LEARN EFFECTIVE REPRESENTATION FOR DEEP REINFORCEMENT LEARNING

Yuan Zhan[†,‡], *Zhiwei Xu*[†,‡], *Guoliang Fan*[†✉]

[†]Institute of Automation, Chinese Academy of Sciences;
[‡]University of Chinese Academy of Sciences, School of Artificial Intelligence;
{zhanyuan2020, xuzhiwei2019, guoliang.fan}@ia.ac.cn.

2022 IEEE International Conference on Multimedia and Expo (ICME) | 978-1-6654-8563-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICME52920.2022.9859768

ABSTRACT

Recent years have witnessed an increasing application of deep reinforcement learning (DRL) on video games. While deeper and wider neural networks have played a crucial role in computer vision and natural language processing, such capacity remain under-explored in most DRL works. Under the fact that feature propagation together with large networks contributes to learning a good representation, we propose an end-to-end Large Feature Extractor Network (LFENet) that uses large neural networks with dense connections to train a high-capacity encoder. Even though the increased dimensionality of input is usually thought to result in poor performance for RL agents, we introduce the information bottleneck to alleviate the problem. Finally, we combine LFENet with Proximal Policy Optimization (PPO) algorithm. Through numerical experiments on Atari 2600 video games, we demonstrate our method matches or outperforms state-of-the-art algorithms.

Index Terms— Representation Learning, Reinforcement Learning, Video Game

1. INTRODUCTION

Recently, DRL has achieved remarkable successes on various sequential decision-making problems, especially on video games such as Atari 2600 [1], StarCraft II [2] and so on. It's a convenient and inexpensive way for RL agents to learn complex control policy directly from high-dimensional raw images. However, under sparse reward signals, a large amount of training data is required to learn good representations. For example, model-free methods on Atari games need to take tens of millions of steps to converge to the optimal policy. These problems cause enormous computing resource consumption and time cost, which severely limit DRL algorithms for real-world applications. One natural approach is to learn good representations for pixels to improve sample efficiency.

For the state-input setting, state representation learning (SRL) [3] generally tends to learn a low-dimensional representation, because the proprioceptive states for lots of physical systems are relatively few. Such a representation is expected to contain sufficient statistics for the current observation, and ideally discards all redundant information [4]. Prior work has attempted to learn state representations from image-based observation with auxiliary task, adding an self-supervised objective that provide the reconstruction signal to the feature extractor [5]. But it may lead to suboptimal policy. An interesting question is whether RL agents can benefit from high-dimensional features by using a proper end-to-end architecture?

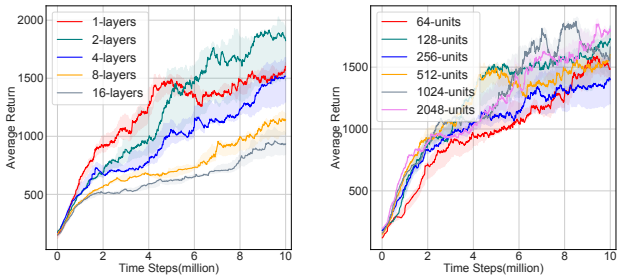

Fig. 1. Training curves of PPO agents with different number of layers (left) and different number of units (right) on AlienNoFrameskip-v4.

Modern computer vision research has a simple intuition that deep and wide neural networks help learn better representations because they increase the search space of possible solutions [6]. In a striking contrast, this intuition may not apply to DRL. Recently reported research [7–9] have shown that DRL algorithms will suffer from unstable training when training with large networks and require more training data. As an example, Fig.1 shows the result of a PPO agent when we fix its unit size to 256 by increasing the number of layers from 1 to 16 layers. Likewise, in Fig.1, we show the effect of increasing dimensionality of representation while the number of layers is fixed to $N^{layer} = 2$. We can see that using deeper networks naively leads to poor performance, and the monotonic improvement with the increase of the dimensionality of representation, until a threshold is reached.

Our contributions are as follows. Firstly, we investigate the problem with deeper and wider network architecture used to learn representations and empirically show that, naively increasing network capacity will decrease perfor-

mance. Secondly, we propose the Large Feature Extractor Network (LFENet) framework, where we train a large network for high-dimensional representaions by modified dense connections. Besides, in LFENet, we introduce the information bottleneck, which is used to constrain the mutual information between the representation variables and observations. Finally, we extend LEFNet to the popular on-policy algorithm PPO. We evaluate it on Atari games, which shows that LEFNet not only outperforms the state-of-the-art algorithms, but also improves the training efficiency.

2. RELATED WORK

Learning Efficient Representations. Generally, we can't obtain the proprioceptive states of video games and must rely on the high-dimensional raw sensory image as observation, so it is key to learn a good representation. One approach to alleviate this problem is incorporating unsupervised or self-supervised auxiliary tasks [5]. Prior research [10–12] utilized autoencoder to learn various constrained representations. Other work had attempted to learn a forward model [13], utilizing a two-step training procedure, where the first encoding from state to representation and using the model that predicts the next state from representation and action, then minimizing the error of prediction and true state. Recently, CPC [14] applied contrastive losses over multiple time steps as an auxiliary task for the convolutional and recurrent layers of RL agents, and it has been extended with future action-conditioning. PBL [15] surpassed these methods with an auxiliary loss of forward and backward predictions in the recurrent latent space using partial agent histories. Where the trend is of increasing complexity in auxiliary networks, these works are also revealed to be correspondingly brittle to hyperparameter settings compared with end-to-end methods.

Learning Very Deep Networks. Deep neural networks achieve great advancements in extracting features from images. However, naively increasing the depth of a feed-forward neural network leads to instability in training due to issues such as vanishing or exploding gradients [16]. To address this problem, ResNet [17] proposes skip connections, which involve an alternate path between layers. DenseNet [18] introduces densely connected block that directly connects each layer to all subsequent layers. Skip-VAEs [19] solve a similar problem of posterior collapse in typical VAE training by adding skip connections to the architecture of the VAE decoder. In the context of DRL, some studies investigate the effect of making networks larger for Atari games using CNN and report that larger networks tend to perform better, but they also become more unstable [20]. To build a large network, OFENet [21] proposes an online feature extractor with intentionally increased input dimensions and demonstrates that larger feature sizes can improve RL performance. Inspired by it, we explore to use larger input for RL agents, while using skip connections for efficient learning.

3. PRELIMINARIES

3.1. Markov Decision Process

We consider the problem of finding the optimal policy for RL agents, which is formalized as a Markov Decision Process (MDP). MDP can be described by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \rho_0, \gamma)$. $\mathcal{S}$ is a finite set of states; $\mathcal{A}$ is a finite set of actions; $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is a transition probability distribution $p(s_{t+1}|s_t, a_t)$, specifying the probability of transitioning from state s_t to s_{t+1} under the action a_t; $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function; $\gamma \in (0, 1)$ is the discount factor. Every episode, the RL agent starts with an initial state $s_0 \sim \rho_0$, receives the state and performs an action according to current policy π at certain time steps, then the agent receives a reward from the environment and transitions to the next state s_{t+1} until the episode terminates. The goal is to find an optimal policy which maximizes the discounted expected return $\mathbb{E}_{\tau_\pi}\left[\sum_{t=0}^{\infty} \gamma^t r\left(s_t, s_t\right)\right]$, where τ is a trajectory.

3.2. Proximal Policy Optimization

Proximal Policy Optimization (PPO) [22] is an on-policy reinforcement learning algorithm that has shown remarkable performance on many tasks. It utilizes a clipped surrogate objective to constrain the updating step in a trust region [23], as:

$$L^{clip} = \hat{\mathbb{E}}_t \left[\min\left(r_t(\theta)\hat{A}_t, \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right)\right], \quad (1)$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta ld}(a_t|s_t)}$ and $\hat{A}_t$ is an estimator of the advantage function at time step t. The clip term prevents $r_t(\theta)$ from moving the outside range$[1-\epsilon, 1+\epsilon]$, which makes the training process more stable. The overall minimization objective is :

$$L_t(\theta) = L^{clip} + \lambda_V L^V - \lambda_H H\left[\pi_\theta\right], \quad (2)$$

where L^V is the loss of value function, $H[\cdot]$ denotes an entropy bonus to encourage exploration.

4. METHOD

Our method is based on two key ideas: adopting well-designed large network architectures for better feature representation learning and introducing the information bottleneck to compress information. In the following, we describe in detail two elements we use to learn better representations for states, then we extend them to PPO.

4.1. Network Architecture

It is well known that deep networks have advantages in extracting features and optimizing representation [6], so we employ them in our network structure. Numerous computer vision reports have provided advanced experience in designing

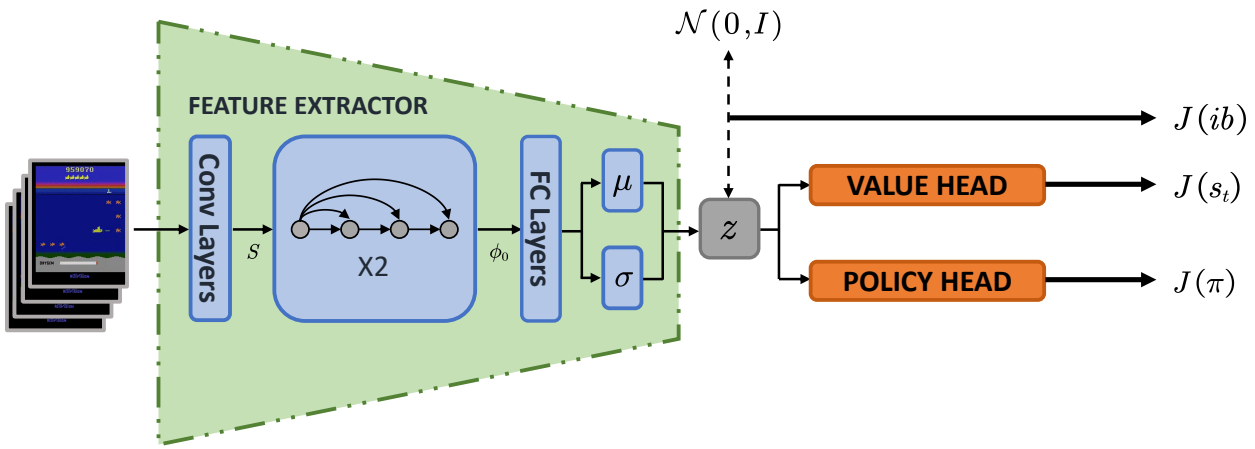

Fig. 2. The overview of LFENet architecture. The inputs are passed to each layer of the neural network through identity mappings and connected with information bottleneck layer to achieve a compressed representation from end-to-end training.

large network structures [17, 18, 24]. Inspired by DenseNet, we propose a slightly modified version of the densely connected mechanism. For the output y of each hidden Multilayer Perceptron layer (MLP), we concatenate the state x to it except the last output linear layer, defined as:

$$y_i = \alpha(W_i[y_{i-1}, h(x)]), \tag{3}$$

where $[x_1, x_2]$ means concatenation, W_i is the weight matrix, α is the activation function, $h(\cdot)$ is an identity mapping, y_i is the output of the the i^{th} layer. To simplify notation, the biases are omitted. If RL agents are directly learning from raw images, we consider the input x to be the output of the convolutional neural network.

The raw sensory observation is mapped as ϕ_0 through above structure, then a bottleneck layer defined in section 4.2 receives the mapping ϕ_0, discards redundant information and generates observation representation z ultimately. The whole process is depicted in Fig.2. RL algorithms take the learned representation as input and compute the optimal policy. Note that our method can be trained *end-to-end*, it learns state representations and trains RL algorithms simultaneously.

4.2. Joint Representation Learning with Information Bottleneck

Several studies have reported that there is a large performance gap when RL agent learns from high-dimensional raw sensory inputs rather than low-dimensional proprioceptive states [11]. It shows that learning a compact representation is crucial to improving RL agent's performance. To incentivize more compressed features, we explore to minimize mutual information between the input and its latent representation. In a Markov process $S \rightarrow Z \rightarrow A$, where S is the input, Z is the learned latent representation of S and A is the predicted actions from S. According to the information bottleneck (IB) principle [25] , we hopes to learn an embedding distribution parameterized as $P(Z|S,\theta)$, such that:

$$P(Z|S;\theta) = \arg\max_{P(Z|S;\theta)} I(Z;A) - \beta I(S;Z), \tag{4}$$

where $I(\cdot;\cdot)$ is mutual information (MI) and $\beta \geq 0$ is a hyperparameter. Eq.4 ensures the predictive power of its latent representation Z on actions A, while omitting task-irrelevant information. However, IB is intractable in general. We approximate estimate it by a variational approximation(VIB), which is applicable to supervised learning tasks. For more details, interested readers are referred to [26]. The VIB can derive variational lower bounds of the two MI terms in the IB objective. We will examine each of these expressions in turn. For the first term $I(Z;A)$:

$$\begin{aligned} I(Z;A) &= \int p(a,z) \log \frac{p(a|z)}{p(a)} dadz \\ &\geq \int p(a,z) \log \frac{q(a|z)}{p(a)} dadz \\ &= \mathbb{E}_{z,a}[\log q(a|z)] + H(A), \end{aligned} \tag{5}$$

where $q(a|z)$ is a variational approximation of $p(a|z)$. It is our policy network, which we will take to compute the optimal policy. The inequality holds because $\mathrm{KL}[p(A|Z)\|q(A|Z)] \geq 0$. The entropy of $H(A)$ can be ignored since it is independent of the optimization procedure. For the second term $\beta I(S;Z)$:

$$\begin{aligned} I(S;Z) &= \int p(z,s) \log \frac{p(z|s)}{p(z)} dzds \\ &\leq \int p(z,s) \log \frac{p(z|s)}{r(z)} dzds \\ &= \mathrm{KL}\left[p(Z|S)||q(Z)\right], \end{aligned} \tag{6}$$

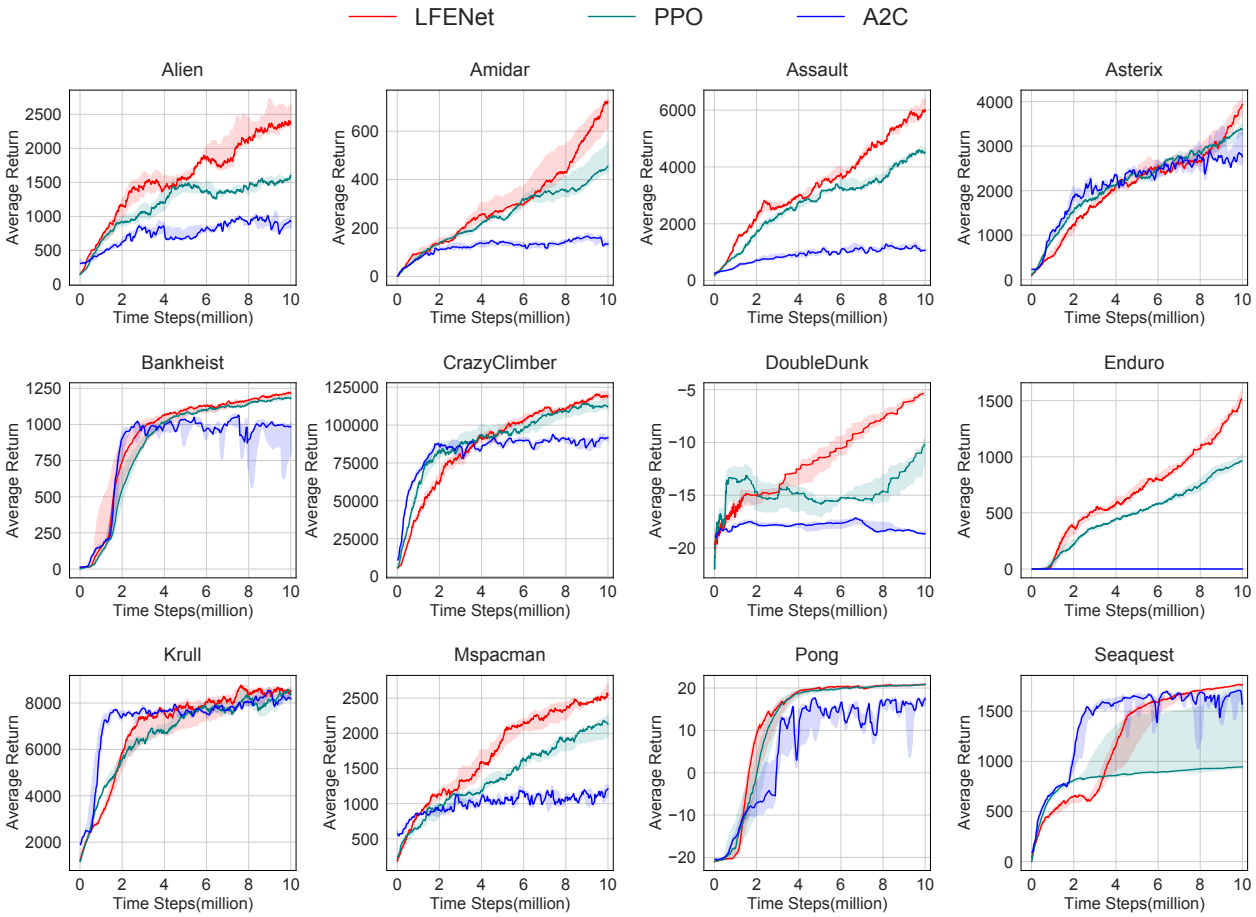

Fig. 3. Learning curves for A2C, PPO, LFENet on 12 Atari games over 10 million time steps. The solid curves show represent the average return across seeds and the shaded regions around each curve represent the standard deviation in measured return.

where $r(z)$ means a variational approximation of the marginal distribution $p(z)$ and the inequality holds because $\mathrm{KL}[p(Z|S)\|r(Z)] \geq 0$.

Now we will introduce how to extend the lower bound L to PPO. The encoder $p_\theta(z|s)$ is trained to map the observation s into a Gaussian distribution: $p_\theta(z|s) = \mathcal{N}(\mu_\theta(s), \sigma_\theta(s))$, where $\mu_\theta(s)$ is the mean and $\sigma_\theta(s)$ is the variance, we use the reparameterization trick to write $p_\theta(z|s) = \mu_\theta(s) + \epsilon\sigma_\theta(s)$ with $\epsilon \sim \mathcal{N}(0, I)$. We set the variational prior $q(z) = \mathcal{N}(0, I)$. Hence a tractable variational approximation to the IB can be defined as:

$$L_{IB} = \mathbb{E}\left[-\log q_\phi(a|z) + \beta \mathrm{KL}\left[p_\theta(z|s)\|r(z)\right]\right], \quad (7)$$

In practice, We find that the latent representation z generated by the mean $\mu_\theta(s)$ performs better than generated by random sample from the distribution, so we take the gradient:

$$\begin{aligned}\nabla L_{\mathrm{VIB}} &= -\mathbb{E}_{\pi_\theta(a|s)}\left[\nabla \log q_\phi\left(a|\mu_\theta(s)\right) A^\pi(s,a)\right] \\ &\quad + \nabla\beta D_{\mathrm{KL}}\left[p_\theta(z|s)\|q(z)\right] \\ &= \nabla_\theta\left(L^{\mathrm{VIB}} + \beta L^{\mathrm{KL}}\right),\end{aligned} \quad (8)$$

Consequently, the overall loss function of the LFENet is:

$$L_t^{\mathrm{VIB}}(\theta,\phi) = L^{\mathrm{VIB}} + \lambda_V L^V - \lambda_H H^{\mathrm{VIB}}\left[\pi_\theta\right] + \beta L^{\mathrm{KL}}, \quad (9)$$

where $H^{\mathrm{VIB}}\left[\pi_\theta\right] = \int p_\theta(s,z) H\left[q_\phi(a|z)\right] dsdz$ and $\mathrm{KL} = \frac{1}{2}\sum_{i=1}^{d} \mu_\theta^2(s) + \sigma_\theta^2(s) - \log\sigma_\theta^2(x) - 1$.

5. EXPERIMENTS

In this section, we combine LFENet with on-policy algorithm PPO [22] on some popular Atari games. Through the experiments, we try to answer the following questions:

- How does LFENet perform compared with the baselines in terms of performance and efficiency?
- Can LFENet learn a compact representation?
- What leads to the performance gain obtained by LFENet?

5.1. Evaluation on Atari Games

To enable fair comparison between the standard baselines and LFENet, We evaluate all agents on 12 challenging Atari games. In LFENet, we use the 2-layer convolution encoder from [11] and 2-layer modified dense block as feature extractor, the dimensionality increments of z from their observations are 1024 in all experiments and the regularization coefficient β takes 1e-6. In our experiments, except for the new parameters introduced by LFENet, the other hyperparameters all follow OpenAI-Baselines. Each agent is trained for 10 million.

In Atari games, We compare the proposed LFENet with **A2C** [27] and **vanilla PPO**. For a reliable comparison, we run 3 random seeds and eight agents work simultaneously for each seed. The solid curves represent the average return across seeds and the shaded regions around each curve represent the standard deviation in measured return. Fig.3 shows LFENet converges to the highest average return in most environments. Note that PPO with LFENet outperforms vanilla PPO on all environments in terms of final performance and sampling efficiency.

5.2. Visualization of Mutual Information

The Mutual Information Neural Estimator (MINE) [28] proposed a method to estimate the MI by introducing a statistics network to transform the calculation of MI to an optimization problem. In light of this, we visualize the MI between input state s and the learned representation z to figure out whether the information bottleneck contributes to learning a compact representation. We use the authors' implementation and the MINE is defined as:

$$\begin{aligned} I(S;Z) &\geq I_{\Theta}(S;Z) \\ &= \sup_{\theta\in\Theta} \mathbb{E}_{\mathbb{P}_{SZ}}\left[T_\theta\right] - \log\left(\mathbb{E}_{\mathbb{P}_S\otimes\mathbb{P}_Z}\left[e^{T_\theta}\right]\right), \end{aligned} \tag{10}$$

where $\mathbb{P}_{SZ}$ means the joint probability distribution and $\mathbb{P}_S \otimes \mathbb{P}_Z$ means the product of the marginals, the optimal functions take the form $\mathbb{E}_{\mathbb{P}}\left[T^*\right] = I(X;Z)$. Please refer to [28] for more details.

Fig.4 shows the mutual information estimation between s and z on the AlienNoFrameskip-v4. We can see that the feature extractor network first stores lots of feature information by increasing the MI, then compresses the input to efficient representation. Note that the architecture with IB is proved to have better information extraction capability due to their smaller MI.

5.3. Ablation Study

In order to verify that the proposed LFENet can learn a compact, high-dimensional representation to improve performance. We perform an ablation study by comparing the performance while gradually removing two key components of

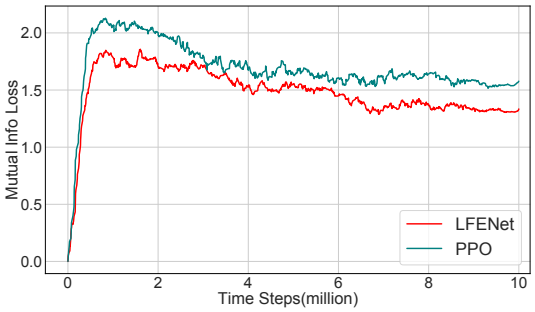

Fig. 4. Visualizing the Mutual Information(MI) during the training on AlienNoFrameskip-v4.

LFENet, namely: modified DenseNet and IB to show what component contributes to the improvement. In addition to this, we compare the result with modified ResNet [17], which has skip connections: $y_i = f_i^{\text{res}}(y_{i-1}) + y_{i-1}$, where y_i means the output of the the i^{th} layer, and f_i^{res} is a residual module architecture. Fig.5 shows the training curves of average return on AlienNoFrameskip-v4 environment.

Only IB replaces the modified DenseNet defined in Sec.4.1 with standard MLP architecture. As dense connections enable information propagation in large networks, we believe that using a large network with modified DenseNet to learn a high-dimensional representation contributes to improving performance.

Only MD removes information bottleneck. The much lower return shows that utilizing information bottleneck is essential to achieve high performance. Since the learned representation contains lots of redundant information. Information bottleneck effectively contributes to learn a compressive representation.

Add Res experiments with a ResNet-like MLP, and the result shows that the modified DenseNet achieves higher scores than other connectivity architectures.

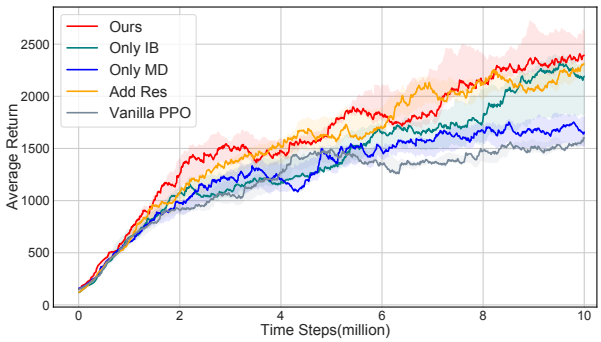

Fig. 5. Training curves of the derived methods of PPO on AlienNoFrameskip-v4. This shows that each element does contribute to the performance gain.

6. CONCLUSION

For DRL to be effective on video games, we need to learn good state representations from image-based observations. Prior work usually prioritizes representation learning where learned features are in low dimension. In this paper, we present an end-to-end Large Feature Extractor Network (LFENet), exploring to learn effective higher-dimensional representation for raw sensory input. In LFENet, we leverage information propagation in the hidden layer, thus improving the performance and sample efficiency. Besides, we introduce the information bottleneck to discard redundant information. We combine LFENet with PPO and conduct experiments on Atari 2600 games. Our experimental results demonstrate that LEFNet can achieve state-of-the-art performance in most environments.

7. REFERENCES

[1] Volodymyr Mnih, Koray Kavukcuoglu, and David Silver et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.

[2] Oriol Vinyals and Igor Babuschkin et al, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, pp. 1–5, 2019.

[3] Timothée Lesort and Natalia Díaz Rodríguez et al, "State representation learning for control: An overview," *Neural networks : the official journal of the International Neural Network Society*, vol. 108, pp. 379–392, 2018.

[4] Manuel Watter and Jost Tobias Springenberg et al, "Embed to control: A locally linear latent dynamics model for control from raw images," in *NeurIPS*, 2015.

[5] Max Jaderberg and Volodymyr Mnih et al, "Reinforcement learning with unsupervised auxiliary tasks," *ArXiv*, vol. abs/1611.05397, 2017.

[6] Alex Krizhevsky and Ilya Sutskever et al, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, pp. 84 – 90, 2012.

[7] Peter Henderson and Riashat Islam et al, "Deep reinforcement learning that matters," in *AAAI*, 2018.

[8] Joshua Achiam, Ethan Knight, and P. Abbeel, "Towards characterizing divergence in deep q-learning," *ArXiv*, vol. abs/1903.08894, 2019.

[9] Samarth Sinha and Homanga Bharadhwaj et al, "D2rl: Deep dense architectures in reinforcement learning," *ArXiv*, vol. abs/2010.09163, 2020.

[10] John-Alexander M. Assael and Niklas Wahlstrom et al, "Data-efficient learning of feedback policies from image pixels using deep dynamical models," *ArXiv*, vol. abs/1510.02173, 2015.

[11] Denis Yarats and Amy Zhang et al, "Improving sample efficiency in model-free reinforcement learning from images," in *AAAI*, 2021.

[12] Samuel Alvernaz and Julian Togelius, "Autoencoder-augmented neuroevolution for visual doom playing," *2017 IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 1–8, 2017.

[13] Deepak Pathak and Agrawal et al, "Curiosity-driven exploration by self-supervised prediction," in *ICML*, 2017.

[14] Aäron van den Oord, Yazhe Li, and Oriol Vinyals, "Representation learning with contrastive predictive coding," *ArXiv*, vol. abs/1807.03748, 2018.

[15] Zhaohan Daniel Guo and Bernardo Ávila Pires et al, "Bootstrap latent-predictive representations for multitask reinforcement learning," in *ICML*, 2020.

[16] Nicol N. Schraudolph, "Centering neural network gradient factors," in *Neural Networks: Tricks of the Trade*, 2012.

[17] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[18] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger, "Densely connected convolutional networks," in *CVPR*, 2017.

[19] Adji B Dieng, Yoon Kim, Alexander M Rush, and David M Blei, "Avoiding latent variable collapse with generative skip models," in *AISTATS*, 2019.

[20] Hado Van Hasselt, Arthur Guez, and David Silver, "Deep reinforcement learning with double q-learning," in *AAAI*, 2016.

[21] Keita Ota and Tomoaki Oiki et al, "Can increasing input dimensionality improve deep reinforcement learning?," in *ICML*, 2020.

[22] John Schulman and Filip Wolski, "Proximal policy optimization algorithms," *ArXiv*, vol. abs/1707.06347, 2017.

[23] John Schulman and Sergey Levine et al, "Trust region policy optimization," in *ICML*, 2015.

[24] François Chollet, "Xception: Deep learning with depthwise separable convolutions," in *CVPR*, 2017.

[25] Naftali Tishby, Fernando C Pereira, and William Bialek, "The information bottleneck method," *ArXiv*, vol. physics/0004057, 2000.

[26] Alexander Amir Alemi and Ian S. Fischer et al., "Deep variational information bottleneck," *ArXiv*, vol. abs/1612.00410, 2017.

[27] Volodymyr Mnih and Adrià Puigdomènech Badia et al, "Asynchronous methods for deep reinforcement learning," in *ICML*, 2016.

[28] Mohamed Ishmael Belghazi et al, "Mutual information neural estimation," in *ICML*, 2018.