Joint State-Action Embedding for Efficient Reinforcement Learning

Paul J. Pritz
Department of Computing
Imperial College London
London, United Kingdom
paul.pritz18@imperial.ac.uk

Liang Ma
Formerly IBM Research
lianglondon@gmail.com

Kin K. Leung
Department of Computing
Imperial College London
London, United Kingdom
kin.leung@imperial.ac.uk

***Abstract*—While reinforcement learning has achieved considerable successes in recent years, state-of-the-art models are often still limited by the size of state and action spaces. Model-free reinforcement learning approaches use some form of state representations and the latest work has explored embedding techniques for actions, both with the aim of achieving better generalisation and applicability. However, these approaches consider only states or actions, ignoring the interaction between them when generating embedded representations. In this work, we propose a new approach for jointly embedding states and actions that combines aspects of model-free and model-based reinforcement learning. We use a model of the environment to obtain embeddings for states and actions and present an algorithm that uses these to learn a policy. The embedded representations obtained through our approach enable better generalisation over both states and actions by capturing similarities in embedding space and thereby improving convergence speed. The efficacy of our approach is evaluated on a small grid world as an initial example. For future work we plan to apply this methodology to SDC – a domain that tends to suffer from state-action space explosion.**

***Index Terms*—reinforcement learning, embedding, representation learning, machine learning**

I. Introduction

Reinforcement learning (RL) has been successfully applied to a wide range of tasks, including game-based scenarios [1]. However, the application of RL in other domains, such as SDC control, is often hindered by their large state and action spaces. The field of RL can be broadly categorised into model-free and model-based algorithms. Model-free approaches construct a policy directly from interaction with the environment, while model-based algorithms learn a model of the environment and then use it for planning. We propose to combine both approaches by learning a model of the environment, which is consequently used to generate state and action representations to be used in a model-free algorithm. While some form of state encoding as well as embedding techniques have been integrated with RL models in previous work, these either neglect states or actions or use an explicit model to encode one of the two. We propose the use of an environment model to jointly embed states and actions and thereby capture the dependencies between them.

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001.

II. Background

We consider problems that can be expressed as a discrete-time *Markov decision process* (MDP), defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$. $\mathcal{S}$ and $\mathcal{A}$ are the sets of all possible states and actions, referred to as the *state space* and *action space*, respectively. The state, action and reward at time t is denoted by $s_t \in \mathcal{S}$, $a_t \in \mathcal{A}$ and $r_t \in [\mathcal{R}_{min}, \mathcal{R}_{max}]$. γ denotes the reward discounting parameter and P is the state transition function, defined as $\forall s_t, s_t, s_{t+1}, t, P(s_t, a_t, s_{t+1}) := P(s_{t+1}|s_t, a_t)$. A policy is a distribution over all actions for each state, such that $\pi : \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$. The goal of RL is to find an optimal policy π^* that maximises the expected sum of discounted future rewards for a given environment. An optimal policy is any policy in the set of all possible policies Π that satisfies $\pi^* \in arg\,max_{\pi \in \Pi} \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t | \pi]$.

III. Related Work

The idea of embeddings originates from natural language processing, where using the context of a word to generate an embedded representation is a common procedure. In [2], a two-layer neural network is trained to predict a word's context and the weights of the first layer of the network are consequently used as the embedded representation of the words in the corpus. Several pieces of literature have applied the idea of embeddings in a RL context. [3] and [4] train action embedding models similar to the CGRAM model proposed in [5]. They then combine the obtained embeddings with *policy gradient* and *Q-learning* agents. [6] use a model of the environment to obtain an abstract representation of the state that captures some of the dynamics of the environment. This abstract state is then used in a model-free Q-learning agent. In contrast to previous work, we consider both state and action in our proposed embedding model and couple this with a policy gradient-based agent.

IV. Joint State and Action Embeddings

Our proposed algorithm consists of two models that are trained separately. The first component is a model of the environment (Figure 1), trained to predict the next state $\mathcal{S}_{t+1}$, given the current state and action, $\mathcal{S}_t$ and $\mathcal{A}_t$. The environment model can be used for discrete as well as continuous states. For discrete states, we use a softmax output activation and

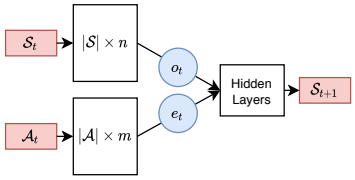

Fig. 1: Architecture of the embedding model.

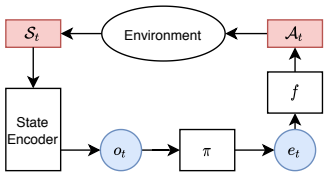

Fig. 2: Architecture of the combined embedding and policy model.

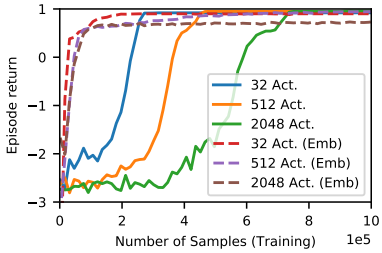

Fig. 3: Performance of our approach compared to a PPO baseline model.

train the model to minimise the negative log likelihood. For discrete states and actions, the weights of the first two layers of the environment model are used as the state and action embeddings respectively. Continuous states or actions are embedded via a forward pass. In the case of continuous states, the embedding model can be trained to minimise a mean sqaured error loss.

The second component can be any policy gradient based RL algorithm. The policy network takes the embedded state representation as an input and outputs a point in the action embedding space. This is illustrated in Figure 2. In order to map this point in embedding space to an executable action, another function $f(e)$ is required. For discrete action spaces, a simple nearest neighbour approach can be used. Otherwise, in the case of continuous action spaces, f can be parameterised by a function approximator.

V. Experiments

As a proof-of-concept, we evaluate the proposed methodology in a simple grid world, given in Figure 4. The agent starts in the bottom-left corner and has to reach a goal state (blue) that leads to a reward of 1. The agent also receives a small penalty of -0.01 for each step and a collision penalty of -0.05 for hitting the walls (shown in gray in Figure 4) or trying to leave the grid. The underlying state is a continuous coordinate, which is discretised in a 40 by 40 grid and represented as a one-hot encoding of the position in the grid. Actions are defined by *actuators*, i.e. displacement vectors that move the agent into a certain direction, with each action being a combination of actuators. Hence, 4 actuators result in 2^4 actions. For the experiments, we use proximal policy optimisation (PPO) to learn the policy and compare the performance with and without embedding states and actions. We evaluate the convergence speed and performance in the same grid world, using different numbers of actuators. For this experiment, we pre-train the embedding module using 20,000 randomly collected samples and keep the embeddings fixed thereafter. The results using embeddings (denoted by *Emb.*) shown in Figure 3 demonstrate the significant improvements in convergence speed that can be achieved with our approach, especially for large action spaces.

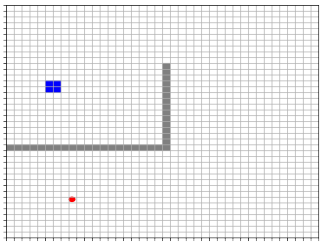

Fig. 4: Grid world environment.

VI. Discussion

We present a new approach for jointly embedding states and actions that can be readily combined with model-free RL algorithms. We demonstrate superior performance, which is the result of better generalisation across states and actions and a reduced dimensionality of their representations, on an exemplary grid world environment.

While we evaluate our approach on a simple experiment, the presented architecture is general. For instance, SDC control would be a suitable testing ground for our methodology as it tends to involve large state-action spaces and is therefore likely to benefit. We plan to study the application to this scenario in future work.

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015. [Online]. Available: http://dx.doi.org/10.1038/nature14236

[2] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'13. Red Hook, NY, USA: Curran Associates Inc., 2013, p. 3111–3119.

[3] Y. Chandak, G. Theocharous, J. Kostas, S. Jordan, and P. S. Thomas, "Learning Action Representations for Reinforcement Learning," *36th International Conference on Machine Learning, ICML 2019*, vol. 2019-June, pp. 1565–1582, jan 2019. [Online]. Available: http://arxiv.org/abs/1902.00183

[4] G. Tennenholtz and S. Mannor, "The natural language of actions," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. Long Beach, California, USA: PMLR, 09–15 Jun 2019, pp. 6196–6205. [Online]. Available: http://proceedings.mlr.press/v97/tennenholtz19a.html

[5] T. Mikolov, Q. V. Le, and I. Sutskever, "Exploiting similarities among languages for machine translation," *CoRR*, vol. abs/1309.4168, 2013. [Online]. Available: http://arxiv.org/abs/1309.4168

[6] V. Francois-Lavet, Y. Bengio, D. Precup, and J. Pineau, "Combined Reinforcement Learning via Abstract Representations," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 3582–3589, sep 2019. [Online]. Available: http://arxiv.org/abs/1809.04506