EXPRESSIVENESS IN DEEP REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Representation learning in reinforcement learning (RL) algorithms focuses on extracting useful features for choosing good actions. Expressive representations are essential for learning well-performed policies. In this paper, we study the relationship between the state representation assigned by the state extractor and the performance of the RL agent. We observe that representations assigned by the better state extractor are more scattered than which assigned by the worse one. Moreover, RL agents achieving high performances always have high rank matrices which are composed by their representations. Based on our observations, we formally define *expressiveness* of the state extractor as the rank of the matrix composed by representations. Therefore, we propose to promote expressiveness so as to improve algorithm performances, and we call it *Expressiveness Promoted DRL*. We apply our method on both policy gradient and value-based algorithms, and experimental results on 55 Atari games show the superiority of our proposed method.

1 INTRODUCTION

Deep reinforcement learning (DRL) algorithms, such as DQN (Mnih et al., 2015), A3C (Mnih et al., 2016), DDPG (Lillicrap et al., 2015), and TRPO (Schulman et al., 2015), have been applied in a range of challenging domains including Atari games (Mnih et al., 2015; Schaul et al., 2015; Van Hasselt et al., 2016), robot locomotion tasks (Schulman et al., 2015; 2017) and the game of Go (Silver et al., 2016; 2017). The combination of RL and high-capacity function approximators such as neural networks holds the promise of automating a wide range of decision making and control tasks.

The DRL model often contains two parts. First, a deep neural network is used to extract stateful information from raw signals, e.g, convolutional neural network for images-based games (Mnih et al., 2015; Vinyals et al., 2017) or recurrent neural network for natural language-based games (Narasimhan et al., 2015; Zhao & Eskenazi, 2016). Without any confusions, we call such raw signals as *observation*, call the extracted stateful information as *state* and consider this deep neural network as *state extractor*. Given the representation of state, a feedforward neural network is further used to select different actions to maximize the potential rewards. From the above description, it is easy to see that the performance of an RL agent depends on two parts. First, it depends on whether the state extractor is good. With a good state extractor, the representation which is a depiction of the observation will retain necessary information for taking actions. Second, it depends on the accuracy of the policy: whether the feed-forward model can correctly take the optimal action given the state.

In this paper, we mainly study the relationship between representations extracted by the state extractor and the performance of the RL agents. We observe that when agents achieve better performances, matrices composed by their representations are approximately higher rank.[1] Firstly, we study representations assigned by two state extractors in trained RL agents. One is better with higher rewards and another is worse with lower rewards. We find that representations assigned by the better extractor are more scattered. Furthermore, given different trajectories that lead to high/low rewards as inputs, matrices of representations from the better extractor are higher rank. Secondly,

[1] Approximately low rank means that most of the singular values for a matrix are close to zero while only little of them have large values, and higher rank corresponds to more large singular values. In this paper, low/high rank refer to approximately low/high rank.

we find changes of the approximate rank are highly consistent with changes of rewards, which also demonstrates the positive correlation between the rank and rewards. These observations motivate us to encourage matrices of representations to be high rank during training.

Based on our observations, we formally define *expressiveness* of the state extractor in reinforcement learning. For given markov decision process (MDP) and the initialization state, we can get a realization of the MDP, i.e., a trajectory. The state extractor extracts representations from observations in this trajectory. Representations compose a matrix, which we call it representation matrix. For given MDP with finite state space, expressiveness in RL for the state extractor model is defined as the rank of the representation matrix. According to the definition, expressiveness in RL is both related to the MDP which generates data and the state extractor model. Based on above experimental studies, we can conclude that higher expressiveness will lead to better performances. As we can see that expressiveness is not easy to calculate because the max operator and the infinite number of columns in representation matrix, so we consider the rank of the representation matrix composed by a mini-batch of representations, which is called empirical representation matrix. Empirical expressiveness in RL is defined as the approximate rank of the empirical representation matrix, i.e., if some singular values are approximately zero, we regard them as zero.

We further propose a novel method *ExP* (**Ex**pressiveness **P**romoted) DRL, which aims improve the expressiveness of the state extractor, so as to promote RL algorithm performances. Based on experimental observations and the definition, the empirical representation matrix is encouraged to be high rank in our method. The idea is implemented by adding a regularization term to the loss, which is computationally efficient and can be applied to multiple kinds of DRL algorithms.

We applied our method to A3C (Mnih et al., 2016) and DQN (Mnih et al., 2015). Evaluation results on Atari games show that our method outperforms the baseline on most of games. Furthermore, we also demonstrate that the expressiveness of the state extractor is significantly enhanced by our proposed ExP DRL.

2 State Extractor Expressiveness

2.1 Experimental Observations

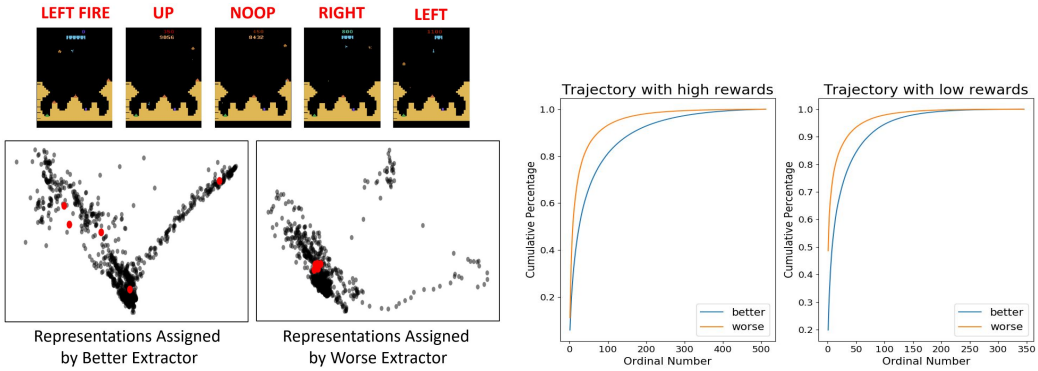
In image classification tasks, many papers show that the representation of images (high-level features from top layers) contains useful and abstract information for decision makings, e.g., predicting the categorical labels (Coates et al., 2013). In reinforcement learning, such high-level features will be used to find better actions in pursuit of larger reward.

However, the representation in reinforcement learning is much harder to learn compared to that in supervised learning problems. In image classification task, e.g., ImageNet, different images with the same label contain the same item and most of the images in different categories contain different items. This makes the neural network can learn discriminative information effectively and the learned features are good with intra-class compactness and inter-class separability (Liu et al., 2016). In reinforcement learning, taking shooter games in Atari as an example, the input images which will be fed into the neural network are similar as most of them contains a group of enemies, bullets, the agent and different objects in the game. Based on similar observations, the neural network has to learn to extract fine-grained local features that contain the positions of enemies, the direction of bullets, the current position of the agent and related objects, which are essential for taking actions.

Therefore, to investigate the characteristic of good representations assigned by the state extractor, we compare two state extractors in RL models with different performances. We find that representations generated good state extractor are more scattered compared with the worse one, and the matrix composed by better representation vectors is higher rank. We observe similar phenomena during training process, which shows that when the matrix formed by representations becomes higher rank, the reward of the RL model become higher. Furthermore, we notice that representations are not always getting more discriminative during training, which also motivates our proposed method.

2.1.1 Comparison between Better and Worse State Extractors

Comparison between representations assigned by two state extractor in differently performed RL models is pictorially depicted in Fig.1. We let two models with different model size but same number



(a) Two-dimensional embedding of the representations and some observations.

(b) Cumulative percentage of singular values.

Figure 1: Comparison between representations assigned by two state extractors in RL models with different performances.

of last hidden layer units play the Atari game Gravitar for 200M frames respectively. After training the larger model achieves 1350 for the game score, while the smaller model gets 0 point because points are hard to obtained in this game.

Fig.1 (a) shows the two-dimensional embedding of the representations in the last hidden layer assigned by two models to game observations in the trajectory played by the better model. Plots are generated by using SVD dimension reduction on the matrix composed by representation vectors. Observations above correspond to red points in embedding figures below. And actions should be taken for these observations are annotated. Points in the embedding figure of the model with higher final reward (i.e., the better model) are more scattered overall. Observations with different actions only have subtle differences on the existence of enemies, the position and direction of the controlled agent. Representations assigned by the better state extractor model are more discriminative and expressive: corresponding representation embedding points of these observations with different actions scatter around the figure of the better model, while they gather on the figure of the worse model. These demonstrate that the better model and can do better in extracting these fine-grained local features, while the worse model cannot distinguish these images well. Representations assigned by the better state extractor is more discriminative.

To get more accurate and solid observations, we regard representations of observations in one trajectory as a matrix, and plot cumulative percentages of singular values of this matrix in Fig. 1 (b). We generate two trajectories with different final reward using the better and the worse model to interact with the environment. Observations in these two trajectories are sent to two state extractor models respectively, and then 4 matrices are obtained. Cumulative percentages are plotted separately according to the input trajectory. It demonstrates that given different trajectories that lead to higher/lower reward, the matrix composed of representations generated by the better model is always higher rank than which generated by the worse model, as the cumulative percentage of singular values from the better model increases to 100% faster in both figures. This phenomenon suggests that regardless of input trajectories, states assigned by the better extractor is more discriminative and expressive comparing with the worse model, because higher rank corresponds to better discrimination. Representations which are not expressive enough cause troubles for decision making. Therefore, representations should be encouraged to be more discriminative and expressive for obtaining good policies.

2.1.2 Trends During Training

In order to investigate whether the found phenomena is common among state extractors in RL models, and also make the observation more convincing, we study representation changes during training in this section. We train a model to play an Atari game for 200M frames. Each time the model is updated with a mini-batch of transitions, representation vectors in the last hidden layer are regarded as a matrix, and singular values of this matrix are recorded. We sort singular values from largest to smallest, and calculate the smallest ordinal number while the cumulative percentage of singular

values is above a certain threshold (i.e., 80%). Large ordinal number means that the matrix composed by representation vectors is high rank, and corresponding representations are discriminative.

Two curves are plotted in Fig.2. The blue one tracks testing rewards, and the yellow one tracks smallest ordinal numbers. It shows that curve trends of testing rewards and smallest ordinal numbers are highly consistent: when smallest ordinal numbers increase/decrease, rewards also become high/low. Same phenomena on more games can also been seen in Fig. 6. This consistency demonstrates that states assigned by the better extractor, which form high rank matrices, are more discriminative and expressive, and holding discriminative representations is necessary for learning good policies. Besides these same conclusions mentioned in last section, we also notice that during training, the smallest ordinal number is not always increase. But representations which are not expressive enough cause troubles for decision making. Thus, the expressiveness of state extractors should be promoted (i.e., formed matrices should be encourage to be high rank) during training.

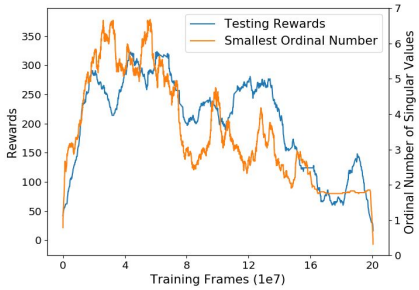


Figure 2: Curves tracking testing rewards and smallest number of singular values during training process. Curves are smoothed.

2.2 Definition of the Expressiveness

Based on previous observations, we formally define a new concept *expressiveness* in RL in this section. Consider an Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, p, \gamma, r)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ denotes the action space, $p : \mathcal{S} \times \mathcal{A} \to \mu(S)$ denotes the transition probability with $\mu(S)$ denoting the space of measures on $\mathcal{S}$, $\gamma \in (0, 1)$ denotes the discount factor and $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function. The goal of reinforcement learning is to learn a policy $\pi(x)$ that maximize the average future reward function $\mathbb{E}_{\pi(x),p(x,a)}[\sum_{k=0}^{\infty} \gamma^k r(x_k, a_k)|x_0]$.

Assume that $\{x_1, \cdots, x_b\}$ is a mini-batch of observations. We use operator $h(x) : \mathcal{X} \to \mathbb{R}^d$ to denote the state extractor. For given observation x_i, we obtain its the representation $h(x_i) = (h_{i1}, \cdots, h_{id})$. The representations of a mini-batch of observation consist a matrix, which we call it representation matrix and denote it as $H = (h(x_1), \cdots, h(x_b))^T = \{h_{ij}\}_{i=1,\cdots,b;j=1,\cdots,d}$, where h_{ij} denotes the j-th representation calculated using the i-th observation in the mini-batch.

Definition 1. ***(Expressiveness in RL)*** *For given MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, p, \gamma, r)$ with finite observation space $\mathcal{S}$ and initial observation x_0, the expressiveness $\mathcal{E}_{\mathcal{M}}(h)$ for extractor model h is defined as the rank of matrix composed by $h(X_t)$, $\forall t$, where $X_t \sim \mathcal{M}$, i.e.,*

$$\mathcal{E}_{\mathcal{M}}(h) = rank\{h(X_1), \cdots, h(X_t), \cdots\}. \tag{1}$$

The defined expressiveness is related to both the MDP $\mathcal{M}$ and the model h. For fixed MDP, the expressiveness is similar to that is defined in supervised learning which is related to the function approximation ability. For fixed h, the expressiveness is related to the MDP $\mathcal{M}$: 1) if the transition $p(x_{t+1}|x_t, a_t)\pi(a_t|x_t)$ of the MDP is not ergodic, i.e., it can only visit a subset of observations, then the matrix $\{h(X_1), \cdots, h(X_t), \cdots\}$ will more possibly be low rank; 2) if the reward is very sparse, the representation matrix will be low rank.

In practical, it is not easy to exactly calculate the rank for a trajectory with infinite length. Thus, we define the following empirical expressiveness.

Definition 2. ***(Empirical expressiveness in RL)*** *For given observations $\{x_1, \cdots, x_b\}$ and representation matrix $H_{b\times d}$, we order singular values of H from large to small as $\sigma_1(H), \cdots, \sigma_{\min\{b,d\}}(H)$. The empirical expressiveness $\mathcal{E}_{b,\epsilon}(h)$ for extractor model h is defined as*

$$\mathcal{E}_{b,\epsilon}(h) = argmin_j \left\{ j : \frac{\sum_{i=1}^{j} \sigma_i(H)}{\sum_{i=1}^{\min\{b,d\}} \sigma_i(H)} > 1 - \epsilon \right\}. \tag{2}$$

Please note that the empirical expressiveness is related to the number of sampled observations, given precision ϵ and the extractor model h. Please note that the rank of the representation matrix is full in practical because the singular values will not be exactly equal to zero. So we introduce ϵ in the definition.

3 ExP (EXPRESSIVENESS PROMOTED) DRL

We propose our method named ExP DRL, which intends to improve the expressiveness of the state extractor, so as to promote DRL algorithm performances. Based on the new concept expressiveness, experimental observations in Sec. 2.1 can be summarized as: state extractors with better expressiveness lead to better policies, and the expressiveness not always becomes better during training. Hence, in order to obtain better policies, we propose to promote the expressiveness of the state extractor, which means encouraging matrices formed by extracted representations to be high rank.

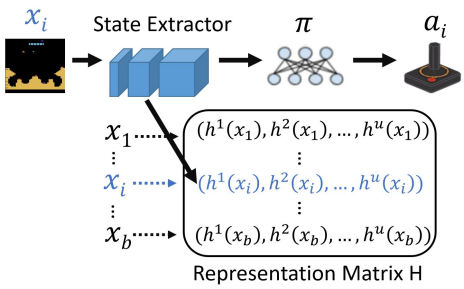


Figure 3: General architecture of a DRL model and its representation matrix H. u is the number of hidden units, and b is the mini-batch size.

To be specific, the general architecture of a DRL model and its representation matrix are shown in Fig. 3. The state extractor extract features from the observation x to generate the representation vector $h(x)$, then these representations are sent to a feed-forward neural network to select the action a. For one observation x_i, a representation vector $h(x_i)$ is generated, and for a mini-batch of observations, $\{h(x_1), ..., h(x_b)\}$ of size b, these vectors compose a representation matrix H. In ExP DRL, this matrix H is encouraged to be high-rank.

To minimize computation costs, we adopt a simple way to encourage high rank representation matrices. A regularization term is added to the policy loss, making matrix H to be high rank. Thus, total loss is denoted as,

$$\mathcal{L} = \mathcal{L}_{policy} + \alpha * \epsilon_{b,\epsilon}(h), \tag{3}$$

where α is the coefficient. Directly optimize the above regularized loss function is not easy because of the argmax operator, thus we propose three types of regularizers which can reflect the scale of $\epsilon_{b,\epsilon}(h)$ to some extent.

Negative Nuclear Norm Minimizing the nuclear norm is usually used as a constrain in low-rank matrix completion problems (Cai et al., 2010). The nuclear norm of a matrix is the sum of all singular values of this matrix, which is defined as

$$||H||_1 = \sum_{i=1}^{\min\{b,d\}} \sigma_i(H), \tag{4}$$

where $\sigma_i(H)$ is the ith singular value of H. Here we try to make the representation matrix to be high rank. Hence, we add a negative nuclear norm to the loss, so $\epsilon_{b,\epsilon}(h) = -||H||_1$.

Max Minus Min Besides enlarging all singular values, a more direct way to improve expressiveness of the state extractor is to reduce the gap between the maximum and the minimum singular value. This gap is defined as,

$$\mathcal{G}(H) = \sigma_{max}(H) - \sigma_{min}(H), \tag{5}$$

and $\epsilon_{b,\epsilon}(h) = \mathcal{G}(H)$.

Condition Number The aforementioned two kinds of rank regularization terms, $||H||_1$ and $\mathcal{G}(H)$, are sensitive with the scale of singular values. So in order to reduce this bias, we also try to use the condition number to be the regularization term.

Condition number is defined as the ratio of the maximum and the minimum singular value:

$$\mathcal{K}(H) = \frac{\sigma_{max}(H)}{\sigma_{min}(H)}. \tag{6}$$

The scale bias can be resolved by the division operator in the condition number. Similar with the *Max Minus Min* term, the condition number should be minimized during training to improve the expressiveness, so $\epsilon_{b,\epsilon}(h) = \mathcal{K}(H)$ here.

In summary, the key improvement in ExP DRL is that we change the loss function to Eq. 3. And we introduce three kinds of regularization terms, which means $\epsilon_{b,\epsilon}(h)$ can be the negative nuclear norm

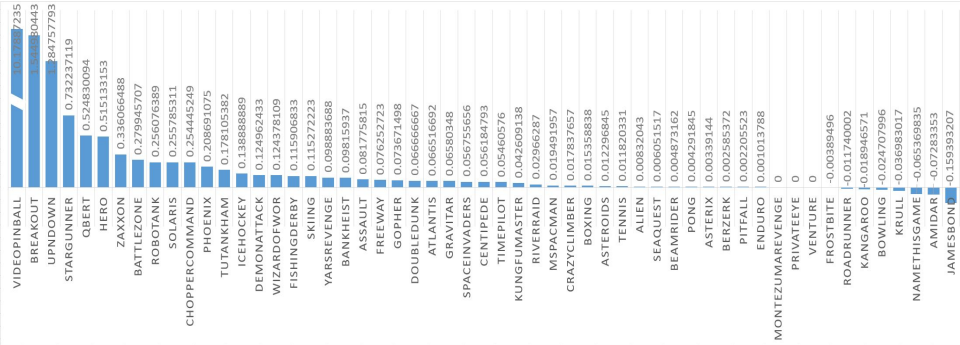


Figure 4: Improvements of our method ExP DRL compared to A3C, using the metric given in Eq. 7.

$-||H||_1$, the max minus min $\mathcal{G}(H)$ or the condition number $\mathcal{K}(H)$. We analyze differences among these 3 terms in experiments.

Note that ExP DRL can be applied to various kinds of DRL algorithms. The expressiveness can be promoted by simply adding a regularization term when the RL model is updated. Excepting these transitions used for updating the RL model, no other data are introduced for encouraging highly expressive representations. Thus, our method can be widely applied to current DRL algorithms, with little extra computational cost.

4 Experiments

The experiment section is designed to answer these questions: (1) Can algorithm performances be improved by applying our proposed expressiveness promoted method? (2) What is the difference between 3 proposed rank regularization terms? (3) Is the expressiveness of the state extractor promoted after using ExP DRL? (4) Can ExP DRL improve performances of multiple kinds of DRL alogirithms? These questions are answered respectively in 4 subsetions below.

4.1 Overall Performances on A3C

4.1.1 Settings

We evaluate our proposed method on 55 Atari games (Bellemare et al., 2013). Atari game learning environment is one of the most popular and challenging RL task because of its high-dimensional and diverse observations. Here we use OpenAI Gym (Brockman et al., 2016) package.

In this section, we use A3C (Mnih et al., 2016) as our baseline, and use the pytorch-a3c package (Kostrikov, 2018), which is also be adopted by Peysakhovich & Lerer (2017), to implement the baseline and our proposed methods. An environment wrapper is utilized to simplify original visual screens, and they are processed to 42×42 gray-scale images. We use a frame-skip of 4 here. The number of processes is set to be 16. Besides, each agent is trained for 200M game frames, and the obtained reward is averaged over 5 runs with different random seeds.

For our proposed method, outputs of the LSTM are taken as representations generated by the state extractor. Each time the A3C model is updated using a mini-batch of transitions, representations of observations in these transitions form the matrix R. To improve expressiveness, this matrix is encouraged to be high rank by adding the regularization term. Expect the coefficient α, other hyper parameters and settings are set as same as the baseline.

4.1.2 Performances

Following the previous work (Wang et al., 2015), we use the measure below to compare the performance of ExP DRL over the baseline.

$$\frac{\text{Score}_{\text{Agent}} - \text{Score}_{\text{Baseline}}}{\max\{\text{Score}_{\text{Human}}, \text{Score}_{\text{Baseline}}\} - \text{Score}_{\text{Random}}}. \tag{7}$$

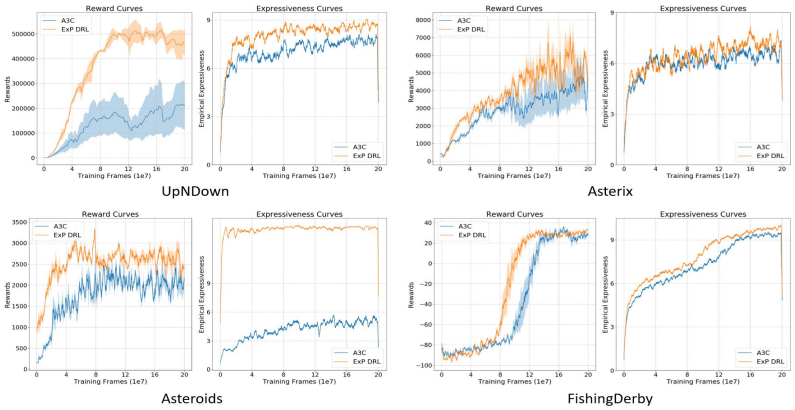


Figure 6: Testing rewards curves (left) and empirical expressiveness curves (right) on 4 Atari games for ExP DRL (yellow) and the baseline (blue).

We summarize the improvement of our method over the A3C baseline in Fig. 4. Among 55 games, ExP DRL outperforms the baseline for 43 games. These demonstrate that encouraging representation matrices to be high rank promotes performances of original algorithms. Improving expressiveness of state extractors benefits learning good policies.

4.2 Three Rank Regularization Terms

To compare different regularization terms, we test each term on the same game with same coefficient α. We empirically set α as 0.01 here. Performances of UpNDown and Qbert are plotted in Fig. 5. In general, the algorithm performance can be promoted largely by applying ExP DRL. And improvements made by Max Minus Min and Condition Number is larger than which made by Negative Nuclear Norm. This is reasonable because that enlarging all singular values may not increase the expressiveness of the state extractor. It is not certain that which singular value gets larger when applying the Negative Nuclear Norm. For example, the expressiveness will decrease when only the largest singular value becomes larger. Instead, the object of Max Minus Min and Condition Number is to reduce the gap between the largest and the smallest singular value, which directly enlarges the expressiveness of state extractors.

We use Max Minus Min and Condition Number with α as 0.01 to cover results of most games in the previous section. However, as the observation type and the intrinsic reward mechanism vary considerably for each game, we also use some other regularization term and coefficient for some of games.

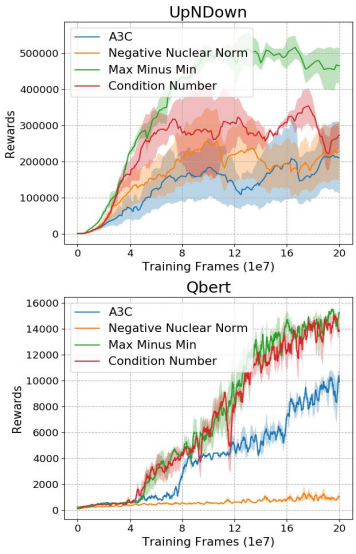


Figure 5: Performances of ExP DRL applying different regularization terms. Shaded areas depict variances.

4.3 Expressiveness Analysis

In this section we analysis the expressiveness of the state extractor learned using the baseline and our proposed method. Curves tracking testing rewards and the empirical expressiveness are plotted in Fig. 6. From testing reward curves we can see that ExP DRL outperforms the A3C baseline on these games. These results show that improving expressiveness significantly benefits policy learning.

Combining testing reward figures and expressiveness figures, we can observe that: (1) the empirical expressiveness of the state extractor is increased by adding the regularization term, which demonstrate

that the proposed method can really enhance the expressiveness. (2) Curve trends of the reward and the expressiveness are highly consistent. This is same as what we observe in Sec. 2.1.2. High rewards and high expressiveness state extractors come out together, which imply that holding good representations is necessary for learning good policies.

4.4 PERFORMANCES ON DQN

Since our method ExP DRL does not contain any algorithm related operations, we investigate whether it can promote performances of another kind of DRL algorithm. Experiments on policy gradient algorithms have been done based on A3C in previous sections, so we choose Q-learning here, and use DQN (Mnih et al., 2015) as our baseline.

Metrics	Average Last Reward
Times Better	20
Times Worse	10

Table 1: Overall performances of 30 Atari games. Average last reward is the average testing reward of the model after training over 5 runs.

We use same network architecture and hyper-parameters as Mnih et al. (2015). Due to the resource and time limitation, we run 30 Atari games (first 30 in alphabetical order) and run each game for 20M frames. We summarize the overall performances of these 30 Atari games in Tab. 1. Results show that ExP DRL outperforms the baseline in most of games. This demonstrates the superiority of ExP DRL over the DQN baseline, and supports our claim that encouraging highly expressive state extractor can promote performances of multiple kinds of DRL algorithms.

5 RELATED WORKS

State representation learning in RL has been studied in many research works. It focus on learning features which can capture useful information for taking good actions (Lesort et al., 2018). In some of proposed methods, auxiliary models, including variational auto-encoder (van Hoof et al., 2016), auto-encoder (Mattner et al., 2012), Generative Adversarial Networks (Donahue et al., 2016; Shelhamer et al., 2016) and some other models (Oh et al., 2017; Weber et al., 2017), are built for refining the representation learning. These models may be designed as a part of the network architecture, and are trained end-to-end (Oh et al., 2017; Pathak et al., 2017; Tamar et al., 2016), or they can also be trained separately and are used for helping decision making (Weber et al., 2017). These auxiliary models help to improve representation learning via completing some certain tasks, such as reconstructing current observation or state (Watter et al., 2015), predicting future observations or states (Oord et al., 2018; Munk et al., 2016), and recovering actions given transitions (Zhang et al., 2018). Some essential information for taking actions can be retained by completing these auxiliary tasks. Different with these methods, none other models need to be built and trained in our method. We improve the expressiveness of the state extractor by simply adding a regularization term.

Task specific prior knowledge or information are utilized to improve representation learning in some proposed methods. For example, detection of moving objects is used in Goel et al. (2018) for better learning video games. Jonschkowski & Brock (2015) proposes to use robotic prior knowledge for robot learning. In some chatting systems (Zhao & Eskenazi, 2016), task-related information, such as extracted named entities, are utilized for dialog state representation learning. For our method, we focus on the expressiveness of the state extractor, which does not contain task-specific informations.

6 CONCLUSIONS

In this paper, we mainly study the relationship between representations extracted by the state extractor and the performance of the RL agents. We observe that when RL agents achieving high rewards, its representations become discriminative, and the representation matrix goes to be high rank. Therefore, we formally define the expressiveness of the state extractor as the rank of the representation matrix. We then further propose a new method ExP DRL, in which algorithm performances are promoted via improving the expressiveness. Experiments of A3C and DQN on 55 Atari games demonstrate that ExP DRL can promote their performances significantly.

REFERENCES

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47: 253–279, 2013.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

Adam Coates, Brody Huval, Tao Wang, David Wu, Bryan Catanzaro, and Ng Andrew. Deep learning with cots hpc systems. In *International Conference on Machine Learning*, pp. 1337–1345, 2013.

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

Vik Goel, Jameson Weng, and Pascal Poupart. Unsupervised video object segmentation for deep reinforcement learning. *arXiv preprint arXiv:1805.07780*, 2018.

Rico Jonschkowski and Oliver Brock. Learning state representations with robotic priors. *Autonomous Robots*, 39(3):407–428, 2015.

Ilya Kostrikov. Pytorch implementations of asynchronous advantage actor critic. `https://github.com/ikostrikov/pytorch-a3c`, 2018.

Timothée Lesort, Natalia Díaz-Rodríguez, Jean-François Goudou, and David Filliat. State representation learning for control: An overview. *arXiv preprint arXiv:1802.04181*, 2018.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *ICML*, pp. 507–516, 2016.

Jan Mattner, Sascha Lange, and Martin Riedmiller. Learn to swing up and balance a real pole based on raw visual input data. In *International Conference on Neural Information Processing*, pp. 126–133. Springer, 2012.

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.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937, 2016.

Jelle Munk, Jens Kober, and Robert Babuška. Learning state representation for deep actor-critic control. In *Decision and Control (CDC), 2016 IEEE 55th Conference on*, pp. 4667–4673. IEEE, 2016.

Karthik Narasimhan, Tejas Kulkarni, and Regina Barzilay. Language understanding for text-based games using deep reinforcement learning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1–11, 2015.

Junhyuk Oh, Satinder Singh, and Honglak Lee. Value prediction network. In *Advances in Neural Information Processing Systems*, pp. 6118–6128, 2017.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning (ICML)*, volume 2017, 2017.

Alexander Peysakhovich and Adam Lerer. Consequentialist conditional cooperation in social dilemmas with imperfect information. *arXiv preprint arXiv:1710.06975*, 2017.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pp. 1889–1897, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Evan Shelhamer, Parsa Mahmoudieh, Max Argus, and Trevor Darrell. Loss is its own reward: Self-supervision for reinforcement learning. *arXiv preprint arXiv:1612.07307*, 2016.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

Aviv Tamar, Yi Wu, Garrett Thomas, Sergey Levine, and Pieter Abbeel. Value iteration networks. In *Advances in Neural Information Processing Systems*, pp. 2154–2162, 2016.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *AAAI*, volume 2, pp. 5. Phoenix, AZ, 2016.

Herke van Hoof, Nutan Chen, Maximilian Karl, Patrick van der Smagt, and Jan Peters. Stable reinforcement learning with autoencoders for tactile and visual data. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pp. 3928–3934. IEEE, 2016.

Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, et al. Starcraft ii: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782*, 2017.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581*, 2015.

Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in neural information processing systems*, pp. 2746–2754, 2015.

Théophane Weber, Sébastien Racanière, David P Reichert, Lars Buesing, Arthur Guez, Danilo Jimenez Rezende, Adria Puigdomenech Badia, Oriol Vinyals, Nicolas Heess, Yujia Li, et al. Imagination-augmented agents for deep reinforcement learning. *arXiv preprint arXiv:1707.06203*, 2017.

Amy Zhang, Harsh Satija, and Joelle Pineau. Decoupling dynamics and reward for transfer learning. *arXiv preprint arXiv:1804.10689*, 2018.

Tiancheng Zhao and Maxine Eskenazi. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. *arXiv preprint arXiv:1606.02560*, 2016.