Procgen Benchmarkにおける汎化性能を高める強化学習

徐 凡超[1,a)]　金子 知適[1,,b)]

概要：強化学習エージェントが高度な多様性を持つ現実環境に対応できるようにするために，ロバストなポリシーを学習する必要がある．本研究は Procgen Benchmark という高度な多様性を持つテスト環境を題材に，汎化性能を高める学習手法の開発が目標である．入力データは画像形式のみとする．観測空間 (画像データ) を潜在表現に変換することができる VAE と，強化学習手法の IMPALA と組み合わせして，同時に学習させることを行った．Procgen Benchmark の限定された数の level set (num_levels=500) で学習して，zero-shot に未知の level set で汎化性能をテストする．提案手法と IMPALA (baseline) の比較実験を行った．提案手法の VAE 部分は強化学習が収集したデータを学習し，入力画像を復元できた．一方，全体的に顕著な性能向上が観測出来なかった．原因について分析して，改善策を議論した．

キーワード：変分自己符号化器，強化学習，汎化性能，Procgen Benchmark，プロシージャルコンテンツ生成

Combining reinforcement learning and VAE to improve generalization in Procgen Benchmark

Fanchao Xu[1,a)]　Tomoyuki Kaneko[1,,b)]

Abstract: For reinforcement learning agents to have an ability to handle real-world tasks with high diversity, agents must be able to learn a robust policy. This study focuses on Procgen Benchmark, an environment that is designed to measure both sample sufficiency and generalization in reinforcement learning. In this paper, we evaluate a method that combines reinforcement learning and VAE, training these two models simultaneously. Our agent that only takes images as input, is trained in a limited set of levels and tested in zero-shot samples (the full distribution of levels). Reinforcement learning can create trainning data efficiently for the VAE part of our algorithm to learn. The VAE part in our method can reconstruct input images successfully. Total performance of RL and VAE, however, we can't observe a clear improvement in generalization scores. We analyze this problem and propose possible solutions as future work.

Keywords: VAE，Reinforcement learning, Generalization, Procgen Benchmark, Procedural content generation

1. はじめに

近年，強化学習の研究者が色々なドメインで素晴らしい成果を収めた．複雑な伝統的なボードゲーム，囲碁で人間を倒した [1]．E スポーツの競技項目としてのスタークラフト II でプロ選手を倒した [2]．しかし，強化学習エージェントが学習段階であったことがない環境だと，例えそれは相似の環境だとしても，性能が顕著に悪くなる場合がある [3]．

汎化性能は強化学習研究の挑戦的な課題であり，現実環境が高度な多様性を持つことを考えると，それに対応可能なロバストなポリシーを学習できる学習手法の開発は有意義である．本研究の目的は Procgen Benchmark という高度な多様性を持つテスト環境を題材に，汎化性能の高い学習手法の開発することである．具体的には画像データを潜在表現に変換することができる Variational AutoEncoder (VAE) [4] と，強化学習手法の Importance Weighted Actor-Learner Architec-ture (IMPALA) [5] の組み合わせを扱う．

[1] 東京大学大学院情報学環
Interfaculty Initiative in Information Studies, the University of Tokyo
[a)] xu-fanchao@g.ecc.u-tokyo.ac.jp
[b)] kaneko@acm.org

2. 関連研究

2.1 Variational AutoEncoder

文献 [6] に従って VAE を紹介する．Variational AutoEncoder (VAE) では，潜在変数を含むモデルを考える．潜在変数は，観測できないものでしたがってデータセットには含まれない．本稿では z で，潜在変数をあらわす．VAE は，通常複雑な観測空間 x-space と相対的に簡単な潜在空間 z-space の確率的な対応関係を学習する．

データセットを X で表現し，その確率モデルは $p(X)$ と表現する．訓練データ X から潜在変数 z への写像 (ニューラルネットワーク) が encoder $q_\psi(z|x)$ と呼ばれる．Encoder は $q_\psi(z|x)$ を計算困難な z における事後分布 $p_\theta(z|x)$ に近似する方向に学習する．

$$q_\psi(z|x) \approx p_\theta(z|x) \tag{1}$$

z を encoder で直接に出力することではなく，$q_\psi(z|x) = N(\mu(x), \sigma^2(x))$，$q_\psi(z|x)$ の平均値 μ と標準偏差 σ を出力する．z を $N(\mu(x), \sigma^2(x))$ からサンプリングできる．しかし，ニューラルネットワークのパラメータを更新するために，確率的勾配降下法と誤差逆伝播を用いる時，ランダム変数である z の誤差逆伝播ができない．故に，ランダムノイズ $\epsilon \sim N(\mathbf{0}, I)$($\epsilon$ は n 次元のベクトルだとすると，$\mathbf{0}$ は n 次元のベクトルで，I は $n \times n$ の単位行列である．) を使って，z のランダム性を維持して，再構成する (reparameterization trick)．それは

$$z = \mu + \sigma \odot \epsilon \tag{2}$$

符号 $\odot$ はアダマール積である．本稿では z, μ, σ, ϵ が全部同じ次元のベクトルで用いられている．

逆に，z から X に復元する部分 (ニューラルネットワーク) が decoder $p_\theta(x|z)$ と呼ばれる．ψ と θ それぞれは encoder と decoder のパラメータである．

全体的な目標は観測データの log-likelihood を最大化することである．

$$\log p_\theta(x) = L(\psi, \theta; x) + D_{KL}[q_\psi(z|x)||p_\theta(z|x)] \tag{3}$$

$D_{KL}[q_\psi(z|x)||p_\theta(z|x)] \geq 0$ であるので，

$$\log p_\theta(x) \geq L(\psi, \theta; x) \tag{4}$$

が成り立つ．

式 3 を式変換で，VAE の目標関数 (観測データの log-likelihood の)evidence lower bound (ELBO)

$$L(\psi, \theta; x) = -D_{KL}[q_\psi(z|x)||p(z)] + E_{q_\psi(z|x)}[\log p_\theta(x|z)] \tag{5}$$

になる．$p(z)$ は潜在変数 z の事前分布を表す．本研究中は $p(z)$ を 64 次元の各次元が独立した標準正規分布とする．

ELBO を最大化することは式 5 の項 $D_{KL}[q_\psi(z|x)||p(z)]$ を小さくして，式 5 の項 $E_{q_\psi(z|x)}[\log p_\theta(x|z)]$ を大きくすることを意味する．一つ目の項は $q_\psi(z|x) \approx p(z)$ の目標を達成するためである．二つ目の項は $q_\psi(z|x)$ に関するデータ x の対数尤度 $\log p_\theta(x|z)$ の期待値を最大化する．

本研究は VAE が observation space (高次元のデータ) を潜在表現 (低次元の特徴) に変換して強化学習の入力に用いることで，汎化性能を高めると期待して VAE を採用した．

2.2 潜在空間での強化学習

本稿と同じ VAE を強化学習に使った研究である文献 [7] は，現実世界のロボット操作問題に取り組んだ．文献 [7] で”the degree of disentanglement” (画像の特徴を潜在空間で分離する) を重視する β-VAE [8] と goal-conditioned reinforcement learning を組み合わせして，goal を潜在空間でサンプルしなおして，一つのエピソードから沢山学習する手法が提案した．

強化学習の探索を加速するための手法の一つは reward に bonus を加える (例えば，state における entropy など) ことである．文献 [9] で低次元の潜在変数における entropy 最大化は，高次元の state でのエントロピー最大化より効果的であるされている．次に，文献 [10] で潜在空間における surprise minimizing reward (SM reward) を，強化学習 (PPO [11]) の reward に加えることで，Procgen Benchmark のような procedurally generated 環境でロバスト性を持つことと予想した．文献 [10] での比較実験はオリジナルな PPO，PPO+状態空間における SM reward，PPO+潜在空間における SM reward，三つの手法の性能を比較した．CoinRun と BossFight での実験，両方は easy の設定で，実験結果により，提案手法は多少の overfitting 現象が観測された．

2.3 IMPALA

Importance Weighted Actor-Learner Architecture [5] (IMPALA) は学習効率と安定性が優れる手法である．

IMPALA は actor-learner の構造で，複数の actor がお互いに独立して，強化学習環境を実行して，効率的に trajectories (observation，action，reward) を収集する．集めた trajectories を learner に送り，learner で勾配を計算して，ネットワークを更新する．Actor が次の実行を始まる前に，learner から最新のネットワークパラメータを複製する．

文献 [5] で，DeepMind Lab [12] と the Atari Learning Environment [13] での評価実験の際に，small architecture と large architecture 2 つのネットワーク構造が用いられた．Procgen Benchmark の論文 [3] では，IMPALA のネットワーク構造は性能と計算資源のバランスを取る構造と評価されている．本研究は large architecture を強化学習の

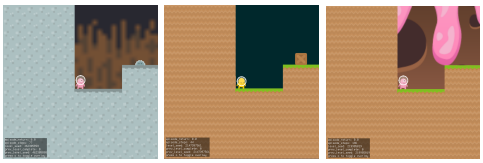

図 1　CoinRun の例: 訓練時に見たことのない表現 (色等) の level でも，安定して振る舞うことが学習の目標
Fig. 1　Different levels from the environment CoinRun.

ネットワーク構造として用いる．

2.4　環境: Procgen Benchmark

Procgen Bnechmark は多様性を持つ環境でる．そこで良い成績をとるためにはエージェントは汎化能力の高い手法で学習する必要がある [3]．Procgen Benchmark は 16 種類の gym environment [14] で構成されでいる．人間がコンテンツを手動で設計する代わりに，アルゴリズムを設計して，変数でゲーム素材と背景の選択を制御して，コンテンツを自動生成できる手法 procedural content generation が用いられている．

Procedural content generation は game-specific details をランダムで生成する (例えば，環境内エンティティの位置と生成するタイミングなど)．故に，特定な trajectories に対応する policy を学習する強化学習エージェントと，環境のバリエーションに対してロバストな policy を学習する強化学習エージェントは Procgen Benchmark で大きな性能の違いが観測できるはずである．

Level とは一つの環境の実例の一種である．**図 1** に，環境 coinrun の 3 levels，それぞれの 1 フレームの画面を示す．環境パラメータの *num_levels* と *start_level* を設定すると，levels の集合が特定できる．これによって，トレーニング段階とテスト段階が異なる levels の集合で実行することが可能になる．難易度を設定するためのパラメータ *distribution_mode* は easy と hard の値をとる．その意味は各 level はハイバリアンスを持って，hard 設定の level は必ず easy 設定の level より難しいとは言い切れないが，hard 設定の level 集合全体的には easy 設定の level 集合より難しいことである．

全部の環境は共通の離散の 15 dimensional action space と $64 \times 64 \times 3$ の RGB 画像の observation space を持っている．

3.　提案手法

VAE は，画像入力をより構造化された表現に変換して，強化学習のために提供することが出来る．この潜在表現が有用な features として，強化学習に入力して，汎化性能を高めることを期待している．一般的な VAE は事前に用意したデータセットで，訓練して，テストする．本研究では文献 [9] [10] と同様に，VAE の学習を強化学習と同時に進めて，強化学習が収集したデータを VAE の訓練データにすることで，VAE の学習で利用可能なデータの多様性と数を改善することが出来る．従って本研究は VAE と IMPALA を同時に学習して，環境の observation と共に，状態の潜在表現を IMPALA への入力として，強化学習を行うモデルを用いる．

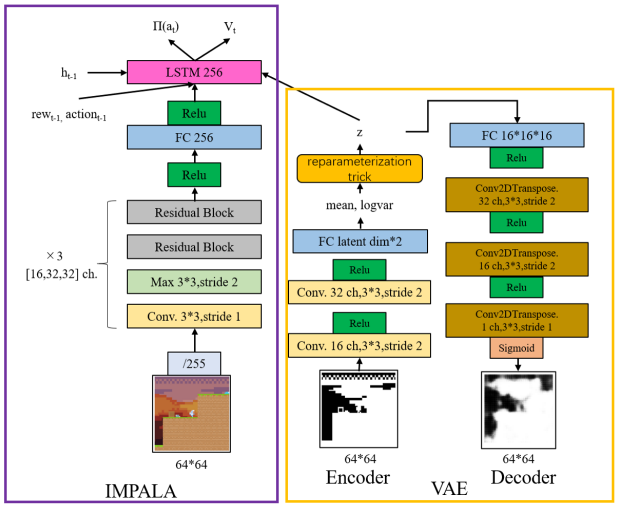

図 2　提案手法 IMPALA と VAE の組み合わせのモデル構造
Fig. 2　Model Architecture.

図 2 で，提案のモデル構造とネットワークの詳細を示す．IMPALA 部分のネットワーク構造は文献 [5] の large architecture を用いた．VAE と IMPALA，この 2 つのモデルは別々に loss function (obejective function) を計算して，独立に更新する．この方法はメモリーの消費が大きいため，VAE の入力は前処理により元の RGB 画像 ($64 \times 64 \times 3$) をグレイスケール画像にして，それから 01 のバイナリの画像 (64×64) に変換して用いた (各ピクセルは独立にベルヌーイ分布で決まり，それぞれの母数 (文献 [6] の式 1.18 の p_j) は潜在変数 (z) と対応しているとモデル化する)．IMPALA のネットワークの入力は $64 \times 64 \times 3$ そのままで，$[0, 255]$ を $[0, 1]$ にした．IMPALA 側では，FC 256 の出力と，サンプリングした潜在変数 z (オリジナルな IMPALA との違い)，1 step 前の reward rew_{t-1}，1 step 前の action $action_{t-1}$ と合わせて LSTM に入力する．本研究で，z には 64 次元のベクトルを用いた．

4.　実験

本章では，提案手法と baseline としての IMPALA (large architecture を用いる) との比較実験で，トレーニング段階とテスト段階の性能を評価する．本研究は以下 4 つの Procgen Benchmark 環境を扱う．

1. **CoinRun:**　ゴールは地図の一番右にあるコインを取ることである．プレーヤーの開始位置は地図の一番左

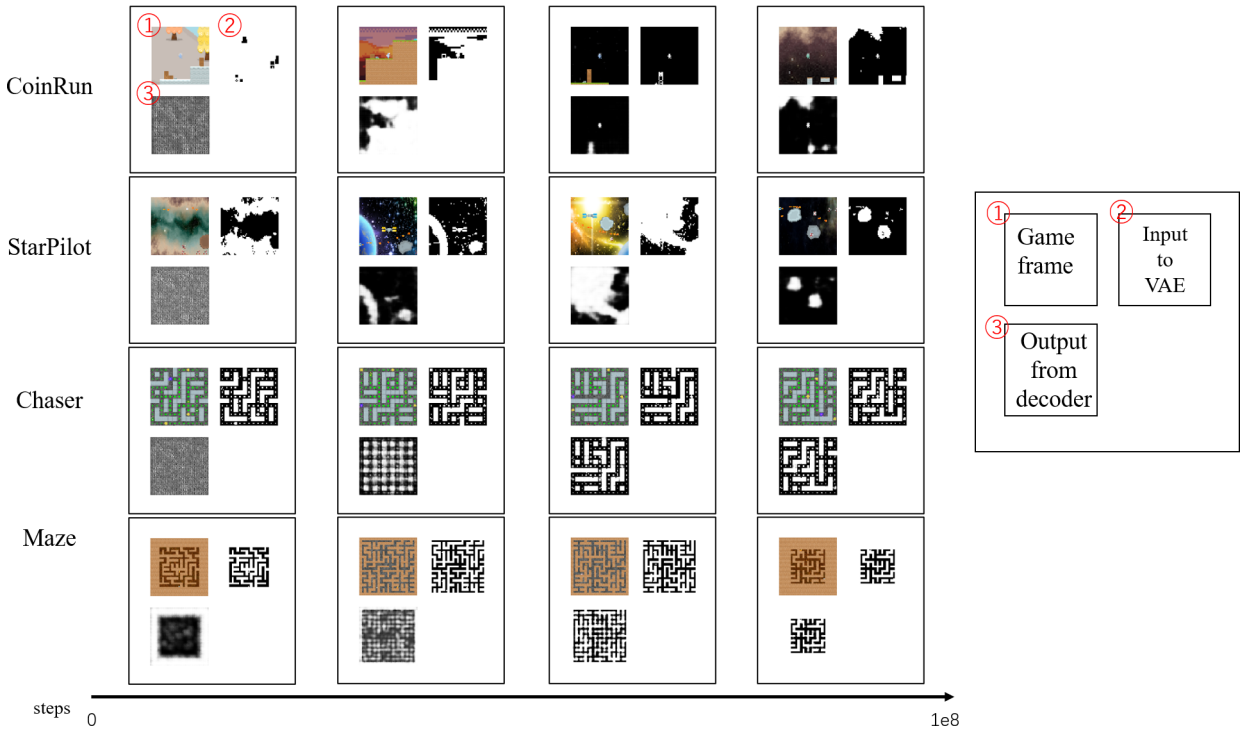

図 3　トレーニング段階で，VAE の可視化性能評価
Fig. 3　Visualization VAE performance evaluation.

で，コインまでの道にランダムに生成された障害物，敵，罠がある．

2. **Starpilot:**　宇宙戦艦を操作して，敵と撃ち合う横スクロールシューティングゲームである．より多くの敵を倒すことが目的で，敵に撃たれた時や，障害物と接触した時などの状況はゲームオーバーとなる．
3. **Chaser:**　Atari game の「MsPacman」と類似するゲームである．迷路にある緑の玉を全部回収することが目標で，迷路に敵が存在して，敵と接触するとゲームオーバーになる．Level 毎に，迷路の構造がランダムに生成される．
4. **Maze:**　プレーヤーがネズミを操作して，迷路の終点にあるチーズを取ることが目標である．Level 毎に，迷路の構造がランダムに生成される．

これらを用いて，IMPALA (baseline) と IMPALA と VAE の組み合わせ (提案手法) の比較実験を行う．文献 [3] の実験では，性能と計算資源の消費のバランスを取るために，500 levels ($num\_levels = 500$, $distribution\_mode = hard$) で学習し，full distributions of levels ($num\_levels = 0$, $distribution\_mode = hard$) でテストを行っている．それにならって，本研究は同じ設定で実験を行う．実験で使用したソフトウェアは，python 3.6.9 と，procgen 0.10.4, gym 0.17.2，gym3 0.3.3，numpy 1.19.2，tensorflow-gpu 1.14.1 (CUDA 10.2)，tensorflow-probability 0.7.0 である．IMAPLA と VAE はすでに open-source[*1][*2]で利用可能であるため，本研究はそれらのソースコードに基づいて実装した．

性能評価はトレーニングとテスト，この 2 つの段階で評価する．

4.1　トレーニング段階

4.1.1　VAE 性能評価

VAE 部分の学習は強化学習と同時に進めているので，強化学習が止まらない限り，VAE が使える訓練データの数が増え続ける．**図 3** で，4 つの環境それぞれの 0 *steps* から 10^8 *steps* まで時間列の順で，一定頻度で記録した，① game frame，② game frame を前処理した VAE の encoder の入力，③ VAE の decoder 部分が復元した可視化結果を示す．また，訓練段階の $-ELBO$ を**図 5** で示す．

図 3 の可視化結果から，全体的に評価すると，強化学習で収集した訓練データを前処理した後，encoder の入力にして，decoder はその入力を確実に復元できた．VAE の目標関数 ELBO が最大化する ($-ELBO$ を最小化する) 方向に学習された．個別に評価すると，CoinRun と StarPilot の結果は比較的に悪い．図 3 で CoinRun と StarPilot の行を見ると，各図の②encoder の入力は① game frame と比べて，前処理した後，元画像の多くの情報が失われている (例えば，steps 順で CoinRun の 1 番目の図だと，game

*1　https://github.com/deepmind/scalable_agent
*2　https://www.tensorflow.org/tutorials/generative/cvae

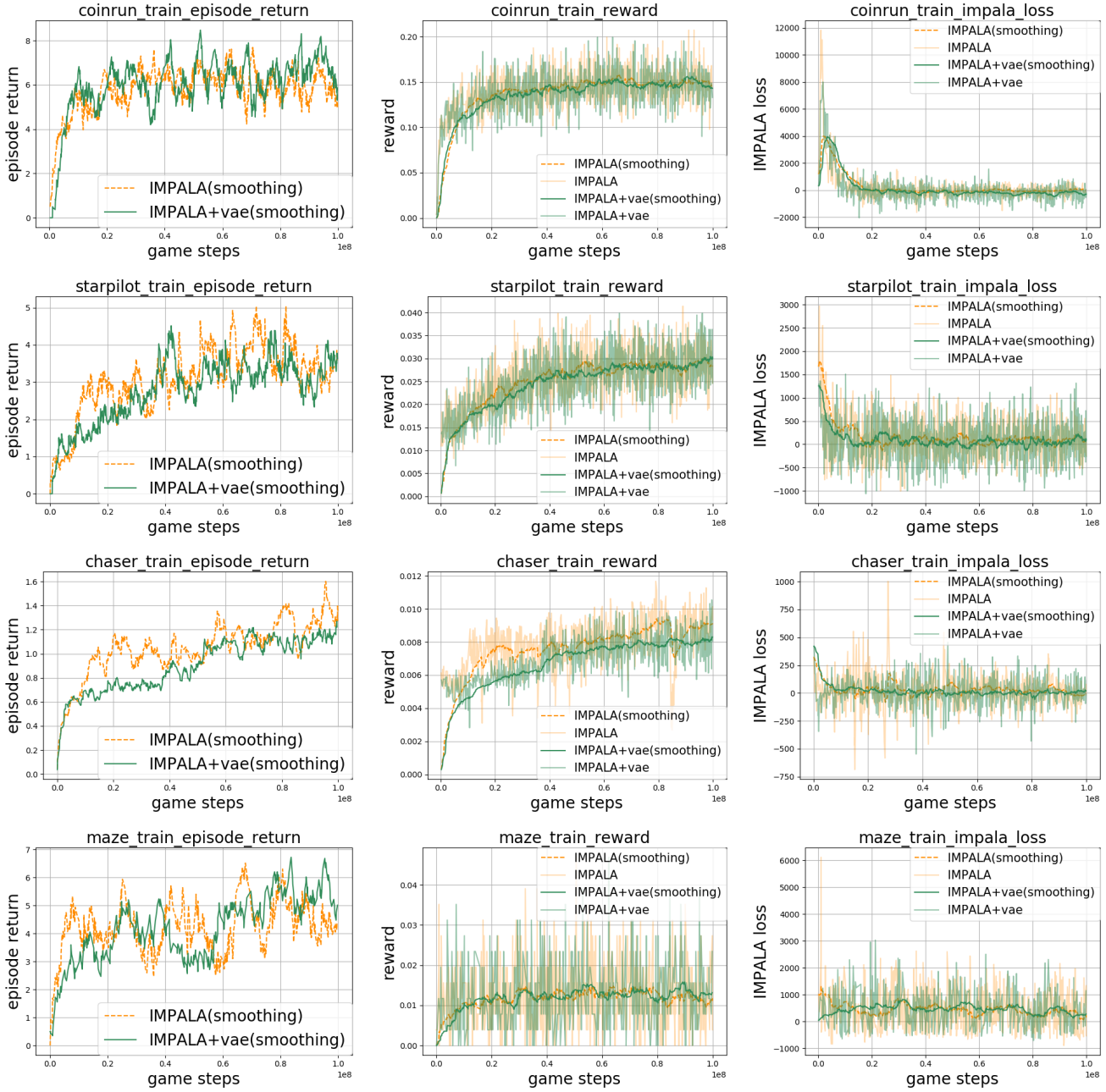

図 4　学習曲線:オレンジ (dark orange) の曲線は baseline，緑 (sea green) の曲線は提案手法で，スムーズしたデータに濃いめの同じ色を用いた．

Fig. 4　Learning curves:The baseline is in dark orange, the proposed method is in sea green.

frame で観察できるキャラクターの部分は，前処理した後，観察できなくなった)．このような encoder の入力で VAE の学習を進めると，自然に復元した結果は元画像との差も大きい．

考えられる原因は色と前処理の影響である．この二つの環境ではランダムで生成された level に，game frame の色が大きく変わることがあって，ゲーム内のエンティティのエッジを検出し難い．故に，前処理した後の画像が元の画像と比べて，より多くの情報が失われた．逆に，Chaser と Maze は迷路と関連する環境であるので，ランダムで生成されても，前処理した後，迷路部分のエッジが保存されて，VAE がより正確に復元できた (元の画像と比べて)．

4.1.2　強化学習の性能評価

既存研究 [3] で，IMPALA の性能が評価されないため，本稿は Procgen Benchmark における IMPALA を用いた強化学習の性能を評価する．図 4 で，各環境での強化学習の episode return，step reward (一回学習する時 *unroll length* · *batch size* の平均 reward)，強化学習での loss function 曲線を示す．全体的に評価すると，訓練段階で，提案手法は baseline の IMPALA より性能が優れるこ

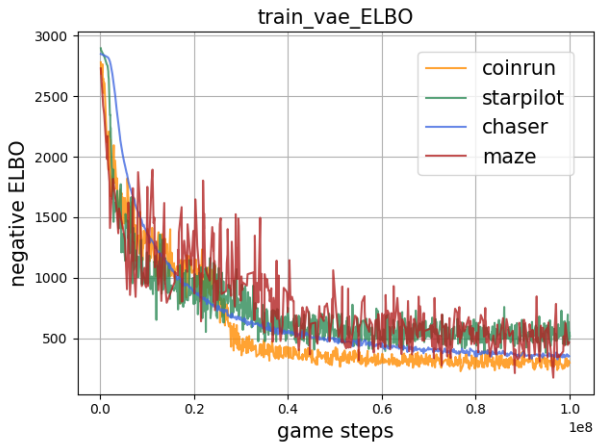

図 5　トレーニング段階で，各実験環境での $-ELBO$

Fig. 5　$-ELBO$ from each experimental environment when training.

とが観測出来なかった．そして，VAE の学習の影響で，訓練初期の VAE はまだ正確に観測空間に対応する潜在表現を得ることができない．状態 (state) の潜在表現を強化学習が入力に取るため (baseline と比べて提案手法の入力には潜在変数の情報が増えている)，baseline より学習速度が遅くなる状況が観測された (特に，StarPilot と Chaser の 0 *steps* から $4 * 10^7$ *steps* までの区間)．この実験結果によって，VAE と IMPALA を同時に学習することができたが，baseline の IMPALA からの性能向上は観測出来なかった．

4.2　汎化性能テスト

500 levels ($num\_levels = 500$, $distribution\_mode = hard$) で学習したエージェントを，full distribution of levels ($num\_levels = 0$, $distribution\_mode = hard$) で汎化性能についてテストを行う．それぞれ 10^5 game steps を実行した．

表 1　各環境の full distribution of levels でのテスト平均点数

Table 1　Mean scores for the full distribution of levels in each experimental environment.

	CoinRun	StarPilot	Chaser	Maze
Baseline	6.51(22.72) [10, 0]	3.473(9.94) [18, 0]	**1.33**(0.50) [3.28, 0.12]	**3.89**(23.77) [10, 0]
提案手法	**6.61**(22.41) [10, 0]	**3.665**(14.97) [18, 0]	1.28(0.46) [3.28, 0.12]	3.69(23.28) [10, 0]

表 1 で，提案手法と baseline の，各環境での mean episode return (variance)[max,min] を示す．理想的な結果は学習段階であったことがない level に対しても，baseline より安定して，高い平均値が得られることだが，CoinRun と StarPilot でごくわずかな性能向上しか観測出来なかった．Chaser と Maze では，VAE の復元はできていたにも関わらず，平均報酬は baseline を少し下回っている．

4.3　まとめと今後の課題

本研究は汎化性能の高い強化学習手法を開発するために，多様性を持つ環境 Procgen Benchmark における，IMPALA と VAE の組み合わせを評価した．

Procgen Benchmark に含まれる 4 つの環境で，それぞれ 500 levels ($num\_levels = 500$, $distribution\_mode = hard$) で訓練した．提案手法の VAE 部分は，可視化結果と目標関数の曲線で評価する．強化学習が収集したデータを VAE の訓練データにすることで，encoder の入力を確実に復元できた．全体的な性能について，提案手法は baseline と比較して，顕著な性能向上が観測出来なかった．状態の潜在表現を強化学習が入力に取るため，baseline より学習速度が遅くなる状況が観測された．テスト段階で，汎化性能を測るために，full distribution of levels ($num\_levels = 0$) で実験を行った．テストの結果から見ると，ごくわずかな性能向上しか観測出来なかった．

今後の課題について，先に提案手法は強化学習に VAE を加えることで，強化学習の性能改善出来なかった点を分析して，改善方法を探す．実験結果から，decoder による復元は潜在変数の学習には成功していると考える．しかし，baseline と比べて提案手法の入力には潜在変数の情報が増えただけで，汎化性能を改善するためにはまた不足であった可能性がある．今後の課題としては，既存研究 [10] は強化学習 (PPO) に潜在空間における SM reward を加えることで汎化性能に効果があると予想したため，提案手法に実装して，汎化性能を検証する．

参考文献

[1] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A. et al.: Mastering the game of go without human knowledge, *nature*, Vol. 550, No. 7676, pp. 354–359 (2017).

[2] Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P. et al.: Grandmaster level in StarCraft II using multi-agent reinforcement learning, *Nature*, Vol. 575, No. 7782, pp. 350–354 (2019).

[3] Cobbe, K., Hesse, C., Hilton, J. and Schulman, J.: Leveraging procedural generation to benchmark reinforcement learning, *arXiv preprint arXiv:1912.01588* (2019).

[4] Doersch, C.: Tutorial on variational autoencoders, *arXiv preprint arXiv:1606.05908* (2016).

[5] Espeholt, L. et al.: Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures, *arXiv preprint arXiv:1802.01561* (2018).

[6] Kingma, D. P.: Variational inference & deep learning: A new synthesis (2017).

[7] Nair, A. V., Pong, V., Dalal, M., Bahl, S., Lin, S. and Levine, S.: Visual reinforcement learning with imagined goals, *Advances in Neural Information Processing Systems*, pp. 9191–9200 (2018).

[8] Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S. and Lerchner, A.: beta-VAE: Learning basic visual concepts with a constrained variational framework, *ICLR* (2017).
[9] Vezzani, G., Gupta, A., Natale, L. and Abbeel, P.: Learning latent state representation for speeding up exploration, *arXiv preprint arXiv:1905.12621* (2019).
[10] Chen, J. Z.: Reinforcement Learning Generalization with Surprise Minimization, *arXiv preprint arXiv:2004.12399* (2020).
[11] Schulman, J., Wolski, F., Dhariwal, P., Radford, A. and Klimov, O.: Proximal policy optimization algorithms, *arXiv preprint arXiv:1707.06347* (2017).
[12] Beattie, C., Leibo, J. Z., Teplyashin, D., Ward, T., Wainwright, M., Küttler, H., Lefrancq, A., Green, S., Valdés, V., Sadik, A. et al.: Deepmind lab, *arXiv preprint arXiv:1612.03801* (2016).
[13] Bellemare, M. G., Naddaf, Y., Veness, J. and Bowling, M.: The arcade learning environment: An evaluation platform for general agents, *Journal of Artificial Intelligence Research*, Vol. 47, pp. 253–279 (2013).
[14] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J. and Zaremba, W.: Openai gym, *arXiv preprint arXiv:1606.01540* (2016).

付　　録

表 **A·1**　ハイパーパラメータ
Table A·1　Hyperparameter.

Hyperparameter	Value
num_actors	8
num_learner	1
Discount (γ)	0.99
Batch size	32
Unroll Length	80
Baseline loss scaling	0.5
Entropy loss scaling	1e-2
Clip global gradient norm	100.0
Adamβ_1	0.9
Adamβ_2	0.999
Adamϵ	1e-8
Adam learning rate	1e-5
Dim of latent space	64

表 **A·2**　Procgen の環境設定
Table A·2　Parameters for Procgen.

Parameter	Value
num_levels	
train	500
test	0
start_level	
train	0
test	1
paint_vel_info	False
use_generated_assets	False
center_agent	True
use_sequential_levels	False
distribution_mode	Hard