CONTINGENCY-AWARE EXPLORATION IN REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper investigates whether learning contingency-awareness and controllable aspects of an environment can lead to better exploration in reinforcement learning. To investigate this question, we consider an instantiation of this hypothesis evaluated on the Arcade Learning Element (ALE). In this study, we develop an attentive dynamics model (ADM) that discovers controllable elements of the observations, which are often associated with the location of the character in Atari games. The ADM is trained in a self-supervised fashion to predict the actions taken by the agent. The learned contingency information is used as a part of the state representation for exploration purposes. We demonstrate that combining A2C with count-based exploration using our representation achieves impressive results on a set of notoriously challenging Atari games due to sparse rewards. For example, we report a state-of-the-art score of >6600 points on MONTEZUMA'S REVENGE without using expert demonstrations, explicit high-level information (*e.g.* RAM states), or supervised data.[1] Our experiments confirm our hypothesis that contingency-awareness is an extremely powerful concept for tackling exploration problems in reinforcement learning and opens up interesting research questions for further investigations.

1 INTRODUCTION

The success of reinforcement learning (RL) algorithms in complex environments hinges on the way they balance *exploration* and *exploitation*. There has been a surge of recent interest in developing effective exploration strategies for problems with high-dimensional state spaces and sparse rewards (Schmidhuber, 1991; Oudeyer and Kaplan, 2009; Houthooft et al., 2016; Bellemare et al., 2016; Osband et al., 2016; Pathak et al., 2017; Plappert et al., 2018; Zheng et al., 2018). Deep neural nets have seen great success as expressive function approximators within RL and as powerful representation learning methods for many domains. However, there is an ongoing open question about the most effective way of using neural network representations for exploration (Tang et al., 2017; Martin et al., 2017; Pathak et al., 2017). For example, count-based exploration with neural density estimation (Bellemare et al., 2016; Tang et al., 2017; Ostrovski et al., 2017) presents one of the state-of-the-art techniques on the most challenging Atari games with sparse rewards.

Despite the success of recent exploration methods, it is still an open question on how to construct an optimal representation for exploration. For example, the concept of visual similarity is used for learning neural density models as a basis for calculating *pseudo-counts* (Ostrovski et al., 2017). However, as Tang et al. (2017) noted, the ideal way of representing states should be based on what is relevant to solving the MDP, rather than only relying on visual similarity. In addition, there remains another question on whether the representations used for recent exploration work are easily interpretable. To address these questions, we investigate whether we can learn a complementary and more intuitive and interpretable high-level abstraction that can be very effective in exploration, by using the ideas of contingency awareness and controllable dynamics.

The key idea we focus on in this work is the notion of *contingency awareness* (Watson, 1966a; Bellemare et al., 2012) — the agent's understanding of the environmental dynamics and recognizing that some aspects of the dynamics are under the agent's control. Intuitively speaking, this can represent the segmentation mask of the agent operating in the 2D or 3D environments (yet one can think of more abstract and general state spaces). In this study, we investigate the concept of contingency-awareness based on *self-localization*, *i.e.* the awareness of where the agent is located in the abstract state space. We are interested in discovering parts of the world that are directly dependent on the agent's immediate action, which often reveals the agent's approximate location.

[1] Examples of the learned policy and the contingent regions are available at http://goo.gl/PPN32n.

To further motivate the problem, we note that contingency-awareness is a very important concept in developmental psychology. For example, being aware of the location of the agent itself is an important property of many observed intelligent systems; we can think of self-localization (and more broadly self-awareness) as a principled, fundamental direction towards intelligent agents. Recent breakthroughs in neuroscience, such as discovering grid cells (Moser et al., 2015; Banino et al., 2018), show that organisms that perform very well in spatially-challenging tasks are self-aware of their location. This allows rats to navigate, remember paths to visited places and important sub-goals, and find shortcuts. In addition, the notion of contingency awareness has been shown as an important factor in developmental psychology (Watson, 1966b; Baeyens et al., 1990).

These discussions suggest that contingency-awareness can be an extremely powerful mechanism for tackling exploration problems in reinforcement learning setting. We consider an instantiation of this hypothesis evaluated on the Arcade Learning Element (ALE). For example, in the context of 2D Atari games, contingency-awareness roughly corresponds to understanding the notion of controllable entities (*e.g.* the player's avatar), which Bellemare et al. (2012) refer to as *contingent regions*. More concretely, as shown in Figure 1, in the game FREEWAY, only the chicken sprite is under the agent's control and not the multiple moving cars. Therefore the chicken's location should be an informative element for exploration (Bellemare et al., 2012; Pathak et al., 2017).

In this study, we also investigate whether contingency-awareness can be learned without any external annotations or supervision. For this, we provide an instantiation of an algorithm for automatically learning such information and using it for improving exploration on a 2D ALE environment (Bellemare et al., 2013). Concretely, we employ an attentive dynamics model (ADM) to predict the agent's action chosen between consecutive states. It allows us to approximate the agent's position in 2D environments, but unlike other approaches such as (Bellemare et al., 2012), it does not require any additional supervision to do so. ADM learns in an online fashion with pure observations as the agent's policy is updated and does not require hand-crafted features, an environment simulator, or supervision labels for training.

In experimental evaluation, our methods significantly improve the performance of A2C on hard-exploration Atari games in comparison with competitive methods such as count-based exploration (Bellemare et al., 2016; Ostrovski et al., 2017) and SimHash (Tang et al., 2017). We report very strong results on sparse reward games, including the state-of-the-art performance on the notoriously difficult MONTEZUMA'S REVENGE, without using expert demonstrations nor explicit high-level information.

We summarize our contributions as follows:

- We demonstrate the importance of learning contingency-awareness for efficient exploration in challenging sparse-reward RL problems.
- We develop a novel instance of attentive dynamics model using contingency and controllable dynamics to provide robust localization abilities across the most challenging Atari environments.
- We achieve a strong performance on difficult sparse-reward Atari games, including a state of the art score on notoriously challenging MONTEZUMA'S REVENGE.

Overall, we believe our experiments confirm the hypothesis that contingency-awareness is an extremely powerful concept for tackling exploration problems in reinforcement learning, and it opens up interesting research questions for further investigations.

2 RELATED WORK

Biological Motivation. The discovery of grid cells (Moser et al., 2015) motivates working on agents that are self-aware of their location. Banino et al. (2018) emphasize the importance of self-localization and train a neural network which learns a similar mechanism to grid cells when performing tasks related to spatial navigation. The presence of grid cells was correlated with high performance. A set of potential approaches to self-localization ranges from ideas specific to a given environment, *e.g.* SLAM (Durrant-Whyte and Bailey, 2006), to methods with potential generalizability. Although grid cells seem tailored to 2D or 3D problems that animals encounter in their life, it is speculated that their use can be extended to more abstract spaces.

Self-supervised Dynamics Model. Several works have used forward and/or inverse dynamics models of the environment: (Oh et al., 2015; Agrawal et al., 2016; Shelhamer et al., 2017). Pathak et al. (2017) . They employ a similar dynamics model to learn a feature representation of states that captures controllable aspects of the environment. This dense representation is used to design

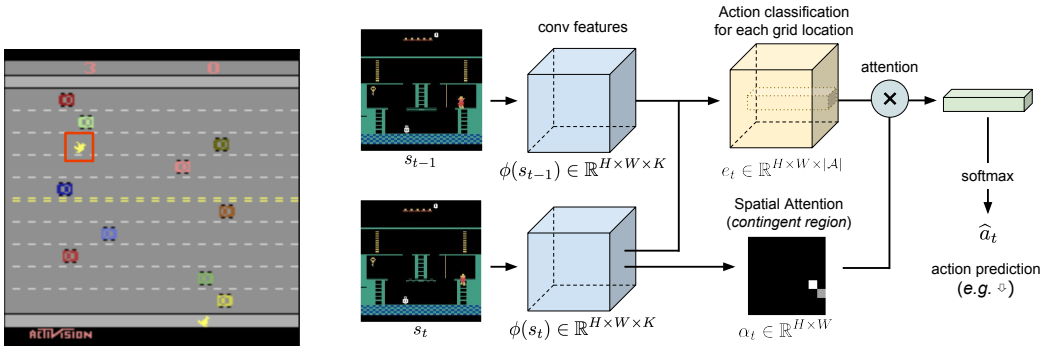

Figure 1: **Left:** Contingent region in FREEWAY; an object in red box denotes what is under the agent's control, whereas the rest is not. **Right:** A diagram for the proposed attentive ADM architecture.

a curiosity-driven intrinsic reward. Our presented approach is different as we focus on explicitly discovering controllable aspects using an attention mechanism, resulting in better interpretability.

Exploration and Intrinsic Motivation. The idea of providing an exploration bonus reward depending on the state-action visit-count was proposed by Strehl and Littman (2008) (MBIE-EB), originally under a tabular setting. Later it has been combined with different techniques to deal with high-dimensional state spaces. Bellemare et al. (2016) use a Context-Tree Switching (CTS) density model to derive a state *pseudo-count*, and Ostrovski et al. (2017) use PixelCNN as a state density estimator. Martin et al. (2017) also construct a visitation density model over a compressed feature space rather than the raw observation space. Alternatively, Tang et al. (2017) proposed a locality-sensitive hashing (LSH) method to cluster states and maintain a state-visitation counter based on a form of similarity between frames. We train an agent with a similar count-based exploration bonus, but the way of maintaining state counter seems relatively simpler in that key feature information (*i.e.* controllable region) is explicitly extracted from the observation and directly used for counting states.

Another popular family of exploration strategies in RL uses intrinsic motivation (Oudeyer and Kaplan, 2009; Pathak et al., 2017). These methods encourage the agent to look for something surprising in the environment dynamics which motivates its search for novel states, such as surprise (Achiam and Sastry, 2017) or curiosity (Pathak et al., 2017; Burda et al., 2018). More comprehensive survey on intrinsic motivation for RL can be found in (Oudeyer and Kaplan, 2009).

3 APPROACH

3.1 DISCOVERING CONTINGENCY VIA ATTENTIVE DYNAMICS MODEL

To discover the region of the observation is controllable by the agent, we develop an instance of *attentive dynamics model* based on inverse dynamics f_{inv}. The model takes two consecutive input frames (observations) $s_{t-1}, s_t \in \mathcal{S}$ as input and aims to predict the action ($a_t \in \mathcal{A}$) taken by the agent to move from s_{t-1} to s_t:

$$\widehat{a}_t = f_{\text{inv}}(s_{t-1}, s_t). \tag{1}$$

Our key intuition is that the inverse dynamics model should attend to the most relevant part of the observation, which is controllable by the agent, to be able to classify the actions. We determine whether each region in a $H \times W$ grid is controllable, or in other words, useful for predicting the agent's action, by using a spatial *attention mechanism* (Bahdanau et al., 2015; Xu et al., 2015).

Model. To perform action classification, we first compute a convolutional feature map $\phi_t^s = \phi(s_t) \in \mathbb{R}^{H\times W\times K}$ based on the observation s_t using a convolutional neural network ϕ. We estimate a set of *logit* (score) vectors, denoted $e_t(i,j) \in \mathbb{R}^{|\mathcal{A}|}$, for action classification from each grid cell (i,j) of the convolutional feature map. The local convolution features and feature difference for consecutive frames are fed into a shared multi-layer perceptron (MLP) to derive the logits as:

$$e_t(i,j) = \text{MLP}\Big(\big[\phi_t^s(i,j) - \phi_{t-1}^s(i,j);\ \phi_t^s(i,j)\big]\Big) \in \mathbb{R}^{|\mathcal{A}|}. \tag{2}$$

We then compute an attention mask $\alpha_t \in \mathbb{R}^{H\times W}$ corresponding to frame t, which indicates the controllable parts of the observation s_t. Such attention masks are computed via a separate MLP from

the features of each region (i, j), and then converted into a probability distribution using softmax or sparsemax operators (Martins and Astudillo, 2016):

$$\alpha_t = \text{sparsemax}(\widetilde{\alpha}_t) \quad \text{where} \quad \widetilde{\alpha}_t(i,j) = \text{MLP}\big(\phi_t^s(i,j)\big) \ , \tag{3}$$

so that $\sum_{i,j} \alpha_t(i,j) = 1$. The sparsemax operator is similar to softmax but yields a sparse attention, leading to more stable performance. Finally, the logits $e_t(i,j)$ from all regions are linearly combined using the attention probabilities α_t:

$$p(\widehat{a}_t \mid s_{t-1}, s_t) = \text{softmax}\Big(\textstyle\sum_{i,j} \alpha_t(i,j) \cdot e_t(i,j)\Big) \in \mathbb{R}^{|\mathcal{A}|}. \tag{4}$$

Training. The model can be optimized with the standard cross-entropy loss $\mathcal{L}_{\text{action}}(a_t^*, \widehat{a}_t)$ with respect to the ground-truth action $a_t^* \in \mathcal{A}$ that the agent actually has taken. Based on this formulation, the attention probability $\alpha_t(i,j)$ should be high only on regions (i,j) that are predictive of the agent's actions. Our formulation enables learning to localize controllable entities in a fully unsupervised way without any additional supervisory signal, unlike some prior work (*e.g.* Bellemare et al. (2012)) that adopts simulators to collect extra supervisory labels.

Optimizing the parameters of ADM on on-policy data is challenging for several reasons. First, the ground-truth action may be unpredictable for given pairs of frames, leading to noisy labels. For example, the actions taken in uncontrollable situations do not have any effect (*e.g.* when the agent is in the middle of jumping in MONTEZUMA'S REVENGE). Second, since we train the ADM online, along with the policy, the training examples are not independently and identically distributed, and the data distribution can shift dramatically over time. Third, the action distribution from the agent's policy can attain a low entropy, being biased towards certain actions. These issues may prevent the ADM from generalization to novel ovservations, which hurts exploration. Generally, we prefer models that quickly adapt to the policy and learn to localize the controllable regions in a robust manner.

To mitigate the aforementioned issues, we adopt a few additional objective functions. We encourage the attention distribution to attain a high entropy by including an *attention entropy regularization loss*, *i.e.* $\mathcal{L}_{\text{ent}} = -\mathcal{H}(\alpha_t)$. This term penalizes over-confident attention masks, making the attention closer to uniform whenever action prediction is not possible. We also train the logits corresponding to each grid cell independently using a separate cross entropy loss, $p(\widehat{\alpha}_t^{i,j} \mid e_t(i,j)) = \text{softmax}(e_t(i,j))$. These additional cross-entropy losses, denoted $\mathcal{L}_{\text{cell}}^{i,j}$. allow the model to learn from novel observations even when attention fails to perform well at first. The entire training objective becomes

$$\mathcal{L} = \mathcal{L}_{\text{action}} + \textstyle\sum_{i,j} \mathcal{L}_{\text{cell}}^{i,j} + \lambda_{\text{ent}} \mathcal{L}_{\text{ent}} \tag{5}$$

where λ_{ent} is a mixing hyperparameter.

3.2 COUNT-BASED EXPLORATION WITH CONTINGENT REGIONS

One natural way to take advantage of discovered contingent regions for exploration is count-based exploration. The attentive dynamics model can be used to localize the controllable entity (*e.g.* the agent's avatar) from an observation s_t experienced by the agent. In 2D environments, a natural discretization (x, y) where $(x, y) = \text{argmax}_{(i,j)} \alpha_t(i,j)$ provides a good approximation of the agent's location within the current observation[2]. This provides a key piece of information about the current state of the agent.

Inspired by privious work (Bellemare et al., 2016; Tang et al., 2017), we add an exploration bonus of r^+ to the environment reward, where $r^+(s) = 1/\sqrt{\#(\psi(s))}$ and $\#(\psi(s))$ denotes the visitation count of the mapped state $\psi(s)$. We want to find a policy π that maximizes the expected discounted sum of environment rewards plus the count-based exploration bonus, denoted $R = \mathbb{E}_\pi\big[\sum_t \gamma^t \left(\beta_1 r(s_t, a_t) + \beta_2 r^+(s_t)\right)\big]$, where $\beta_1 \geq 0$ and $\beta_2 \geq 0$ are hyperparameters that balances the weight of environment reward and the exploration bonus. For every state s_t encountered at time step t, we increase the counter value $\#(\psi(s_t))$ by 1 during training. The overall procedure of count-based exploration algorithm is summarized in Algorithm 1 in the Appendix.

[2] To obtain more accurate localization by taking temporal correlation into account, we can use exponential smoothing as $\overline{\alpha}_t(i,j) = (1 - c_t)\overline{\alpha}_{t-1}(i,j) + c_t \alpha_t(i,j)$, where $c_t = \max_{(i,j)}\{\alpha_t(i,j)\}$.

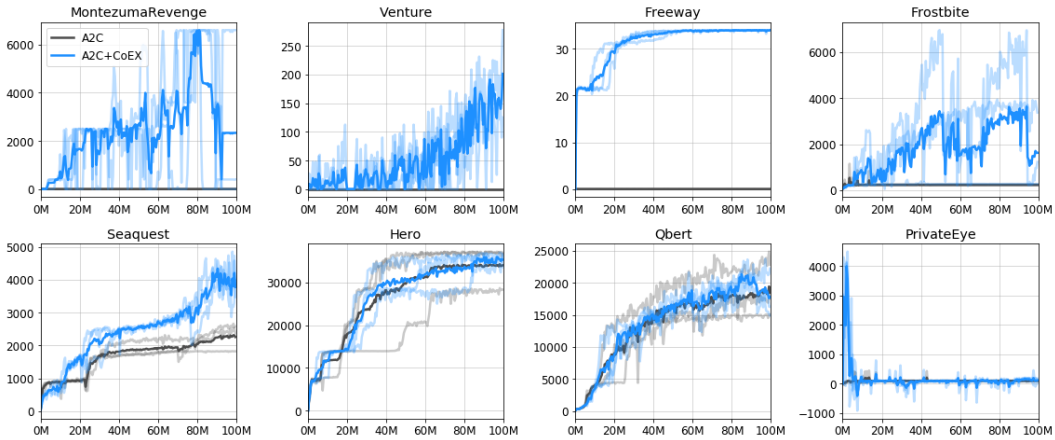

Figure 2: Learning curves on several Atari games: A2C+CoEX and A2C. (Best viewed zoomed-in)

4 EXPERIMENTS

In the experiments below we investigate the following key questions:

- Does the contingency awareness in terms of self-localization provide a useful state abstraction for exploration?
- How well can an unsupervised model discover the ground truth abstract states?
- How well does the proposed exploration strategy perform against other exploration methods?

4.1 EXPERIMENTAL SETUP

We evaluate the proposed exploration strategy on several difficult exploration Atari 2600 games from the Arcade Learning Environment (ALE) (Bellemare et al., 2013). We focus on 8 Atari games including FREEWAY, FROSTBITE, HERO, PRIVATEEYE, MONTEZUMA'S REVENGE, QBERT, SEAQUEST, and VENTURE. In such games an agent without an effective exploration strategy can often converge to a suboptimal policy. For example, as depicted in Figure 2, our Advantage Actor-Critic (A2C) baseline (Mnih et al., 2016) achieves a reward close to 0 on MONTEZUMA'S REVENGE, VENTURE, FREEWAY, FROSTBITE, and PRIVATEEYE, even after 100M steps of training. By contrast, our proposed technique, which augments A2C with count-based exploration with the location information learned by the inverse dynamics model, denoted **A2C+CoEX** (CoEX stands for "Contingency-aware Exploration"), significantly outperforms the A2C baseline on 6 out of the 8 games, presenting a new state of the art for exploration in difficult Atari games (see Table 2).

We compare our proposed A2C+CoEX technique against the following baselines:

- **A2C**: an implementation adopted from OpenAI baselines (Dhariwal et al., 2017) using the default hyperparameters, which serves as the building block of our more complicated baselines.
- **A2C+Pixel-SimHash**: Following (Tang et al., 2017), we map 52×52 gray-scale observations to 128-bit binary codes using random projection followed by quantization (Charikar, 2002). Then, we add a count-based exploration bonus based on quantized observations.

As control experiment, we also evaluate **A2C+CoEX+RAM***, our contingency-aware exploration method with the ground truth location information from game RAM, which roughly shows upper-bound performance of our proposed method.

4.2 IMPLEMENTATION DETAILS

For the A2C (Mnih et al., 2016) algorithm, we use 16 parallel actors to collect the agent's experience, with 5-step rollout, which yields a minibatch of size 80 for on-policy transitions.We use the last 4 observation frames stacked as input, each of which is resized to 84×84 and converted to grayscale as in Mnih et al. (2015; 2016). We set the end of episode to be when the game ends, rather than when the agent loses a life. More implementation details can be found in Appendix B.

For the inverse dynamics model, we take observation frames of size 160×160 as input (resized from the raw observation of size 210×160).[3] We employ a 4-layer convolutional neural network that

[3] In some games such as Venture, the agent is depicted in very small pixels, which might be hardly recognizable in rescaled 84×84 images.

	Freeway	Frostbite	Hero	Montezuma	PrivateEye	Qbert	Seaquest	Venture
A2C	7.17	1099	34352	12.5	574	19620	2401	0
A2C+Pixel-SimHash	0.0	829	28181	412	276	18180	2177	31
A2C+CoEX	**34.0**	**4260**	**36827**	**6635**	**5316**	**23962**	**5169**	**204**
A2C+CoEX+RAM*	34.0	4418	36765	6600	24296	24422	6113	1100

Table 1: Performance of our method and its baselines on Atari games: maximum scores achieved over 100M timesteps (400M frames) of training, averaged over 3 seeds. The best entry in the group of experiments without supervision is shown in bold. * denotes that A2C+CoEX+RAM acts as a control experiment, which includes some supervision. More experimental results about A2C+CoEX+RAM are shown in Appendix C.

Approach	#Steps	Freeway	Frostbite	Hero	Montezuma	PrivateEye	Qbert	Seaquest	Venture
A2C+CoEX (Ours)	50M	33.9	3900	31367	4100	5316	17724	2620	128
A2C+CoEX (Ours)	100M	**34.0**	4260	**36827**	**6635**	5316	**23962**	**5169**	204
DDQN+	50M	29.2	1450	-	3439	-	-	-	369
A3C+	50M	27.3	507	15210	142	100	15805	2274	0
TRPO-AE-SimHash	50M	33.5	**5214**	-	75	-	-	-	445
Sarsa-ϕ-EB†	25M	0.0	2770	-	2745	-	4112	-	1169
DQN-PixelCNN*	37.5M	31.7	-	-	2514	**15806**	5501	-	**1356**
Curiosity-Driven†	25M	32.8	-	-	2505	3037	-	-	416

Table 2: Performance of our method and state of the art exploration methods on Atari games. For fair comparison, we report maximum scores achieved over the specific number of timesteps during training, averaged over 3 seeds. The best entry is shown in bold. For the baselines (for reference), DDQN+ and A3C+ are from (Bellemare et al., 2016). TRPO-AE-Simhash, Sarsa-ϕ-EB, and Curiosity-Driven are from (Tang et al., 2017), (Martin et al., 2017), and (Burda et al., 2018) respectively.

produces a feature map $\phi(s_t)$ with spatial grid size of $H \times W = 9 \times 9$. As a result, the prediction of location coordinates lies in the 9×9 grid.

In some environments, the contingent regions within the visual observation alone are not sufficient to determine the exact location of the agent within the game; for example, the coordinate alone is not enough to distinguish between different rooms in HERO, MONTEZUMA'S REVENGE, PRIVATEEYE, etc. Therefore, we introduce a discrete context representation $c \in \mathbb{Z}$ that summarizes the high-level visual context in which the agent currently lies. We use a simple clustering method which we refer to as observation embedding clustering that clusters the random projection vector of the input frames as in (Tang et al., 2017), so that different contexts are assigned to different clusters. We explain this heuristic approach more in detail in the Appendix D.

In sparse reward problems the act of collecting a reward is rare but frequently instrumental for the future states of the environment. The cumulative reward, $R_t = \sum_{t'=0}^{t-1} r(s_{t'}, a_{t'})$ from the beginning of the episode up to the current step t, can provide a useful high-level *behavioral context*, because collecting rewards can trigger significant changes to the agent's state and as a result the optimal behavior can change as well. In this sense, the agent should revisit the previously visited location for exploration when the context changes. For example, in MONTEZUMA'S REVENGE, if the agent is in the first room and the cumulative reward up to now is 0, we know the agent has not picked up the key, and the optimal policy is to reach the key. However, if the cumulative reward in the first room is 100, it means the agent has picked up the key and the next optimal goal is to open a door and move on to the next room. Therefore, we include the cumulative reward as part of state abstraction for exploration, which leads to empirically better performance.

To sum up, for the purpose of count-based exploration, we utilize the location (x, y) of the controllable entity (*i.e.* the agent) in the current observation discovered by attentive dynamics model (Section 3.1), a context representation $c \in \mathbb{Z}$ that denotes the high level visual context, and cumulative environment reward $R \in \mathbb{Z}$ that represents the exploration behavioral state, as will be described below. In such setting, we may denote $\psi(s) = (x, y, c, R)$.

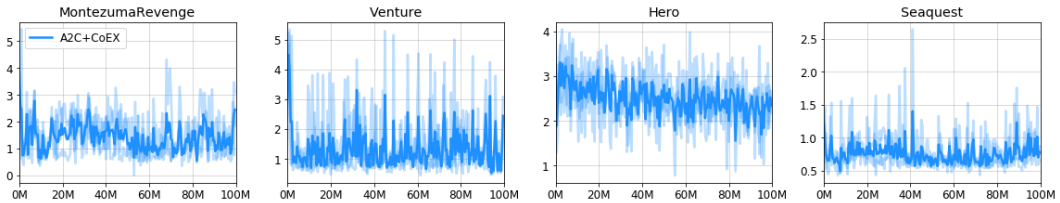

Figure 3: Performance plot of the inverse dynamic models, trained online using on-policy transition samples from the A2C agent.

4.3 Performance of Count-based Exploration

Figure 2 shows the learning curves of the proposed methods on 8 Atari games. The performance of our method A2C+CoEX and A2C+CoEX+RAM as well as baselines A2C and A2C+Pixel-SimHash are summarized in Table 1. In order to find a balance between environment reward and exploration bonus reward, we perform a hyper-parameter search for the proper weight of environment reward β_1 and exploration reward β_2 for A2C+CoEX+RAM, as well as for A2C+CoEX. The hyper-parameters for the two ended up being the same, which is consistent with our results. For fair comparison, we also search for the proper weight of environment reward for A2C baseline. The best hyper-parameters for each game are shown in Table 4 in Appendix B.

Compared to the vanilla A2C, the proposed exploration strategy improves the score on all the hard-exploration games. In Table 1, if representation (x, y, c, R) is perfect, A2C+CoEX+RAM achieves a significant improvement over A2C by encouraging the agent to visit novel locations, and could nearly solve these hard exploration games, especially the games with sparse reward, as training goes on.

Furthermore, A2C+CoEX using representation learned with our proposed inverse dynamic model and observation embedding clustering, also outperforms the A2C baseline. Especially on Freeway, Frostbite, Hero, Montezuma's Revenge, Qbert and Seaquest, the performance is comparable with A2C+CoEX+RAM, which demonstrates the usefulness of ADM.

Comparison to other count-based exploration methods. Table 2 contrasts the proposed method with previous state of the art results. We outperform the other methods on 5 out of 8 games. DQN-PixelCNN is the strongest alternative achieving a state of the art performance on some of the most difficult sparse-reward games. We argue that using Q-learning as the base learner with DQN-PixelCNN makes the direct comparison with A2C+CoEX not completely adequate. Please note that the closest count-based alternative to A2C+CoEX would be A3C+ (Bellemare et al., 2016), which augments A3C (Mnih et al., 2016) with exploration bonus derived from pseudo-count, because A2C and A3C share the similar policy learning method. With that in mind, one can observe a clear improvement of A2C+CoEX over A3C+ on all of the Atari games.

4.4 Analysis of Inverse Dynamics Model

We also analyze the performance of the inverse dynamics model that learns the controllable dynamics of the environment. As a performance metric, we report the average distance between the ground-truth agent location (x^*, y^*) and the predicted location (x, y) within the 9×9 grid: $\|(x, y) - (x^*, y^*)\|_2$. The ground-truth location of the agent is extracted from RAM[4], then rescaled so that the observation image frame fits into the 9×9 grid.

Figure 3 shows the result on 4 Atari games (Montezuma's Revenge, Seaquest, Hero, Venture). The inverse dynamic model is able to quickly capture the location of the agent without any supervision of localization and despite the agent constantly visiting new places. Typically the predicted location is on average 1 or 2 grid cells away from the ground-truth location. Whenever a novel scene is encountered (*e.g.* the second room in Montezuma's Revenge at 10–15M frames), the average distance temporarily increases, but quickly drops again as the model learns the new room. We provide with the video of playing agents and localization information.[5]

4.5 Analysis of Observation Embedding Clustering

For games such as Montezuma's Revenge and Venture, there is a change of high-level visual contexts (*i.e.* rooms in Atari games). To make the agent aware of it, we obtain a representation of the

[4] Please note that the location from RAM is used only for analysis and evaluation purposes.

[5] A demo video of the learnt policy and localization is available at http://goo.gl/PPN32n.

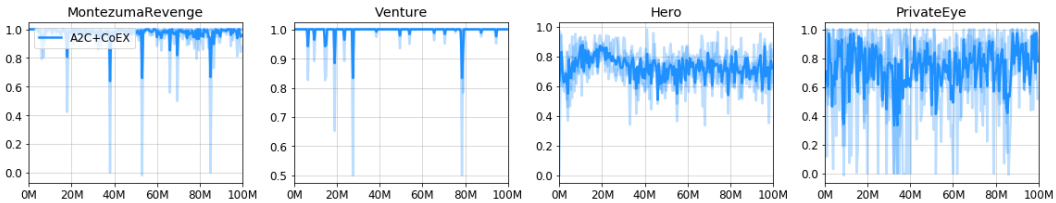

Figure 4: Curves of Adjusted Rand Index on several Atari games during training, averaged over the 100 most recent observations.

high-level context and use it for exploration. The high-level visual contexts are different from each other (different layouts, objects, colors, etc.), so the embedding generated by a random projection is quite distinguishable and clustering is accurate and robust.

For evaluation, given an observation in Atari games, we get the ground truth room number from RAM, and discrete representation (i.e. which cluster it's assigned in) based on the embedding from our random projection and we compare the discrete representation to the ground truth room number. Adjusted Rand Index (ARI) (Rand, 1971) measures the similarity between these two data clusterings. The ARI may only yield a value between 0 and 1, and is exactly 1 when the clusterings are identical.

The curves of Adjusted Rand Index is shown in Figure 4. For MONTEZUMA'S REVENGE and VENTURE, the discrete representation as room number is roughly as good as the ground truth. For HERO and PRIVATEEYE, because there are many rooms quite similar with each other, it's more challenging to cluster the embedding perfectly. From the samples shown in Figure 6 in Appendix D, we can see the reasonable performance of the clustering method on all these games.

4.6 DISCUSSIONS AND FUTURE WORK

This paper investigates whether discovering controllable dynamics via an attentive dynamics model (ADM) can help exploration in challenging sparse reward environments. We demonstrate the effectiveness of this approach by achieving significant improvements on notoriously difficult video games. That being said, we acknowledge that our approach has certain limitations. Our currently presented instance of state abstraction method mainly focuses on controllable dynamics and employs a simple clustering scheme to abstract uncontrollable elements of the scene. In more general setting, one can imagine using attentive (forward or inverse) dynamics models to learn an effective and compact abstraction of the controllable and uncontrollable dynamics as well, but we leave this to future work.

Key elements of the current ADM method include the use of spatial attention and modelling of the dynamics. These ideas can be generalized by a set of attention-based dynamics models (ADM) operating in forward, inverse, or combined mode. Such models could use attention over a lower-dimensional embedding that corresponds to an intrinsic manifold structure from the environment (i.e., intuitively speaking, this also corresponds to *being self-aware of (e.g., locating) where the agent is in the abstract state space*). Our experiments with the inverse dynamics model suggest that the mechanism does not have to be perfectly precise, allowing for some error in practice. We speculate that mapping to such subspace could be obtained by techniques of embedding learning.

We note that RL environments with different visual characteristics may require different forms of attention-based techniques and properties of the model (*e.g.* partial observability). Even though this paper focuses on 2D video games, we believe that the presented high-level ideas of learning contingency-awareness (with attention and dynamics models) are more general and coule be applicable to more complex 3D environments with some extension. We leave this as future work.

5 CONCLUSION

We proposed a method of providing contingency-awareness through an attentive dynamics model (ADM). It enables approximate self-localization for an RL agent in 2D environments (as a specific perspective). The agent is able to estimate its position in the space and therefore benefits from a compact, informative representation of the world. This idea combined with a variant of count-based exploration achieves strong results in various sparse reward Atari games. Furthermore, we report state of the art results of 6600+ points on the infamously challenging MONTEZUMA'S REVENGE without using expert demonstrations or supervision. Though in this work we focus mostly on 2D environments in the form of sparse reward Atari games, we view our presented high-level concept and approach as a stepping stone towards more universal algorithms capable of similar abilities in various RL environments.

REFERENCES

Joshua Achiam and Shankar Sastry. Surprise-Based Intrinsic Motivation for Deep Reinforcement Learning. 2017.

Pulkit Agrawal, Ashvin Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to Poke by Poking: Experiential Learning of Intuitive Physics. In *NIPS*, 2016.

Frank Baeyens, Paul Eelen, and Omer van den Bergh. Contingency awareness in evaluative conditioning: A case for unaware affective-evaluative learning. *Cognition and emotion*, 4(1):3–18, 1990.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*, 2015.

Andrea Banino, Caswell Barry, Benigno Uria, Charles Blundell, Timothy Lillicrap, Piotr Mirowski, Alexander Pritzel, Martin J Chadwick, Thomas Degris, Joseph Modayil, Greg Wayne, Hubert Soyer, Fabio Viola, Brian Zhang, Ross Goroshin, Neil Rabinowitz, Razvan Pascanu, Charlie Beattie, Stig Petersen, Amir Sadik, Stephen Gaffney, Helen King, Koray Kavukcuoglu, Demis Hassabis, Raia Hadsell, and Dharshan Kumaran. Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557(7705):429–433, 2018.

Marc G Bellemare, Joel Veness, and Michael Bowling. Investigating Contingency Awareness Using Atari 2600 Games. In *AAAI*, 2012.

Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The Arcade Learning Environment: An Evaluation Platform for General Agents. *Journal of Artificial Intelligence Research 47*, 2013.

Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying Count-Based Exploration and Intrinsic Motivation. In *NIPS*, 2016.

Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A. Efros. Large-scale study of curiosity-driven learning. In *NIPS*, 2018.

Moses S Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 380–388. ACM, 2002.

Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, and Yuhuai Wu. OpenAI Baselines. https://github.com/openai/baselines, 2017.

Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. VIME: Variational Information Maximizing Exploration. In *NIPS*, 2016.

Jarryd Martin, Suraj Narayanan Sasikumar, Tom Everitt, and Marcus Hutter. Count-Based Exploration in Feature Space for Reinforcement Learning. 2017.

André F T Martins and Ramón Fernandez Astudillo. From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification. In *ICML*, 2016.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 2015.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous Methods for Deep Reinforcement Learning. In *ICML*, 2016.

May-Britt Moser, David Rowland, and Edvard I Moser. Place cells, grid cells, and memory. 5, 2015.

Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L Lewis, and Satinder P Singh. Action-Conditional Video Prediction using Deep Networks in Atari Games. In *NIPS*, 2015.

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep Exploration via Bootstrapped DQN. In *NIPS*, 2016.

Georg Ostrovski, Marc G. Bellemare, Aaron van den Oord, and Remi Munos. Count-Based Exploration with Neural Density Models. In *ICML*, 2017.

Pierre-Yves Oudeyer and Frederic Kaplan. What is intrinsic motivation? A typology of computational approaches. *Frontiers in Neurorobotics*, 2009.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven Exploration by Self-supervised Prediction. In *ICML*, 2017.

Matthias Plappert, Rein Houthooft, Prafulla Dhariwal, Szymon Sidor, Richard Y Chen, Xi Chen, Tamim Asfour, Pieter Abbeel, and Marcin Andrychowicz. Parameter Space Noise for Exploration. In *ICLR*, 2018.

William M. Rand. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

Jürgen Schmidhuber. Adaptive confidence and adaptive curiosity. 1991.

Evan Shelhamer, Parsa Mahmoudieh, Max Argus, and Trevor Darrell. Loss is its own Reward: Self-Supervision for Reinforcement Learning. 2017.

Alexander L. Strehl and Michael L. Littman. An analysis of model-based interval estimation for markov decision processes. *Journal of Computer and System Sciences*, 74(8), 2008.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. #Exploration: A Study of Count-Based Exploration for Deep Reinforcement Learning. In *NIPS*, 2017.

John S Watson. The development and generalization of "contingency awareness" in early infancy: Some hypotheses. *Merrill-Palmer Quarterly of Behavior and Development*, 12(2):123–135, 1966a.

John S Watson. The development and generalization of" contingency awareness" in early infancy: Some hypotheses. *Merrill-Palmer Quarterly of Behavior and Development*, 12(2):123–135, 1966b.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C . Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *ICML*, 2015.

Zeyu Zheng, Junhyuk Oh, and Satinder Singh. On Learning Intrinsic Rewards for Policy Gradient Methods. In *NIPS*, 2018.

APPENDIX

A SUMMARY OF TRAINING ALGORITHM

Algorithm 1 Actor-Critic with count-based exploration

Initialize parameter θ_{inv} for inverse dynamics model f_{inv}
Initialize parameter θ_{a2c} for actor critic network
Initialize parameter θ_c for context embedding projector if applicable (which is not trainable)
Initialize transition buffer $\mathcal{E} \leftarrow \emptyset$
for each iteration **do**
  $\triangleright$ *Collect on-policy transition samples, distributed over K parallel actors*
  for each step t **do**
    $s_t \leftarrow$ Observe state
    $a_t \sim \pi_\theta(a_t|s_t)$
    $s_{t+1}, r_t \leftarrow$ Perform action a_t in the environment
    $\triangleright$ *Increment state visitation counter based on the representation*
    $\psi(s_{t+1}) \leftarrow (\text{argmax}_{(i,j)}\, \overline{\alpha}_t(i,j),\, c(s_{t+1}),\, \lfloor \sum_{k=0}^{t} r_t \rfloor)$
    $\#(\psi(s_{t+1})) \leftarrow \#(\psi(s_{t+1})) + 1$
    $r_t^+ \leftarrow \frac{\beta}{\sqrt{\#(\psi(s_{t+1}))}}$
    Store transition $\mathcal{E} \leftarrow \mathcal{E} \cup \left\{(s_t, a_t, \beta_1 \text{clip}(r_t, -1, 1) + \beta_2 r_t^+)\right\}$
  end for
  $\triangleright$ *Perform actor-critic using on-policy samples in $\mathcal{E}$*
  $\theta_{\text{a2c}} \leftarrow \theta_{\text{a2c}} - \eta \nabla_{\theta_{\text{a2c}}} \mathcal{L}^{\text{a2c}}$
  $\triangleright$ *Train inverse dynamic network using on-policy samples in $\mathcal{E}$*
  $\theta_{\text{inv}} \leftarrow \theta_{\text{inv}} - \eta \nabla_{\theta_{\text{inv}}} \mathcal{L}^{\text{inv}}$
  Clear transition buffer $\mathcal{E} \leftarrow \emptyset$
end for

B ARCHITECTURE AND HYPERPARAMETER DETAILS

The architecture details of the inverse dynamics network, A2C policy network, and hyper-parameters are as follows.

Table 3: Network architecture and hyperparameters

Hyperparameters	Value	
A2C policy Architecture	Input: 84x84x1	
	-Conv(32-8x8-4)	/ReLU
	-Conv(64-4x4-2)	/ReLU
	-Conv(64-3x3-1)	/ReLU
	-FC(512)	
Inverse Dynamic Encoder Architecture	Input: 160x160x3	
	-Conv(8-4x4-2)	/LeakyReLU
	-Conv(8-3x3-2)	/LeakyReLU
	-Conv(16-3x3-2)	/LeakyReLU
	-Conv(16-3x3-2)	/LeakyReLU
MLP architecture for $e_t(i,j)$	FC(1296,256)	/ReLU
	-FC(256,128)	/ReLU
	-FC(128,18)	
MLP architecture for $\widetilde{\alpha}_t(i,j)$	FC(1296,64)	/ReLU
	-FC(64,64)	/ReLU
	-FC(64,1)	
λ_{ent} for loss	0.001	
Discount factor γ	0.99	
Learning rate (RMSProp)	0.0007	
Number of parallel environments	16	
Number of roll-out steps per iteration	5	
Entropy regularization of policy (α)	0.01	

Table 4: The list of hyperparameters used in each game.

Games	β_1 in A2C+CoEX	β_2 in A2C+CoEX	β_1 in A2C	τ for clustering
FREEWAY	10	10	10	-
FROSTBITE	10	10	10	-
HERO	1	0.1	1	0.7
MONTEZUMA'S REVENGE	10	10	10	0.7
PRIVATEEYE	10	10	10	0.55
QBERT	1	0.5	1	-
SEAQUEST	1	0.5	10	-
VENTURE	10	10	10	0.7

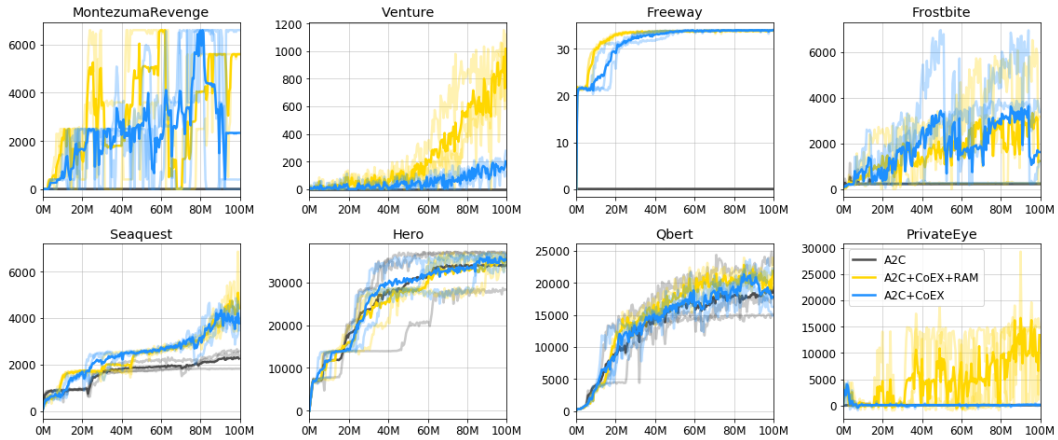

Figure 5: Learning curves on several Atari games: A2C, A2C+CoEX, and A2C+CoEX+RAM.

C EXPERIMENT WITH RAM INFORMATION

In order to understand the performance of exploration with perfect representation, we extract the ground truth location of agent, ground truth room number from RAM, and exploit count-based exploration with the perfect (x, y, c, R). Figure 5 shows the learning curves of the experiments, and we could see A2C+CoEX+RAM acts as an upper bound performance of our proposed method.

D OBSERVATION EMBEDDING CLUSTERING

We describe the detail of a method to obtain the observation embedding. Given an observation of shape $(84, 84, 3)$, we flatten the observation and project it to an embedding of dimension 64. We randomly initialize the parameter of fully-connected layer for projection, and keep the values unchanged during the training to make the embedding stationary.

For the embedding of these observations, we cluster them based on a threshold value τ. The value of τ for each game with change of rooms is listed in Table 4. If the distance between the current embedding and the center $\mathrm{mean}(c)$ of one cluster c is less than the threshold, we assign this embedding into this cluster and update the center as the mean value of all embedding belonging to this cluster. If the distance between the current embedding and the center of any cluster is larger than the threshold, we create a new cluster and this embedding is assigned to this new cluster.

Algorithm 2 Clustering Observation Embedding

Initialize parameter θ_c for context embedding projector if applicable (which is not trainable)
Initialize threshold τ for clustering
Initialize clusters set $C \leftarrow \emptyset$
for each observation s **do**
  $\triangleright$ *Get embedding of the observation from the random projection*
  $v \leftarrow f_{\theta_c}(s)$
  $\triangleright$ *Find a cluster to which the current embedding fits, if any*
  Find a cluster $c \in C$ such that $\|\mathrm{mean}(c) - v\| \leq \tau$, or NIL if there is no such
  if $c \neq$ NIL **then**
    $c \leftarrow c \cup v$
  else
    $\triangleright$ *if there's no existing cluster that v should be assigned in, create a new one*
    $C \leftarrow C \cup \{v\}$
  end if
end for

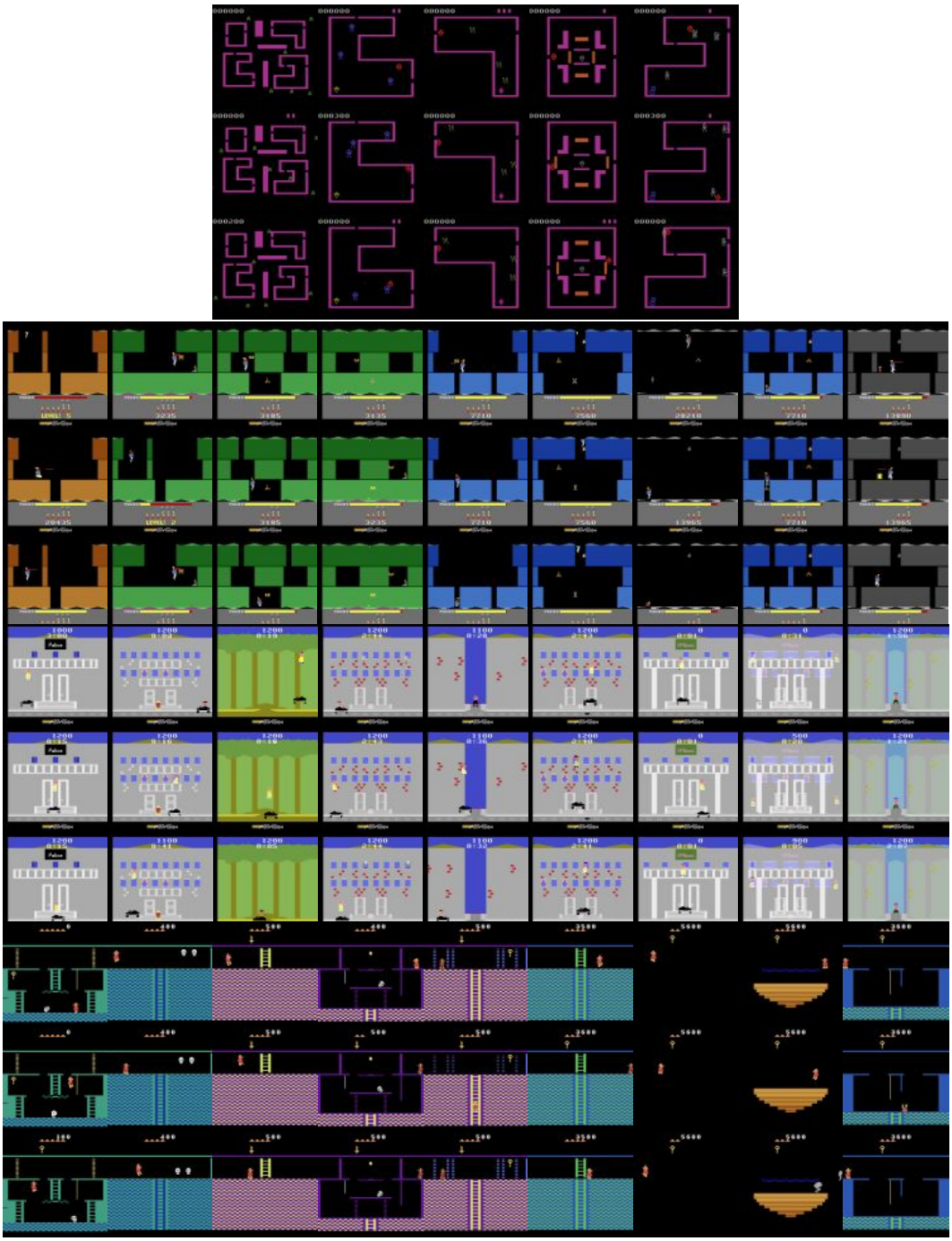

Figure 6: Sample of clustering result for VENTURE, HERO, PRIVATEEYE, and MONTEZUMA'S REVENGE. Each column is one cluster, and we show 3 random samples assigned into this cluster.

In Figure 6 we also show the samples of observation in each cluster, and we could see observations of the same room are assigned to the same cluster, and different clusters corresponds to different rooms.