An Extended Reinforcement Learning Framework to Model Cognitive Development With Enactive Pattern Representation

Zhenping Xie and Yaochu Jin, *Fellow, IEEE*

***Abstract*—In order to make machines more intelligent, it is inevitable to understand human-like cognitive development, in which adaptive, autonomous and progressive evolution of cognitive decision-making in interacting with the environment plays a key role. Inspired by enactive artificial intelligence and evolutionary sampling learning, a new cognitive development learning model termed evolutionary enactive learning (EEL) is proposed in this paper. The proposed model is constructed by extending the reinforcement learning framework and introducing the utility-selection theory to guide the coevolution of pattern representation and decision-making policies. Theoretical analysis on the model's validity of EEL is given. To further demonstrate the effectiveness of the proposed method, two simulated cognitive decision-making tasks are designed, in which pattern representation and decision-making must be jointly developed to achieve good cognitive performance. Our experimental results clearly demonstrate that the resulting learning process is rational and effective. Finally, we indicate that the proposed EEL could be readily further extended by introducing existing machine learning techniques to solve more practical applications.**

***Index Terms*—Cognitive decision-making, cognitive development, enactive artificial intelligence, evolutionary utility-selection, pattern representation.**

I. Introduction

RECENTLY, research on computational modeling of human-like intelligence have been attracting increasing attention in the field of artificial intelligence [1]–[3]. In such models, decision-making and performing suitable behaviors by properly recognizing the state of the agent itself and environment [4], [5] is a fundamental. Meanwhile, modeling the cognitive development process, in particular how machines can autonomously develop their cognitive decision abilities, is crucial for understanding human intelligence [6]–[10]. In general, an effective cognitive decision system should comprise two key components: 1) an efficient sensory-state pattern representation model and 2) a good decision-making model [11], [12].

Over the past 30 years, reinforcement learning (RL) has become the most basic way for achieving autonomous decision-making capabilities in artificial systems [13]–[15]. Traditional RL methods mainly focus on how to obtain a convergent reward value-function with respect to a set of decision policies [15]. Accordingly, the learned reward value-function for state-action pairs will be used as *a prior* experience to choose the optimal action for a future perception state. However, how to construct efficient sensory state pattern representation, and how to define optimal decision policies model must be separately considered by means of other machine learning methods. For example, a Gaussian multilayer network is employed to manually integrate the state-action policies and the corresponding value function in [14] for the car-pole balancing problem. In [11] and [12], deep convolutional neural networks [16] are adopted to efficiently represent high-dimensional sensory inputs and the value function of a massive number of decision policies. However, these predefined black-box representation structures cannot be adapted to satisfy with the requirements for explicitly understanding the environmental states and performing sufficiently robust decision-making like the human brain.

According to the viewpoints of enactive artificial intelligence, the constitutive autonomy and adaptivity are necessary for the cognitive development of human life [17]. Here, the constitutive autonomy and adaptivity must contain the variability of pattern representation structures of sensory-states and decision-making policies. To imitate the above-mentioned characteristics of human cognitive development, evolutionary enactive learning (EEL) is put forward in this paper by extending the RL framework. In EEL, an extra utility probability value is allocated to each possible pattern representation item and decision policy item, where utility probability values indicate the statistical usefulness of all possible items according to historical experience. Then, the utility probability values can be used to select the most useful memory items, which is different from the value function used in conventional RL. Moreover, evolutionary exploration and selective memory can be implemented to evolve state pattern representation and

Manuscript received May 15, 2017; revised November 20, 2017; accepted January 16, 2018. Date of publication January 23, 2018; date of current version September 7, 2018. This work was supported in part by the China Scholarship Program for Visiting Scholar, in part by the Jiangsu Province Natural Science Foundation of China under Grant BK20130161, in part by the National Key Technology Support Program of China under Grant 2015BAH54F01, and in part by the National Natural Science Foundation of China under Grant 61572236. *(Corresponding author: Yaochu Jin.)*

Z. Xie is with the School of Digital Media, Jiangnan University, Wuxi 214122, China, and also with the Jiangsu Key Laboratory of Media Design and Software Technology, Jiangnan University, Wuxi 214122, China (e-mail: xiezhenping@hotmail.com).

Y. Jin is with the Department of Computer Science, University of Surrey, Guildford GU2 7XH, U.K. (e-mail: yaochu.jin@surrey.ac.uk).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the author.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCDS.2018.2796940

2379-8920 © 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

decision policies by means of integrating a utility-selection strategy into the evolutionary sampling learning (ESL) method.

Similar to considerations in [11] and [12], the core design of EEL is to close the loop between the objectives of machine learning and action rewards. In deep RL [11] and alphaGo [12], action rewards are converted into an expectation value function, and the learning objective is defined as the optimal approximation to the value function using deep neural network. In EEL, not only reward value functions but also utility probability value functions are designed. Here, the definitions of utility probability value functions conjointly inherit from the ideas of statistical machine learning and RL. Thus, machines may carry out more autonomous statistical learning guided by cognitive action rewards from the surviving environment instead of direct supervised learning [18], which should be pivotal for creating more autonomous and robust artificial intelligence systems.

The rest of this paper is organized as follows. In Section II, related machine learning methods are briefly introduced. In Section III the implementation of EEL algorithm and some theoretical analysis are described. In Section IV, experimental results on two simulated cognitive decision-making games are presented and analyzed. Finally, in Section V some conclusions of this paper are provided.

II. Related Learning Methods

This paper mainly explores the coevolution of cognitive decision-making capabilities along with enactive pattern representation driven by individual actions under environmental constraints, which has so far not yet been studied in the literatures. Even so, some aspects of questions have been touched upon in several machine learning methods. In the following, we will briefly introduce and discuss these methods.

A. Deep Reinforcement Learning

For the decision-making process to be adapted to complex input states like those in Atari games, deep RL method was put forward in [11]. There, deep neural network modeling and Q-learning are seamlessly integrated into a unified learning framework. In the framework, two key strategies are designed: 1) a batch of latest state-action-reward tuples (s_t, a_t, r_t) are considered as the updating training objective of deep neural network and 2) current decision-making system is viewed as the optimal representation of past experience, and new experiences are gathered by autonomously exploring and exploiting possible actions and corresponding environmental rewards. In deep RL, the Q-value function in RL

$$\hat{Q}(s,a) \leftarrow \hat{Q}(s,a) + \alpha\left(r + \gamma \max_{a'} \hat{Q}\left(s',a'\right) - \hat{Q}(s,a)\right) \quad (1)$$

is replaced as follows:

$$Q(\phi(s),a;\theta) \leftarrow r + \gamma \max_{a'} Q\left(\phi\left(s'\right),a';\theta^{-}\right) \quad (2)$$

where s' is the subsequent state of s after action a is executed, $\phi(\bullet)$ refers to a pretransform for primitive system states, and θ^{-}, θ represent the parameters of deep neural networks on consecutive stages. Specifically, the updating approximation objective can be defined to optimize θ by collecting a batch of latest $Q(\phi(s_j), a_j; \theta^{-})$.

Although deep RL can progressively develop and optimize individual decision-making in cases, where the input states are high-dimensional, it could only support a limited number of actions. That is, deep RL may fail to scale up to a huge number of possible actions as seen in human life. In alphaGo [12], extra policy networks and Monte Carlo tree search strategy are introduced to cope with decision evaluation in the presence of huge possible actions. Even so, explicit pattern and policy representation still cannot be realized.

In this paper, a new mechanism, termed utility probability-based evolutionary selection learning strategy is introduced to support the coevolution of explicit pattern and policy representation learning. We consider that the environmental rewards for an agent's actions could be converted into utility probability values for related pattern representation and decision policies. Thus, all learning and optimization objectives can be defined by transforming, distributing, and propagating utility probability values.

B. Evolutionary Optimization

Evolutionary computation [19], [20] is a powerful tool for optimization in control systems [7], [21], [22], machine learning [16], [23], [24], evolutionary robotics [25], and many other applications [26], [27]. In general, optimization problems can be described as the following form:

$$\text{Minimize } f(X), \ \text{ s.t. } X \in S \subseteq R^{D} \quad (3)$$

where $f(X)$ is an evaluable function, S is the corresponding decision space of dimension D. For an cognitive learning problem, the objective is to seek optimal actions or action probabilities from all possible actions for all possible observed inputs. Here, the dimension D should equal to the number of possible inputs multiplied by the dimension of the action space, which may become very large. Conventional evolutionary optimization methods may become less effective because only a limited number of fitness evaluations may be allowed. To address this issue, special evolutionary optimization strategies should be designed for the evolutionary optimization of cognitive development learning, e.g., surrogate-assisted evolutionary optimization [28].

C. Pattern Analysis

Pattern analysis [29], [30] is a fundamental element of human cognitive capabilities, mainly consisting of pattern representation and pattern classification [31]. Pattern representation [32]–[35] refers the way how a pattern is described using numerical forms, while pattern classification [36]–[38] performs the task of distinguishing and recognizing different patterns. In general, the classification of an observed datum x can be formally defined as

$$C_j = f_c(\phi(x),\theta) \quad (4)$$

where, C_j is the index of the pattern class in pattern domain C, $f_c(\bullet)$ is a classification function with parameter θ, and $\phi(\bullet)$

is a pattern representation transform with respect to original physical signals. Moreover, a pattern analysis model should contain pattern feature representation, classification model, and learning method of parameter θ.

For pattern feature representation, it is often manually designed [39], [40] or extracted using statistical learning methods [41], [42]. Nowadays, common classification models include decision tree methods [43], [44], support vector machines [45], [46], and neural networks [16]. In order to train the model parameter θ, unsupervised or supervised learning strategies can be employed [47].

Though large amount of researches have been carried out in past several decades, there are little work on how to develop a type of pattern analysis systems that are able to completely autonomously evolve based on the active interaction with the environment in which the individual is. Recently, deep neural network methods [16], [34] offered a possible way of jointly learning pattern representation and classification provided that a huge number of labeled samples are available. Despite this, the huge number of labeled samples must be prepared by hand or other tools. Lately, deep RL [11] and alphaGo [12] further improved the learning autonomy by combing RL, in which supervision information could be created by consecutively simulating decision actions and getting subsequent environmental rewards.

In this paper, an enactive pattern representation strategy is proposed for explicit modeling of complex pattern representation, which is well complementary to neural network modeling.

D. Evolutionary Sampling Learning

In our previous research [48], a novel machine learning approach within a probabilistic framework, termed ESL was put forward. It was shown that ESL can be used to acquire an approximation representation to any point-wise computable probability function. The learning process of ESL can be formulated as

$$p^t\left(\phi(x);\theta^t\right) \to \pi^t(\phi(x)) \tag{5}$$

for the learning objective $p^*(\phi(x);\theta^*) \equiv \pi^*(\phi(x))$, where $p^t(\bullet;\theta^t)$ is the approximation representation of $\pi^t(\phi(x))$ at time t, and θ^t is composed of a subset of past perceptual data. When the learning procedure of ESL terminates, there has $\pi^{t\to\infty}(\phi(x)) \equiv \pi^*(\phi(x))$ or $\pi^{t\to\infty}(\phi(x)) \to \pi^*(\phi(x))$ for different cases.

Interestingly, ESL could be viewed as a dynamic evolutionary process or an online statistical learning process. Therefore, ESL may be widely applicable to problems those can be converted into probability representation learning problems. In this paper, we will employ ESL to evolve cognitive system by introducing utility-selection strategy.

III. Evolutionary Enactive Learning

This section will first introduce the framework of EEL, followed by the designed details of the evolutionary process in EEL. We then describe the core algorithms used in EEL. Finally, some theoretical discussions on the model are given.

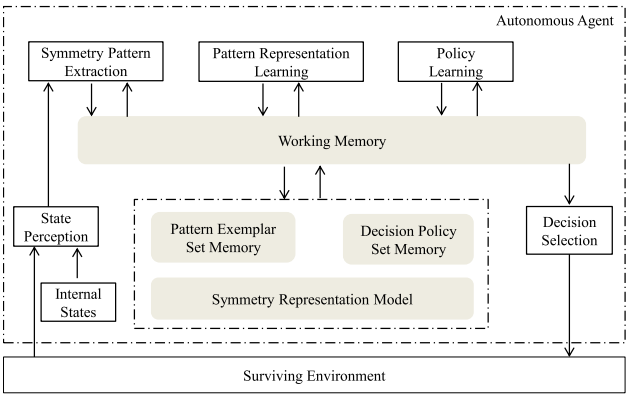

Fig. 1. Framework of EEL.

A. Conceptual Framework

For an autonomously learnable cognitive system and its surviving environment, we suppose that it can continuously perceive two types of information: 1) the environmental states and 2) the internal states of itself. We also assume that a cognitive system may make decision actions that will result in some causal changes to the environmental states and/or internal states. According to the viewpoints of enactive AI [17], there is no other information that could be used in the developmental learning of a fully autonomous cognitive system.

For sake of the convenience, we introduce *cognitive space* as a concept to represent the integrated system composed of intelligent agents and their surviving environment. Thus, a cognitive space will be a closed system, and the cognitive development of an agent in a cognitive space can be seen as a procedure of continuously executing individual utility-selection. Here, utility-selection goals can include, e.g., the optimization of certain function [11], [12], [14], statistically optimal prediction on future system states [4] or optimal surviving robustness [6], [7]. Though cognitive development may have diverse utility-selection goals, it should be an essential task for cognitive development learning to discover and memorize the intrinsic laws embedded in a cognitive space. Accordingly, the core task of pattern representation learning is to computationally model these intrinsic laws, while the results of decision policy learning reflect personalized utility-selection preference.

The generic framework of EEL is illustrated in Fig. 1. The proposed autonomous agent comprises three kinds of memories and five process modules. It evolves its cognitive behaviors by continually interacting with its surviving environment. In addition, a symmetry representation model (SRM) as *a prior* experience is introduced to enactively construct explicit pattern and decision policy representation.

In the proposed EEL framework, an SRM is designed to model all possible pattern exemplars of given input sensory-states and decision policies that may be necessary for the agent to fulfill desired autonomous tasks. In SRM, the symmetry representation strategy aims to realize the enactive constitution

TABLE I
FIVE COGNITIVE FUNCTION MODULES IN EEL

State Perception	Receiving current sensory signals by observing individual surviving environment and the individual itself
Symmetry Pattern Extraction	Transforming sensory signals into formal pattern representation based on symmetry representation model
Pattern Representation Learning	Learning and memorizing most useful pattern representation exemplars by means of the utility-selection learning strategy
Policy Learning	Learning and memorizing most useful decision policies by means of the utility-selection learning strategy
Decision Selection	Selecting suitable behavior policy to perform certain actions according to individual utility-selection preference

of explicit pattern representations related to decision policies. To be specific, two types of patterns are considered in SRM, namely, ground patterns and abstract patterns. Here, ground patterns refer to those patterns that can be explicitly represented by means of symmetry information structures on space, time and/or movement positions, which is inspired by the symmetry theory of physical laws in the real world. By contrast, abstract patterns are considered to be represented by a logic combination of ground patterns also with respect to space, time and movement correlation like in our physical world.

Among three types of memories in the EEL framework, working memory is used to store temporary, shared variables required for four process modules as illustrated in Fig. 1. In contrast, pattern exemplar set and decision policy set memories are used to store two types of persistent cognitive knowledge data (individual experience), respectively. For five cognitive process modules, their main functions are explained as in Table I.

To explain the role of each component of the proposed EEL framework in Fig. 1 in plain language, we take a football player starting to learn playing football as an example. As a beginner, he/she has little memory/skill for playing football. As the player practices playing football, he/she must perceive the environmental and internal states for instance, by observing the positions of the football and players, and experiencing the amount of force when touching the football. The above information is represented by the module "state perception" in Fig. 1. To improve the skills, the player must learn the physical laws governing the movements of the football and the playing strategies or behaviors of other players. These are denoted by the module "symmetry pattern extraction" and "pattern representation learning." In addition, the module "policy learning" includes a collection of possible skills for playing football, while the module "decision selection" is the decision-making process that aims to maximize the reward. Finally, different memory units are needed, including "working memory," "pattern exemplar set memory," and "decision policy set memory."

B. Evolutionary Process of EEL

1) Formal Definitions: According to the above conceptual framework of EEL, some formal definitions will be first provided. Without loss of generality, we use $\{x^t\}_{t\leq T}$ to represent all observable states of a cognitive space within a period of time T. Let $x^t = \{x_i^t\}_{i=1,2,\ldots n}$, where x_i^t is a subset of x^t and different subsets may overlap, and n is the number of all subsets. Similarly, we use $\{a^t\}_{t\leq T}$ to represent all actions performed in the past, and let $a^t = \{a_l^t\}$, where $a_l^t \in \mathrm{A}$ and A is the set of all possible actions. Meanwhile, we use $\{r^t\}_{t\leq T}$ to represent obtained reward from the cognitive space, also r^t may contain multiple components. It should be noted that, r^t may be invalid or null in some time instants.

Moreover, we use Φ to denote the set of all possible pattern representation items, where $\Phi = \{<\Phi_k, u_k>\}$ and $\Phi_k = <\{\phi_j(\{x^t\}_{t\leq T}, \theta_j)\}_{j\in C_k}>$, $\tilde{\phi} = \{\phi_j(\{x^t\}_{t\leq T}, \theta_j)\}_{j\in C}$ is called as the set of ground pattern representation, u_k is an utility probability value and C_k is a subset of C (the set of all possible ground pattern representation items). We denote the decision policy set as $\Re = \{R_l = <\Phi_l, a_l, u_l, \hat{r}_l>\}_{l\in L}$, where $\Phi_l \in \Phi$, $a_l \in \mathrm{A}$, u_l, and $\hat{r}_l$ are the utility probability value and the expectation reward value of decision policy R_l. The definition of $\hat{r}_l$ is similar to the expectation reward value designed in RL.

According to above definitions, the learning parameters of a cognitive system will contain pattern representation parameters $\{\theta_l\}$, combination index sets $\{C_k\}$, and decision policy set $\{R_l\}$, if the forms of all possible ground pattern transforms $\{\phi_j(\bullet; \theta_j)\}$ and all optional actions could be predefined as prior knowledge. It should be noticed that, the decision policy set $\Re = \{R_l = <\Phi_l, a_l, u_l, \hat{r}_l>\}_{l\in L}$ could as well be implicitly expressed using a deep neural network as in [11], [12], and [14]. Obviously, $\{r^t\}_{t\leq T}$ is the only supervision information that can be used to guide the evolution of an autonomous cognitive system.

In above definitions, two types of pattern representation are considered, one is ground pattern representation, and the other is abstract pattern representation. Ground patterns reflect the basic pattern structures like a corner or a line considered in machine vision, while abstract patterns reflect high-level pattern structures like a flower or a house. In an autonomous cognitive system, the above considerations are inevitable [34], [49], [50]. Furthermore, decision policy set $\{R_l\}$ may be considered as explicit logic reasoning rules [51], interpretable fuzzy rules [52], or representative experience tuples [11]. In addition, utility probability variables u are newly designed for each pattern representation item and each decision policy item. They are used to reflect the relative memory importance of every required memory item in a cognitive system.

2) Evolutionary Learning Process of EEL: As illustrated in Fig. 2, a cognitive agent can perceive environmental states x_E^t and internal states x_A^t, perform decision behaviors a^t, and

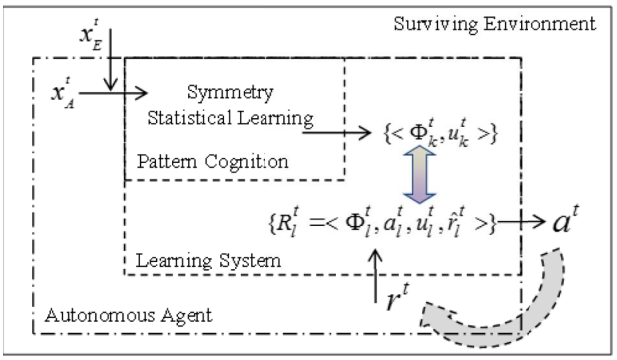

Fig. 2. Evolutionary learning process of EEL.

receive action reward r^t from the cognitive space. Here, we suppose that the r^t value should be ultimately evaluated by the cognitive agent itself. Moreover, the pattern representation transforms $\{\Phi_k\}$ and decision policies $\{<\Phi_l, a_l, u_l, \hat{r}_l>\}$ are all internal components of a cognitive system. They may be gradually evolved by adapting to its surviving environment. So, there exists implied information exchange between the inner and the outer of a module denoted by dash lines.

The above evolutionary learning process contain three core parts: 1) getting new decision experience according to latest states of the cognitive space; 2) updating utility probability values of the current decision policies; and 3) adjusting pattern representation and decision policies. Some detailed implementations will be described below.

In this paper, the symmetry representation is characterized by relation tuples $<\Phi_l^t(x^t), a_l^t, r_l^t>$ and $\Phi_k^t = <\phi_j(\{x^t\}_{t\le T}, \theta_j)>_{j\in C_k}$. We assume that all symmetry relations could be extracted in advance by artificially analyzing causalities existing in the cognitive space. Furthermore, for the satisfaction of computability, all causalities are divided into two types: 1) complete or 2) incomplete. For a given decision policy, the complete causality means that, optimal actions and corresponding expectation rewards can be completely determined by the input. By contrast, input information in a decision policy is insufficient to decide an optimal action for an incomplete case. No doubt that incomplete cases may be more common and widely useful in many practical applications especially in random systems. For example, first-order Markov processes are usually considered to approximately model conditional causalities in which true causalities may be very complicated [53], [54]. In this paper, we consider that all necessary symmetry relations could be predefined, and further research on how to adaptively learn them will be our future work.

Next, because a decision policy $R_l = <\Phi_l, a_l, u_l, \hat{r}_l>$ relates to a valid pattern object Φ_l, then pattern representation learning has to be jointly done together with decision policy learning. The following learning steps are carried out.

1) Newly perceived action rewards r^{t+1} and cognitive space states x^{t+1} are used to construct one new experience $<\Phi^{t+1}(x^{t+1}), a^{t+1}, r^{t+1}>$ by means of given SRM.
2) New reference decision rewards r_l^{t+1} are evaluated for every leant decision policies to form $<R_l^t, r_l^{t+1}>$.
3) decision policies $\{R_l^{t+1}\}$, pattern representation $\Phi_k = <\phi_j(\bullet, \theta_j)>_{j\in C_k}$ and $\Phi = \{<\Phi_k, u_k>\}$ are jointly optimized by means of utility-selection learning strategy.

In summary, the proposed framework makes it feasible to guide the joint learning of pattern representation and decision policies only using decision reward as supervised information. Clearly, the above learning process is considerably different from existing traditional machine learning methods. Specifically, the utility-selection learning strategy is introduced to solve explicit learning of pattern representation and decision policies.

C. Core Algorithms

In this section, we will introduce the core algorithms necessary for implementing the framework of EEL. According to the above discussions, all required symmetry relationships could be predefined. Here, the learning updating could be triggered when a new experience item $<\Phi^t(x^{t+1}), a^{t+1}, r^{t+1}>$ is acquired. For the updating of the expectation reward $\hat{r}_l^{t+1}$ in R_l^{t+1}, we directly borrow the updating strategy of Q-value function $Q^{t+1}(\Phi_l^{t+1}, a_l^{t+1})$ designed in RL. Moreover, Inspired by the process of human decision development [55], utility-selection learning strategies will be used to progressively improve decision-making and pattern representation. Several studies in experimental psychology by Rieskamp and Otto [55] indicated that human participants could use strategy selection theory to gain nearly the best predictions after sufficient RL loops. Next, based on utility-selection learning strategy, several core algorithms of EEL are discussed.

1) Utility Value Updating: In EEL, utility probability values are introduced to guide memory selection of pattern representation and decision policies. In principle, those items with higher utility probability values should be preferentially stored. On the contrary, those items with low utility probability values may be forgotten or replaced by new useful item. The utility probability value updating equations for pattern items and decision policy items can be defined as follows, respectively:

$$u_k^{t+1} = \lambda_1 \times u_k^t + \mu_1\left(<\Phi^t\left(x^{t+1}\right), a^{t+1}, r^{t+1}>, \Phi_k^t\right) \quad (6)$$

and

$$u_l^{t+1} = \lambda_2 \times u_l^t + \mu_2\left(<\Phi^t\left(x^{t+1}\right), a^{t+1}, r^{t+1}>, R_l^t\right) \quad (7)$$

where $\lambda_1, \lambda_2 \in [0, 1)$ are forgetting factors. Furthermore, we may consider $\mu_1(<\Phi^t(x^{t+1}), a^{t+1}, r^{t+1}>, \Phi_k^t) = \mu_1(r^{t+1})$ if $\text{dist}(\Phi^t(x^{t+1}), \Phi_k^t(x^{t+1}))$ is the minimum distance for all pattern items in $\{\Phi_k^t\}$ and it is less than ε_1, otherwise $\mu_1(<\Phi^t(x^{t+1}), a^{t+1}, r^{t+1}>, \Phi_k^t) = 0$. Similarly, we may consider that $\mu_2(<\Phi^t(x^{t+1}), a^{t+1}, r^{t+1}>, R_l^t) = \mu_2(r^{t+1})$ if $\text{dist}(<\Phi^t(x^{t+1}), a^{t+1}>, <\Phi_l^t(x^{t+1}), a_l^t>)$ is the minimum distance for all decision policies in $\{R_l^t\}$ and it is less than ε_2, otherwise, $\mu(<\Phi^t(x^{t+1}), a^{t+1}, r^{t+1}>, R_l^t) = 0$. In practice, we may consider $\mu_{1,2}(r^{t+1})$ equal to constants or r^{t+1}. Thus, the utility probability value may be viewed

Algorithm 1 Decision Policy Learning Algorithm in EEL

1 Given a distance measure on the pair $<\Phi_l^t, a_l^t>$, prior parameters ε, and N_R, let decision policy set $\Re^t = \{R_l^t\} = \emptyset$ in which $t = 0$.
2 For every newly generated experience $R_n^{t+1} = <\Phi_n^t(x^{t+1}), a_n^{t+1}, r_n^{t+1}>$, perform steps 3-8.
3 If $|\Re^t| < N_R$, $\Re^{t+1} = \Re^t \cup R_n^{t+1}$ and go to step 2; otherwise, go to next step.
4 Find the decision policy item R_o^t in $\Re^t$ that has the minimal distance to R_n^{t+1}, then compute $u_n^{t+1} = \mu_2(r^{t+1})$.
5 If $dist_R(R_o^t, R_n^{t+1}) < \varepsilon$, update R_o^t using statistical combination between R_o^t and R_n^{t+1}, and go to step 2; otherwise, go to next step.
6 Re-choose the decision policy item R_o^t in $\Re^t$ with minimum π_o^t, then compute p_o^t, p_n^t, π_n^t.
7 Get a uniform random number z on [0, 1]; if $z < \min\left\{1, \frac{\pi_n^t}{\pi_o^t}\right\}$ receive R_n^{t+1} provisionally, then go to the next step; otherwise exit learning loop and go to step 2.
8 Generate another uniform random number z' on [0, 1], if $z' < \max\{0, 1 - \frac{\pi_o^t}{\pi_n^t} \times \frac{p_n^t}{p_o^t}\}$, replace R_o^t with R_n^{t+1} in $\Re^t$ to form $\Re^{t+1}$.
9 END

as a discounted cumulant on the historical usefulness of a memory item.

2) Decision Policy Learning Algorithm: Here, we discuss how to update decision policies based on utility-selection learning strategy. When the number of storable decision policy items denoted by N_R is bounded, there will exist two type of possible updating cases: 1) slightly adjusting parameters of certain decision policy and 2) replacing an old policy with newly generated policy. Here, ε-approximation is used to decide whether newly coming experience $<\Phi^t(x^{t+1}), a^{t+1}, r^{t+1}>$ could be integrated into existing decision policies. More generally, we employ ESL proposed in [48] to implement utility-selection learning for decision policy updating.

Let the approximation objective $\pi^t(\bullet) \equiv u_l^{t+1}(\bullet)$ and introduce $p^t(\bullet)$ defined as follows:

$$p_l^t = \sum_j \left(\exp\left(-\text{dist}_R^2\left(R_l^t, R_j^t\right)/\sigma_R\right) \times u_j^t\right) \tag{8}$$

where, $\text{dist}_R(\bullet, \bullet)$ denotes the distance between two decision policies, $\sigma_R > 0$ is a scalar parameter, and $\exp(\bullet)$ is the exponential function. So, $\exp(-\text{dist}_R^2(R_l^t, R_j^t)/\sigma_R)$ could be viewed as a correlation coefficient between two decision policies. Then, the ESL objective is defined as to make $p^t(\bullet) \to \pi^t(\bullet)$ on all decision policies. In addition, let p_n^t and π_n^t denote the evaluated and expected utility probability values of newly acquired experience $<\Phi^t(x^{t+1}), a^{t+1}, r^{t+1}>$, respectively, in which $\pi_n^t = u_n^{t+1} = \mu_2(r^{t+1})$.

In summary, Algorithm 1 is proposed for decision policy learning.

In above decision policy learning algorithm, the decision policy replacing strategy originates from ESL approach [48], which enables the decision policies to strike a good balance between different possible options when the number of policy items to be stored is limited. Moreover, this strategy may produce two benefits: 1) discarding those decision policies with small utility probability values and 2) preserving high diversity among different decision policies. In addition, the function $\text{dist}_R(\bullet, \bullet)$ and the statistical combination strategy on R_o^t and R_n^{t+1} should be extra designed for practical applications.

Algorithm 2 Statistical Decision Output Algorithm in EEL

1 Given a distance measurement on (Φ_{oi}, Φ_l), and a current decision policy set $\Re^t = \{R_l^t\}$.
2 For current observed input $\Phi_{ob}(x^{t+1})$, compute w_l^t for every feasible policy.
3 Randomly choose a decision policy R_c^t with probability $\frac{w_c^t}{\sum_l w_l^t}$.
4 Output the action of R_l^t as the current decision action.
5 END

3) Pattern Representation Learning: In EEL, we use utility probability value u_k to determine whether a possible pattern representation Φ_k should be stored in the cognitive agent's memory. For most cases, we may select top N_P items with the highest utility probability values if the number of storable pattern representation items N_P is limited. This will be our default consideration in this paper and could be viewed as a simple variant of the strategy considered in classical RL methods, where all possible pattern representation items are as possible as memorized. Nevertheless, ESL presented in our previous study [48] also could be employed to subtly keep most useful pattern representation items like the decision policy learning algorithm presented above. Finally, existing unsupervised pattern representation learning could also be integrated into the framework of EEL to offer more flexible learning abilities.

4) Decision Selection Algorithm: Here, we further discuss the problem how to use decision policy to choose behavior actions. When input states are continuous, there may be no exact match between new input observation Φ_{ob}^{t+1} and pattern representation items in $\{\Phi_l^t\}$. Usually, there will exist multiple optional decision policies partially suited for the current input. Inspired by statistical inference theory, the statistical decision output strategy is presented. And the following weight value evaluation for every decision policy is designed:

$$w_l^t = \exp\left(-\text{dist}_\Phi^2\left(\Phi_{\text{ob}}^{t+1}, \Phi_l^t\right)/\sigma_\Phi\right)\left(u_l^t\right)^\gamma \tag{9}$$

where, $\gamma \in (0, \infty)$ is a deformation factor, $\text{dist}_\Phi(\bullet, \bullet)$ denotes the distance between two pattern representation items, σ_Φ is a scalar factor. In particular, $\text{dist}_\Phi(\bullet, \bullet)$ may only output ∞ or 0 for the cases, where only exact input match is valid.

Thus, we consider w_l^t as a relative probability value to choose corresponding decision policy R_l^t. Moreover, we propose the decision selection algorithm in EEL in Algorithm 2.

Equation (9) is compatible to the cases that multiple actions are available for the same input state, especially in case of finite discrete input states. In addition, the observation Φ_{ob}^{t+1}

may be information-incomplete. That is, Φ_l^t in R_l^t should contain more information to make an absolutely credible decision. In addition, some variable definitions on $\text{dist}_\Phi(\bullet, \bullet)$ may be allowed for practical problems.

D. Theoretical Discussions

The proposed EEL framework aims to realize a newly autonomous development framework of cognitive capabilities. Theoretical performance analysis on EEL is provided in Appendix A in the supplementary material.

As a preliminary instantiation of the basic idea in enactive AI, the evolutionary mechanism of EEL mainly contains the following three aspects: 1) two persistent memory sets (pattern representation set and decision policy set), which are the representation of the solutions; 2) the utility maximum priority criterion, which is designed as the objective of the evolutionary optimization; and 3) ESL strategy employed as the search and selection strategies of the evolutionary optimization.

Compared to DRL [11], EEL has strong capability of learning explicit pattern and policy representation, which is also motivated from enactive AI model. On top of this, several new evolutionary mechanisms are designed as described above. Consequently, EEL can take into account of not only complex sensory inputs as in DRL, but also a huge number of optional actions as in [12]. In addition, deep representation networks designed in DRL is also available in EEL to model highly complex observation inputs. In summary, EEL could be seen as attempt to imitate the developmental process of human cognitive control. It should be mentioned that the decision policy learning algorithm developed in EEL can be extended to DRL to optimally select a small number of experience tuples.

Compared to other existing RL methods, EEL aims to realize more flexible and explicit (white-box) pattern and policy representation learning framework envisioned in enactive AI [17]. Note that for problems that can be well solved by traditional RL, EEL does not necessarily have advantage in performance.

In addition, as a generic learning process, EEL is open to extensions and specialization for different practical applications. For example, the distance measure methods on pattern items and decision policy items, and combination of representation forms for abstract pattern, and pattern item memory strategy should be flexibly defined in practical applications.

IV. Experimental Studies

A. Simulation Scenarios

In this part, two simulated cognitive decision-making scenarios are designed to examine the performance of EEL. Scenario I is a color stripe sequence cognition game and Scenario II is an optimal coverage selection game. Note that the cognitive space of Scenario I is discrete while that of Scenario II is continuous. Scenario I is constructed by abstracting the main logic rules used in the building blocks game. Scenario II is inspired by skill learning in agricultural planting, in which some natural laws must be considered. For example, the seasons change periodically, the crop yield may be related to the seasonal temperatures, and only one effective planting may be allowed in one year.

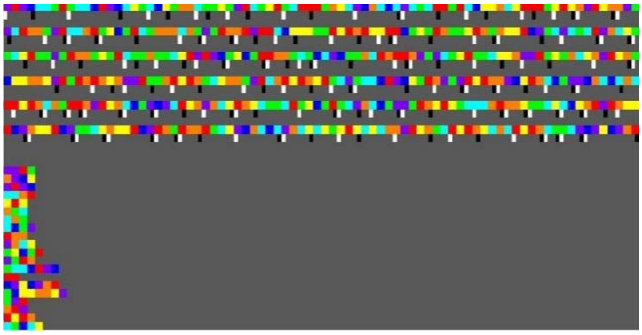

Fig. 3. Illustration of color stripe sequence cognition game.

In the two simulated cognitive tasks, we assume that an intelligent agent is able to develop its cognitive behaviors by continuously perceiving the environmental states, making cognitive decisions, and receiving rewards.

1) Color Stripe Sequence Cognition Game: The color stripe sequence cognition game is illustrated in Fig. 3. In this game, a pattern is represented as a sequence of color stripes with certain sequence. The bottom-left part in Fig. 3 shows 20 such sequence patterns with different sequence lengths.

The rules of the game are described as follows. An intelligent agent can continuously receive sequenced color stripes generated by a random strategy. At any time, it can select a group of consecutive color stripes with a certain length from the current observed scope, and then submit the selected sequence to get an evaluation reward. The top half part in Fig. 3 shows a sequence of color stripes continually generated by randomly jointing true color stripe pattern sequences and noisy color stripes. In Fig. 3, some small white and black blocks below the original color stripes are plotted to visualize the starts and ends of true color stripe pattern sequences.

Thus, the cognitive space of the color stripe sequence cognition game comprises the following three components: 1) a group of true color stripe sequence patterns; 2) positive evaluation rewards for any cognitive submission; and 3) the rule of generating observable sequenced color stripes. The evaluation reward for every submission is 1 if the submitted color stripe sequence is identical to the embedded true pattern sequence; otherwise, the reward value is 0.

The cognitive learning objectives include: 1) to find as many true sequence patterns as possible and 2) to pick out as many as possible correct pattern sequences embedded in observed sequenced color stripes. In addition, we assume that when a submission is performed, all color stripes before submission point will disappear and cannot be used again. Thus, the previous decision submissions will affect future decisions. Note that the above two cognitive tasks may be slightly conflicting with each other.

In addition, seven different colors are used to define color stripes, and the length of each color stripe pattern sequence ranges from 2 to 9. The total number of different true sequence patterns is denoted by N_C, and the total number of true sequence patterns in once simulation is denoted by N_L.

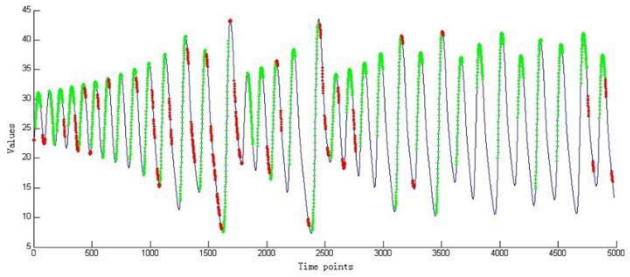

Fig. 4. Illustration of the optimal coverage selection game.

2) Optimal Coverage Selection Game: In this simulated cognitive game, we assume an agent can perceive an 1-D time series, in which one real number can be observed at one time instant. It is expected in the game that an agent can perform optimal coverage selection on every peak zone, in which only once coverage selection will be effective for one peak period.

At first, we employ variable $z(t)$ of the following standard Lorenz chaos system to simulate 1-D time series:

$$\begin{aligned}\frac{dx}{dt} &= \sigma(y-x)\\ \frac{dy}{dt} &= x(\rho-z)-y\\ \frac{dz}{dt} &= xy-\beta z\end{aligned} \tag{10}$$

where $\sigma = 10$, $\rho = 28$, $\beta = (8/3)$, and initial states $[xyz]_{t=0} = [011.05]$. The sampling interval between two subsequent time points is set as$\Delta\ t = 0.005$ (called as a time interval).

Fig. 4 shows a segment of the simulated value curve with a sequence of coverage selection results being attached, in which green segments denote effective coverage selection results while red segments denote failed coverage selection results. For the above periodic curve, we consider that a local varying period starts at one valley point (a local minimal point) and ends at its sequential valley point. Moreover, we call the left half part of a local varying period as up half-period, and the right half part of a local varying period as down half-period. In the following we will introduce the rules of the game in detail.

1) An agent can start a coverage selection at the current time instant if there is no unclosed previous selection, and will get rewards after some time instants for once operation. Once effective coverage selection must start from the up half-period. Otherwise, coverage selection becomes failing and no positive rewards will be given.
2) If once coverage selection is effective, then we set the coverage range as 50 time intervals. If one coverage selection is failing, the coverage range is set to ten intervals. That is to say, if an agent performs a correct coverage selection, then it can gain positive rewards after 50 time intervals. By contrast, the agent will be able to perceive the effect of a failing selection after ten time intervals. Some effective or failing selections' results indicated by green or red color, respectively, are illustrated in Fig. 4.
3) For one effective coverage selection c_i, the reward is defined as

$$\mathrm{Rwd}(c_i) = \sum_{j=1}^{50} Z(Tc_i(j)) \tag{11}$$

where $Tc_i(j)$ denotes the jth time instant after the coverage start of coverage selection c_i. $Z(\bullet)$ is the variable value with respect to every time instant.

For this game, the developmental self-intention of individual cognitive evolution aims to get as much reward as possible and as less failing selection as possible. This is however, under the constraint that only limited memory capacity for storing experience is available to the agent. Usually, there exists one optimal coverage start point for every local varying period and doing one effective selection for one local period only is the best.

3) Discussions: In the two scenarios, we introduce more realistic considerations for real-world problem solving, including requirements on flexible and explicit pattern/policy representation and unknown environmental changes. DRL or existing RL methods will have difficulties in modeling these two problems. For example, for Scenario I, it will be very difficult for DRL to construct balanced training sample set without an explicit pattern representation learning mechanism. Besides, DRL is not able to extract explicit pattern representation, which does not satisfy with the problem requirements. Although the classical RL methods do not have the above issues they are very likely subject to combinatorial rule explosion problem.

Solving Scenario II requires open system states prediction cognition, where decisions need to be made in the presence of uncertainty. Existing RL methods have no joint learning strategies for state prediction in the presence of uncertainty and policy decision required in enactive AI.

By contrast, EEL is equipped with several newly added mechanisms including flexible and explicit pattern/policy representation, and a utility-selection learning strategy, which are essential requirements in enactive AI.

B. EEL Algorithm Settings for the Two Scenarios

As previously indicated, EEL is a generic learning framework for modeling collaborative cognitive development learning. To apply EEL to the two cognitive learning tasks discussed above, problem-specific settings need to be defined. Mainly, three modules: 1) pattern representation learning; 2) decision action selection; and 3) decision policy. learning should be implemented in detail for above two cognitive learning problems.

1) Settings for the Color Stripe Sequence Cognition Game: In the EEL framework, we use tuples $\{<\Phi_k, u_k>\}$ to model pattern representation learning, and those pattern representation items with high utility values should be memorized with a higher priority. Here, the ground pattern representation is denoted by $\phi_j, j = 1, 2, \ldots, 7$, in which the total number of possible color stripes is 7. And a pattern Φ_k is autonomously constructed by combining another pattern Φ_{kj} and a ground pattern. That is $\Phi_k = <\phi_j>$ or $\Phi_k = <\Phi_{kj}, \phi_j>$, in which it

might be possible that $\Phi_0 = \{\}$. In addition, only two possible actions, either to submit or not to submit, can be performed for every observed pattern sequence. Thus, the representation model of decision polies is fully complete according to the Definition 1 in Appendix A in the supplementary material.

Moreover, u_k could be defined as follows:

$$u_k(t+1) = \lambda_1 \times u_k(t) + \mathrm{IS}_k(t+1) \times (1-\lambda_1) \tag{12}$$

where

$$\mathrm{IS}_k(t+1) = \begin{cases} 1, & \text{perceived} \\ 0, & \text{otherwise} \end{cases}$$

indicates whether pattern item Φ_k is perceived at current time instant t or not. The forgetting factor $\lambda_1 = 0.995$ is used in our simulations.

For the decision policies updating, we assume that $\mu(<\Phi^t(x^{t+1}), a^{t+1}, r^{t+1}>, <\Phi_l^t, a_l^t>)$ equals $(1-\lambda_2)$ or 0 for effective or failing decision submission, respectively. Thus, we can define that

$$u_l(t+1) = \begin{cases} \lambda_2 \times u_l(t) + Iu_l(t) \times (1-\lambda_2), & \text{do submit} \\ u_l(t), & \text{do not submit} \end{cases} \tag{13}$$

where

$$Iu_l(t) = \begin{cases} 1, & \text{effective decision} \\ 0, & \text{failing decison} \end{cases}$$

indicates whether the decision is effective or failing. The updating factor $\lambda_2 = 0.6$ is adopted by our experimental comparisons in this paper. Naturally, $u_l(0) = 0.5$ could be set as the default value for any sequence pattern items. It is easy to see $u_l(t) \in (0, 1)$ for any t. In addition, we assume that different pattern sequences are uncorrelated, so $\exp(-\mathrm{dist}_\Phi^2(\Phi_{\mathrm{ob}}^{t+1}, \Phi_l^t)/\sigma_w)$ should be set to 1 if $\Phi_{\mathrm{ob}}^{t+1} \equiv \Phi_l^t$, or 0 if not. Accordingly, $w_l^t = (u_l^t)^{0.5}$ is considered according to Appendix A in the supplementary material. Thus, if one observed color stripe sequence matches the conditional pattern of a decision policy, the cognitive submission may be determined with a probability $[((u_l^t)^{0.5})/((u_l^t)^{0.5} + (1-u_l^t)^{0.5})]$. If more than one pattern sequences in decision policies could be matched at current observation, we may select the one having the highest u_k^t as the candidate decision policy.

2) Settings for Coverage Selection Game: For the optimal coverage selection game, several parts of settings also need to be specified include: pattern representation learning, coverage selection decision policy learning, and decision output strategy.

For the pattern representation learning, a ground pattern item ϕ_j is used to express a variable changing pattern of an up or down half-period, and for simplicity, is denoted as a 2-D vector $(\text{slope}, \text{duration})_j$, where, "slope" denotes the average changing gradient, and "duration" the time length of the corresponding half-period. Moreover, an abstract pattern Φ_k can be represented by a ground pattern or the combination of tuples $<\phi_{k1}, \phi_{k2}, \phi_{k2}>$. u_k in $\{<\Phi_k, u_k>\}$ is defined as follows:

$$u_k(t+1) = \lambda_1 \times u_k(t) + \mathrm{IS}_k(t+1) \times (1-\lambda_1) \tag{14}$$

where

$$\mathrm{IS}_k(t+1) = \begin{cases} 1, & \text{perceived} \\ 0, & \text{otherwise} \end{cases}$$

and $\lambda_1 = 0.995$ is adopted from our experimental comparisons. Moreover, the distance metric on two pattern items is defined by

$$\mathrm{dist}(\Phi_1, \Phi_2) = \max\left(\frac{abs(\Phi_1 - \Phi_2)}{abs(\Phi_1) + abs(\Phi_2)}\right) \tag{15}$$

where, Φ_1 and Φ_2 are two vectors, function $abs(\bullet)$ returns a new vector by getting the absolute values of the input vector's values in each dimension, and function $\max(\bullet)$ outputs the maximum value of a vector. In addition, $\exp(-\mathrm{dist}_\Phi^2(\Phi_{\mathrm{ob}}^{t+1}, \Phi_l^t)/\sigma_\Phi)$ is simplified to be 1 or 0 according to $\varepsilon-$approximate criterion, where $\varepsilon = 0.1$ is used according to our experimental analysis.

For coverage selection policy learning, the decision policies are described as the tuple $<\Phi_l(\phi_{l1}, \phi_{l2}, \phi_{l3}), a_l, u_l, \hat{r}_l>$ in terms of the framework of EEL, where Φ_l is the combination of three consecutive ground pattern items $\phi_{l1}, \phi_{l2}, \phi_{l3}$, a_l is an optimal coverage selection decision vector related to Φ_l, u_l is the decision utility probability value, and $\hat{r}_l$ denotes corresponding weighted reward according to historical experiences. Here, we define

$$u_l(t+1) = \lambda_2 \times u_l(t) + \hat{r}_l(t+1) \tag{16}$$

and

$$\hat{r}_l(t+1) = \beta \times \hat{r}_l(t) + (1-\beta) \times \mu_{\mathrm{Rwd}}(t+1) \tag{17}$$

where, $\mu_{\mathrm{Rwd}}(t+1)$ represents the reward from the current valid coverage selection experience $<\Phi^{t+1}, a^{t+1}, \mu_{\mathrm{Rwd}}(t+1)>$ at time $t+1$. We set $\lambda_2 = 0.995$, $\beta = 0.99$, and $\hat{r}_l(1) = \mu_l(t_0)$ by our experimental analysis, where t_0 is the time instant when the rule R_l was created. We use the relaxed distance threshold $\varepsilon = 0.35$ on Φ to determine whether a near decision policy R_l is updated based on new experience $<\Phi^{t+1}, a^{t+1}, \mu_{\mathrm{Rwd}}(t+1)>$.

Decision variable a_l should also be gradually optimized for this cognitive problem. Here, we assume a_l to be a 30-D vector, in which each element denotes a probability of starting coverage selection at a related time instant within the time covered by $\Phi_l(\phi_{l1}, \phi_{l2}, \phi_{l3})$. Next, the following action optimization strategy is introduced:

$$\mathrm{prob}(a_l, j, t+1) = \begin{cases} \mathrm{prob}(a_l, j, t) \times 1.2, & \text{effective selection} \\ \mathrm{prob}(a_l, j, t) \times 1.0, & \text{no selection} \\ \mathrm{prob}(a_l, j, t)/1.2, & \text{failing selection} \end{cases} \tag{18}$$

where $j \in \{1, 2, \ldots, 30\}$, and 30 intervals are uniformly distributed in the range related to Φ_l.

In addition, the representation form used in coverage selection policies must involve unknown future states. Here, the partial distance metric match is considered between the conditional pattern of decision policies and current observation, and the subpart $<\phi_{l1}, \phi_{l2}>$ of Φ_l is used to compute the match degree between a decision policy and the current observation. Moreover, if more than one decision policy is matched, we choose the one with the maximum u_l as the referenced policy.

TABLE II
PERFORMANCE RESULTS OBTAINED BY EEL AND COMPARED METHODS ON PROBLEM I WITH $N_L = 20000$ AND $N_C = 40$

	Discovery ratio(%)	Recall ratio (%)	Precision (%)
GRC	78.45±6.85	30.44±3.28	42.05±7.85
BRC	93.75±3.98	25.15±2.69	42.60±7.63
EEL	**98.20±2.18**	**54.11±5.04**	**59.32±3.48**

C. Experimental Results

1) Results on the Color Stripe Sequence Cognition Game: In this section, the experimental results on color stripe sequence cognition game will be reported. For comparison, two random cognitive strategies, greedy random cognition (GRC) and balanced random cognition (BRC) will be introduced before we discuss the experimental result.

The GRC learning strategy first seek whether there is a pattern sequence that can be matched to stored pattern sequences. If there is a match, that matched color stripe sequence will be submitted; otherwise, a pattern sequence ranging from 2 to 9 in latest observation will be randomly select to explore new possible color stripe pattern. The BRC learning strategy uses a similar random searching strategy, but it does not always submit matched color stripe sequence. We set the submission probability as 0.8 to balance the requirement of exploring new possible patterns and picking out known patterns. EEL will automatically create all new possible pattern items according to the latest observations. The earlier a pattern is stored, the lower the utility value u will be. In our simulations, limited storable pattern items $N_P = 200$ is set as default, which we found sufficient in our experiments.

In all simulations, we set the number of different true pattern sequences $N_C = 40$, in which the sequence length ranges from 2 to 9. The ratio between the numbers of two types of sequences with lengths of $i + 1$ and i is set as 0.85. Between two embedded true patterns, noisy sequences are added with a probability of 50%. For those sequences with noise being added, the noisy stripe subsequence length ranges from 1 to 5, and possible color stripes are uniformly selected from seven different color stripes.

Three performance evaluation indices are adopted in the comparisons. The index "discovery ratio" indicates the ratio of the number of totally discovered true patterns to N_C. The index "recall ratio" denotes the ratio of successfully picked true patterns, and the index "precision" indicates the correct ratio of cognitive submissions.

Table II lists the simulation results obtained by GRC, BRC, and EEL. The simulation results are averaged over 50 independent runs.

The results in Table II demonstrate that EEL has achieved significantly better cognitive performance compared to the two common random cognitive strategies. By comparing the results obtained by GRC and BRC, we find that there is a negative correlation between the performance of discovery ratio and recall ratio, which is rational. By contrast, EEL can achieve a good balance between exploring new possible patterns and picking out known patterns.

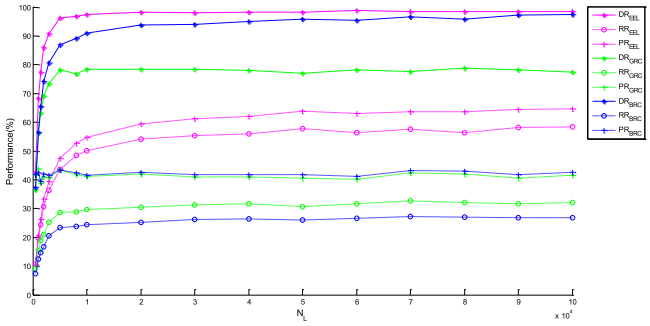

Fig. 5. Performance curves obtained by EEL, GRC, and BRC for the color stripe sequence cognition game with different N_L.

Fig. 5 shows a group of performance curves obtained by EEL, GRC, and BRC on the color stripe sequence cognition game with different N_L. In the figure, DR, RR, and PR stands for discovery ratio, recall ratio, and precision, respectively. By comparing the performance curves of the three cognitive learning strategies, it is clearly that EEL achieves the more significant improvement than the two random strategies GRC and BRC in terms of the overall cognitive development. The tendency of performance curves of EEL demonstrates that it can progressively improve its own cognitive performance, and converge to a stable value. These experimental results indicate that the learning procedure of EEL is effective for color stripe sequence cognition game.

For BRC and GRC, we also find that, the performance indices discovery ratio and recall ratio are conflicting with each other. BRC using the balanced random strategy obtains a better discovery ratio but a worse recall ratio, compared to GRC using greedy random strategy. This means that the decision criterion used in the random cognitive strategies are unable to well balance the requirement of exploring new patterns and exploiting old patterns. More adaptive or flexible decision policies must be considered like in human-like cognitive behavior. Naturally, the flexible cognitive decision policies composed of precondition, actions, and expectation reward originated from standard RL might be useful. This consideration also be retained in EEL. In addition, EEL can further improve the flexibility of pattern representation and decision policy learning by combining utility-selection theory and the ESL strategy. In this sense, EEL can be seen as an extended learning framework compared to traditional RL.

In addition, when the maximum length of a true color stripe pattern is limited to 9, the total number of all possible pattern $N_{C\_\text{all}} = \sum_{i=2}^{9} 7^i = 47079200$, which is a huge value when completely storing them like in classical RL frameworks. Our results indicate that EEL has achieved a significant improvement on cognitive performance compared to the random cognitive strategies by only using $N_P = 200$ pattern memory items.

2) Results on Coverage Selection Game: Like in the first task, we introduce a random strategy algorithm (RSA) for comparison in evaluating the cognitive performance of EEL. In RSA, the probability of starting coverage selection is set to 0.1 according to our pilot studies. The above random strategy

TABLE III
PERFORMANCE RESULTS OBTAINED BY EEL AND RSA ON PROBLEM II WITH $T_L = 200000$

	Coverage efficiency (%)	Coverage accuracy (%)
RSA	81.99±0.30	15.96±0.07
EEL	**91.39±0.67**	**56.83±1.23**

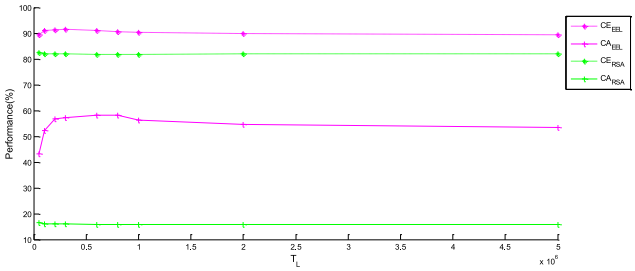

Fig. 6. Performance curves obtained by EEL and RSA on coverage selection with different T_L.

will also be used in EEL if no decision policy matches the current observations. Moreover, we set the maximum number of storing pattern items $N_P = 100$, the maximum number of storing coverage selection decision policies $N_{R\_cs} = 30$, which have shown to be adequate according to our preliminary studies.

To evaluate the cognitive learning performance, the following two quantitative indices are considered in our simulations. Coverage accuracy (CA) denotes the ratio of effective coverage selection in all selections. Coverage efficiency (CE) denotes the reward efficiency of coverage selection decision strategy. CA and CE can be calculated as follows:

$$\text{CA} = \text{Nce} \div Nc \times 100\%$$
$$\text{CE} = R_{\text{gtc}} \div R_{i\text{gtc}} \times 100\%$$

where Nce denotes the effective coverage selection times, Nc the total coverage selection times, R_{gtc} the gained total coverage rewards, and $R_{i\text{gtc}}$ the ideal total coverage rewards.

Different from the color stripe sequence cognition game, the above two performance indices are independent. Here, "effective coverage times" refers to the total number of effective coverage selection in one simulation, in which the total number times of coverage selection are performed. "ideal total coverage rewards" refers to the optimal total reward value for one simulation. "gained total coverage rewards" is the sum of the rewards of all effective coverage selections. The ideal values of these performance indices are all 100%. Table III lists the typical experimental results obtained by EEL and RSA on the coverage selection task with $T_L = 200000$, and corresponding number of periods $N_{\text{periods}} = 1271$. In Table III, the average values and standard deviation values of performance indices are calculated using the results from 20 independent simulations.

Similar to the color stripe sequence cognition game, EEL also achieves considerable performance improvement compared to optimal random strategies. EEL has not only improved the coverage selection accuracy more than two times, but also achieved higher overall reward. These results indicate that EEL is effective not only in reducing the failing coverage selections but also in improving effective coverage selections with the help of evolutionary cognitive development.

Performance curves for EEL and RSA for the coverage selection task with different T_L are plotted in Fig. 6. Overall, EEL has obtained the best performance, and can converge to stable results as simulation time increases. The performance curves obtained by EEL gradually rises at the beginning and then remains steady. For RSA, its performance curves keep almost constant for different T_L. These results indicate that the designed cognitive task is statistically stable. The above experimental results also demonstrate the effectiveness of EEL in developing individual cognitive decision behaviors.

It should be noted that decision actions in the coverage selection task have a huge space and needs to be dynamically optimized, which is very different from classical RL. In classical RL, all possible actions are considered to be finitely countable, and the total number is not larger than 10 in most studies [11], [13]. However, action variables of decision policies for coverage selection game designed in this paper are continuous. This task is very meaningful extension to traditional RL as considered in [12] and [14]. Our experimental results on the coverage selection game task show that EEL is a feasible algorithm for solving such types of RL tasks.

D. Discussions

To demonstrate the effectiveness of the proposed EEL, two novel cognitive decision tasks are designed. The color stripe sequence cognition game is motivated by the cognitive combination intelligence that humans usually use in, e.g., playing "building blocks" game. The optimal coverage selection game is abstracted from the prediction selection abilities found in intelligent cognitive behaviors. No doubt, the above two types of cognitive tasks are very valuable for studying human-like cognitive learning. Meanwhile, they are inevitable for measuring whether an intelligent machine possesses human-like cognitive developmental learning capabilities.

To the best of our knowledge, existing RL methods are not able to directly model the above two simulated cognition game problems. Therefore, we can only introduce naïve random cognitive algorithms for empirical comparisons. The experimental results reported in this paper indicate that EEL is remarkably effective.

It should be mentioned that the traditional RL would be sufficient if all useful decision policies are finite and can be fully recorded. However, in the real world, there are many cases, where the conditional states and/or actions in decision policies are infinite or huge. In this case, constrained learning strategies have to be adopted due to the limited computational resources [11], [12], [14]. In this paper, we investigate the learning strategy of selectively memorizing a bounded number of pattern representation and decision policies, in which utility probability values are introduced to guide selective memory. Our theoretical and experimental results demonstrate that the utility-selection learning strategy proposed in this

paper is very effective and can be complementary to existing RL strategies [11], [12], [14].

V. Conclusion

A novel machine learning method, EEL was proposed to model collaborative cognitive development process. In the proposed framework, two cognitive tasks, enactive pattern representation, and decision-making, are jointly learned driven by environmental action rewards. Specifically, three important tools, namely, "SRM," "utility-selection learning strategy," and "statistical decision output strategy" were introduced to realize above targets.

Two cognitive decision tasks were designed to examine the performance of proposed method. Those tasks cannot be nicely solved by existing RL methods. Our experimental results demonstrated that EEL was effective and could significantly enhance the cognitive performance with evolutionary developmental learning.

From our results, two important observations can be made. First, EEL presented in this paper expands the applicability of traditional RL methods and provides a novel machine learning framework as an implementation of enactive artificial intelligence. Second, pattern representation learning guided by utility-selection criterion related to decision reward from the individual cognitive space may be a feasible way to design autonomous intelligent systems.

Nevertheless, according to enactive artificial intelligence, an autonomous intelligent system should have the ability of adaptively regulating its own sensorimotor interaction, which is still missing from the proposed EEL framework. It will be our future work to extend EEL by adding and evolving decision policies for sensorimotor control. In addition, some extended implementations of EEL may be feasible for practical cognitive development problems. For example, multiple groups of associated decision policies can be defined and jointly evolved. Deep network structure can also be employed to model decision policy and complicated interpretive relationships over ground patterns. These aspects will also be explored in our future research.

References

[1] S. J. Gershman, E. J. Horvitz, and J. B. Tenenbaum, "Computational rationality: A converging paradigm for intelligence in brains, minds, and machines," *Science*, vol. 349, no. 6245, pp. 273–278, 2015.

[2] P. Johnson-Laird, S. S. Khemlani, and G. P. Goodwin, "Logic, probability, and human reasoning," *Trends Cogn. Sci.*, vol. 19, no. 4, pp. 201–214, 2015.

[3] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.

[4] A. Clark, "Whatever next? Predictive brains, situated agents, and the future of cognitive science," *Behav. Brain Sci.*, vol. 36, no. 3, pp. 181–204, 2013.

[5] M. T. Jones, *Artificial Intelligence: A Systems Approach*. Sudbury, MA, USA: Jones & Bartlett Learn., 2015.

[6] A. Cully, J. Clune, D. Tarapore, and J.-B. Mouret, "Robots that can adapt like animals," *Nature*, vol. 521, no. 7553, pp. 503–507, 2015.

[7] J. Bongard and H. Lipson, "Evolved machines shed light on robustness and resilience," *Proc. IEEE*, vol. 102, no. 5, pp. 899–914, May 2014.

[8] R. Pfeifer, M. Lungarella, and F. Iida, "Self-organization, embodiment, and biologically inspired robotics," *Science*, vol. 318, no. 5853, pp. 1088–1093, 2007.

[9] D. Floreano, A. J. Ijspeert, and S. Schaal, "Robotics and neuroscience," *Current Biol.*, vol. 24, no. 18, pp. R910–R920, 2014.

[10] G. Infantes, M. Ghallab, and F. Ingrand, "Learning the behavior model of a robot," *Auton. Robots*, vol. 30, no. 2, pp. 157–177, 2011.

[11] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[12] D. Silver *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[13] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[14] K. Doya, "Reinforcement learning in continuous time and space," *Neural Comput.*, vol. 12, no. 1, pp. 219–245, 2000.

[15] M. L. Littman, "Reinforcement learning improves behaviour from evaluative feedback," *Nature*, vol. 521, no. 7553, pp. 445–451, 2015.

[16] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[17] T. Froese and T. Ziemke, "Enactive artificial intelligence: Investigating the systemic organization of life and mind," *Artif. Intell.*, vol. 173, nos. 3–4, pp. 466–500, 2009.

[18] D. Martínez, G. Alenyà, and C. Torras, "Relational reinforcement learning with guided demonstrations," *Artif. Intell.*, vol. 247, pp. 295–312, Jun. 2015.

[19] Y. Jin and J. Branke, "Evolutionary optimization in uncertain environments-a survey," *IEEE Trans. Evol. Comput.*, vol. 9, no. 3, pp. 303–317, Jun. 2005.

[20] R. Cheng and Y. Jin, "A social learning particle swarm optimization algorithm for scalable optimization," *Inf. Sci.*, vol. 291, pp. 43–60, Jan. 2015.

[21] D. Rus and M. T. Tolley, "Design, fabrication and control of soft robots," *Nature*, vol. 521, no. 7553, pp. 467–475, 2015.

[22] Y. Jin, H. Guo, and Y. Meng, "A hierarchical gene regulatory network for adaptive multirobot pattern formation," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 3, pp. 805–816, Jun. 2012.

[23] M. Reif, F. Shafait, and A. Dengel, "Meta-learning for evolutionary parameter optimization of classifiers," *Mach. Learn.*, vol. 87, no. 3, pp. 357–380, 2012.

[24] L. Zhang and P. Suganthan, "A survey of randomized algorithms for training neural networks," *Inf. Sci.*, vols. 364–365, pp. 146–155, Oct. 2016.

[25] J. C. Bongard, "Evolutionary robotics," *Commun. ACM*, vol. 56, no. 8, pp. 74–83, 2013.

[26] Z.-H. Zhou, N. V. Chawla, Y. Jin, and G. J. Williams, "Big data opportunities and challenges: Discussions from data analytics perspectives [discussion forum]," *IEEE Comput. Intell. Mag.*, vol. 9, no. 4, pp. 62–74, Nov. 2014.

[27] A. E. Eiben and J. Smith, "From evolutionary computation to the evolution of things," *Nature*, vol. 521, no. 7553, pp. 476–482, 2015.

[28] Y. Jin, "Surrogate-assisted evolutionary computation: Recent advances and future challenges," *Swarm Evol. Comput.*, vol. 1, no. 2, pp. 61–70, 2011.

[29] H. Niemann, "Pattern analysis and understanding," in *Springer Series in Information Sciences*, vol. 4. Heidelberg, Germany: Springer, 1990.

[30] R. Cappelli and D. Maltoni, "Multispace KL for pattern representation and classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 9, pp. 977–996, Sep. 2001.

[31] D. Mumford, "Pattern theory: The mathematics of perception," in *Proc. Int. Conf. Math.*, vol. 1. Beijing, China, 2002, pp. 401–422.

[32] T. Watanabe, K. Sugawara, and H. Sugihara, "A new pattern representation scheme using data compression," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 579–590, May 2002.

[33] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[34] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[35] G. Alain and Y. Bengio, "What regularized auto-encoders learn from the data-generating distribution," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 3563–3593, 2014.

[36] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967.

[37] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. New York, NY, USA: Wiley, 2001.

[38] J.-Y. Zhu, J. Wu, Y. Xu, E. Chang, and Z. Tu, "Unsupervised object class discovery via saliency-guided multiple class learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 4, pp. 862–875, Apr. 2015.

[39] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. 7th IEEE Int. Conf. Comput. Vis.*, vol. 2, 1999, pp. 1150–1157.

[40] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.
[41] R. Goroshin, J. Bruna, J. Tompson, D. Eigen, and Y. LeCun, "Unsupervised feature learning from temporal data," *arXiv preprint arXiv:1504.02518*, 2015.
[42] J. Yang, A. F. Frangi, J.-Y. Yang, D. Zhang, and Z. Jin, "KPCA plus LDA: A complete kernel Fisher discriminant framework for feature extraction and recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 2, pp. 230–244, Feb. 2005.
[43] R. C. Barros, M. P. Basgalupp, A. C. P. L. F. De Carvalho, and A. A. Freitas, "A survey of evolutionary algorithms for decision-tree induction," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 3, pp. 291–312, May 2012.
[44] L. Rokach and O. Maimon, *Data Mining With Decision Trees: Theory and Applications*. Hackensack, NJ, USA: World Sci., 2014.
[45] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.
[46] G. Huang, S. Song, J. N. D. Gupta, and C. Wu, "Semi-supervised and unsupervised extreme learning machines," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2405–2417, Dec. 2014.
[47] S. Maji, A. C. Berg, and J. Malik, "Efficient classification for additive kernel SVMs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 66–77, Jan. 2013.
[48] Z. Xie, J. Sun, V. Palade, S. Wang, and Y. Liu, "Evolutionary sampling: A novel way of machine learning within a probabilistic framework," *Inf. Sci.*, vol. 299, pp. 262–282, Apr. 2015.
[49] O. Sigaud and A. Droniou, "Towards deep developmental learning," *IEEE Trans. Cogn. Develop. Syst.*, vol. 8, no. 2, pp. 99–114, Jun. 2016.
[50] J. B. Tenenbaum, C. Kemp, T. L. Griffiths, and N. D. Goodman, "How to grow a mind: Statistics, structure, and abstraction," *Science*, vol. 331, no. 6022, pp. 1279–1285, 2011.
[51] L. Bottou, "From machine learning to machine reasoning," *Mach. Learn.*, vol. 94, no. 2, pp. 133–149, 2014.
[52] C.-F. Juang, T.-L. Jeng, and Y.-C. Chang, "An Interpretable fuzzy system learned through online rule generation and multiobjective ACO with a mobile robot control application," *IEEE Trans. Cybern.*, vol. 46, no. 12, pp. 2706–2718, Dec. 2016.
[53] K. Etessami and M. Yannakakis, "Recursive Markov decision processes and recursive stochastic games," *J. ACM*, vol. 62, no. 2, p. 11, 2015.
[54] S. Ross, J. Pineau, B. Chaib-Draa, and P. Kreitmann, "A Bayesian approach for learning and planning in partially observable Markov decision processes," *J. Mach. Learn. Res.*, vol. 12, no. 5, pp. 1729–1770, 2011.
[55] J. Rieskamp and P. E. Otto, "SSL: A theory of how people learn to select strategies," *J. Exp. Psychol. Gen.*, vol. 135, no. 2, pp. 207–236, 2006.

Zhenping Xie received the B.Eng. and Ph.D. degrees from Jiangnan University, Wuxi, China, in 2002 and 2008, respectively.

He is currently an Associate Professor with the School of Digital Media, Jiangnan University. He has authored or co-authored of about 30 research papers in academic journals. His current research interest includes evolutionary behavior learning, knowledge modeling and decision, and cognitive physics.

Yaochu Jin (M'98–SM'02–F'16) received the B.Sc., M.Sc., and Ph.D. degrees from Zhejiang University, Hangzhou, China, in 1988, 1991, and 1996, respectively, and the Dr.-Ing. degree from Ruhr University Bochum, Bochum, Germany, in 2001.

He is a Professor in computational intelligence with the Department of Computer Science, University of Surrey, Guildford, U.K., where he heads the Nature Inspired Computing and Engineering Group. He is also a Finland Distinguished Professor funded by the Finnish Agency for Innovation (Tekes) and a Changjiang Distinguished Visiting Professor appointed by the Ministry of Education, China. He has co-authored over 250 peer-reviewed journal and conference papers and been granted eight patents on evolutionary optimization. He has delivered 30 invited keynote speeches at international conferences. His current research interests include data-driven surrogate-assisted evolutionary optimization, evolutionary multiobjective optimization, evolutionary learning, interpretable and secure machine learning, and evolutionary developmental systems.

Dr. Jin was a recipient of the 2018 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award, the 2015 and 2017 *IEEE Computational Intelligence Magazine* Outstanding Paper Award, and the Best Paper Award of the 2010 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology. He is the Editor-in-Chief of the IEEE TRANSACTIONS ON COGNITIVE AND DEVELOPMENTAL SYSTEMS and the Co-Editor-in-Chief of *Complex and Intelligent Systems*. He is an IEEE Distinguished Lecturer from 2013 to 2015 and from 2017 to 2019 and the past Vice President for Technical Activities of the IEEE Computational Intelligence Society from 2014 to 2015.

APPENDIX A: THEORETICAL ANALYSIS ON EEL

This appendix provides some theoretical analysis on the availability of proposed EEL. To this end, we will first introduce a number of novel concepts to evaluate the learning performance for a cognitive development system.

A. Representation completeness

Definition 1: The representation model of decision policies in a cognitive system is *fully complete*, if all true causal relations in that system's cognitive space can be completely covered by the representation model. A causal relation commonly comprises state inputs, actions and corresponding system responses.

In standard reinforcement learning, we normally assume that representation models are fully complete, in which one or multiple fixed input patterns are pre-defined and all useful policies can be stored in the decision system. Obviously, it could be feasible only when input states and decision actions are all discrete. Even so, because many real applications may contain huge possible decision policies, completely storing them will become unrealistic given a finite memory capacity. In addition, it is very difficult to define cognitive rules with strict causality for many real-world problems. Therefore, we introduce a weaken form of full completeness, i.e., statistical completeness.

Definition 2: The representation model of decision policies in a cognitive system is *statistically complete*, if all true causal relations in that system's cognitive space can be statistically interpreted by the representation model with a limited memory capacity.

Compared to full completeness, statistical completeness tries to represent all required causal rules using a small number of rules but not all possible rules. This implies that some original rules must be statistically reduced. The internal representation model of decision policies in EEL is supposed to be statistically complete.

Next, we will discuss the evaluation of the learning performance of a whole cognitive system.

B. Evolvable completeness

We further introduce the concept "evolvable completeness" to measure the evolvable intelligence of a cognitive system. A few related definitions are first given below.

Definition 3: A cognitive development system is said to be of *evolvable completeness*, if the system can be asymptotically developed by itself towards the optimal consistency to all causal relations of that system's cognitive space. Here, we assume that a cognitive agent surviving in an environment can perceive all meaningful states, perform some behavioral actions, and receive action rewards from individual cognitive space.

In definition 3, the term "evolvable" means that the cognitive performance of an agent can be improved by enactive interaction to the environment. The "optimal consistency" indicates that all stored policies with limited capacity can optimally interpret all causal relations of individual cognitive space.

According to definition 3, we may believe that human cognitive development system satisfies evolvable completeness. Also, evolvable completeness may be a necessary condition for realizing human-like cognitive development system for intelligent machines. By combining the definitions 1 and 2, we can find that if a cognitive system satisfies evolvable completeness, then its representation model of decision policies should also be fully or statistically complete. Thus, for a human-like cognitive agent, another learning ability should be equipped, and the following definition is introduced.

Definition 4: A cognitive development system is said to be of *ergodic completeness*, if the system has the ability of exploring all possible causal relations existed in that system's cognitive space.

Definitions 3 and 4 indicate that if a cognitive system is of evolvable completeness, then it should also satisfy ergodic completeness. In addition, ergodic completeness is a basic requirement for almost all search-based optimization algorithms including evolutionary computation and other meta-heuristics algorithms. However, this concept has not yet been attached enough importance in machine learning. Based on above discussions, we believe that ergodic completeness should be a base point to constitute a human-like cognitive development system. Here, suitable behavior decision strategy may also be very important to find all possible causal exemplars, and it should have the ability of exploring and exploiting all meaningful causal pairs as quickly as possible. According to the probabilistic principles, if a behavior decision strategy has non-zero probability values on all meaningful actions, it could visit all possible causal pairs. In traditional reinforcement learning, the strategies with balanced exploration and exploitation on decision output are used to achieve ergodic completeness. A similar strategy is also used in EEL.

From definitions 3 and 4, we find that ergodic completeness should be necessary but not sufficient to get evolvable completeness when memory capacity is limited for a cognitive development system. Undoubtedly, selective memory has to be considered.

Definition 5: A cognitive developmental system is said to be of *reduction completeness*, if the system can dynamically reduce all perceived causal examples into limited number of causal exemplars that are statistically consistent with the original examples on decision inference in a cognitive space.

In definition 5, the concept "decision inference" refers to effectively predicting concomitant results for given inputs and actions. From definitions 3 to 5, we can conclude that evolving completeness may be the combination of ergodic completeness and reduction completeness. Furthermore, reduction completeness should be a core condition for gradually developing cognitive intelligence guided by environmental action rewards.

C. Core characteristics of EEL

In EEL, we consider that the representation forms of decision policies are pre-constructed by hand, consequently

the statistical completeness is supposed to be directly satisfied.

Moreover, because the action output is selected based on one extra weighted factor attached to every feasible decision policy in terms of their own utility values, EEL has the ability of simultaneously exploring and exploiting new possible causal examples. That is, EEL satisfies ergodic completeness according to definition 4. Here, proposed adaptive weighted factor evaluation strategy can enable EEL to own more flexible adaptivity. In comparison, traditional reinforcement learning methods can only adopt constant probability value to balance the requirements between exploration and exploitation.

On the basis of the above analysis, we further examine reduction completeness of EEL. At first, reduction completeness could be viewed as a core characteristic originated from online statistical learning, a kind of extension of classical statistical learning. Wherein, trained samples can only be dynamically or incrementally gathered, while the memory capacity may be limited and insufficient to store all perceived samples. According to our previous study, as a very unique dynamic sampling process, evolutionary sampling learning is particularly well suited for online statistical learning problems. According to slightly strict analysis, the following property for evolutionary enactive learning can be obtained.

Proposition 1: In evolutionary enactive learning, given a proper distance metric on decision policies, the learning algorithm satisfies reduction completeness if the perceived examples of decision policies are uniformly generated according to the probability values linearly related to their true utility values.

For *Proposition 1*, if a distance metric on the decision policies can be properly pre-defined, we can then construct a Hilbert space including all feasible decision policies. Also, correlation factor metric between any two decision policies are valid. In addition, if we assume the correlation degree between two decision policies obeys with Gaussian distribution function on the distance, we may define a probability space on decision policies using their utility values. Moreover, we use $\pi^*(R_l)$ to denote the true probability of different decision policies in a cognitive space, and $\pi^t(R_l)$ to denote the estimation at time t based on past experience. Accordingly, $p^t(R_l)$ can be seen as the current approximation representation of $\pi^t(R_l)$. By integrating the above considerations, we know that the decision policy learning algorithm of EEL tries to perform the evolution with the objective $p^t(R_l) \to \pi^t(R_l)$, which is consistent with the standard evolutionary sampling approach. Moreover, the definition of $\pi^t(R_l)$ is tightly related to the environmental rewards produced by the laws in a cognitive space, and it will tend to $\pi^*(R_l)$ with the cognitive development. Thus, in terms of the learning characteristic owned by the evolutionary sampling approach, the decision policy learning algorithm used in EEL can achieve the goal $p^t(R_l) \to \pi^*(R_l)$. By summarizing above analysis, *Property 1* can be concluded.

Next, we can further deduce the following property for evolutionary enactive learning.

Proposition 2: If the representation model of decision policies pre-defined by hand is statistically complete and it contains a proper distance metric on decision policy space, then evolutionary enactive learning is evolvable completeness.

For the decision policy learning algorithm designed in EEL, we have

$$u^{t+1}(R_l) = \lambda_2 \times u^t(R_l) + \pi^t(R_l \mid \Phi_{ob}^{t+1}) \psi(\Phi_{ob}^{t+1}) \mu_2^{t+1}(R_l)$$

and

$$\pi^t(R_l \mid \Phi_{ob}^{t+1}) \sim \exp(-dist_{\Phi}^2(\Phi_{ob}^{t+1}, \Phi_l^t) / \sigma_w) \left(u^t(R_l)\right)^{\gamma}.$$

where, $\psi(\Phi_{ob}^{t+1})$ represents the prior probability that the input Φ_{ob}^{t+1} may occur in the cognitive space, and $\mu_2^{t+1}(R_l) = \mu_2^{t+1}(\hat{r} \; in \; R_l)$. So, if $\mu_2^{t+1}(R_l) \to \mu_2^*(R_l)$, then there have $\pi^t(R_l) \to \pi^*(R_l)$, $u^t(R_l) \to u^*(R_l)$, and $\int \pi^t(R_l \mid \Phi_{ob}^{t+1}) \psi(\Phi_{ob}^{t+1}) d\Phi_{ob}^{t+1} \to \int \pi^*(R_l \mid \Phi_{ob}) \psi(\Phi_{ob}) d\Phi_{ob}$ $= m_{\pi}(R_l) \left(u^*(R_l)\right)^{\gamma}$.

Moreover, we have

$$u^*(R_l) = \lambda_2 \times u^*(R_l) + \left(u^*(R_l)\right)^{\gamma} \times m_{\pi}(R_l) \times \mu_2^*(R_l),$$

that is, $u^*(R_l) = \left(\frac{\mu_2^*(R_l) \times m_{\pi}(R_l)}{1 - \lambda_2} \right)^{\frac{1}{1-\gamma}}$.

So, there is

$$\pi^t(R_l \mid \Phi_{ob}^{t+1}) \sim \exp(-dist_{\Phi}^2(\Phi_{ob}^{t+1}, \Phi_l^t) / \sigma_w) \left(\frac{\mu_2^*(R_l) \times m_{\pi}(R_l)}{1 - \lambda_2} \right)^{\frac{\gamma}{1-\gamma}}$$

Similar to reinforcement learning, $\mu_2^{t+1}(R_l) \to \mu_2^*(R_l)$ could be satisfied with the reinforcement development of a cognitive system. So, if $\gamma = 0.5$ is considered as default, the *Proposition 2* will be satisfied.