Survey of Exploration in Reinforcement Learning

Jongkook Kim *jkkim123@gmail.com

1 Introduction

Exploration versus exploitation is a critical topic in reinforcement learning. Although the ultimate goal of RL problems is to maximize the expected reward of the policy, committing to solutions too early on without enough exploration has significant opportunity costs. Modern RL algorithms such as [41], [54], [55] that optimize for the best returns can achieve good exploitation quite efficiently, while exploration remains more like an open topic. Such RL algorithms that proved successful in dense reward environments utilized ϵ-greedy exploration which leaves space only to how much randomness will be allowed, but not to how the randomness will be directed.

[27] was among the first algorithms to apply entropy of the policy as part of the loss function, thus encouraging diverse actions to be considered within a single policy. Furthermore, [40] incorporated Tsallis entropy, a generalized entropy term into the actor-critic framework, thereby allowing the degree of exploration to be manipulated. However, this still could not shed light on how the stochasticity of the policy will unfold as exploration in a directed fashion.

The biggest problem can be elucidated on their performance on hard-exploration problems with sparse reward environments. A concrete example of this would be *montezuma's revenge*, where the player needs to take hundreds of actions(i.e. moving to the next room, going down the ladder and jumping over the monsters) to get the first reward signal. These environments require much "wandering around" to find which line of actions has the highest possibility of a reward before concluding that all actions seem pretty much equal reward-wise. Another problem was evident in the "Noisy-TV" problem first presented in [11]. Imagine that an RL agent is rewarded with seeking novel experience, however a TV with uncontrollable and unpredictable random noise outputs would be able to attract the agent's attention forever. The agent obtains new rewards from noisy TV consistently, but it fails to make any meaningful progress and becomes a "couch potato".

As we will explore throughout this report, intrinsic motivation was one of the biggest clues in the area. Agents formulated their own rewards based on counts over observations, past memories, predictions over the future states, etc. Variational inference (information gain) was also utilized in computing the intrinsic rewards. Moreover, exploration was also studied on meta-reinforcement learning and multi-agent environments.

2 Preliminaries

2.1 Reinforcement Learning Setting

A Markov decision process(MDP) is defined as a tuple $M = S, A, d, P, \gamma, r$, where S is the state space, F is the corresponding feature space, A is the action space, d(s) is the distribution of an initial state, $P(s'|s, a)$ is the transition probability from $s \in S$ by taking $a \in A, \gamma \in (0, 1)$ is a discount factor, and r is the reward function defined $r(s, a, s') \triangleq \mathbb{E}[\mathbf{R}|s, a, s']$ with random reward $\mathbf{R}$.

Then, the MDP problem can be formulated as :

$$max_{\pi \in \Pi} \mathbb{E}_{\pi}[\Sigma_t^{\infty} \gamma^t \mathbf{R}_t], \tag{1}$$

where $\Sigma_t^{\infty} \gamma^t \mathbf{R}_t$ is a discounted sum of rewards, also called a return, $\Pi = \{\pi | \forall s, a \in \mathcal{S} \times \mathcal{A}, \pi(a|s) \geq 0, \Sigma_a \pi(a|s) = 1\}$ is a set of policies, and τ is a sequence of state-action pairs sampled from

*College of Business Administration, Seoul National University

the transition probability and policy, i.e., $s_{t+1} \sim P(\cdot|s_t, a_t), a_t \sim \pi(\cdot|s_t)$ for $t \in [0, \infty]$ and $s_o \sim d$. For a given π, we can define the state value and state-action (or action) value as $V^\pi(s) \triangleq \mathbb{E}_{\tau\sim P,\pi}[\Sigma_t^\infty \gamma^t \mathbf{R}_t | s_o = s]$ and $Q^\pi(s,a) \triangleq \mathbb{E}_{\tau\sim P,\pi}[\Sigma_t^\infty \gamma^t \mathbf{R}_t | s_o = s, a_o = a]$, respectively. The solution of an MDP is called the optimal policy π^*. The optimal value $V^* = V^{\pi^*}$ and the action-value $Q^* = Q^{\pi^*}$ satisfy the Bellman optimality equation as follows: For $\forall s, a$,

$$Q^*(s,a) = \mathbb{E}_{s'\sim P}[\mathbf{r}(s,a,s') + \gamma V^*(s')] \tag{2}$$

$$V^*(s) = max_{a'} Q^*(s,a'), \pi^* \in argmax_{a'} Q^*(s,a') \tag{3}$$

where $argmax_{a'} Q^*(s,a')$ indicates a set of the policy π satisfying $\mathbb{E}_{a\sim\pi}[Q^*(s,a') = max_{a'} Q^*(s,a')$ and $a \sim \pi^*$ indicates $a \sim \pi^\star(\cdot|s)$. Note that there may exist multiple optimal policies if the optimal action value has multiple maximum with respect to actions.

3 Exploration Strategies

3.1 Classic exploration strategies

3.1.1 ϵ-greedy exploration

ϵ-greedy exploration is where the agent randomly takes exploratory "random" actions with probability ϵ and takes the optimal action with $1-\epsilon$ probability. Recent variations of this exploration strategy is [14], where a temporally extended form of ϵ-greedy exploration is proposed. The work shows that temporal persistence with the selected random action is much more time-efficient than vanilla ϵ-greedy exploration and can show results when combined with existing algorithms such as [35] or [28] that surpasses existing intrinsically motivated algorithms on continuous control, tabular environments and Atari Learning Environment(ALE)[8].

3.1.2 Upper confidence bound(UCB) exploration

Since random exploration can give us a bad action which we have confirmed in the past, an alternative approach is to be optimistic about options with high uncertainty and thus to prefer actions for which we haven't had a confident value estimation yet(principle of "optimism in the fact of uncertainty"). Thus The agent selects the greediest action to maximize the upper confidence bound:

$$A_t = \underset{a}{argmax} \left[Q_t(a) + c\sqrt{\frac{logt}{N_t(a)}} \right], \tag{4}$$

where $Q_t(a)$ is the estimated value of action 'a' at time step 't,' $N_t(a)$ is the number of times that action a has been selected, prior to time t, and c is a confidence value that controls the level of exploration. Here exploration is represented by the second part of the equation, and the first part represents the exploitation scheme. $N_t(a)$ will be small for actions that hasn't been taken as often and will lead to large uncertainty measure, leading to higher chance of selecting such action. One of the most recent works that utilizes UCB as part of exploration strategy include UCLS[39], where least-squares TD-learning is combined with exponential averaging over the estimation of upper confidence bound. State-dependent noise variance was added to focus exploration on a subset of variable states while less dwelling on other states.

3.1.3 Noise-based exploration

Adding noise to observation, action or even parameter space was experimented and produced noticeable results. [20] added stochasticity to the agent's policy parameters and replaced entropy reward and ϵ-greedy exploration heuristics in A3D and DQN to achieve faster learning attributable to efficient exploration. [49] added Gaussian noise to agent's policy parameter vector(both on- and off-policy) with adaptive noise scaling:

$$\sigma_{k+1} = \begin{cases} \alpha\sigma_k & \text{if } d(\pi, \hat{\pi}) \leq \delta, \\ \frac{1}{\alpha}\sigma_k & \text{otherwise}, \end{cases} \tag{5}$$

where $\alpha \in \mathcal{R}_{>0}$ is a scaling factor and $\delta \in \mathcal{R}_{>0}$ a threshold value. The concrete realization of $d(\cdot,\cdot)$ depends on the algorithm.

3.2 Intrinsic rewards

Intrinsic motivation which was first shed light on from psychological viewpoint [46] has provided a great insight into how an agent could learn in a sparse reward environment. Thus the policy is trained with a reward composed of two terms, $r_t = r_t^e + \beta r_t^i$ where β is a hyperparameter adjusting the balance between exploitation and exploration. One pioneering work on intrinsic motivation was [34] which proposed the idea of empowerment of the agent as intrinsic reward utilizing Shannon channel capacity.

3.2.1 Count-based rewards

Density estimation models Using density models for the count, [9] [45] proposed counting the number of states the agent has visited and using it as additional intrinsic motivation. The less it has visited in the past, the more chance the agent has to take action towards such state. [9] introduced two concept: pseudo-count function $\hat{N}_n(s)$ and pseudo-count total $\hat{n}$. Combined, they were designed to imitate an empirical count function:

$$\rho_n(s) = \frac{\hat{N}_n(s)}{\hat{n}} \leq \rho'_n(s) = \frac{\hat{N}_n(s) + 1}{\hat{n} + 1} \tag{6}$$

Thus the pseudo-count can be computed as:

$$\hat{N}_n(s) = \hat{n}\rho_n(s) = \frac{\rho_n(s)(1 - \rho'_n(s))}{\rho'_n(s) - \rho_n(s)} \tag{7}$$

where $\rho'_n(s) = \rho_{n+1}(s)$ due to online training.

Using raw image input Following works on count-based intrinsic reward [45] used PixelCNN[60] as density models. Hashing after counting[59] was proposed which used SimHash[13], a type of locality-sensitive hashing that measures similarity by angular distance for low-dimensional state-spaces and used auto-encoders for high-dimensional raw pixel inputs. [21] trains a classifier to discriminate states against each other which corresponds to density estimation and further uses it to compute count-based internal rewards.

Recent works which utilizes density estimation not directly using count-based reward include [65], where density estimation is computed using Variational Gaussian Mixture Model(V-GMM) to prioritize trajectories in the replay buffer.

3.2.2 Prediction-based rewards

Prediction of the environment dynamics could provide a measure of agent's knowledge of the environment, so improvements of the agent's knowledge was another proxy for intrinsic reward. First proposed in [53], the idea of curiosity(predictability) was widely used to generate intrinsic motivation. Intelligent adaptive curiosity(IAC)[46] first outlined the idea of using a forward dynamics prediction model for intrinsic exploration. IAC constructed the intrinsic reward so that the agent would pick an action that would most decrease the prediction error rate of the dynamics predictor and learn quickly about the environment.

Utilizing neural networks as forward dynamics Utilizing deep predictive models, [57] trained a forward dynamics model in the encoding space defined by $\phi, f_\phi : (\phi(s_t), a_t) \rightarrow \phi(s_{t+1})$ and computed intrinsic reward in the new space using model's prediction error as $r_t^i = (\frac{\bar{e}_t(s_t, a_t)}{t \cdot C})$ where $\bar{e}_t = \frac{e_t}{max_{i \leq t} e_i}$ is the model's prediction at time t and $C > 0$ is a decay constant. The encoding function was learned via an auto-encoder using ϵ-greedy explored data collection. Given a forward model f, an inverse dynamics model g and an observation s_t, a_t, s_{t+1}:

$$g_{\psi_I}(\phi(s_t), \phi(s_{t+1})) = \hat{a}_t \tag{8}$$

$$f_{\psi_F}(\phi(s_t), a_t) = \hat{\phi}(s_{t+1}) \tag{9}$$

$$r_t^i = \|\hat{\phi}(s_{t+1}) - \phi(s_{t+1})\|_2^2 \tag{10}$$

Intrinsic Curiosity Module(ICM) In an attempt to replace auto-encoders, [47] proposed Intrinsic Curiosity Module(ICM) which learns the state space encoding with a self-supervised inverse dynamics model $g : (\phi(s_t), \phi(s_{t+1})) \rightarrow a_t$. The feature space only captures those changes in the environment related to the actions of the agent, and ignores the rest, based on the paper's proposition that good state feature space should exclude such factors because they cannot influence the agent's behavior and thus the agent has no incentive for learning them. Following the same curiosity-driven approach, [10] compared four encoding functions over various environments with *pure* intrinsic reward at large-scale:

- Raw image pixels : $\phi(x) = x$
- Random features(RF): Each state is compressed through a fixed random neural network.
- VAE: The probabilistic encoder is used for encoding, $\phi(x) = q(z|x)$.
- Inverse dynamic features (IDF) : The same feature space as used in ICM

where the intrinsic reward was computed as $r_t = r_t^i = \|f(s_t, a_t) - \phi(s_{t+1})\|_2^2$. Random features have shown competitive results in some environments, but inverse dynamics feature performed better in feature transfer experiments. Multiple forward dynamics with ICM was used in [48] to use disagreements between the models as intrinsic reward, which was differentiable thus allowing gradient descent through the ensembles.

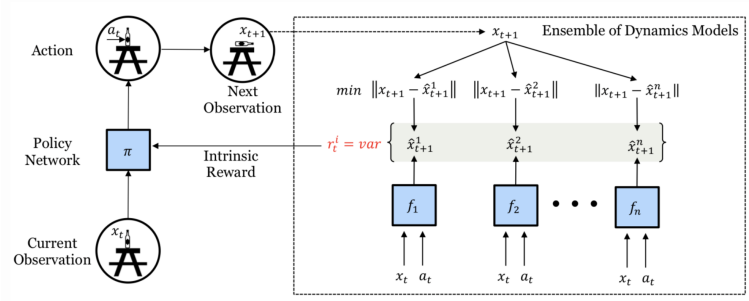

Figure 1: Illustration of training architecture for self-supervised exploration via disagreement. (Image source: Pathak, et al. 2019)

Random Network Distillation(RND) Random network distillation(RND) was used[11] to capture the error between the predicting neural network $\hat{f}(s_t)$ and the fixed randomly initialized neural network $f(s_t)$. If the discrepancy is high it means the state the agent is currently in is novel and needs to be further explored. The exploration bonus is computed as $r^i(s_t) = \|\hat{f}(s_t; \theta) - f(s_t)\|_2^2$. Recent works on prediction-based rewards include RIDE(Rewarding Impact-Driven Exploration) [50] where intrinsic reward which encourages the agent to take actions that lead to significant changes in its learned state representation is applied to the agent. The overall intrinsic reward is calculated as

$$R_{IDE}(s_t, a_t) \equiv r_t^i(s_t, a_t) = \frac{\|\phi(s_{t+1} - \phi(s_t)\|_2}{\sqrt{N_{ep}(s_{t+1})}} \tag{11}$$

where $\phi(s_{t+1})$ and $\phi(s_t)$ are the learned representations of consecutive states, resulting from the agent transitioning to state a_{t+1} after taking action a_t in state s_t. Since the proposed agent will not receive rewards for reaching states that are inherently unpredictable, exploration was made robust with respect to distractor objects or other inconsequential sources of variation in the environment, as was made clear in the Noisy-TV and mini-grid experiments.

3.2.3 Entropy-based rewards

One notable work that utilizes state entropy itself as intrinsic reward was Random Encoders for Efficient Exploration (RE3)[56]. In environments with high-dimensional observation k-nearest neighbor entropy estimator in the low-dimensional representation space of convolutional encoder is utilized. This work is surprising in that although the encoder is randomly initialized *and* fixed throughout training without any representation learning, sample efficiency is achieved especially in sparse reward environments.

3.3 Variational inference-based exploration

3.3.1 Variational Information Maximizing Exploration(VIME)

Variational Information(Information Gain) of the forward dynamics model was maximized in [31] to provide intrinsic motivation. The dynamics model is paremeterized as a Bayesian neural network, as it maintains a distribution over its weights. The BNN weight distribution $q_\phi(\theta)$ is modeled as a fully factorized Gaussian with $\phi = \{\mu, \sigma\}$ and we can easily sample $\theta \sim q_\phi(\cdot)$. KL-Divergence was computed using the Fisher Information Matrix, leading to the intrinsic reward: $r_t^i = D_{KL}[q_{\phi_{t+1}}(\theta) \| q_{\phi_t}(\theta)]$.

3.3.2 Mutual Information based strategies

MINE and Deep-Infomax Mutual Information was also shed light upon as a tool for exploration by [7], [29]. In the context of generative adversarial networks MINE[7] aims at maximizing the approximation of mutual information between the latent code and the raw data. [29] builds on the idea and trains a decoder-free encoding representation maximizing the mutual information between the input image and the representation. Furthermore, the method uses f-divergence information for better numerical stability.

MUSIC In Mutual Information State Intrinsic Control(MUSIC)[64], MI quantity $I(S^s; S^a)$ was approximated using lower bound in the Donsker-Varadhan representation with the compression lemma in the PAC-Bayes literature. Interesting point here is that mutual information was not only used as an intrinsic motivation but was also applied as prioritization scheme as in [65] to attain similar results. In [37], mutual information was computed to reduce uncertainty of embedding representation in between consecutive states and state-action pair to find a pair of optimal embedding functions $\phi(s)$ and $\psi(a)$. This work focused on how to learn an effective linear dynamics, i.e. $\phi(s') = \phi(s) + \psi(a)$ and leaving the nonlinear aspects of the dynamics onto the neural networks. Embedding functions were then used to construct intrinsic reward which showed exceeding performance in some Atari-games and continuous control tasks. [15] proposed feature control as intrinsic motivation and shows state-of-the-art results in *montezuma's revenge*.

Drop-Bottleneck Notable recent work includes Drop-Bottleneck[38] where the input information is compressed by discretely dropping a subset of input features. Using Deep-Infomax[29] discriminator and the notion of episodic memory[51], the Drop-Bottleneck is trained as to encourage compression by dropping unnecessary features, but still be able to predict the next state compressed representation each using a mutual information term in the objective loss. This strategy was particularly effective in noisy environments such as VizDoom[36] and DM-Lab[6].

3.4 Memory-based exploration

Never Give Up and Agent57 The idea of intrinsic reward was further extended by Never Give Up(NGU)[3] and Agent57[2] where the former structured the intrinsic reward in two levels: episodic reward utilized random network distillation as lifelong novelty reward and inverse dynamics feature to compute episodic reward as shown in figure 2. The total intrinsic reward is expressed as $r_t^i = r_t^{episodic} \cdot min\{max\{\alpha_t, 1\}, L\}$, where L is a chosen maximum reward scaling and α_t a life-long curiosity factor that measures novelty of the action across multiple episodes. Using UVFA [52] framework, the proposed algorithm learns a family of policies each varying with the degree of exploration as opposed to exploitation. Using the intrinsic reward, *augmented reward* is further computed with a hyperparamter β: $r_t = r_t^e + \beta r_t^i$. [3] also utilized used LSTM[30] in the form of Recurrent Experience Replay[35] to store episodic memories. NGU was improved by [2], the first deep RL agent that outperforms the standard human benchmark on all 57 Atari games. Two major improvements in Agent57 over NGU are:

- A population of policies are trained in Agent57, each equipped with a different exploration parameter pair $\{(\beta_j, \gamma_j)\}_{j=1}^N$. Recall that given β_j, the reward is constructed as $r_{j,t} = r_t^e + \beta r_t^i$ and γ_j is the reward discounting factor. If is natural to expect policies with higher β_j and lower γ_j to make more progress early in training, while the while the opposite would be expected as training progresses. A meta-controller (sliding-window UCB bandit algorithm[22]) is trained to select which policies should be prioritized.

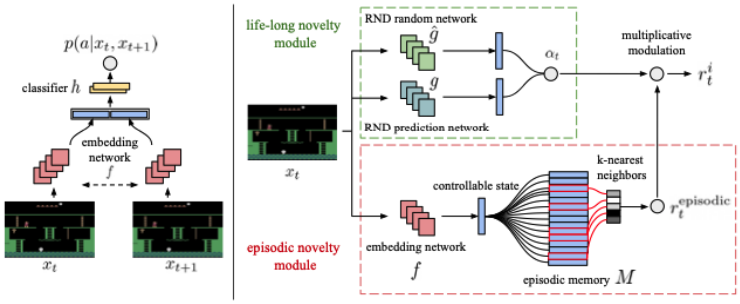

Figure 2: (left) Training architecture for the embedding network (right) NGU's reward generator. (Image source: Badia, et al. 2020)

- The second improvement is a new parameterization of Q-value function that decomposes the contributions of the intrinsic and extrinsic rewards in a similar form as the combined reward: $Q(s, a; \theta_j = Q(s, a; \theta_j^e + \beta_j Q(s, a; \theta_j^i$. During training, $Q(s, a; \theta_j^e)$ and $Q(s, a; \theta_j^i$ are optimized separately with rewards $r_j^e$and r_j^i, respectively.

Episodic Curiosity In order to measure the closeness of states in episodic memory, [51] took the transition between states into consideration rather than using Euclidean distance between states. The proposed method measure the number of steps needed to visit one state from other states in memory. If the number of steps is bigger than a threshold, the state is considered novel and is rewarded by bonus. Siamese network containing one embedding network $\phi : \mathbf{S} \to \mathbb{R}^n$ and a comparator network $\mathbf{C} : \mathbb{R}^n \times \mathbb{R}^n \to [0, 1]$ to output a binary label on whether two states are close enough in the transitional graph $C(\phi(s_i), \phi(s_j)) \to [0, 1]$.

First of the two notable recent works, PolyRL[1] incorporates the idea of local history into finite (short-term) persistence of the agent's behavior. Introduced as *locally self-avoiding random walks(LSA-RWs)*, the stiffness is structured via computing radius of gyration as the measure of spread in the states. The persistence which resembles that of [14], achieved comparable results in in 2D Navigation tasks and Mujoco locomotion experiments and outperformed in several environments with sparse or delayed reward structures. Second work is RAPID[63] which proposed an episode-level exploration method for procedurally-generated environments. RAPID regards each episode as a whole and gives an episodic exploration score from both per-episode(local) and long-term(global) views where those highly scored episodes are stored in a small ranking buffer and used as imitation learning data. Although it is very much similar to the episodic and life-long reward concept of NGU, this one used episodic reward as a prioritization score for imitation learning.

3.5 Option/Skill discovery methods

Options Options are policies with termination conditions. There are a large set of options available in the search space and they are independent of an agent's intentions. By explicitly including intrinsic options into modeling, the agent can obtain intrinsic rewards for exploration. Variational intrinsic control[23] is such a framework for providing the agent with intrinsic exploration bonuses based on modeling options and learning policies conditioned on options.

Deep covering options[32] builds upon covering options [33] which could not be easily combined with modern representation learning techniques and succeeds in discovering a small set of options that encourage exploration by minimizing the agent's expected cover time—the expected number of steps required to visit every state in the environment.

Skill Discovery Unsupervised learning provides a fresh view on learning skills and using goal-conditioned learning. Skill discovery can be thought of as an *undirected* exploration where the task is not given. Various skills found along the process could later be matched with the appropriate task. DIAYN[18] was successful at discovering skills that are task-agnostic through sampling a latent skill variable $z \sim p(z)$ and training a separate discriminator $q_\phi(z|s_t)$ to maximize the diversity of the skills discovered. The pretrained skills were then later composed hierarchically to perform specific

given tasks. DISCERN[62] learns a MI objective between states and goals to discover new skills. MUSIC[64] was combined with DISCERN to discover skills that maximized mutual information between the surrounding state and the agent state, and mutual information between the state overall and the goal combined. Explore, discover and learn[12] tests the limitation of existing option-based exploration method regarding poor coverage of the state space. While most assumptions and settings are similar to DIAYN(i.e. latent skill variable $z \sim p(z)$), EDL relies on a fixed distribution over states $p(s)$ and makes use of variational inference (VI) techniques to model $p(s|z)$ and $p(z|s)$.

Deep skill graphs One interesting latest works that directly uses skills as a direct method of exploration is deep skill graphs[4], developed over skill chaining[5]. The algorithm seamlessly interleaves discovering skills and planning using them to gain unsupervised mastery over ever increasing portions of the state-space. Taking advantage of deep covering options[33] mentioned above, *salient* events are sequentially generated and is distinguished using a binary classifier. The constructed skill graph is constantly extended with focus on these *salient* events while exploring as shown in Figure 3.

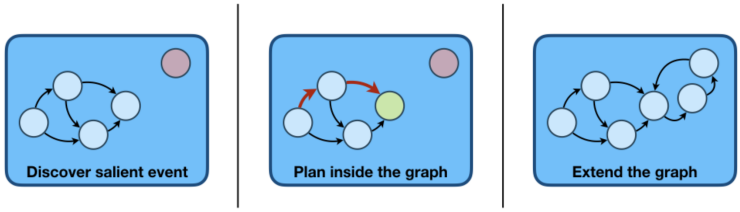

Figure 3: When the discovered salient event (red) is outside the graph, the agent uses planning inside the graph to reach the node closest to its goal (green). It then expands the graph by constructing a series of skills that connect the salient event to the graph. (Image source: Bagaria, et al. 2020)

3.6 Q-value exploration

Bootstrapped DQN[44] modifies DQN to approximate a distribution over Q-values via the bootstrap. A classic DQN is altered into multi-headed DQN but with bootstrapping method to measure uncertainty of q-value for each action and choose the one with the highest uncertainty. Multiple Q-value heads are trained in parallel but each only consumes a bootstrapped sub-sampled set of data and each has its own corresponding target network, although the results leaned towards complete sharing of data over all heads. Deep exploration is well structured via MDP chain experiments followed by the ALE. Although not dominant in every games, Bootstrapped DQN proved to be a scalable exploration scheme. However, this kind of exploration is still restricted, because uncertainty introduced by bootstrapping fully relies on the training data. Following work[43] utilizes randomized prior function to inject some prior information independent of the data. Bootstrapped DQN using randomized prior function is described as analogous to randomized least-squares value iteration and aids to propagate a temporally consistent sample of Q-value.

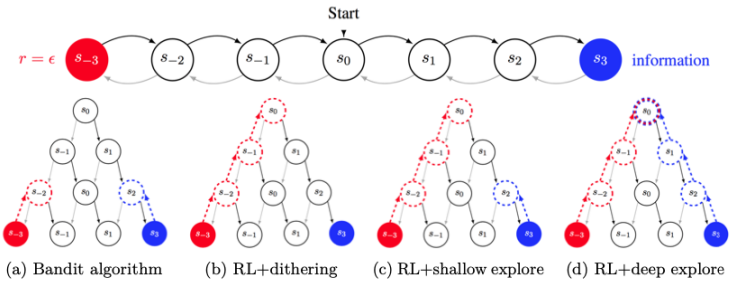

Figure 4: MDP chain experimental setup for deep exploration. The actual experiment was done using extended version of the chain (Image source: Osband, et al. 2016)

3.7 Direct exploration

Direct exploration was first proposed in Go-Explore[16] to solve the *hard-exploration* problems. The method involved exploring until a task is solved but required going back to past states that was stored in its memory as "novel." Then those trajectories were used for imitation learning for "robustification." Calling for a resettable simulator, the algorithm was unrealistic. An improved version, policy-based Go-Explore [17] learned a goal-conditioned policy and uses that to access a known state in memory repeatedly. The goal-conditioned policy is trained to follow the best trajectory that previously led to the selected states in memory. They include a Self-Imitation Learning[42] loss to help extract as much information as possible from successful trajectories. DTSIL[24] and directed exploration[25] also used goal-conditioned policies but with different definition of goals.

3.8 Multi-agent environment exploration

As multi-agent reinforcement learning gained much light, exploration that was well directed and coordinated among agents was studied. Measuring influence among agents was structured in [61] with two specific methods: exploration via information-theoretic influence (EITI) and exploration via decision-theoretic influence (EDTI). EDTI uses a novel intrinsic reward called Value of Interaction(VoI) to disentangle transition and reward influences of one agent's behavior on expected returns of other agents by disentangling both. The EITI reward is an intrinsic motivation that encourages agent 1 to visit more frequently the state-action pairs where it can influence the trajectory of agent 2. The paper furthermore proposed augmented policy gradient formulations which allows to draw a connection between coordinated exploration and the distribution of individual intrinsic rewards among team members. By optimizing EITI or EDTI objective as a regularizer, agents are encouraged to coordinate their exploration and learn policies to optimize the team performance.

3.9 Exploration for Meta-RL

Meta-learning, or learning to learn, refers to the problem of learning strategies for fast adaptation by using prior tasks. Model agnostic exploration with structure noise(MAESN) [26] uses prior experience both to initialize a policy and to learn a latent exploration space from which it can sample temporally coherent structured behaviors. This proposed method allows producing exploration strategies that are stochastic, but still informed by prior knowledge, and thus more effective than random noise. Importantly, per-task latent variable distribution is explicitly trained as Gaussian $\mathcal{N}(\mu_i, \sigma_i)$ and the update is similar to those of MAML[19].

HyperX[66] is one of the most interesting meta-learning exploration strategy. The work focuses on what the agent could learn during meta-training and explores the idea of *meta-exploration*. Meta-exploration refers to the challenge of exploring across tasks and adaptation behaviours during meta-training. The agent has to (a) explore across individual tasks since the same state can have different values across tasks, and (b) learn about the shared structure between tasks to extract information about how to adapt. Thus the algorithm concentrates more on efficiently conducting exploration on the hyper-state space using RND[11] so as to gather more data to find the most Bayes-optimal task-exploration strategy.

3.10 Pessimism on bonus-based exploration schemes

Most of the recent works on exploration have focused on hard-exploration problems, the most representative being the *montezuma's revenge* from ALE[8]. There was a *inspritiing* paper[58] pointing out the trade-off in performance over other easy-exploration games that is prevalent in most exploration schemes that have performed well on *montezuma's revenge*. This was quite evident in NGU[3] and Agent57[2]. The paper carefully compares the results of rainbow[28] combined with ϵ-greedy exploration with the results of 1) count-based exploration[9],[45], 2) curiosity driven exploration[47], 3) random network distillation and 4) NoisyNet [20]. The paper claims that recent gains in *montezuma's revenge* may be better attributed to architecture change, rather than better exploration schemes; and that the real pace of progress in exploration research for Atari 2600 games may have been obfuscated by good results on a single domain.

4 Conclusion

This concludes the paper's journey on exploration strategies in RL. Classic ϵ-greedy exploration still seems to perform comparatively to more advanced exploration schemes, except on hard-exploration environments. UCB and noise-based explorations still add useful insights to estimation of uncertainty and sampling. Intrinsic rewards was started out with count-based strategies, but more engagement with the transition model and the environment were added, as shown in ICM and RND. Variational information allowed for various uses of neural network as function approximators. Memory-based intrinsic rewards, although requiring large compute and memory, achieved post-human performance on all 57 Atari games with pixel inputs. Skill discovery were also another path to finding new novel states. Exploration was also useful in meta-learning and multi-agent RL. Last but not least, there were concerns raised on the utility of exploration schemes that had to sacrifice performance in certain easy tasks for fewer harder ones. These trade-offs were quite evident from these papers to me, but the fact that careful experiment proved ϵ-greedy experiments as competitive as other bonus-based exploration was quite refreshing. Although originally sparked from biological inspirations, exploration is now by far an exploding area of research within RL, and will continue to be the leading driver in the advances in RL.

References

[1] Susan Amin, M. Gomrokchi, Hossein Aboutalebi, Harsh Satija, and Doina Precup. Locally persistent exploration in continuous control tasks with sparse rewards. *ArXiv*, abs/2012.13658, 2020.

[2] Adrià Puigdomènech Badia, Bilal Piot, Steven Kapturowski, P. Sprechmann, Alex Vitvitskyi, Daniel Guo, and C. Blundell. Agent57: Outperforming the atari human benchmark. *ArXiv*, abs/2003.13350, 2020.

[3] Adrià Puigdomènech Badia, P. Sprechmann, Alex Vitvitskyi, Daniel Guo, Bilal Piot, Steven Kapturowski, O. Tieleman, Martín Arjovsky, A. Pritzel, Andew Bolt, and C. Blundell. Never give up: Learning directed exploration strategies. *ArXiv*, abs/2002.06038, 2020.

[4] Akhil Bagaria, J. Crowley, Jing Wei, N. Lim, and G. Konidaris. Skill discovery for exploration and planning using deep skill graphs. 2020.

[5] Akhil Bagaria and G. Konidaris. Option discovery using deep skill chaining. In *ICLR*, 2020.

[6] Charlie Beattie, Joel Z. Leibo, Denis Teplyashin, Tom Ward, M. Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, A. Sadik, Julian Schrittwieser, Keith Anderson, Sarah York, Max Cant, Adam Cain, A. Bolton, Stephen Gaffney, Helen King, D. Hassabis, S. Legg, and Stig Petersen. Deepmind lab. *ArXiv*, abs/1612.03801, 2016.

[7] Ishmael Belghazi, Sai Rajeswar, Aristide Baratin, R. Devon Hjelm, and Aaron C. Courville. MINE: mutual information neural estimation. *CoRR*, abs/1801.04062, 2018.

[8] Marc G. Bellemare, Yavar Naddaf, J. Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents (extended abstract). In *IJCAI*, 2015.

[9] Marc G. Bellemare, S. Srinivasan, Georg Ostrovski, Tom Schaul, D. Saxton, and R. Munos. Unifying count-based exploration and intrinsic motivation. In *NIPS*, 2016.

[10] Yuri Burda, Harrison Edwards, Deepak Pathak, A. Storkey, Trevor Darrell, and Alexei A. Efros. Large-scale study of curiosity-driven learning. *ArXiv*, abs/1808.04355, 2019.

[11] Yuri Burda, Harrison Edwards, A. Storkey, and Oleg Klimov. Exploration by random network distillation. *ArXiv*, abs/1810.12894, 2019.

[12] Víctor Campos, Alexander Trott, Caiming Xiong, R. Socher, Xavier Giro i Nieto, and Jordi Torres. Explore, discover and learn: Unsupervised discovery of state-covering skills. In *ICML*, 2020.

[13] M. Charikar. Similarity estimation techniques from rounding algorithms. In *STOC '02*, 2002.

[14] Will Dabney, Georg Ostrovski, and André Barreto. Temporally-extended e-greedy exploration. *arXiv: Learning*, 2020.

[15] Nat Dilokthanakul, Christos Kaplanis, Nick Pawlowski, and M. Shanahan. Feature control as intrinsic motivation for hierarchical reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 30:3409–3418, 2019.

[16] Adrien Ecoffet, J. Huizinga, J. Lehman, Kenneth O. Stanley, and J. Clune. Go-explore: a new approach for hard-exploration problems. *ArXiv*, abs/1901.10995, 2019.

[17] Adrien Ecoffet, J. Huizinga, J. Lehman, Kenneth O. Stanley, and J. Clune. First return then explore. *Nature*, 590 7847:580–586, 2021.

[18] Benjamin Eysenbach, A. Gupta, J. Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *ArXiv*, abs/1802.06070, 2019.

[19] Chelsea Finn, P. Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017.

[20] Meire Fortunato, M. G. Azar, Bilal Piot, Jacob Menick, Ian Osband, A. Graves, Vlad Mnih, R. Munos, D. Hassabis, O. Pietquin, C. Blundell, and S. Legg. Noisy networks for exploration. *ArXiv*, abs/1706.10295, 2018.

[21] Justin Fu, John D. Co-Reyes, and Sergey Levine. Ex2: Exploration with exemplar models for deep reinforcement learning. In *NIPS*, 2017.

[22] A. Garivier and É. Moulines. On upper-confidence bound policies for non-stationary bandit problems. *arXiv: Statistics Theory*, 2008.

[23] K. Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. *ArXiv*, abs/1611.07507, 2017.

[24] Yijie Guo, Jongwook Choi, M. Moczulski, Shengyu Feng, S. Bengio, Mohammad Norouzi, and Honglak Lee. Memory based trajectory-conditioned policies for learning from sparse rewards. In *NeurIPS*, 2020.

[25] Z. Guo and Emma Brunskill. Directed exploration for reinforcement learning. *ArXiv*, abs/1906.07805, 2019.

[26] A. Gupta, R. Mendonca, Yuxuan Liu, P. Abbeel, and Sergey Levine. Meta-reinforcement learning of structured exploration strategies. In *NeurIPS*, 2018.

[27] Tuomas Haarnoja, Aurick Zhou, P. Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *ICML*, 2018.

[28] Matteo Hessel, Joseph Modayil, H. V. Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, M. G. Azar, and D. Silver. Rainbow: Combining improvements in deep reinforcement learning. In *AAAI*, 2018.

[29] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization, 2019.

[30] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997.

[31] Rein Houthooft, Xi Chen, Yan Duan, John Schulman, F. Turck, and P. Abbeel. Vime: Variational information maximizing exploration. In *NIPS*, 2016.

[32] Yuu Jinnai, J. W. Park, David Abel, and G. Konidaris. Discovering options for exploration by minimizing cover time. In *ICML*, 2019.

[33] Yuu Jinnai, J. W. Park, Marlos C. Machado, and G. Konidaris. Exploration in reinforcement learning with deep covering options. In *ICLR*, 2020.

[34] T. Jung, D. Polani, and P. Stone. Empowerment for continuous agent—environment systems. *Adaptive Behavior*, 19:16 – 39, 2011.

[35] Steven Kapturowski, Georg Ostrovski, John Quan, R. Munos, and Will Dabney. Recurrent experience replay in distributed reinforcement learning. In *ICLR*, 2019.

[36] Michal Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. Vizdoom: A doom-based ai research platform for visual reinforcement learning. *2016 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 1–8, 2016.

[37] HyoungSeok Kim, Jaekyeom Kim, Yeonwoo Jeong, Sergey Levine, and Hyun Oh Song. Emi: Exploration with mutual information. In *ICML*, 2019.

[38] Jaekyeom Kim, Minjung Kim, Dongyeon Woo, and Gunhee Kim. Drop-bottleneck: Learning discrete compressed representation for noise-robust exploration. *ArXiv*, abs/2103.12300, 2021.

[39] Raksha Kumaraswamy, M. Schlegel, Adam White, and Martha White. Context-dependent upper-confidence bounds for directed exploration. *ArXiv*, abs/1811.06629, 2018.

[40] Kyungjae Lee, Sungyub Kim, Sungbin Lim, Sungjoon Choi, and Songhwai Oh. Tsallis reinforcement learning: A unified framework for maximum entropy reinforcement learning. *ArXiv*, abs/1902.00137, 2019.

[41] T. Lillicrap, Jonathan J. Hunt, A. Pritzel, N. Heess, T. Erez, Yuval Tassa, D. Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *CoRR*, abs/1509.02971, 2016.

[42] Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. Self-imitation learning. In *ICML*, 2018.

[43] Ian Osband, J. Aslanides, and Albin Cassirer. Randomized prior functions for deep reinforcement learning. In *NeurIPS*, 2018.

[44] Ian Osband, C. Blundell, A. Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. In *NIPS*, 2016.

[45] Georg Ostrovski, Marc G. Bellemare, Aäron van den Oord, and R. Munos. Count-based exploration with neural density models. *ArXiv*, abs/1703.01310, 2017.

[46] Pierre-Yves Oudeyer, F. Kaplan, and V. Hafner. Intrinsic motivation systems for autonomous mental development. *IEEE Transactions on Evolutionary Computation*, 11:265–286, 2007.

[47] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 488–489, 2017.

[48] Deepak Pathak, Dhiraj Gandhi, and A. Gupta. Self-supervised exploration via disagreement. *ArXiv*, abs/1906.04161, 2019.

[49] Matthias Plappert, Rein Houthooft, Prafulla Dhariwal, S. Sidor, Richard Y. Chen, Xi Chen, T. Asfour, P. Abbeel, and Marcin Andrychowicz. Parameter space noise for exploration. *ArXiv*, abs/1706.01905, 2018.

[50] Roberta Raileanu and Tim Rocktäschel. Ride: Rewarding impact-driven exploration for procedurally-generated environments. *ArXiv*, abs/2002.12292, 2020.

[51] Nikolay Savinov, Anton Raichuk, Raphaël Marinier, Damien Vincent, M. Pollefeys, T. Lillicrap, and S. Gelly. Episodic curiosity through reachability. *ArXiv*, abs/1810.02274, 2019.

[52] Tom Schaul, Dan Horgan, K. Gregor, and D. Silver. Universal value function approximators. In *ICML*, 2015.

[53] J. Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. 1991.

[54] John Schulman, Sergey Levine, P. Abbeel, Michael I. Jordan, and P. Moritz. Trust region policy optimization. *ArXiv*, abs/1502.05477, 2015.

[55] John Schulman, F. Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.

[56] Younggyo Seo, Lili Chen, Jinwoo Shin, Honglak Lee, P. Abbeel, and Kimin Lee. State entropy maximization with random encoders for efficient exploration. *ArXiv*, abs/2102.09430, 2021.

[57] Bradly C. Stadie, Sergey Levine, and P. Abbeel. Incentivizing exploration in reinforcement learning with deep predictive models. *ArXiv*, abs/1507.00814, 2015.

[58] Adrien Ali Taiga, W. Fedus, Marlos C. Machado, Aaron C. Courville, and Marc G. Bellemare. On bonus based exploration methods in the arcade learning environment. In *ICLR*, 2020.

[59] Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, Xi Chen, Yan Duan, John Schulman, F. Turck, and P. Abbeel. #exploration: A study of count-based exploration for deep reinforcement learning. In *NIPS*, 2017.

[60] Aäron van den Oord, Nal Kalchbrenner, Lasse Espeholt, K. Kavukcuoglu, Oriol Vinyals, and A. Graves. Conditional image generation with pixelcnn decoders. In *NIPS*, 2016.

[61] Tonghan Wang, Jianhao Wang, Yi Wu, and Chongjie Zhang. Influence-based multi-agent exploration. *ArXiv*, abs/1910.05512, 2020.

[62] David Warde-Farley, T. Wiele, Tejas D. Kulkarni, Catalin Ionescu, S. Hansen, and Volodymyr Mnih. Unsupervised control through non-parametric discriminative rewards. *ArXiv*, abs/1811.11359, 2019.

[63] D. Zha, Wenye Ma, L. Yuan, Xia Hu, and Ji Liu. Rank the episodes: A simple approach for exploration in procedurally-generated environments. *ArXiv*, abs/2101.08152, 2021.

[64] Rui Zhao, Yang Gao, P. Abbeel, Volker Tresp, and W. Xu. Mutual information state intrinsic control. *ArXiv*, abs/2103.08107, 2021.

[65] Rui Zhao and Volker Tresp. Curiosity-driven experience prioritization via density estimation. *ArXiv*, abs/1902.08039, 2019.

[66] Luisa M. Zintgraf, L. Feng, Maximilian Igl, Kristian Hartikainen, Katja Hofmann, and S. Whiteson. Exploration in approximate hyper-state space for meta reinforcement learning. *ArXiv*, abs/2010.01062, 2020.