Count-Based Exploration in Feature Space for Reinforcement Learning*

Jarryd Martin, Suraj Narayanan S., Tom Everitt, Marcus Hutter

Research School of Computer Science, Australian National University, Canberra
jarrydmartinx@gmail.com,surajx@gmail.com,tom.everitt@anu.edu.au,
marcus.hutter@anu.edu.au

Abstract. We summarise recent work on a new count-based optimistic exploration algorithm for reinforcement learning (RL) that is feasible in environments with high-dimensional state-action spaces. The success of RL algorithms in these domains depends on generalisation from limited training experience. Function approximation techniques enable RL agents to generalise in order to estimate the *value* of unvisited states, but at present few methods enable generalisation regarding *uncertainty*. We present a new method for computing a generalised state visit-count, which allows the agent to estimate the uncertainty associated with any state. Our *ϕ-pseudocount* achieves generalisation by exploiting the same feature representation of the state space that is used for value function approximation. States that have less frequently observed features are deemed more uncertain. The *ϕ-Exploration-Bonus* algorithm rewards the agent for exploring in feature space rather than in the untransformed state space. The method is simpler and less computationally expensive than some previous proposals, and achieves near state-of-the-art results on high-dimensional RL benchmarks.

1 Introduction

Reinforcement learning (RL) methods have recently enjoyed widely publicised success in domains that once seemed far beyond their reach [8]. Much of this progress is due to the application of modern function approximation techniques to the problem of policy evaluation for Markov Decision Processes (MDPs) [14]. These techniques address a key shortcoming of tabular MDP solution methods: their inability to generalise what is learnt from one context to another. This sort of generalisation is crucial if the state-action space of the MDP is large, because the agent typically only visits a small subset of that space during training.

Comparatively little progress has been made on the problem of efficient *exploration* in large domains. Even algorithms that use sophisticated nonlinear methods for policy evaluation tend to use very old, inefficient exploration techniques, such as the ϵ-greedy strategy [4, 7, 9]. There are more efficient tabular *count-based* exploration algorithms for finite MDPs, which drive the agent to reduce its uncertainty by visiting states that have low visit-counts [13]. However,

* This work was supported in part by ARC DP150104590.

these algorithms are often ineffective in MDPs with high-dimensional state-action spaces, because most states are never visited during training, and the visit-count remains at zero nearly everywhere.

Count-based exploration algorithms have only very recently been successfully adapted for these large problems [2, 15]. The breakthrough has been the development of *generalised state visit-counts,* which are larger for states that are more *similar* to visited states, and which can be nonzero for unvisited states. The key challenge is to compute an appropriate similarity measure in an efficient way, such that these exploration methods can be combined with scalable RL algorithms. It soon becomes infeasible, for example, to do so by storing the entire history of visited states and comparing each new state to those in the history. The most promising proposals instead compute generalised counts from a compressed representation of the history of visited states.

This paper presents a new count-based exploration algorithm that is feasible in environments with large state-action spaces. It can be combined with any value-based RL algorithm that uses linear function approximation (LFA). Our principal contribution is a *new method for computing generalised visit-counts.* We construct a visit-density model in order to measure the similarity between states. To do so, we exploit the feature map that is used for value function approximation, and construct a density model over the transformed *feature space.* This model assigns higher probability to state feature vectors that *share features* with visited states. Generalised visit-counts are then computed from these probabilities; states with frequently observed features are assigned higher counts. These counts serve as a measure of the uncertainty associated with a state. *Exploration bonuses* are then computed from these counts in order to encourage the agent to visit regions of the state-space with less familiar features.

Our density model can be trivially derived from *any* feature map used for LFA, regardless of the application domain, and requires little or no additional design. In contrast to existing algorithms, there is no need to perform a special dimensionality reduction of the state space in order to compute our generalised visit-counts. Our method uses the same lower-dimensional feature representation to estimate value *and* to estimate uncertainty. This makes it simpler to implement and less computationally expensive than some existing proposals. Our evaluation demonstrates that this simple approach achieves near state-of-the-art performance on high-dimensional RL benchmarks.

2 Technical Background

The reinforcement learning (RL) problem is usually formulated as an MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$, where $\mathcal{S}$ is the set of states of the environment, $\mathcal{A}$ is the set of available actions, $\mathcal{P} : (S \times \mathcal{A}) \times S \to [0, 1]$ is the state transition distribution, $\mathcal{R} : (\mathcal{S} \times \mathcal{A}) \times \mathcal{S} \to \mathbb{R}$ is the reward function, and γ is the discount factor. The agent is formally a *policy* $\pi : \mathcal{S} \to \mathcal{A}$ that maps a state to an action. At timestep t, the agent is in a *state* $s_t \in \mathcal{S}$, receives a reward r_t, and takes an *action* $a_t \in \mathcal{A}$. We seek a policy π that maximises the *expected* sum of future rewards, or *value.* The

action-value $Q^\pi(s,a)$ of a state-action pair (s,a) under a policy π is the expected discounted sum of future rewards, given that the agent takes action a from state s, and follows π thereafter: $Q^\pi(s,a) = \mathbb{E}_\pi\big[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s,\ a_t = a\big]$. RL methods that compute a *value function* are called *value-based* methods. *Tabular* methods store the value function as a table having one entry for each state(-action). This representation of the state space does not have sufficient structure to permit generalisation based on the *similarity* betweeen states. *Function approximation* methods achieve generalisation by approximating the value function by a parameterised functional form. In LFA the *approximate action-value function* $\hat{Q}^\pi_t(s,a) = \boldsymbol{\theta}_t^\top \boldsymbol{\phi}(s,a)$ is a linear combination of state-action features, where $\boldsymbol{\phi} :\ \mathcal{S} \times \mathcal{A} \to \mathcal{T} \subseteq \mathbb{R}^M$ is an M-dimensional feature map and $\boldsymbol{\theta}_t \in \mathbb{R}^M$ is a parameter vector.

3 Method

Here we introduce the *ϕ-Exploration Bonus* (ϕ-EB) algorithm, which drives the agent to visit states about which it is uncertain. Following other optimistic count-based exploration algorithms [2, 10], we use a (generalised) state visit-count in order to estimate the uncertainty associated with a state. A generalised count is a *novelty* measure that quantifies how dissimilar a state is from those already visited. Measuring novelty therefore involves choosing a similarity measure for states. Of course, states can be similar in myriad ways, but not all of these are relevant to solving the MDP. If the solution method used is value-based, then states should only be considered similar if they share the features that are determinative of value. This motivates us to construct a similarity measure that exploits the feature representation that is used for value function approximation. These features are explicitly designed to be relevant for estimating value. If they were not, they would not permit a good approximation to the true value function. This sets our method apart from the alternative approaches to deriving a novelty measure presented in [2, 10, 15]. They measure novelty with respect to a separate, exploration-specific representation of the state space, one that bears no relation to the value function or the reward structure of the MDP. We argue that measuring novelty in feature space is a simpler and more principled approach, and hypothesise that more efficient exploration will result.

3.1 A Visit-Density over Feature Space

Our exploration method is designed for use with LFA, and measures novelty with respect to a fixed feature representation of the state space. The challenge is to measure novelty without computing the distance between each new feature vector and those in the history. That approach becomes infeasible because the cost of computing these distances grows with the size of the history.

Our method constructs a *density model over feature space* that assigns higher probability to states that share more features with more frequently observed states. Let $\boldsymbol{\phi} : \mathcal{S} \to \mathcal{T} \subseteq \mathbb{R}^M$ be the feature mapping from the state space

into an M-dimensional feature space $\mathcal{T}$. Let $\boldsymbol{\phi}_t = \boldsymbol{\phi}(s_t)$ denote the state feature vector observed at time t. We denote the sequence of observed feature vectors after t timesteps by $\boldsymbol{\phi}_{1:t} \in \mathcal{T}^t$, and denote the set of all finite sequences of feature vectors by $\mathcal{T}^*$. Let $\boldsymbol{\phi}_{1:t}\boldsymbol{\phi}$ denote the sequence where $\boldsymbol{\phi}_{1:t}$ is followed by $\boldsymbol{\phi}$. The i-th element of $\boldsymbol{\phi}$ is denoted by ϕ_i, and the i-th element of $\boldsymbol{\phi}_t$ is $\phi_{t,i}$.

Definition 1 (Feature Visit-Density). *Let $\rho : \mathcal{T}^* \times \mathcal{T} \to [0,1]$ be a density model that maps a finite sequence of feature vectors $\boldsymbol{\phi}_{1:t} \in \mathcal{T}^*$ to a probability distribution over $\mathcal{T}$. The* feature visit-density $\rho_t(\boldsymbol{\phi}) = \prod_{i=1}^{M} \rho_t^i(\phi_i)$ *at time t is the distribution over $\mathcal{T}$ that is returned by ρ after observing $\boldsymbol{\phi}_{1:t}$, where each of the ρ_t^i are independent factor distributions over individual features $\phi_i \in \mathcal{U} \subseteq \mathbb{R}$.*

If $\mathcal{U}$ is countable we can use a count-based estimator for the factor models $\rho_t^i(\phi_i)$, such as the empirical estimator $\rho_t^i(\phi_i) = \frac{N_t(\phi_i)}{t}$, where $N_t(\phi_i)$ is the number of times ϕ_i has occurred. In our implementation we use binary-valued features and the Krichevsky-Trofimov (KT) estimator $\rho_t^i(\phi_i) = \frac{N_t(\phi_i)+\frac{1}{2}}{t+1}$.

This density model induces a similarity measure on the feature space. Loosely speaking, feature vectors that share component features are deemed similar. This enables us to use $\rho_t(\boldsymbol{\phi})$ as a novelty measure for states, by comparing the features of newly observed states to those in the history. If $\boldsymbol{\phi}(s)$ has more novel component features, $\rho_t(\boldsymbol{\phi})$ will be lower. By modelling the features as independent, and using count-based estimators as factor models, our method learns reasonable novelty estimates from very little data.

3.2 Reinforcement Learning with the ϕ-pseudocount

Here we adopt a recently proposed method for computing generalised visit-counts from density models [2, 10]. We derive the *ϕ-pseudocount* from our feature visit-density.

Definition 2 (ϕ-pseudocount). *Let $\rho_t(\boldsymbol{\phi})$ be the feature visit-density after observing $\boldsymbol{\phi}_{1:t}$. Let $\rho'_t(\boldsymbol{\phi})$ denote the same density model after $\boldsymbol{\phi}_{1:t}\boldsymbol{\phi}$ has been observed. The* ϕ-pseudocount $\hat{N}_t^{\phi}(s)$ *for $s \in \mathcal{S}$ at time t is $\hat{N}_t^{\phi}(s) = \frac{\rho_t(\boldsymbol{\phi}(s))(1-\rho'_t(\boldsymbol{\phi}(s)))}{\rho'_t(\boldsymbol{\phi}(s))-\rho_t(\boldsymbol{\phi}(s))}$.*

Following traditional count-based exploration algorithms, we drive optimistic exploration by computing a bonus from the ϕ-pseudocount.

Definition 3 (ϕ-Exploration Bonus). *Let $\beta \in \mathbb{R}$ be a hyperparameter. The* ϕ-exploration bonus *for a state-action pair $(s,a) \in \mathcal{S} \times \mathcal{A}$ is $\mathcal{R}_t^{\phi}(s,a) = \frac{\beta}{\sqrt{\hat{N}_t^{\phi}(s)}}$.*

This bonus is added to the reward r_t. The agent is trained on the augmented reward $r_t^+ = r_t + R_t^{\phi}(s,a)$ using any value-based RL algorithm with LFA. At each timestep our algorithm performs updates for at most M estimators, one for each feature. The cost of our method is therefore independent of the size of the state-action space, and scales only in the number of features. If the feature vectors are sparse, we can maintain a single prototype estimator for all the features that have not yet been observed. Under these conditions our method scales only in the number of *observed* features.

Algorithm 1 Reinforcement Learning with LFA and ϕ-EB.

Require: β, t_{end}
  while $t < t_{\text{end}}$ **do**
    Observe $\boldsymbol{\phi}(s)$, r_t
    Compute $\rho_t(\boldsymbol{\phi}) = \prod_i^M \rho_t^i(\phi_i)$
    for i in {1,...,M} **do**
      Update ρ_{t+1}^i with observed ϕ_i
    end for
    Compute $\rho_{t+1}(\boldsymbol{\phi}) = \prod_i^M \rho_{t+1}^i(\phi_i)$
    Compute $\hat{N}_t^{\phi}(s) = \frac{\rho_t(\boldsymbol{\phi})(1-\rho_{t+1}(\boldsymbol{\phi}))}{\rho_{t+1}(\boldsymbol{\phi})-\rho_t(\boldsymbol{\phi})}$
    Compute $\mathcal{R}_t^{\phi}(s,a) = \frac{\beta}{\sqrt{\hat{N}_t^{\phi}(s)}}$
    Set $r_t^+ = r_t + \mathcal{R}_t^{\phi}(s,a)$
    Pass $\boldsymbol{\phi}(s)$, r_t^+ to RL algorithm to update $\boldsymbol{\theta}_t$
  end while
  return $\boldsymbol{\theta}_{t_{\text{end}}}$

4 Empirical Evaluation

Here we summarise the results of the evaluation of our algorithm on five hard exploration games from the Arcade Learning Environment (ALE) [1]. (Theoretical results are presented in [6].) Three of the chosen games have sparse rewards (Montezuma's Revenge, Venture, Freeway) and two have dense rewards (Frostbite, Q*bert). We conducted five independent learning trials for Montezuma and Venture, and two trials for the remaining three games. All agents were trained for 100 million frames on the no-op metric using the stochastic version of the ALE [1]. Trained agents were then evaluated for 500 episodes. We implement Algorithm 1 using Sarsa(λ) with replacing traces and LFA as our RL method. To implement LFA in the ALE we use the Blob-PROST feature set presented in [5] and the same parameter values for Sarsa(λ) algorithm. We also evaluate a baseline implementation of ϵ-greedy Sarsa(λ) with the same feature set, denoted Sarsa-ϵ. The β coefficient in the ϕ-exploration bonus was set to 0.05 for *all games*, after a coarse parameter sweep across a range of ALE games.

Learning curves are reported in Figure 1. ϕ-EB with $\beta = 0.05$ outperforms Sarsa-ϵ on all tested games except Freeway. (Note that with $\beta = 0.035$ it performs well in

	Vent.	**Mont.**	**Free.**	**Frost.**	**Qbert**
ϕ-**EB** (100)[6]	1169.2	2745.4	0.0	2770.1	4111.8
Sarsa-ϵ (100)	0.0	399.5	29.9	1394.3	3895.3
PC (100)[2]	N/A	**3459**	N/A	N/A	N/A
A3C+ [2]	0	142	27	507	15805
Hash [15]	445	75	**34**	**5214**	N/A
MPEB (20)[12]	N/A	0	12	380	N/A
DDQN [4]	98	0	33	1683	15088
DQNPA [3]	1172	0	33	3469	5237
Gorila [9]	**1245**	4	12	605	10816
TRPO [11]	121	0	16	2869	7733
Dueling [16]	497	0	0	4672	**19220**

Table 1: Evaluation scores after 200M frames, unless indicated in parentheses.

Freeway. Figure 1 shows the learning curve for the latter.) Since both algorithms use the same feature set and RL algorithm, and differ only in their exploration policies, this is strong evidence that ϕ-EB produces improvement over random exploration across a range of environments. Table 1 reports the average evaluation scores for leading algorithms. On Venture, ϕ-EB's score is the second highest reported to date, and the third highest on Montezuma's Revenge. Note that no other algorithm achieves good scores on *both* these sparse reward games. On the dense reward games (Frostbite and Qbert), nonlinear methods perform better.

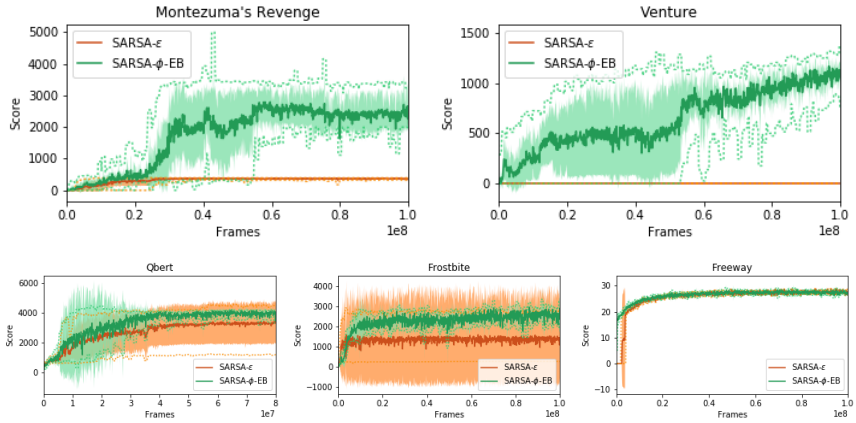

Fig. 1: Average training scores for ϕ-EB and the baseline Sarsa-ϵ. Dashed lines are min/max scores. Shaded regions describe one standard deviation.

5 Conclusion

We have introduced the ϕ-Exploration Bonus method, a count-based optimistic exploration strategy that scales to high-dimensional environments. It is simpler to implement and less computationally demanding than comparable proposals, since it does not require an exploration-specific state representation, but rather exploits the features used in the approximate value function. Our evaluation shows that it improves upon ϵ-greedy exploration on a variety of games, and that it is competitive with leading exploration techniques developed for deep RL. This supports our conjecture that using the same task-relevant features for value function approximation and novelty estimation is an efficient and principled way to generalise visit-counts to the high-dimensional setting. We conclude by noting that this reliance on the feature representation used for LFA is also a limitation. It is not obvious how a method like ours could be combined with the nonlinear function approximation techniques that have driven recent progress in RL. We hope the success of our simple method will inspire future work in this direction.

References

1. Bellemare, M.G., Naddaf, Y., Veness, J., Bowling, M.: The arcade learning environment: An evaluation platform for general agents. Journal of Artificial Intelligence Research 47, 253–279 (2013)
2. Bellemare, M.G., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., Munos, R.: Unifying count-based exploration and intrinsic motivation. CoRR abs/1606.01868 (2016), http://arxiv.org/abs/1606.01868
3. van Hasselt, H., Guez, A., Hessel, M., Silver, D.: Learning values across many orders of magnitude. CoRR abs/1602.07714 (2016), http://arxiv.org/abs/1602.07714
4. van Hasselt, H., Guez, A., Silver, D.: Deep reinforcement learning with double Q-learning. In: AAAI (2016)
5. Liang, Y., Machado, M.C., Talvitie, E., Bowling, M.: State of the art control of Atari games using shallow reinforcement learning. In: Autonomous Agents and Multi-Agent Systems (2016)
6. Martin, J., Narayanan S., S., Everitt, T., Hutter, M.: Count-Based Exploration in Feature Space for Reinforcement Learning. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. AAAI Press (2017), http://arxiv.org/abs/1706.08090
7. Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T.P., Harley, T., Silver, D., Kavukcuoglu, K.: Asynchronous methods for deep reinforcement learning. In: International Conference on Machine Learning (2016)
8. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.a., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D.: Human-level control through deep reinforcement learning. Nature 518(7540), 529–533 (2015)
9. Nair, A., Srinivasan, P., Blackwell, S., Alcicek, C., Fearon, R., Maria, A.D., Panneershelvam, V., Suleyman, M., Beattie, C., Petersen, S., Legg, S., Mnih, V., Kavukcuoglu, K., Silver, D.: Massively parallel methods for deep reinforcement learning. CoRR abs/1507.04296 (2015), http://arxiv.org/abs/1507.04296
10. Ostrovski, G., Bellemare, M.G., van den Oord, A., Munos, R.: Count-based exploration with neural density models. CoRR abs/1703.01310 (2017), http://arxiv.org/abs/1703.01310
11. Schulman, J., Levine, S., Moritz, P., Jordan, M.I., Abbeel, P.: Trust region policy optimization. CoRR abs/1502.05477 (2015), http://arxiv.org/abs/1502.05477
12. Stadie, B.C., Levine, S., Abbeel, P.: Incentivizing exploration in reinforcement learning with deep predictive models. CoRR abs/1507.00814 (2015), http://arxiv.org/abs/1507.00814
13. Strehl, A.L., Littman, M.L.: An analysis of model-based interval estimation for Markov decision processes. Journal of Computer and System Sciences 74(8), 1309–1331 (2008)
14. Sutton, R., Barto, A.: Reinforcement Learning: An Introduction, vol. 1. MIT press Cambridge (1998)
15. Tang, H., Houthooft, R., Foote, D., Stooke, A., Chen, X., Duan, Y., Schulman, J., Turck, F.D., Abbeel, P.: #Exploration: A study of count-based exploration for deep reinforcement learning. CoRR abs/1611.04717 (2016), http://arxiv.org/abs/1611.04717
16. Wang, Z., de Freitas, N., Schaul, T., Hessel, M., van Hasselt, H., Lanctot, M.: Dueling network architectures for deep reinforcement learning. In: International Conference on Machine Learning (2016)