LEARNING TO CONTROL VISUAL ABSTRACTIONS FOR STRUCTURED EXPLORATION IN DEEP REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Exploration in environments with sparse rewards, even in simple environments, is a key challenge. How do we design agents with generic inductive biases so that they can temporally explore instead of just location exploration schemes? We propose an unsupervised reinforcement learning agent which simultaneously learns a discrete pixel abstractions model that preserves spatial geometry of the environment, derives geometric intrinsic reward functions from such abstractions to induce a basis set of behaviors (options) trained with off-policy learning, and finally learns to compose and explore in this options space to optimize for extrinsically defined tasks. We propose an agent that learns a structured exploration algorithm end-to-end using discrete visual abstractions model from raw pixels. We show that our approach can scale to a variety of domains with competitive performance, including navigation in 3D environments and Atari games with sparse rewards.

1 INTRODUCTION

Exploration in environments with sparse feedback is a key challenge for deep reinforcement learning (DRL) research. In DRL, agents typically explore with local exploration strategies like epsilon greedy or entropy based schemes. We are interested in learning structured exploration algorithms, grounded in spatio-temporal visual abstractions given raw pixels. We achieve this via novel generic inductive biases, that if optimized with the proposed loss function, gives rise to structured visual abstractions or entities. Through such abstract representations, behaviors can be derived and composed to output final task related behaviors.

What are good inductive biases to yield structured representations and exploration? In human perception and its developmental trajectory (Spelke & Kinzler, 2007), spatio-temporal pixel groupings is one of the first visual abstractions to emerge. Our key result is to develop a neural network architecture and loss to enforce such intermediate representations and then ground behavior exploration in them via a compositional options model.

Due to the discrete nature of such visual entities, they are tolerant to small optical variations in the image, track pixels along the temporal dimension in groups, and are naturally interpretable by our visual systems. We learn a vector quantized encoder (VQ) model with 1 of E discrete components on a spatial grid, inspired from the VQ-VAE model (van den Oord et al., 2017) but with significant differences. We enforce a spatial VQ layer to preserve the affine geometry of the environment in its internal states. A non-parametric discriminative loss maximizes the lower bound of the mutual information between VQ and image patches. We sample pairs of VQ and image patches with positive and negative labels. The positive pair consists corresponding patches and the negative pair consists of misaligned patches that are either sampled from the same image or different (batch or time dimension). In effect, the discrete patch representations are forced via optimization to represent information to disambiguate the corresponding image patch from the rest. As shown in section 4, this model is surprisingly effective at locally tracking intuitive chunks of visual elements like walls, objects, ground, background and so on.

Given such structured representations, how can we use them to explore better? Our agent architecture automatically derives temporal abstractions of behaviors from these underlying visual entities. Options models have been studied extensively in (deep) reinforcement learning research (Sutton

et al., 1999; Kulkarni et al., 2016). However, prior work is limited in its ability to need hand-crafted knowledge in expressing options or option discovery schemes have not utilized grounded spatio-temporal abstractions. We propose an agent architecture which induces a set of options grounded in discrete visual abstractions.

This options bank consists of a matrix of Q functions for each visual entity segment and its corresponding geometric measurement (segment area and centroid positions in the image plane). A hierarchical meta Q function acts by picking one of the subpolicies and committing to it for a fixed number of environment steps. The subpolicies it can select are either: (1) A behavior from the options bank by following actions outputted by its corresponding Q-epsilon greedy policy or (2) a Q_{task} function trained to predict expected discounted extrinsic reward. Early on in training, the options bank typically serves as a good directed exploration strategy, whose experienced is then subsumed by the Q_{task} function later on in training, to solve the extrinsic task via off-policy learning through a shared replay buffer.

We demonstrate that our approach can scale to two diverse domains – navigation in 3D environments and Atari games – just given raw pixels. Although much work remains in improving the visual and temporal abstraction discovery models, our results indicate that it is possible to learn bottom-up structured exploration schemes with simple spatial inductive biases and loss functions.

2 Related Work

Learning visual abstractions has a long history in computer vision with some of the earlier successes relying on clustering either for inference Shi & Malik (2000) or learning Ren & Malik (2003). More recently some of these intuitions were adapted to neural networks as in Xia & Kulis (2017). Instance segmentation algorithms have been developed to output spatio-temporal groupings of pixels from raw videos (Romera-Paredes & Torr, 2016). However, most of the existing deep learning based approaches require supervised data. Structured deep generative models (van den Oord et al., 2017) is another approach to learning disentangled representations from raw videos. Very recently segmentation has been cast as a mutual information maximization problem in Ji et al. (2018) where the mutual information is computed between the original image segmentation and the output of a transformed one. That approach uses the discrete mutual information estimate which makes it only applicable to enforce pixel-label constraints. For continuous variables recent papers have proposed very promising techniques. Most relevant to our work are van den Oord et al. (2018) and Belghazi et al. (2018).

In reinforcement learning research, semi-MDPs and the options framework (Sutton et al., 1999) have been proposed as a model for temporal abstractions of behaviors. Our work is most similar to hierarchical-DQN (Kulkarni et al., 2016). However, this approach required hand-crafted instance segmentation and the agent architecture is not distributed to learn about many intrinsic rewards learners at the same time. Object-Oriented-MDPs (Diuk et al., 2008) uses object oriented representations for structured exploration but requires prebuilt symbolic representations. A recent paper also demonstrates the important of object based exploration when humans learn to play video games Dubey et al. (2018). HRA (Van Seijen et al., 2017) is an agent that used prebuilt object representations to obtain state of the art policies on Pacman using object based structured exploration. Another interesting line of work is Gregor et al. (2016) which formalizes a notion of empowerment which ends up as a mutual information between options in the same MDP. Count-based exploration algorithms have yielded impressive results on hard exploration Atari games Ostrovski et al. (2017).

The Horde architecture (Sutton et al., 2011) proposed learning many value functions, termed as Generalized Value Functions (GVFs), using off-policy learning. Our approach automatically constructs GVFs using the abstract entity based representations. Our work is also related to pixel control (Jaderberg et al., 2016) as an auxiliary task. However, we learn to control and compose abstract discrete representations.

3 Model

Consider a Markov Decision Process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathbb{P}, r)$ (MDP) represented by states $s \in \mathcal{S}$, actions $a \in \mathcal{A}$, transition function $\mathbb{P} : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$ and an extrinsic reward function defined as $r : \mathcal{S} \rightarrow \mathbb{R}$.

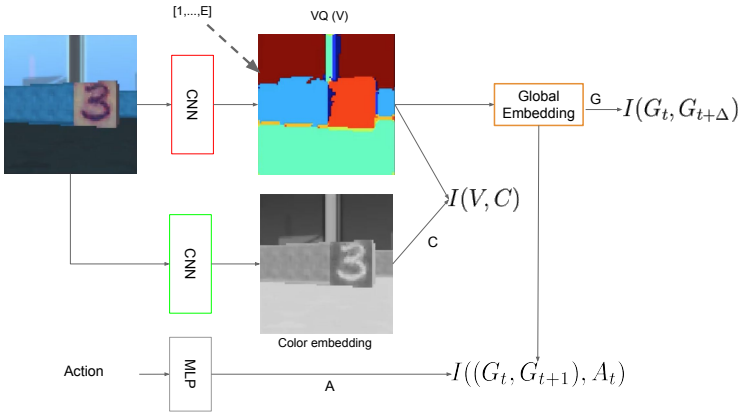

Figure 1: **Unsupervised Visual Abstraction Model:** There are three mutual information losses being computed in this process: $I(V, C)$, $I(G_t, G_{t+\Delta})$, and $I((G_t, G_{t+1}), A_t)$. $I(V, C)$ enforces the VQ to represent color information. $I(G_t, G_{t+\Delta})$ enforces the VQ to distinguish frames in the same unroll or large temporal segment with frames outside of this window. Finally, $I((G_t, G_{t+1}), A_t)$ encourages the VQ to encode action controllable elements in the frame. The agent, as shown in Figure 2, factors spatial VQ layers into E segmentation masks in order to compute geometric intrinsic rewards, which then induces a set of options model grounded in V.

In a discrete MDP an agent observes a state s_t at time t, produces and action a_t then the agent can observe s_{t+1} produced according to $\mathbb{P}$ and $r_{t+1} = r(s_{t+1})$. The agent's objective is to maximize the expected sum of rewards over time. In this particular work we are focusing on visual inputs thus we assume $\mathcal{S} \subset \mathbb{R}^{H \times W}$ but we assume some of the intuition and machinery we develop will carry on to different domains.

The agent learns an abstract representation we call *visual entities* $v \in \mathcal{V}$, where $\mathcal{V} \subset \{0 \dots E\}^{H \times W}$, useful for deriving state s and intrinsic rewards $r^{e,m}$. Here $e \in \{0 \dots E\}$ denotes a discrete entity and E is the maximum number of possible entities, and $m \in \{0 \dots M\}$ denotes a geometric feature of e's derived segmentation mask, from M possible measurements. These reward functions induce $M \times E$ additional MDPs $\mathcal{O}^{e,m} = (\mathcal{S}, \mathcal{A}, \mathbb{P}, r^{e,m})$. With the correct intrinsic rewards, the optimal policies for these MDPs can be composed to produce near-optimal behavior in the the original MDP of interest. In particular, we are interested in intrisic rewards that induce behaviors that can be used for exploration in the original MDP, and assist in computing an optimal policy.

Our agent architecture tries to leverage this intuition see Figure 2. The top level MDP $\mathcal{M}$ is represented by Q_{meta} which outputs action a_t^M at time t, where the action space is discrete 1 of $(E \times M) + 1$ possible actions. In our implementation this is modelled by composite actions $E + 1$ and M i.e. $a_t^M = (e_t, m_t)$ with $e_t \in \{1, E + 1\}$ and $m_t \in \{1, M\}$. We also learn $(E \times M) + 1$ separate Q functions $Q^{e,m}$ that solve each of the $\mathcal{O}^{e,m}$ and one for the original MDP Q_{task}. These Q functions work over the environment action set.

3.1 Visual Abstractions

Our agent relies on an abstraction model that assigns each pixel in the image to one of E separate abstractions or entities. To obtain this representation the image is passed through a convolutional network (CNN) encoder to output a spatial grid of the same resolution as the original image. Then a vector quantized layer V (see van den Oord et al. (2017)) assigns the planar activations to 1 of E entities. From the agents perspective this means all the pixels that are being grouped together become indistinguishable above this layer thus providing the visual abstraction we desire.

Let us define $f : \mathbb{R}^{H \times W} \rightarrow \{0, E\}^{H \times W}$ the function that takes the observation at time t and computes the abstract representation corresponding to it $v_t = f(s_t)$. The function f can be degenerate without constraints, unless it is made bijective. We do this by training a function $g : \{0, E\}^{H \times W} \rightarrow \mathbb{R}^{H \times W}$ such that $g(f(s_t)) \simeq s_t$. This ensures that the structure of $\mathcal{S}$ is preserved

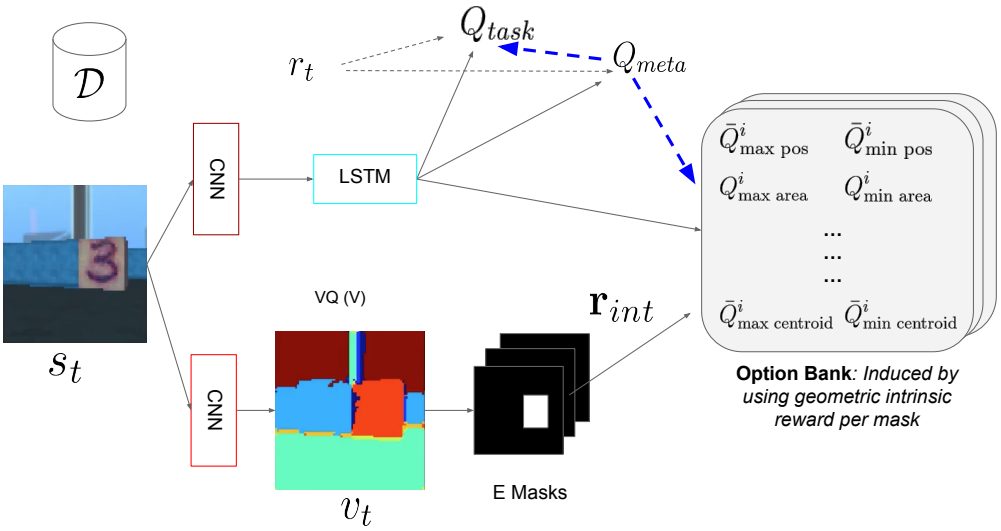

Figure 2: **Agent Architecture: (a)** The discrete E-VQ layer as described in Figure 1 outputs E segmentation masks. The agent derives intrinsic rewards based on affine geometric properties of these masks such as: area and centroid positions. An options bank learns Q functions to minimize and maximize these measurements to represent a basis set of useful and interpretable behaviors. Each option has a fixed termination length. **(b)** The original image is also passed through a CNN and LSTM network, which learns to predict two sets of variables: Q_{task} and Q_{meta}. The Q_{meta} and Q_{task} both get external task reward. Q_{meta} acts every T steps, which is the fixed temporal commitment window, and outputs an action to select and execute either: (1) composition over Q function from the option bank indexed by a particular entity and an intrinsic reward function or (2) the Q_{task} policy which outputs raw actions. All Q functions are trained with off-policy learning with a shared replay buffer.

in $\mathcal{V}$. In this work we aim for learning abstractions thus we aim for an injective function f. So what we need to define is $\forall x, y, f(x) \neq f(y)$.

The basis of the estimation is the following lower bound on the mutual information which holds for in the case of discrete variables[1]:

$$I(X,Y) = H(X) - H(X|Y) \geq \mathbb{E}_{X,Y}\left[\log q(X|Y)\right] + H(X) \tag{1}$$

which allows us to maximize a lower bound on the mutual information between two variables by minimizing a classification loss.

By choosing the appropriate joint distribution and marginals we can derive the right representation invariances as follows (see Figure 1 for an overview):

Preserving global information. The main term driving the representation learning is a global information term. To estimate it, the VQ layer output at time t, v_t, is further processed by another CNN encoder to output a frame level embedding vector G_t. We train a non-parametric classifier to distinguish pairs of frames from the same unroll from pairs of frames from different unrolls. For that we form pairs $(G_t, G_{t+\Delta})$, where Δ is sampled randomly, from pairs (G_t, G'), where G' is sampled randomly from a different unroll. Training this classifier lets us indirectly maximize a lower-bound on $I(G_t, G_{t+\Delta})$ because of equation (1). This forces V to preserve enough information the distinguishes this particular unroll from other ones which tends to remove all irrelevant information like textures and unchanging "background" elements and preserve useful moving elements. Note that as long as predictions are stable across time there is no pressure exerted by this cost to simplify the

[1]For continuous variables similar arguments lead to classification objectives e.g. Barber & Agakov (2004) and Belghazi et al. (2018).

representation. This is important because we will add more losses that will bring out other aspects of variation in the input and they should not affect this optimization significantly.

Preserving controllable information. Secondly, we want the controllable information to be preserved in the abstract representation. That is, we want to know what aspects of the input were changed as a result of the action, or equivalently which action was taken in a particular transition. For that we add another loss, denoted by $I((G_t, G_{t+1}), A_t)$, that maximizes mutual information between the a pair of consecutive frames and the action that was taken in the transition. We optimize this loss in the same way as the previous one using a non-parametric classifier.

Preserving local appearance information. Finally, for hard exploration Atari games, where inputs change very little, we add an appearance preservation term[2]. This term is meant to align the abstract representation with appearance changes. For that we feed the input image into a shallow CNN encoder which outputs an embedding C of the local color and texture structure of the image. We would like for the abstract representation to follow the appearance changes thus we maximize mutual information $I(V, C)$ in the same way as before. The positive pairs spatially aligned V and C embeddings, and the negative pairs are obtained by sampling color embeddings from other spatial locations and pairing them with non-sampled V. This promotes V representations that represent well colors changes under the representational constraints of the VQ representation.

To learn the agents' abstract representation we minimize a weighted sum of these classification losses

$$\mathcal{L}_{abs}(\theta_{abs}) = -\alpha_g \mathbb{E}_{G'} \log q_g(Z_g|(G', G_t)) - \alpha_c \log q_c(A_t|(G_t, G_{t+1})) - \alpha_a \mathbb{E}_{C'} \log q_a(Z_c|(C', V_t)) \tag{2}$$

where $Z_g = 1$ if G' is in the same unroll and 0 otherwise and C' is sampled according to the paragraph above.

3.2 Behavior Abstraction

The agent is represented primarily by three sets of Q functions: Q_{meta}, Q_{task} and $\{Q^{1,1}, \dots Q^{e,m}\}$ from the options bank. All policies are epsilon greedy under the corresponding Q functions. We denote T the fixed temporal commitment window for Q_{meta}, which means that it acts every T steps. Note that Q_{task} and $Q^{e,m}$s act at each environment time step. We can express all three Q functions as: (1) $Q_{meta}(s, (e, m)) = \mathbb{E}\big[\sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'} | s_t = s, a_t^M = (e, m), \pi_{meta}\big]$, (2) $Q_{meta}(s, (e, m)) = \mathbb{E}\big[\sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'} | s_t = s, a_t^M = (e, m), \pi_{meta}\big]$ and (3) $Q^{e,m}(s, a) = \mathbb{E}\big[\sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'}^{e,m} | s_t = s, a_t = a, \pi^{e,m}\big]$.

We represent each Q function with a deep Q network (Mnih et al., 2015), for instance $Q_{task}(s, a) \approx Q_{task}(s, a; \theta_{task})$. Each $Q \in \{Q_{task}, Q_{meta}, Q^{1,1}, ..., Q^{e,m}\}$ can be trained by minimizing corresponding loss functions – $\mathcal{L}_{task}(\theta_{task}), \mathcal{L}_{meta}(\theta_{meta})$ and $\{\mathcal{L}_{bank}^{1,1}(\theta^{1,1}), ..., \mathcal{L}_{bank}^{e,m}(\theta^{e,m})\}$. We store experiences $(s_t, (e_t, m_t, a_t), (\mathbf{r}_{int}, r_{ext}), s_{t+1})$ in $\mathcal{D}$ a shared buffer from which all the different Q functions can sample to perform their updates. The transitions are stored in such a way as to be able to sample trajectories.

The Q-learning Sutton & Barto (1998) objective can be written as minimizing the loss:

$$\mathcal{L}_{task}(\theta_{task}) = \mathbb{E}_{(s_\tau, a_\tau, r_\tau, s_{\tau+1})_\tau \sim D}[(R_\tau - Q_{task}(s_\tau, a_\tau; \theta_{task}))^2] \tag{3}$$

where $R_\tau = \sum_{t=\tau}^{\tau+U-1} \gamma^{t+\tau} r_t + \gamma^{t+U} \max_{a'} Q_{task}(s_{\tau+U}, a'; \theta_{task})$, where U is the length of the unroll. The loss function $\mathcal{L}_{meta}$ and all the $\mathcal{L}_{bank}^{e,m}$ can be written in a similar fashion. In our experiments we use $Q(\lambda)$ and learn all parameters with stochastic gradient descent by sampling experience trajectories from the shared replay buffer (see Algorithm 1 for details).

4 Experiments

The implementation has learning setup inspired by the batched actor critic agent Espeholt et al. (2018), with a central GPU learner and multiple actors (64 in most experiments). See supplementary

[2]We found that for tasks that have richer visual variation this was not necessary.

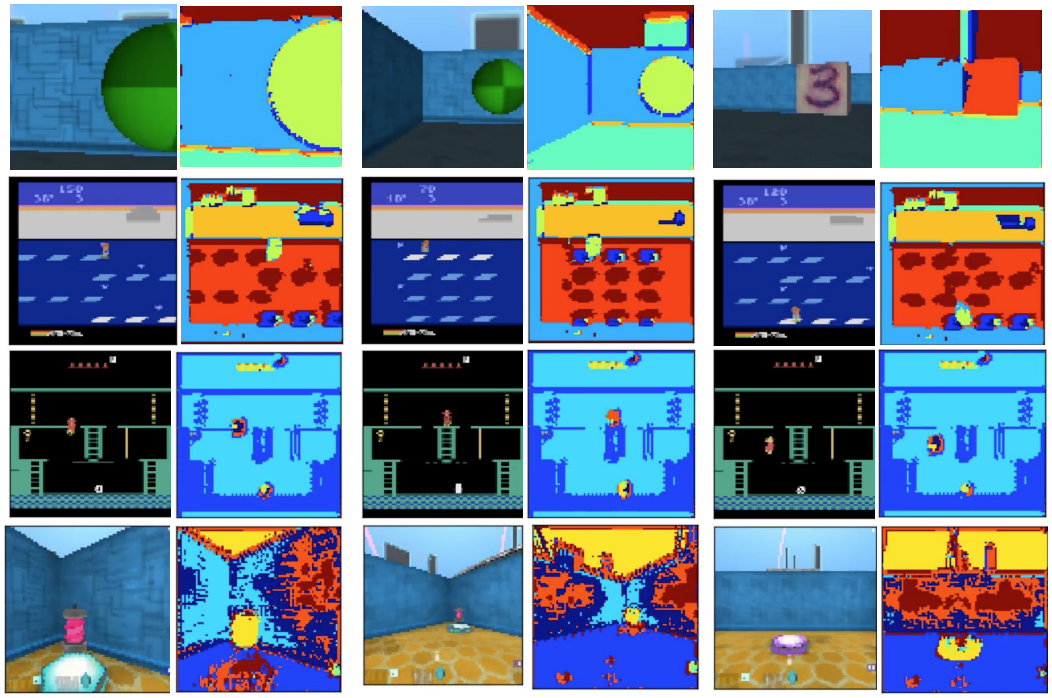

Figure 3: **Unsupervised Visual Abstractions:** The first row shows images and corresponding learnt visual abstractions for the navigation domain. The colors in the segmentation masks corresponds to instances without any semantic meaning. In the second row, we show that the abstraction model is fairly robust to seed variations. Note that the colors change but the similar pixel groups are merged together. The last row shows additional results when jointly training across multiple games. The background, moving and controllable groups of pixel are similarly grouped.

Algorithm 1 Learning algorithm

Inputs: N number of episodes, ϵ_{base} and ϵ_{meta} exploration parameters, T the commitment length, $\lambda_{task}, \lambda_{meta}, \lambda_{bank}$ cost function parameters
Initialize experience replay buffer $\mathcal{D}$ and parameters $\{\theta_{meta}, \theta_{task}, \theta^{1,1}, ..., \theta^{e,m}\}$ for the meta-control agent, task agent and options models respectively.
for $i = 1$ to N **do**
    Initialize environment and get start state s
    $s_0 \leftarrow s$
    while s_t is **not** terminal **do**
        if $t \equiv 0 \mod T$ **then**
            $a_t^M = (e_t, m_t) \leftarrow \text{EPSGREEDY}(s_t, \epsilon_{meta}, \theta_{meta})$
        end if
        Compute abstract features v_t from s_t (Section **??**).
        Compute intrinsic rewards $\mathbf{r}_{int} = (r^{e,m} | \forall e \leq E, m \leq M)$ from v_t and v_{t-1}
        if $e_t \leq E$ **then**
            $a_t \leftarrow \text{EPSGREEDY}(s_t, \epsilon_{base}, Q^{e_t,m_t})$ % selected Q function from bank
        else
            $a_t \leftarrow \text{EPSGREEDY}(s_t, \epsilon_{base}, Q_{task})$ % selected Q_{task}
        end if
        Execute a_t and obtain next state s_{t+1} and extrinsic reward r_t from environment
        Store transition $(s_t, (e_t, m_t, a_t), (\mathbf{r}_{int}, r_{ext}), s_{t+1})$ in $\mathcal{D}$
        Use RMSProp to optimize $\mathcal{L}_{abs}(\theta_{abs})$ %see eq. (2))
        Use RMSProp to optimize $\lambda_{task}\mathcal{L}_{task} + \lambda_{meta}\mathcal{L}_{meta} + \lambda_{bank}\frac{1}{ME}\sum_{e=1}^{E}\sum_{m=1}^{M}\mathcal{L}_{bank}^{e,m}$ (see eq. (3))
    end while
end for

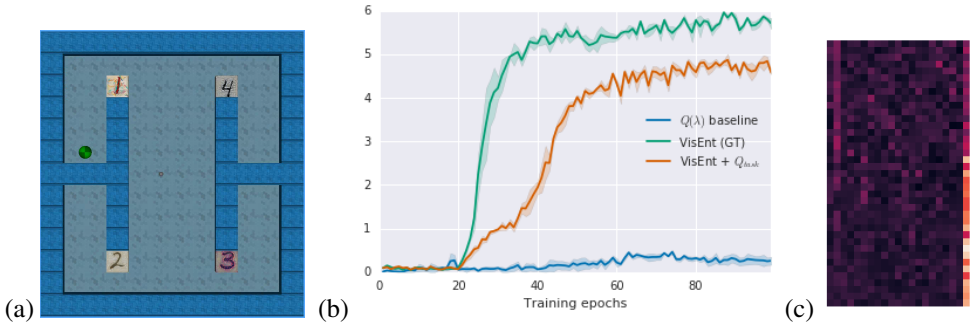

Figure 4: **Navigation domain.** (a) Top-down view of a 3D Maze. The agent observes pixels from the first person view. The task is described in section 4.1. (b) The green curve denotes an optimal agent with access to ground truth visual abstractions. Our learnt model achieves close to optimal performance while the baseline fails to solve the task. (c) Plot showing how the meta control policy switches between options bank and task control policies. Time increases top to bottom. The right most column is the task policy and all other columns denote option policies. Initially the agent uses the options policies to explore and then gradually shift over to the task policy as training progresses.

A for details regarding visual abstraction architecture. See supplementary B for details regarding agent network architecture.

The exploration setup. For the baseline we have 3 types of actors differing only in the exploration parameter corresponding to *pure exploitation*, *high exploration* and *medium exploration*. From the total number of 64 actors that corresponds to 20, 10 and 34 respectively with epsilon values of .001, .5, .01. Both the meta agent and base policies require exploration so we keep the same split but the *high exploration* is considered independently for the base and meta policies i.e. half the actors have $\epsilon_{meta} = .5$ and $\epsilon_{base} = .01$ and half have $\epsilon_{meta} = .1$ and $\epsilon_{base} = .5$. Furthermore we split equally the medium exploration actors in 3 groups with $(\epsilon_{meta} = .01, \epsilon_{base} = .1)$, $(\epsilon_{meta} = .33, \epsilon_{base} = .001)$ and $(\epsilon_{meta} = .01, \epsilon_{base} = .1)$ respectively. This provides much more stable learning at both the higher and lower levels of the behavior hierarchy.

4.1 Navigation

We tested our approach on a 3D navigation domain with sparse rewards and hard exploration requirements. The domain, whose top down view (not visible to the agent) is shown in figure 4, contains four rooms each having a textured number at the entrance. The agent receives the image observation as well as *hint*, a number from 1 to 4, which indicates the number of the room which contains the target object, a green sphere. The goal is to reach the target as often as possible in the limited time budget (250 steps). Once the goal is reached the target is relocated in another randomly chosen room and the hint is updated accordingly. To solve the task the agent needs to associate the hint with its position in the environment, which in turn means exploring far away regions in the maze because the target may move in a far away place. The policy bank provides exactly this type of exploration basis set. A baseline agent with the same architecture but without the policy bank cannot solve this task in 100M steps see Figure 4. In this domain we can also query the environment representation to determine if a given pixel comes from either a wall, the skyline, floor, one of the textured numbers or the target sphere. An agent with our proposed architecture but computing intrinsic rewards based on this priviledged information solves the tasks easily. Most interestingly, an agent that uses the abstraction inference method we propose can also solve it too, though with slightly worse performance.

To gain an insight into the evolution of the agent policy we plot the meta control policy actions during training. The top 20 actions represent the policy bank and the bottom most one represents Q_{task}. We can see that the agent uses most of the policies in the bank but as training progresses learns to rely more and more on Q_{task} which is the optimal behavior.

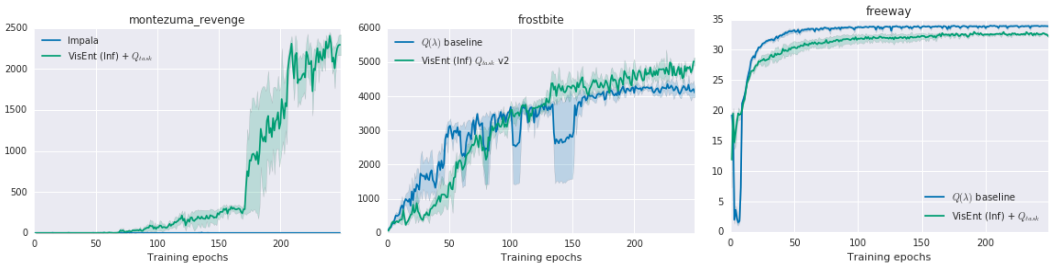

Figure 5: **Quantitative results on the Atari domain:** Average returns per episode on 3 Atari games where exploration is considered very challenging. On these domains we achieve better or comparable results to a strong baseline.

Figure 6: **Quantitative results on the DMLab domain:** Average returns per episode on 2 DMLab levels.

4.2 ATARI

We have run our agent with the same architecture and parameters on hard exploration Atari games where we can show that our agent has better performance than the baseline with a comparable architecture and losses (see Figure 5).

4.3 DM LAB

In order to see if our model scales to more visually challenging environments we have run our agent on 3 varied "DMLab-30" levelsBeattie et al. (2016). We show the training curves as a function of time in Figure 6. In the "non-match" task the agent teleports through pairs of rooms first a room with one object then a room with two, one of which is the same as before. To get a positive reward the agent has to move on top of the other object. The sequence continues for a fixed number of steps. Humans achieve 66 points and a state of the art agent gets 26. Our Q learner baseline achieves only 9 points whereas our proposed agent achieves 33. Though this is meant to be a memory task structured exploration seems to help achieve much better scores than the baselines.

We have also considered an experiment on the "keys doors" task. In this case the agent has to successively pickup keys to unlock doors which seems like a task structure were our representation could be more effective than the baseline. We found that, though both methods are competitive, our representation was not enough to learn a better policy. We think that this may be due to noise in the abstraction inference as well as a planning aspect that is not well enough handled in our agent in its current form. Finally, our agent is outperformed by the baseline on the challenging watermaze task. This is most likely due to the policy bank exploring mostly straight trajectories rather than circular ones which are more appropriate on this task.

5 DISCUSSION

We have shown that it is possible to design unsupervised structured exploration schemes for model-free DRL agents, with competitive performance on a range of environments given just raw pixels.

One of the biggest open question moving forward is to find strategies to balance structure or inductive biases and performance. Our current solution was to augment the meta-controller with Q_{task} along with the options bank as sub-behaviors. The typical strategy that agents follow is to rely on the options bank early in training and then use this experience to train the Q_{task} policy for optimality as training progresses. This is reasonable given that the options models may not cover the optimal policy but could serve as a good exploration algorithm throughout training. As new unsupervised architectures and losses are discovered, we expect to narrow the gap between the optimal desired behaviors and the options bank.

Learning visual entities from pixels in still a challenging open problem in unsupervised learning and computer vision. We expect novel sampling schemes in our proposed architecture to improve the entity discovery results. Other unsupervised video segmentation algorithms and discrete latent variable models could also be used to boost the discovery process.

REFERENCES

David Barber and Felix V. Agakov. Information maximization in noisy channels : A variational approach. In S. Thrun, L. K. Saul, and B. Schölkopf (eds.), *Advances in Neural Information Processing Systems 16*, pp. 201–208. MIT Press, 2004.

Charles Beattie, Joel Z. Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, Julian Schrittwieser, Keith Anderson, Sarah York, Max Cant, Adam Cain, Adrian Bolton, Stephen Gaffney, Helen King, Demis Hassabis, Shane Legg, and Stig Petersen. Deepmind lab. *CoRR*, abs/1612.03801, 2016.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Devon Hjelm, and Aaron Courville. Mutual information neural estimation. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 531–540, Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL `http://proceedings.mlr.press/v80/belghazi18a.html`.

Carlos Diuk, Andre Cohen, and Michael L Littman. An object-oriented representation for efficient reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pp. 240–247. ACM, 2008.

Rachit Dubey, Pulkit Agrawal, Deepak Pathak, Thomas L Griffiths, and Alexei A Efros. Investigating human priors for playing video games. *arXiv preprint arXiv:1802.10217*, 2018.

Lasse Espeholt, Hubert Soyer, Rémi Munos, Karen Simonyan, Volodymyr Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. IMPALA: scalable distributed deep-rl with importance weighted actor-learner architectures. In *ICML*, volume 80 of *JMLR Workshop and Conference Proceedings*, pp. 1406–1415. JMLR.org, 2018.

Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. *CoRR*, abs/1611.07507, 2016. URL `http://arxiv.org/abs/1611.07507`.

Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397*, 2016.

Xu Ji, João F. Henriques, and Andrea Vedaldi. Invariant information distillation for unsupervised image segmentation and clustering. *CoRR*, abs/1807.06653, 2018.

Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Advances in neural information processing systems*, pp. 3675–3683, 2016.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

Georg Ostrovski, Marc G. Bellemare, Aäron van den Oord, and Rémi Munos. Count-based exploration with neural density models. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2721–2730. PMLR, 2017.

Xiaofeng Ren and Jitendra Malik. Learning a classification model for segmentation. In *ICCV*, pp. 10–17. IEEE Computer Society, 2003.

Bernardino Romera-Paredes and Philip Hilaire Sean Torr. Recurrent instance segmentation. In *European Conference on Computer Vision*, pp. 312–329. Springer, 2016.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, August 2000. ISSN 0162-8828. doi: 10.1109/34.868688. URL `https://doi.org/10.1109/34.868688`.

Elizabeth S Spelke and Katherine D Kinzler. Core knowledge. *Developmental science*, 10(1):89–96, 2007.

Richard S. Sutton and Andrew G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998. ISBN 0262193981.

Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Richard S Sutton, Joseph Modayil, Michael Delp, Thomas Degris, Patrick M Pilarski, Adam White, and Doina Precup. Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pp. 761–768. International Foundation for Autonomous Agents and Multiagent Systems, 2011.

Aaron van den Oord, Oriol Vinyals, et al. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pp. 6306–6315, 2017.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

Harm Van Seijen, Mehdi Fatemi, Joshua Romoff, Romain Laroche, Tavian Barnes, and Jeffrey Tsang. Hybrid reward architecture for reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 5392–5402, 2017.

Xide Xia and Brian Kulis. W-net: A deep model for fully unsupervised image segmentation. *CoRR*, abs/1711.08506, 2017.