Disentangled Predictive Representation for Meta-Reinforcement Learning

Sephora Madjiheurem [1] **Laura Toni** [1]

Abstract

A major challenge in reinforcement learning is the design of agents that are able to generalise across tasks under common dynamics. A viable solution is meta-reinforcement learning, which identifies common structures among past tasks to be then generalised to new tasks (meta-test). Prior works learn meta-representation jointly while solving a task, resulting in representations that do not generalise well across different policies, leading to sampling-inefficiency during meta-test phases. In this work, we introduce *state2vec*, an efficient and low-complexity unsupervised framework for learning disentangled representation that are more general. The state embedding vectors learned with state2vec capture the geometry of the underlying state space, resulting in high-quality basis functions for linear value function approximation.

1. Introduction

Finding high-quality representations remains a major challenge in reinforcement learning (RL). Recent efforts have focused on learning representations from experience. Depending on whether the focus is on prediction accuracy, data efficiency, stability or biological plausibility, there are different ways of hypothesising on what constitutes a good representation: Bellemare et al. (2019) learn a state representation from which we can best approximate the value function of any stationary policy for a given task; Gelada et al. (2019) learn a representation for data efficiency by solving auxiliary tasks; Ghosh & Bellemare (2020) discover representations that guarantee the stability of temporal difference learning; and Stachenfeld et al. (2017) learn a predictive representation that captures many aspects of place cell responses found in rodents' brain.

While representation learning in the context of single task RL has been vastly studied, these methods do not trivially extend to multi-task RL since the learned representations are highly task-specific. In the case in which an agent must learn different tasks that share some characteristics (*e.g.,* taking the train to work and taking the train back home), it is clearly desirable for the agent to be able to leverage some of the knowledge acquired while exploring one task to speed up the solving of the other similar task. Recent works tackling multi-task RL use deep reinforcement learning methods and augment their main objective with auxiliary tasks (Barreto et al., 2017; 2018). However, since these auxiliary tasks are learned in parallel while solving a specific task, the data used is on-policy, hence not guaranteed to generalise to a significantly different policy (Lehnert et al., 2017a). Higgins et al. (2017) and Du & Narasimhan (2019) address the issue of adaptation and transfer in RL by proposing to disentangle the representation learning phase from the task solving phase. While their work is limited to tasks with visual inputs and rely on deep learning methods, we propose a simple and general framework for learning task agnostic representations in an unsupervised way.

We consider a meta-reinforcement learning (meta-RL) problem in which tasks are characterized by the same environment (shared structure) but the reward function changes arbitrarily across tasks. Here, the agent learns at two different time scales: slow unsupervised meta-learning, exploiting the large experience accumulated while exploring the domain (learning of the shared structure), and fast learning on individual tasks. This enables learning how to quickly adapt to a previously unseen task with little data.

We propose *state2vec*, an efficient yet reliable framework for learning a representation that effectively captures the underlying geometry of the state space. In particular, the representation generated by state2vec exhibits the following properties: *(i)* learned from data rather than handcrafted − to avoid structural bias, see Madjiheurem & Toni (2019); *(ii)* low-dimensional − to ensure a fast adaptation during meta-testing; *(iii)* geometry-aware rather than task-aware − generalise across optimal policies. State2vec encodes states in low-dimensional embeddings, defining the similarity of states based on the discounted future transitions. Moreover, to ensure generalisation, the learning of the representation is fully unsupervised: we impose that the data used for training is entirely exploratory and independent of any specific task (it is reward agnostic). This allows us to use the same

[1]Department of Electrical and Electronic Engineering, University College London, United Kingdom. Correspondence to: Sephora Madjiheurem <sephora.madjiheurem.17@ucl.ac.uk>.

Proceedings of the 38^{th} International Conference on Machine Learning, 2021.

representation without any retraining of the features to solve tasks with varying reward functions. In the meta-testing phase, the agent will need to simply learn a task-aware coefficient vector to derive a value function approximation. We show experimentally that state2vec captures with high accuracy the structural geometry of the environment while remaining reward agnostic. The experiments also support the intuition that off-policy state2vec representations are robust low dimensional basis functions enabling accurate the value function approximation.

2. Background

2.1. Meta-Reinforcement Learning

In RL, a decision maker, or agent, interacts with an environment by selecting actions with the goal to maximise some long term reward. This is typically modelled as a Markov Decision Process (MDP). A discrete MDP is defined as the tuple $M = (S, A, P, R, \gamma)$, where S is a finite set of discrete states, A a finite set of actions, P describes the transition model $-$ with $P(s, a, s')$ giving the probability of visiting s' from state s once action a is taken, R describes the reward function and $\gamma \in (0, 1]$ defines the discount factor. We consider *finite* MDPs, in which the sets of states, actions, and rewards have a finite number of elements. A policy π is a mapping from states to probabilities of selecting each action in A. Formally, for a stochastic policy, $\pi(a|s)$ is the probability that the agent takes action a when the agent is in state s. Given a policy π, an action-value function Q^π is a mapping $S \times A \mapsto \mathbb{R}$ that describes the expected long-term discounted sum of rewards observed when the agent is in a given state s, takes action a, and follows policy π thereafter. Solving an MDP requires to find a policy that defines the optimal action-value function Q^*, which satisfies the following constraints:

$$Q^*(s, a) = R(s, a) + \gamma \sum_{s' \in S} P(s, a, s') \max_{a'} Q^*(s', a') \tag{1}$$

This recursive equation is known as *Bellman's optimal equation*. The optimal policy is a unique solution to Bellman's equation.

Here, we are interested in the set of MDPs spanned by the tuple (S, A, P, γ):

$$\mathcal{M} = \{M_1, M_2, \ldots, M_n\} \tag{2}$$

where each task M_i is an MDP defined by (S, A, P, R_i, γ), with $R_i : S \times A \mapsto \mathbb{R}$. In other words, we investigate the meta-learning problem in which tasks M_i share the same MDP components, except the reward function. This is of obvious interest as this formalism can be used to model real life applications. This is the same setting adopted in prior related works (Barreto et al., 2017; 2018; Borsa et al., 2019; Lehnert et al., 2017a). In this setting, the main challenge is to find an efficient way of learning the underlying dynamics that is shared across all tasks, such that once this information is known, solving a specific MDP becomes a much easier problem.

2.2. Successor Representation

In order to address the meta learning problem, we need to decouple the dynamics of the MDP (common across tasks) from the reward function (task discriminant) in the value function approximation. This decoupling motivates the adoption of the successor representation, or SR, (Dayan, 1993). With the SR, we can factor the action-value function into two independent terms:

$$Q^\pi(s, a) = \sum_{s'} \Psi^\pi(s, a, s') R(s, a), \tag{3}$$

where the SR $\Psi^\pi(s, a, s')$ is defined, for $\gamma < 1$, as:

$$\Psi^\pi(s, a, s') = \mathbb{E}_\pi\Big[\sum_{t=0}^{\infty} \gamma^t \mathbb{I}(s_t = s') | s_0 = s, a_0 = a\Big], \tag{4}$$

where $\mathbb{I}(s_t = s') = 1$ if $s_t = s'$ and 0 otherwise.

The provided interpretation, is that the SR is a predictive type of representation, which represents a state action pair as a feature vector $\mathbf{\Psi}^\pi_{s,a}$ such that, under policy π, the representation $\mathbf{\Psi}^\pi_{s,a}$ is similar to the feature vector of successors states. Computing the action-value function given the SR is computationally easier as it becomes a simple linear computation. Furthermore, given the SR, the re-computation of the action value function is robust to changes in the reward function: the new action value function can be quickly recomputed using the current SR. The SR is therefore a natural tool to consider for transfer in reinforcement learning.

2.3. Successor Feature

Barreto et al. (2017) proposed a generalisation of the SR called *successor feature* (SF). They make the assumption that the reward function can be parametrised with

$$R(s, a) = \phi(s, a)^\top \boldsymbol{w}\,, \tag{5}$$

where $\phi(s, a)$ is a feature vector for (s, a) and $\boldsymbol{w} \in \mathbb{R}^d$ is a vector of weights. Because no assumption is made about $\phi(s, a)$, the reward function could be recovered exactly, hence (5) is not too restrictive. Under this assumption, the action-value function for the task defined by $\boldsymbol{w}$ can be rewritten as:

$$Q^\pi(s, a) = \boldsymbol{\psi^\pi}(s, a)^\top \boldsymbol{w}\,. \tag{6}$$

where the successor feature $\boldsymbol{\psi}^\pi(s, a)$ is defined as

$$\boldsymbol{\psi}^\pi(s, a) \doteq \mathbb{E}_\pi\Big[\sum_{t=0}^{\infty} \gamma^{t-1} \phi_{i+1} | s_t = s, a_t = a\Big]\,. \tag{7}$$

In Barreto et al. (2017), authors also define the *generalized policy improvement* (GPI) theorem, which shows that given a previously computed SF approximation $\hat{\psi}_{\pi_k}(s,a)$ for some tasks $M_k \in \mathcal{M}$, the agent can derive a policy π_j for a new task $M_j \in \mathcal{M}$ which is guaranteed to preform at least as well as any previously learned policy. In practice, this means that across all the observed tasks, we will consider the best value function when deciding our policy. The main limitation, however, is the double dependency of the value function on M_k, which defines the task as well as the policy. Consequently, if all previous tasks are significantly different than the new task M_j (i.e. have significantly different optimal policies), the derived policy π_j will be far from the optimal policy for task M_j, meaning the knowledge of the previous task is not transferable to the new task.

3. Off-policy Successor Features Approximators

The main limitation of the proposed methodologies is that they proposed a representation that is transferable only across *similar* policies (Lehnert et al., 2017b). In the following, we define state2vec and describe how we learn the representations in an unsupervised way such that they can generalised to different tasks.

3.1. Meta-training : State2vec

The successor features are learned while taking decision, and is therefore intrinsically connected to the task. Therefore, we propose to approximate the SF off-policy and to use the same representation across all tasks in $\mathcal{M}$. We proposed an efficient unsupervised framework for learning continuous feature representations of states, such that, similarly to the SF, states that are neighbours in time should have similar representation. Our method is directly inspired by Grover & Leskovec (2016)'s *node2vec*, and hence we refer to it as *state2vec*.

State2vec learns state representations based on sample episodes' statistics. It optimises the representations such that states that are successors have similar representation. It does so by first collecting a data set $\mathcal{D}_\pi$ of n walks $L = \{(s_0,a_0),(s_1,a_1)\dots,(s_n,a_n)\}$ by following a sampling strategy π for maximum T steps (terminating earlier if it results in an absorbing goal state). Then, optimise the following objective function:

$$\max_{\Psi} \sum_{L\in\mathcal{D}_\pi} \sum_{(s,a)\in L} \log Pr(N(s,a)|\Psi(s,a)), \qquad (8)$$

where

$$N(s_i,a_i) = \{(s_{i+1},a_{i+1}),(s_{i+2},a_{i+2}),\dots,(s_{i+T},a_{i+T})\}$$

defines the succession of state action pair (s_i,a_i) of size T. Similarly to Grover & Leskovec (2016), we model the conditional likelihood as

$$Pr(N(s,a)|\Psi(s,a)) = \prod_{(s_j,a_j)\in N(s,a)} Pr(s_j,a_j|\Psi(s,a)), \qquad (9)$$

Unlike the *node2vec* algorithm, we account for the fact that neighbours that are further in time should be further discounted. We do so by modelling the the likelihood of every source-neighbour pair as a sigmoid weighted by a discount factor:

$$Pr(s_j,a_j|\Psi(s,a)) = \gamma^{|-j|}\sigma(\Psi(s_j,a_j)\cdot\Psi(s,a)) \qquad (10)$$

where σ denotes the sigmoid function.

3.2. Meta-testing with state2vec

Once the state2vec representation are learned, we can use them for solving any task in $\mathcal{M}$ without needing to do any retraining. The solving of task $M_{\boldsymbol{w}} \in \mathcal{M}$ given the structural representation Ψ reduces to optimising the following value function approximation for the weight vector θ_w:

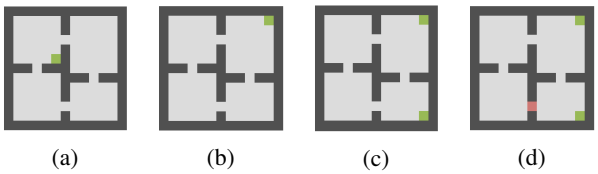
$$\hat{Q}^{\pi_{\boldsymbol{w}}}(s,a) = \Psi(s,a)^\top \boldsymbol{\theta_w}\,. \qquad (11)$$

This can be achieve using any parametric RL algorithm, such as fitted Q-learning or LSPI (Riedmiller, 2005; Lagoudakis & Parr, 2004).

4. Experiments

4.1. Case study

We consider the four-room domain (Sutton et al., 1999) shown in Figure 1. It is a two-dimensional space quantized into 169 states, 4 of which are doorways. The agent starts at a random location, and must collect a goal object at a location defined by the task. Depending on the task, the environment also contains "dangerous" zones. The goal object's location is shown in green in Figure 1, while the dangerous states are depicted in red. Collecting an object gives an instantaneous reward of $+100$, and entering a dangerous state gives an instantaneous penalty of -10. The the episode terminates when a goal object is collected.



(a) (b) (c) (d)

Figure 1. Four-room environment with different configurations.

4.2. Results

4.2.1. META-TRAINING

In the feature learning phase, we collect 300 sample walks of length 100 and run state2vec with $T = 50$ and discount factor $\gamma = 0.8$ for varying dimensions d. Figure 2 visualises the low dimensional (projection onto the first two principal components) representation of the states in the successor representation and in the state2vec feature spaces. As seen in Figure 2, is a close approximation to the exact successor representation under a uniform policy. In both cases, we clearly see that the representations have clustered the states within the same room together, while isolating the doorway states. The learned embeddings are shown to preserve the geometry of the state space and identify states that have a special structural role (e.g. doorways).

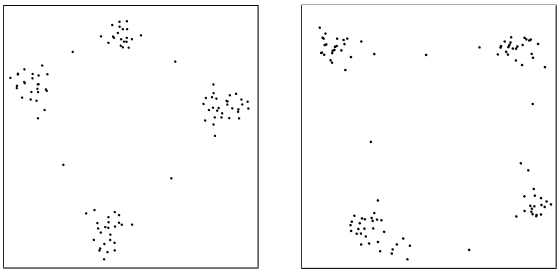


Figure 2. Visualisation of the states representation in feature space (2D PCA projection). **Left**: the exact successor representation, each vector in the original feature space has dimension 169. **Right**: the state2vec approximation of dimension 50 in the embedding space.

4.2.2. META-TESTING

In meta-testing phase, we use the learned state2vec features to learn the optimal policy of each individual. We collect sampled realisations of the form (s, a, s') by simulating 50 episodes of maximum length 200 (terminating earlier if the goal is reach) and run LSPI Lagoudakis & Parr (2004) with state2vec representations as basis vectors to learn the weights $\boldsymbol{\theta_w}$ in 11. Figure 3 shows the performance in terms of average cumulative reward for varying value of d. As it can be seen, we are able to achieve strong performance (maximum reward) for all tasks when using the pre-computed state2vec representations of dimensionality 100 with minimal additional exploration per task.

We compare the quality of state2vec embeddings with a state-of-the-art low dimensional basis function for linear value function approximation (Madjiheurem & Toni, 2019) Figure 4 shows an improved performance of state2vec over node2vec in terms of average cumulative reward. We suspect that the gain in performance comes for the fact that state2vec is design for RL, whereas node2vec is a generic graph embedding algorithm. Specifically, in the objective function, the notion of neighborhood in state2vec is such that further states in time are discounted more than the immediate successors.

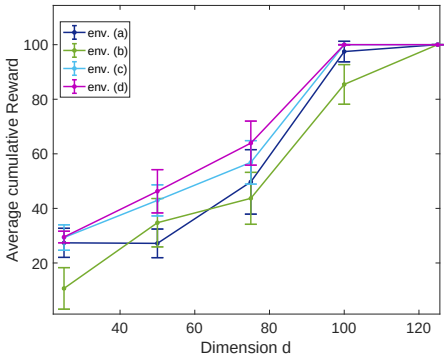


Figure 3. Average cumulative reward after meta-testing using pre-trained state2vec for each of the environment in Figure 1.

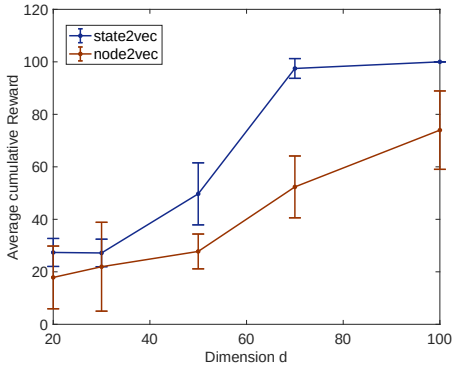


Figure 4. Comparison between node2vec and state2vec on environment (1a) (one goal at the corner).

5. Conclusion

In this work, we relied on an unsupervised approach to address the problem generalisation in RL. We proposed state2vec, an efficient and low-complexity unsupervised framework for learning state representation. We showed that state2vec results in embeddings that capture the geometry of the state space and ensure sample-efficiency during meta-testing. We hope that the proposed idea will pave the way for developing unsupervised meta-reinforcement learning systems that are capable of generalising across tasks.

References

Barreto, A., Dabney, W., Munos, R., Hunt, J. J., Schaul, T., van Hasselt, H. P., and Silver, D. Successor features for transfer in reinforcement learning. In

Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30, pp. 4055–4065. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/350db081a661525235354dd3e19b8c05-Paper.pdf.

Barreto, A., Borsa, D., Quan, J., Schaul, T., Silver, D., Hessel, M., Mankowitz, D., Zidek, A., and Munos, R. Transfer in deep reinforcement learning using successor features and generalised policy improvement. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 501–510, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL http://proceedings.mlr.press/v80/barreto18a.html.

Bellemare, M., Dabney, W., Dadashi, R., Ali Taiga, A., Castro, P. S., Le Roux, N., Schuurmans, D., Lattimore, T., and Lyle, C. A geometric perspective on optimal representations for reinforcement learning. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32, pp. 4358–4369. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/3cf2559725a9fdfa602ec8c887440f32-Paper.pdf.

Borsa, D., Barreto, A., Quan, J., Mankowitz, D. J., van Hasselt, H., Munos, R., Silver, D., and Schaul, T. Universal successor features approximators. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1VWjiRcKX.

Dayan, P. Improving Generalization for Temporal Difference Learning: The Successor Representation. *Neural Computation*, 1993. ISSN 0899-7667. doi: 10.1162/neco.1993.5.4.613.

Du, Y. and Narasimhan, K. Task-agnostic dynamics priors for deep reinforcement learning. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1696–1705. PMLR, 09–15 Jun 2019. URL http://proceedings.mlr.press/v97/du19e.html.

Gelada, C., Kumar, S., Buckman, J., Nachum, O., and Bellemare, M. G. DeepMDP: Learning continuous latent space models for representation learning. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2170–2179. PMLR, 09–15 Jun 2019. URL http://proceedings.mlr.press/v97/gelada19a.html.

Ghosh, D. and Bellemare, M. G. Representations for stable off-policy reinforcement learning. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3556–3565. PMLR, 13–18 Jul 2020. URL http://proceedings.mlr.press/v119/ghosh20b.html.

Grover, A. and Leskovec, J. node2vec. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, pp. 855–864, 2016. ISBN 9781450342322. doi: 10.1145/2939672.2939754. URL http://dl.acm.org/citation.cfm?doid=2939672.2939754.

Higgins, I., Pal, A., Rusu, A., Matthey, L., Burgess, C., Pritzel, A., Botvinick, M., Blundell, C., and Lerchner, A. DARLA: Improving zero-shot transfer in reinforcement learning. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1480–1490, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL http://proceedings.mlr.press/v70/higgins17a.html.

Lagoudakis, M. G. and Parr, R. Least-squares policy iteration. *Journal of Machine Learning Research*, 2004. ISSN 15324435. doi: 10.1162/1532443041827907.

Lehnert, L., Tellex, S., and Littman, M. Advantages and limitations of using successor features for transfer in reinforcement learning. *ArXiv*, abs/1708.00102, 2017a.

Lehnert, L., Tellex, S., and Littman, M. L. Advantages and limitations of using successor features for transfer in reinforcement learning. *CoRR*, abs/1708.00102, 2017b. URL http://arxiv.org/abs/1708.00102.

Madjiheurem, S. and Toni, L. Representation learning on graphs: A reinforcement learning application. In Chaudhuri, K. and Sugiyama, M. (eds.), *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pp. 3391–3399. PMLR, 16–18 Apr 2019. URL http://proceedings.mlr.press/v89/madjiheurem19a.html.

Riedmiller, M. Neural fitted q iteration – first experiences with a data efficient neural reinforcement learning method.

In *Proceedings of the 16th European Conference on Machine Learning*, ECML'05, pp. 317–328, Berlin, Heidelberg, 2005. Springer-Verlag. ISBN 3-540-29243-8, 978-3-540-29243-2. doi: 10.1007/11564096_32. URL http://dx.doi.org/10.1007/11564096_32.

Stachenfeld, K. L., Botvinick, M. M., and Gershman, S. J. The hippocampus as a predictive map. *bioRxiv*, 2017. doi: 10.1101/097170. URL https://www.biorxiv.org/content/early/2017/07/27/097170.

Sutton, R. S., Precup, D., and Singh, S. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artif. Intell.*, 112(1-2):181–211, August 1999. ISSN 0004-3702. doi: 10.1016/S0004-3702(99)00052-1. URL http://dx.doi.org/10.1016/S0004-3702(99)00052-1.