Cognitive Architecture for Video Games

Hongming Li*, Ying Ma†, Jose Principe‡, *Life Fellow, IEEE*
Department of Electronic and Computer Engineering
University of Florida
Gainesville, FL, 32611, USA
hongmingli@ufl.edu*, mayingbit2011@gmail.com†, principe@cnel.ufl.edu‡

***Abstract*—There has been an increasing interest in Frame-oriented reinforcement learning (FORL) in recent year. However, most of the works in the literature show little inspiration from human's perception action reward cycle (PARC) and causation.**

Inspired by human's vision system and learning strategy, we propose a novel architecture for FORL that understands the content of raw frames. The architecture achieves four objectives:

1.Extracting information from the environment by exploiting only unsupervised learning and reinforcement learning.

2.Understanding the content of a raw frame.

3 Exploiting a Folvea vision strategy which is analogous to human's vision system.

4.Establishing self-awareness and collecting new training data subset automatically to learn new objects without forgetting previous ones..

The architecture is developed in the Super Mario Brothers video game.. At first, Mario is the only object recognized by the architecture. After automatic data subset collection and memory update, the architecture can recognize both Goomba and Mario and classify them using incremental training.We exemplify performance of each piece of the architecture with snippets obtained from the video game.

***Index Terms*—Brain-inspired cognitive architectures, visual system, Frame-oriented reinforcement learning**

I. Introduction

Frame-oriented reinforcement learning (FORL), which means the model inputs are raw video frames, is gradually becoming the norm in reinforcement learning (RL)thanks to the introduction of deep Q learning [1]. Recent developments have heightened the need for understanding the contents of frames in an unsupervised way. To this end, recent trends in FORL have led to a proliferation of studies that utilize prior knowledge, e.g. motion of objects or for reducing the complexity of the input imagery using feature representations [2] [3] [4] [5]. Several other attempts have been made to augment rewards with auxiliary objectives [2] [6].

Theoretically, most of the works in the FORL literature are rooted in optimization and show little inspiration of how the brain exploits the perception action reward cycle (PARC). In fact, RL is based on point estimations of a function in a high dimension space to improve future rewards. It is amazing that this simple procedure was proven to converge to the optimal policy [22], but experience has shown that it is very far from efficient and requires huge number of interactions with the environment. In contrast, humans learn from the environment in one shot!

Human' brain architecture is built from the information in the DNA. But the parameters in babies' brains at birth are not set at the proper values. To simplify we can assume that at birth babies brains are in a state that we can call "Tabula rasa". Tabula rasa is a Latin phase meaning "clean state", and it is via the PARC that brain circuits are trained using correlation, as first demonstrated by Donald Hebb [7]. So, babies have to discover the world themselves, gradually and then continue to learn through interaction with the environment using the PARC. Life-long learning is necessary for all of us, and we keep learning throughout our life span, so parameters are continuously being learned online, very quickly, and most of the time without a teacher.

Human learning also exploits causation, while machine learning is based on correlation. The ingredients to reach causation are being discovered. Wiener showed that causation can be mathematically defined between two models of time series [23], which was after exploited by Granger [24]. In statistics it is also known that we need at least 3 random variables to compute conditional mutual information and define causality [25], which is compatible with Wiener's idea because time is a causal third variable. Indeed, Wiener's approach is the one that is most similar to how humans interact with the environment. In our opinion, it is not sensible to ignore the learning strategy that evolved from animals for thousands of years. While exploiting the PARC, humans initiate the interaction with the dynamic environment (time series) using internal models of past experience, which are necessarily causal because the brain is a physical system. So, if a change is observed by the human in the external world after the human action, it may be causally related to the action, provided that the human focuses on the proper subset of variables. This is not an easy task (see Pearl [25]), but using attention, humans properly select the subset of variables for which the causality can be inferred. For instance, every parent has seen babies throwing objects to the floor multiple times. They are effectively learning the rules of gravity, and they can predict the fall after a very few (sometimes one experiment) which explains the fun they have doing this. So, we submit that one tool for one shot learning is to base learning on causality.

Human's visual system is also inspiring. To simplify the

This work was supported in part by the Lifelong Learning Machines Program of the Defense Advanced Research Projects Agency, Microsystems Technology Office, under Grant FA9453-18-1-0039.

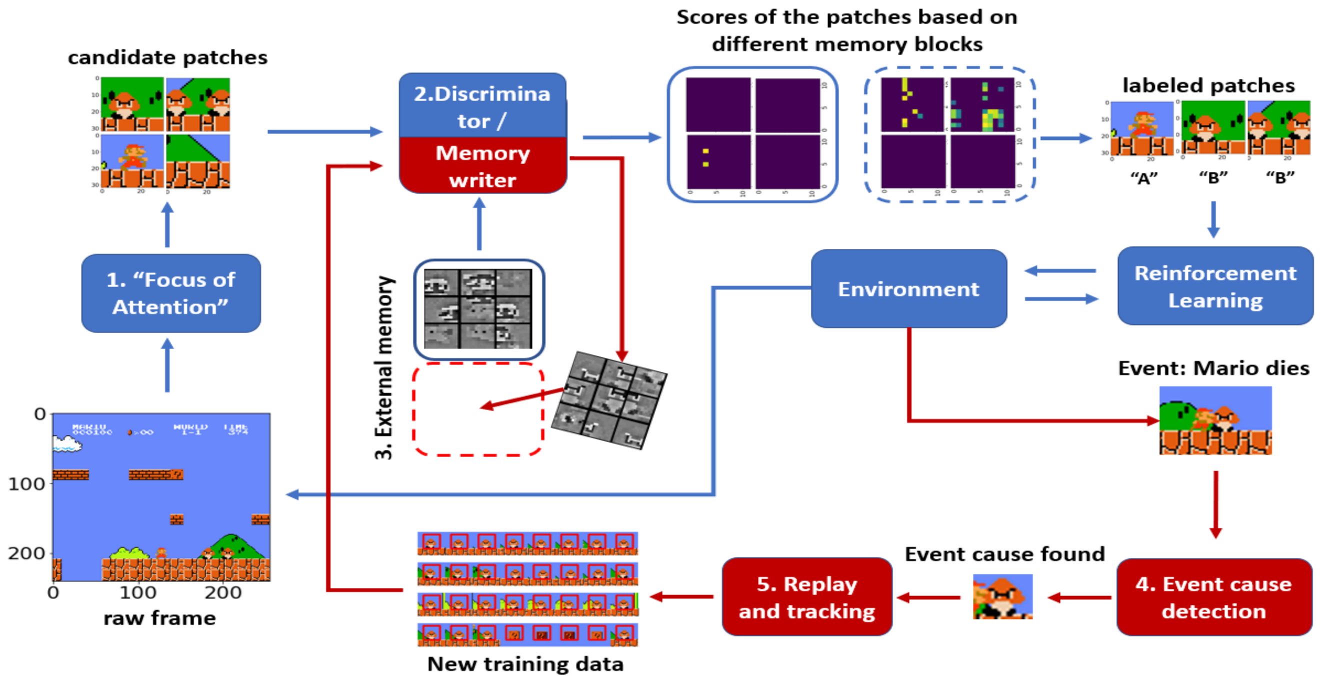

Fig. 1. The Cognitive Architecture

analysis of the real-world complexity, nature cut corners when it 'designed' the human visual system. Only the central portion of the field of view when we gaze (5 degree) is a high–resolution "camera", while the peripheral vision is blurry and used for context and to find where the next eye saccade is going to be targeted. That is the reason we develop a 'focus of attention' to implement the "focus of attention" part of the saccade direction control. There is also a discrimination influence that includes past experience stored in memory, the scene type and the cognitive goal.

In this exploratory paper, we will present our FORL Tabula Rasa Architecture inspired by the human visual system and the PARC. We assume that the vision module is working, as well as the memory, but the system does not know anything about the world (Mario Video game). The architecture achieves the following objectives:

Objective 1: A FORL architecture that extracts information from the environment without labels, i.e. only unsupervised and reinforcement learning. The input of our architecture consists of raw frames while the output is the action of the agent, as is being proposed in Deep Q Networks and its variants.

Objective 2: Understanding the content of a raw frame. Our architecture can detect and even discriminate objects and agents in a frame, in an unsupervised way, and learn a good policy according to their past behavior to maximize reward.

Objective 3: Folvea vision strategy and External memory. Inspired by human's vision strategy, our architecture possesses a readable and writable external memory module coding the learned objects, and agents (e.g. enemies, Mario), in the video game. The memory can be updated based on the interaction between agent and the environment as analogous to life-long learning.

The "focus of attention" in our architecture is implemented by the Gamma saliency [8] algorithm, which predicts the number of candidate patches in a frame and estimates the size of objects. The vision processing cycle is initiated by a saccade. The most salient candidate patch is examined and sent to the Deep predictive coding network (DPCN) [9] for discrimination. The process is repeated until there are no more salient patches above threshold. Therefore, the x,y coordinates of each patch are known and store along with the time they were looked at. The first layer parameters of the DPCN (the columns of a matrix) become the dictionary atoms that represent the learned object. Each column is stored in an external memory of the content addressable type.

Objective 4: Self-awareness and collecting training data automatically from raw videos for memory updating. Humans do not have an image of themselves when born [10] (Babies cannot recognize their image in a mirror). We postulate in our agent-initiated game that for interaction in the world, the first person to be recognized should be the agent. Because the agent can move and recognize movement in the x,y location coincident with its location, it can infer that the recognized object from the DPCN corresponds to itself, using causality (the agent initiated the movement). Therefore, the first step to understand the world in our architecture is identification of the agent, i.e. Mario. We can identify Mario naturally without any additional training because the RL algorithm controls Mario. In practice, we collect a training set containing only Mario and use it to train the DPCN, and the codes are stored in an external CAM memory. After learning the agent, the program captures new objects when Mario is rewarded, cannot move (obstacles) or dies. These events mean that there was an interaction in the x,y coordinates of the agent with another (unknown) agent or obstacle in the environment. The system must then go back in the frames from the point of interaction and collect frames that lead to the interaction. Since its coordinates are known, a patch is drawn around this point and sent to the DPCN for further learning. Notice also that the system knows (causality) if it was good or bad, which can immediately be translated in affordances (avoid or repeat the action). This implements "one-shot learning". Of courses, the procedure is still unrealistic because the movie has to be rewound, but the existence of a working memory (as in the brain) would not require this step. Finally, we train the DPCN using the new data consisting of new objects to populate the dictionary or for memory update. In summary, the architecture is tabula rasa and totally automatic in its interaction with the world for learning, except the "learning of the self" procedure.

II. Related works

Much of the current literature pours attention into extracting better feature representation [5] [11] [12] [13]. The paper [14] extracts key spatial feature points with a pre-trained model back to 2016.

More recent approaches have emphasized the use of specific patches in a raw frame, objects especially. [5] proposed object-oriented framework and reason the interactions for reinforcement learning. [2] proposed a motion-oriented model for reducing the sample complexity through unsupervised learning of optical flow. With the same motivation, [3] highlights object keypoints in an image through the model of [4].

There are also works sharing similar motivation with ours. [15] understood the raw image by disentangling the representation of the frame. These features are used for domain adaptation, and "improve zero-shot transfer in reinforcement learning". [6] built the architecture being analogues to human's learning strategy. Believing that "the image of the world around us in our head is a model", the architecture consists of vision, memory, and controller model. However, the key motivation is still reducing the sample complexity.

III. Methodology

We first give an overview of the general architecture in section III-A. The objective of each module is introduced, then we discuss algorithms of each module in detail in the following subsections. Results of each module are illustrated which is helpful for understanding the architecture.

A. Overview of the architecture

Fig.1 depicts the general architecture pipelines. We divided it into two parts: 1) working pipeline, 2) updating pipeline. Table.I summarizes the method for each module illustrated in Fig.1. Algorithm.1 and Algorithm.2 are pseudocodes of working and updating pipeline, respectively.

TABLE I
Algorithms of the modules

Modules	Algorithms
1."focus of attention" algorithm	Gamma saliency
2.discrimination algorithm / Memory Writer	DPCN
3.External Memory	Dictionary Matrices of DPCN
4.Event cause detection	Objectness Estimation
5.Replay and tracking	CSRT

Algorithm 1: Working pipeline

```
Requirement: Self- awareness.Initializing memory
  containing only Mario;
while not achieving goal of the game do
    1. extract candidate patches using "focus of
      attention";
    2. discriminating and labeling the candidate
      patches using discriminator based on memory;
    3. the Mario interacts with the environment
      controlled by object-based reinforcement learning;
    if the same event happens 50 times, e.g. Mario
      dies then
        update memory using Algorithm.2;
    end
end
```

Algorithm 2: Updating pipeline

```
Requirement: The patches whose center is the Mario
  when the event happens, 50*20 frames before the
  event happens;
for 50 do
    detect objects in Mario vicinity using objectness
      estimation;
    if there is a unknown object then
        1. track the object backward using CRST in
          frames before the event happens;
        2. append the tracked patches to new training
          set;
    end
end
Do Updating the memory by training new dictionary
  matrices of DPCN with new dataset;
```

First of all, we train the DPCN with a dataset containing only Mario for formation of self- awareness. The dictionary matrices become the initial memory.The external CAM memory and discrimination algorithm are rooted in the DPCN. The details of it will be discussed in section III-C.

The blue arrows and blocks in Fig.1 represents working pipeline modules in the following and the Algorithm.1:

(1) After a raw frame is fed, the "focus of attention" algorithm extracts candidate patches from the frame which is analogous to human's visual detection system.

(2) The candidate patches and the external memory are the

input of discrimination algorithm. The algorithm scores the patches and labels them based on the content of the memory. This process is quite similar to discrimination flow of human's vision system [16].

Before we introduce the updating pipeline, let's review a crucial problem: How do we define an object in an environment, i.e. the Mario game in our experiment? If we follow the idea of vision-based definition, there will always be irrelevant 'background' which looks like objects, e.g. small bushes. Inspired by PARC, we define the object as the patches which interact with the agent (Mario in this paper) and enjoy high objectness descriptors [17].

The red arrows and blocks in Fig.1 represent modules of updating pipeline in the following:

(1) When events happen (e.g. Mario dies), the architecture determines if there is a known object (stored in the DPCN matrix) in Mario's vicinity. If not, the system does the following steps; (Else, Reinforcement learning process is continued).

(2) Event causes detection: objectness estimation. We must set a high threshold to avoid detection errors though it increases the probability of missing objects.

(3) Replay and tracking: Track the candidate object using Channel and Spatial Reliability-Tracking (CSRT) [18] in the 20 frames from the point of the event (Mario's death).

(4) Repeat (2) and (3) for fifty scenarios of death (does not need to be contiguous).

(5) Using few shot learning to train new DPCN dictionary matrices which represents only the new object. The new matrices become 'new memory'. Activation of different Matrices indicates different objects naturally without an additional classifier.

(6) After update, continue the reinforcement learning process.

B. "Focus of attention" algorithm: Gamma Saliency

As mentioned above, only a portion of visual field of human's eye an form high-quality images while the peripheral portion forms blurry context. In computer vision, the topic saliency can be considered as bionics of this design.

Gamma saliency is a fixed measurement. It gives a high score if the region is different from the neighborhood. To this end, the algorithm convolves images with 2D Gamma kernels for emphasizing center as well as comparing it with regions in vicinity. The basic kernel can be written as:

$$g_{k,\mu}(x,y) = \frac{\mu^{k+1}}{2\pi k!}\sqrt{x^2+y^2}^{k-1} exp(-\mu\sqrt{x^2+y^2}) \quad (1)$$

where x and y are the local support grid while μ and k are parameters controlling the shape of the kernel. Center kernels are subtracted by a surround kernels, and kernels are summed up for multi-scale adaption. More formally:

$$g_{total} = \sum_{m=0}^{M-1}(-1^m)g_m(k_m,\mu_m) \quad (2)$$

A multi-ring or multi-scale gamma kernel allows for the potential to intrinsically approximate the size of an object by knowing the relationship between the order of the kernel and the number of pixels separating each ring. A crucial advantage of Gamma saliency is its speed, because the computation complexity is equal to a single convolution with a mask.

After initialization, raw frames are fed into the architecture during the process of reinforcement learning. Gamma saliency ("focus of attention" algorithm) predicts eight candidate patches of objects illustrated in Fig.2

In this example, we can find that the algorithm is sensitive to any high contrast point in the backgrounds. It proposes non-objects, e.g. bushes, which is different from it neighborhood. Gamma Saliency has modest performance as a feature extractor/map generator. It only takes color and local contrast into consideration, which is not sufficient for human level attention. Therefore, a discrimination algorithm is introduced to offset it.

C. External Memory and discrimination algorithm: Deep Predictive Coding Network (DPCN)

External Memory and discrimination algorithm are synthesized according to the procedure in the paper [9], using a single layer DPCN in our architecture which is a dynamic variant of sparse coding on image sequences. Sparse coding has been extremely popular feature extractor and wildly applicated in visual neuroscience [19]. Combining the predictive coding, DPCN relies on an efficient inference process to get more accurate latent features from image sequences [9].

A general predictive coding architecture is a state-space model in the form of:

$$\begin{aligned} \widetilde{y_t} &= F(x_t) + n_t, \\ x_t &= G(x_t - 1, u_t) + v_t \end{aligned} \quad (3)$$

Where $\widetilde{y_t}$ is the data and F and G are functions whose parameters are able to be learned, e.g.θ. u_t is 'cause' which are encouraged to have a non-linear relationship with the observation $\widetilde{y_t}$ compared to original sparse coding. The hidden state x_t mediates the influence of u_t. Both v_t and n_t are noise or other stochastic uncertainty.

In detail, the objective function of the single layer DPCN is:

$$\begin{aligned} &E(x_t,u_t,\theta) = \textstyle\sum_{n=1}^{N}(\frac{1}{2}\left\|\widetilde{y_t}^{(n)} - Cx_t^{(n)}\right\|_2^2) \\ &+\lambda\left\|x_t^{(n)} - Ax_{t-1}^{(n)}\right\|_1 \\ &+\textstyle\sum_{k=1}^{K}\left|\gamma_{t,k}\cdot x_{t,k}^{(n)}\right|) + \beta\left\|u_t\right\|_1 \\ &where \\ &\gamma_{t,k} = \gamma_0\left[\frac{1+exp(-(Bu_t)_k)}{2}\right] \end{aligned} \quad (4)$$

$\theta = \{A,B,C\}$ is the set of dictionary matrices of the DPCN and used as the external memory of our architecture. $n \in \{1,2...N\}$ represents contiguous patches.$k \in \{1,2...K\}$ represents patches for average pooling. In our experiment, N = 1 and K = 4.

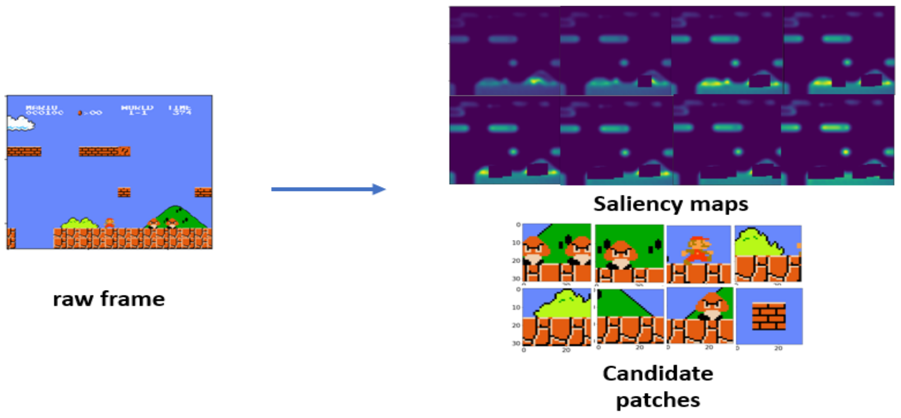

Fig. 2. Saliency maps and candidate patches predicted by Gamma saliency

The first term of Eq.4 is the reconstruction loss identical to sparse coding while the second term is the recurrent term mediating the influence of time delay. The third term is the L1 regulation term of x_t and the essence of DPCN. This term makes the cause u_t control the sparsity of state x_t through it. This design encourages the non-linearity of the DPCN as well as decreasing the dimensionality of the output. Normally, the dimensionality of the sparse coding model is larger than the input. The final term is the L1 regulation of causes u_t which encourages sparsity of causes. Notice that, each u_t corresponds to four x_t through average pooling, and the structure of DPCN is illustrated in Fig.3 [9].

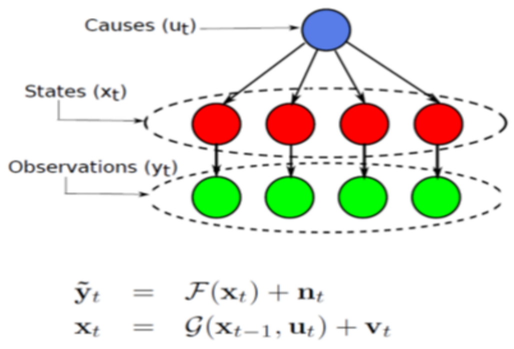
Causes (u_t)
States (x_t)
Observations (y_t)
$\tilde{\mathbf{y}}_t = \mathcal{F}(\mathbf{x}_t) + \mathbf{n}_t$
$\mathbf{x}_t = \mathcal{G}(\mathbf{x}_{t-1}, \mathbf{u}_t) + \mathbf{v}_t$

Fig. 3. The basic Architecture of DPCN

Learning

Inference: Fixed θ, x_t and u_t are jointly inferred through minimizing Eq.4, using an improved FISTA [20]. In other words, we update x_t and u_t alternatively using a single update step to minimize Eq.4, while keeping taking the other variable as constant (x_t,or u_t). After convergence, an iteration of inference is finished.

Dictionary updating: After a number of iterations, all the x_t and u_t are fixed and are used as 'training data' batch for updating $\theta = \{A, B, C\}$ by minimizing Eq.4 using conjugate gradient method. The dictionary matrices are updated independently. B and C are column normalized after update. Notice that, the process of dictionary updating happens in updating pipeline (red part of the Fig1.), but the inference in both.

Inference and dictionary updating are operated alternatively until the dictionary updating coverage.

Test mode

Dictionary matrices are fixed during testing (blue parts in 1), while the inference is almost identical to the learning process except the recurrent term. Notice that, the input of working DPCN consists of candidate patches extracted using Gamma saliency. It is obvious that these patches are not time-related. Therefore, the recurrent term of Eq.4 is removed in the inference process of working mode, and the DPCN is degraded to a sparser coding algorithm.

In the following, we show how the DPCN can 'remember' the objects. We start with the initial memory trained from dataset consisting the Mario itself.

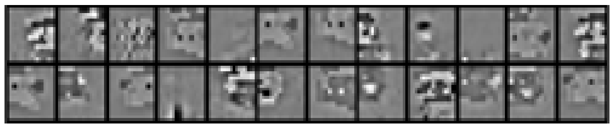

Fig. 4. The receptive fields of the causes of Mario

In Fig.4, the receptive fields of the causes, i.e. dictionary matrices, are visualized using the same method that was detailed for [9]. After initialization, only the features of Mario were stored in the dictionary matrices. The corresponding blocks of the dictionary are activated when the input matches. In other words, only Mario can be detected by DPCN at the beginning.

Observing the score map in Fig.5, some background regions are also activated, because there are also blocks representing

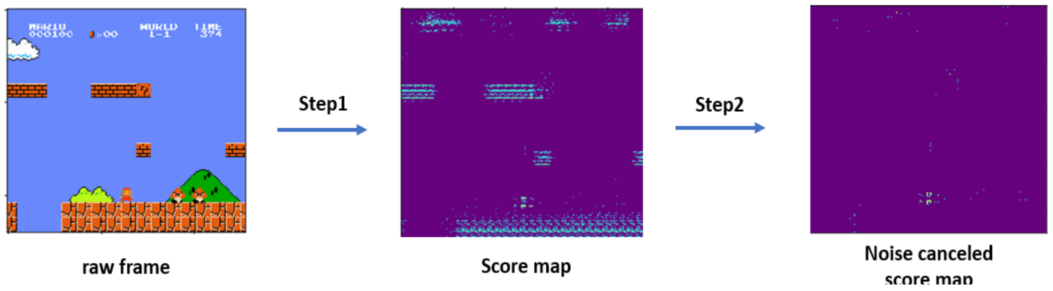

Fig. 5. Procedures of Noise Canceling in two steps: 1. Scan a random image based on Memory blocks consisting of Mario; 2. Get rid of Memory blocks activated too often

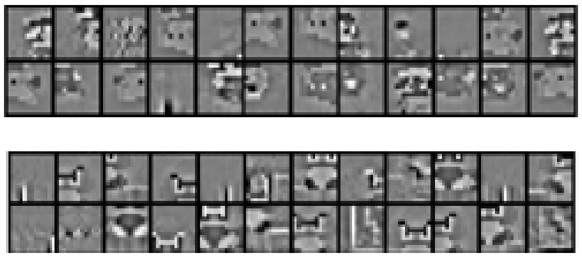

Fig. 6. The receptive fields of the causes of Mario and Goomba

general image features.. As illustrated in Fig.4, some of blocks are edges instead of Marios. The probability that the general feature blocks are activated is extremely high.

Therefore, the noise from background can be canceled by applied the DPCN on a random frame and remove the blocks of dictionary matrices activated too often (illustrated in Fig.5).

The noise canceled score map of Fig.5 illustrates what the architecture is ‘seeing’ in a raw frame. DPCN scores each patch under the condition of dictionary matrices (external memory). The different matrices correspond to specific objects; hence the architecture classifies objects naturally. They can be seen as binary classifiers in parallel.

Learning the “Mario-style” features in Fig.4: The object size is small in our experiments. Batches of 20*20 patches are the DPCN input. Each of them is divided into 12*12 sub-patches overlapping by 4 pixels. The sub-patches are inferred into 150-dimensional state vectors. The four vectors representing four contiguous neighboring sub-patches were pooled for inferring a single 60-dimensional cause vector. Notice that, each element of the vector corresponds to a unique dictionary block. The object sizes are from 16*16 to 24*24, so this makes image primitives [16] cover almost the entire patch in our experiment.

A more popular choice is representing an object as a configuration consisting of small primitives [16], e.g. edges or shapes. Considerable researches believe CNN sharing the same motivation. This is suitable for natural images but too much for the pixel-style games according to the results of our experiments. Expanding single layer to multi-layered model through greedy layer-wise learning, DPCN can also achieve this, but it is not discussed in this paper.

With the help of the visual front-end, reinforcement learning algorithms predict actions based on information processed above. The environment feedback, a new state, leads to the next iteration of reinforcement learning.

D. *Event causes detection: Objectness estimation*

After formation of self-awareness, i.e. locating the Mario in our environment, it is necessary to propose an algorithm to find the event causes when the state of the agent changes, e.g. Mario dies, is trapped or is rewarded.

A branch of unsupervised object detection algorithms dedicated to objectness estimation [17] have been well-known for their performance and handiness in computer vision before convolution neural network. The cue, superpixels straddling (SS), is reported to outperform most of others in the field of objectness estimation [17]. The idea is that an object window

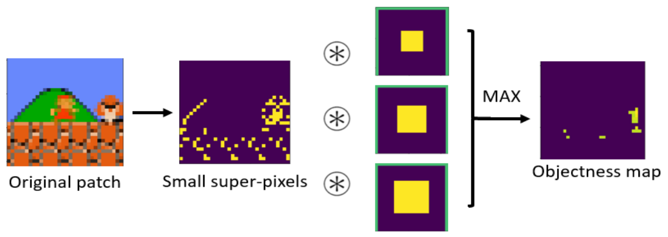

Fig. 7. Procedures of Objectness estimation

is occupied by superpxiels contained entirely inside it instead of straddling it. The SS for superpixels s in a window w is written as:

$$SS(w) = 1 - \sum_{s \in S} \frac{min(|s \backslash w|, |s \bigcap w|)}{|w|} \quad (5)$$

Where S is the set of superpixels obtained by quick-shift clustering [21]. The cost of computation of SS cannot support online operation (20FPS). We propose an efficient approximation of it in the following steps:

1) Generating superpixels using quick shift clustering from raw images.

2) Get rid of superpixels smaller than a threshold and set the left pixels to be 1. The result of this step is illustrated in Fig.7. Besides, the score of Mario is constricted.

3) Convolve the output of 2) with kernels whose center region is covered by positive values while boundary is negative values and the left is -α. We implement kernels in various sizes and keep the largest score. It makes the algorithm immune to scale variance of the object. Examples of kernels are illustrated in in Fig.7. The yellow part indicates 1 while the green indicates -α.

More formally:

$$\tilde{SS}(w,\alpha) = \sum\{p|p \in center\} - \alpha \sum\{p|p \in boundary\} \tag{6}$$

where p is pixel of the output of 2) The steps above convert the computation of Eq.5 to a single convolution computation while share the same motivation with SS. With the objectness estimation, we extract the patch with highest scores (should be higher than a threshold) in vicinity of the agent as the detected object leading to the death of the agent.

Obviously, not only death can be considered as an event, various background features of the game can also be included into this module. In other words, it is convenient to add more rule-based strategy. Another example is obstacle detection. Usually, a small reward is given as long as the agent move forward in side-scrolling video game using RL algorithms. Therefore, the obstacle can be defined as the patches in front of the agent if no reward is given for a while.

E. Replay and track for new training data

Past 20 frames are recorded as a queue before the point of the event. Once the events happen and their causes are detected, the patches of the causes are tracked in the these frames.We use CSR-tracking for its better performance based on experiments.

Examples of new training samples are illustrated in Fig.8. The new training set contains noise (questioning blocks), but they are canceled naturally during the learning of DPCN.

F. Updating memory

In order to update memory in our architecture, DPCN is trained for new dictionary matrices. Training the DPCN on the new dataset, new dictionary matrices are appended to the external memory. The receptive fields of the causes of Mario and Goomba are illustrated in Fig.6.

A significant advantage of our memory is visualization which is illustrated in Fig.6. Therefore, we know exactly what the DPCN has learned rather than considering it as a black box. Since different objects are represented by corresponding dictionary matrices, objects are classified without any additional algorithms. The result of the vision system after memory update is illustrated in Fig.9. Now, the DPCN can detect both Mario and Goomba if we repeat the Algorithm.1, i.e. blue parts of Fig.1.

IV. Conclusion

Returning to the motivation discussed at the beginning of this study, our aim is to propose an architecture which can understand the content of raw frames in the process of frame-oriented reinforcement learning, being analogous to human vision system and learning strategy. That is, learning from ”tabula rasa” and keeping learning for the whole ”life”. It is now possible to state that the proposed architecture, though preliminary, achieves the objectives.

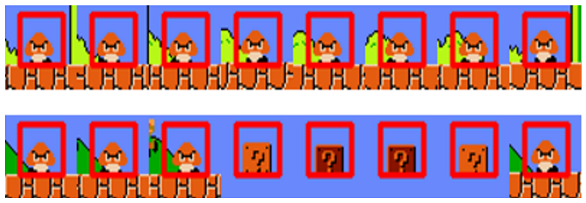
Fig. 8. Examples of new data collected automatically

In this paper, we proposed a new architecture inspired by human vision system and human learning strategy. The architecture mainly consists of ”focus of attention” algorithm, discrimination algorithm, external memory and event cause detection module. Each part of the architecture has been individually trained directly from data, but the overall algorithm training has only now started. Using Gamma saliency, DPCN, dictionary matrices of DPCN and objectness estimation corresponding the four module, respectively, we showed examples of memory update and proved the feasibility of our idea.

At first, Mario is the only object recognized by the architecture. After fifty different scenarios of Mario’s death, the cause (i.e. Goomba) is detected, and a new data subset is collected for memory update, which can be easily automated..

After update, the architecture can recognize both Goomba and Mario and classify them using incremental training. The architecture is totally automatic except the self-awareness procedure,

Another critical advantage of our architecture is model interpretability. Each parameter including the dictionary matrix has its interpretable ”physical meaning”, e.g. size of window, thresholds. We can even know what the DPCN is learning through visualizing the dictionary matrices instead of analyzing it as a black box.

In summary, our architecture remains a promising avenue for the application of life-long learning and ”tabula rasa” theory on reinforcement learning.

References

[1] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... Petersen, S. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529-533.

[2] Goel, V., Weng, J., Poupart, P. (2018). Unsupervised video object segmentation for deep reinforcement learning. In Advances in Neural Information Processing Systems (pp. 5683-5694).

[3] Jakab, T., Gupta, A., Bilen, H., Vedaldi, A. (2018). Unsupervised learning of object landmarks through conditional image generation. In Advances in Neural Information Processing Systems (pp. 4016-4027).

[4] Kulkarni, T. D., Gupta, A., Ionescu, C., Borgeaud, S., Reynolds, M., Zisserman, A., Mnih, V. (2019). Unsupervised learning of object keypoints for perception and control. In Advances in Neural Information Processing Systems (pp. 10723-10733).

[5] Yuezhang Li, Katia Sycara, and Rahul Iyer. Object-sensitive deep reinforcement learning. In Global Conference on Artificial Intelligence, volume 50, pages 20–35, 2017.

[6] world modelsHa, D., Schmidhuber, J. (2018). World models. arXiv preprint arXiv:1803.10122..

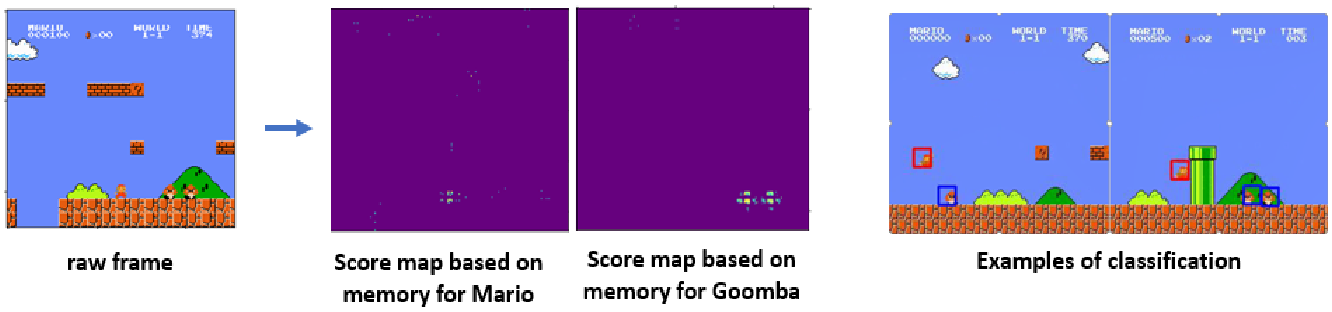
raw frame
Score map based on memory for Mario
Score map based on memory for Goomba
Examples of classification

Fig. 9. Results of the cognitive system after memory update

[7] Hebb, D. O. (2005). The organization of behavior: A neuropsychological theory. Psychology Press.

[8] Burt, R., Santana, E., Principe, J. C., Thigpen, N., Keil, A. (2016, March). Predicting visual attention using gamma kernels. In 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 1606-1610). IEEE.

[9] Chalasani, Rakesh and Principe, Jose C. Deep predictive coding networks. CoRR, abs/1301.3541, 2013.

[10] Gallup Jr, G. G., Anderson, J. R., Shillito, D. J. (2002). The mirror test. The cognitive animal: Empirical and theoretical perspectives on animal cognition, 325-333.

[11] Kulkarni, T. D., Narasimhan, K., Saeedi, A., Tenenbaum, J. (2016). Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In Advances in neural information processing systems (pp. 3675-3683).

[12] Schulman, J., Levine, S., Abbeel, P., Jordan, M., Moritz, P. (2015, June). Trust region policy optimization. In International conference on machine learning (pp. 1889-1897).

[13] Lange, S., Riedmiller, M. (2010, July). Deep auto-encoder neural networks in reinforcement learning. In The 2010 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.

[14] Finn, C., Tan, X. Y., Duan, Y., Darrell, T., Levine, S., Abbeel, P. (2016, May). Deep spatial autoencoders for visuomotor learning. In 2016 IEEE International Conference on Robotics and Automation (ICRA) (pp. 512-519). IEEE.

[15] Higgins, I., Pal, A., Rusu, A., Matthey, L., Burgess, C., Pritzel, A., ... Lerchner, A. (2017, August). Darla: Improving zero-shot transfer in reinforcement learning. In Proceedings of the 34th International Conference on Machine Learning-Volume 70 (pp. 1480-1490). JMLR. org.

[16] Zhu, S. C., Mumford, D. (2007). A stochastic grammar of images. Foundations and Trends® in Computer Graphics and Vision, 2(4), 259-362.

[17] Alexe, B., Deselaers, T., Ferrari, V. (2012). Measuring the objectness of image windows. IEEE transactions on pattern analysis and machine intelligence, 34(11), 2189-2202.

[18] Lukezic, A., Vojir, T., ˇCehovin Zajc, L., Matas, J., Kristan, M. (2017). Discriminative correlation filter with channel and spatial reliability. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 6309-6318).

[19] Gregor, K., LeCun, Y. (2010, June). Learning fast approximations of sparse coding. In Proceedings of the 27th International Conference on International Conference on Machine Learning (pp. 399-406).

[20] Chalasani, R., Principe, J. C., Ramakrishnan, N. (2013, August). A fast proximal method for convolutional sparse coding. In The 2013 International Joint Conference on Neural Networks (IJCNN) (pp. 1-5). IEEE.

[21] Vedaldi, A., Soatto, S. (2008, October). Quick shift and kernel methods for mode seeking. In European conference on computer vision (pp. 705-718). Springer, Berlin, Heidelberg.

[22] Bellman, R. (1954). The theory of dynamic programming (No. RAND-P-550). Rand corp santa monica ca.

[23] Wiener, N (1956). The Theory of Prediction. Modern Mathematics for engineers.

[24] Granger, C. W. (1969). Investigating causal relations by econometric models and cross-spectral methods. Econometrica: journal of the Econometric Society, 424-438.

[25] Pearl, J. (2009). Causality. Cambridge university press.