview*, Springer Berlin Heidelberg, 2013, pp. 1–14.

[25] V. G. Santucci, G. Baldassarre, and M. Mirolli, "Grail: a goal-discovering robotic architecture for intrinsically-motivated learning," *IEEE Trans. Cogn. Dev. Syst.*, vol. 8, no. 3, pp. 214–231, 2016.

[26] R. J. Duro, J. A. Becerra, J. Monroy, and L. Calvo, "Context nodes in the operation of a long term memory structure for an evolutionary cognitive architecture," in *GECCO 2017 - Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2017, pp. 1172–1176, doi: 10.1145/3067695.3082465.

[27] N. Hawes, "A survey of motivation frameworks for intelligent systems," *Artif. Intell.*, vol. 175, no. 5–6, pp. 1020–1036, 2011, doi: 10.1016/j.artint.2011.02.002.

[28] A. Romero, F. Bellas, J. A. Becerra, and R. J. Duro, "Motivation as a tool for designing lifelong learning robots," *Integr. Comput. Aided. Eng.*, vol. 27, no. 4, pp. 353–372, 2020, doi: 10.3233/ICA-200633.

[29] A. Romero, A. Prieto, F. Bellas, and R. J. Duro, "Simplifying the creation and management of utility models in continuous domains for cognitive robotics," *Neurocomputing*, vol. 353, 2019, doi: 10.1016/j.neucom.2018.07.093.

[30] A. Prieto, A. Romero, F. Bellas, R. Salgado, and R. J. Duro, "Introducing Separable Utility Regions in a Motivational Engine for Cognitive Developmental Robotics," *Integr. Comput. Aided. Eng.*, vol. 26, no. 1, pp. 3–20, 2019.

[31] R. J. Duro, J. A. Becerra, J. Monroy, and F. Bellas, "Perceptual Generalization and Context in a Network Memory Inspired Long-Term Memory for Artificial Cognition," *Int. J. Neural Syst.*, vol. 29, no. 6, 2019, doi: 10.1142/S0129065718500533.

[32] A. Romero, F. Bellas, A. Prieto, and R. J. Duro, "Developmental Learning of Value Functions in a Motivational System for Cognitive Robotics," 2020, doi: 10.1109/IJCNN48605.2020.9206931.

[33] F. Bellas *et al.*, *The robobo project: Bringing educational robotics closer to real-world applications*, vol. 630. 2018.

[34] B. Meden, A. Prieto, P. Peer, and F. Bellas, "First Steps Towards State Representation Learning for Cognitive Robotics," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2020, vol. 12344 LNAI, doi: 10.1007/978-3-030-61705-9_41.