Diversity-Driven Extensible Hierarchical Reinforcement Learning Supplementary Material

Hyperparameters	*Value*
Horizon (T)	128
Adam stepsize	$2.5 \times 10^{-4} \times 2$
Learning rate	7×10^4
Number epochs	4
Minibatch size	32×8
Discount (γ)	0.99
GAE parameter (λ)	0.95
Number of actors	8
Clipping parameter (ϵ)	0.1×2
VF coefficient (c^1)	0.5
Entropy coefficient (c^2)	0.01

Table 1: PPO hyperparameters used for DEHRL at each level on OverCooked.

Hyperparameters	*Value*
Horizon (T)	128
Adam stepsize	$2.5 \times 10^{-4} \times 2$
Learning rate	7×10^4
Number epochs	4
Minibatch size	32×8
Discount (γ)	0.99
GAE parameter (λ)	0.95
Number of actors	1
Clipping parameter (ϵ)	0.1×2
VF coefficient (c^1)	0.5
Entropy coefficient (c^2)	0.01

Table 2: PPO hyperparameters used for DEHRL at each level on MineCraft.

Hyperparameters

The *policy* at each level is trained with Proximal Policy Optimization (PPO) algorithm (Schulman et al. 2017). Detailed settings of the hyper-parameters are shown in Table 1 and 2 for OverCooked and MineCraft respectively. Detailed settings of DEHRL framework for OverCooked and MineCraft are shown in Table 3 and 4 respectively.

l	0	1	2
$\mathbb{A}^l$	16	5	5
T^l	1	$1*4$	$1*4*12$

Table 3: DEHRL Settings on OverCooked.

l	0	1	2	3	4	5
$\mathbb{A}^l$	11	8	8	8	8	8
T^l	1	$1*4$	$1*4^2$	$1*4^3$	$1*4^4$	$1*4^5$

Table 4: DEHRL Settings on MineCraft.

Neural Network Details

The details of network architecture for *policy* and *predictor* at each level is shown in Table 5 and 6 respectively. Fully connected layer is denoted as FC and flatten layer is denoted as Flatten. We use leaky rectified linear units (denoted as LeakyRELU) (Maas, Hannun, and Ng 2013) with leaky rate 0.01 as the nonlinearity applied to all the hidden layers in our network. Batch normalization (Ioffe and Szegedy 2015) (denoted as BatchNorm) is applied after hidden convolutional layers (denoted as Conv) in *predictor*. For the *predictor* at each level, the integration of the two inputs, i.e., *state* and *action*, is accomplished by approximated multiplicative interaction (Oh et al. 2015) (the **dot-multiply** operation in Table 6), so that any predictions made by the *predictor* are conditioned on the *action* input. Deconvolutional layers (denoted as DeConv) (Zeiler, Taylor, and Fergus 2011) are applied for predicting the state after T^l steps.

Performance on OverCooked

Here we include a comparison of DEHRL against Option-critic(Bacon and Precup 2017), PPO(Schulman et al. 2017), State Novelty(Şimşek and Barto 2004) and Transition Novelty(Pathak et al. 2017) on OverCooked of 6 settings. Table 7 shows the final performance and Table 8 shows the learning speed.

Input 1: current *state* (s_t), as 84×84 gray scaled image	**Input 2**: *action* from level $l+1$ (a_t^{l+1}), as one-hot vector
Conv: kernel size 8×8, number of features 16, stride 4 **lRELU** **Conv**: kernel size 4×4, number of features 32, stride 2 **lRELU** **Conv**: kernel size 3×3, number of features 16, stride 1 **lRELU** **Flatten**: $16 \times 16 \times 7$ is flatten to 1792 **FC**: number of features 256 **lRELU**	
Output 1: multiple policy functions, each one for one a_t^{l+1}	**Output 2**: multiple value functions, each one for one a_t^{l+1}

Table 5: Network architecture of the *policy* at each level (π^l).

Input 1: current *state* (s_t), as 84×84 gray scaled image	**Input 2**: *action* from level l (a_t^l), as one-hot vector
Conv: kernel size 8×8, number of features 16, stride 4 **BatchNorm** **lRELU** **Conv**: kernel size 4×4, number of features 32, stride 2 **BatchNorm** **lRELU** **Conv**: kernel size 3×3, number of features 16, stride 1 **BatchNorm** **lRELU** **Flatten**: $16 \times 16 \times 7$ is flatten to 1792 **FC**: number of features 256	**FC**: number of features 256
Dot-multiply	
FC: number of features 256 **FC**: number of features 1792 **Reshape**: 1792 is reshaped to $16 \times 16 \times 7$ **DeConv**: kernel size 3×3, number of features 32, stride 1 **BatchNorm** **lRELU** **DeConv**: kernel size 4×4, number of features 16, stride 2 **BatchNorm** **lRELU** **DeConv**: kernel size 8×8, number of features 1, stride 4 **Sigmoid**	**FC**: number of features 1
Output 1: predicted *state* after T^l steps ($\hat{s}_{t+T^l}$)	**Output 2**: predicted *bounty* at downer level ($\hat{b}_t^{l-1}$)

Table 6: Network architecture of the *predictor* at each level (β^l).

reward-level / goal-type	*1 / any*	*1 / fix*	*1 / random*	*2 / any*	*2 / fix*	*2 / random*
DHERL	**1.00**	**1.00**	**1.00**	**0.95**	**0.93**	**0.81**
Option-critic(Bacon and Precup 2017)	**1.00**	**1.00**	0.00	0.00	0.00	0.00
PPO(Schulman et al. 2017)	0.98	0.97	0.56	0.00	0.00	0.00
State Novelty(Şimşek and Barto 2004)	**1.00**	0.96	0.95	0.00	0.00	0.00
Transition Novelty(Pathak et al. 2017)	**1.00**	**1.00**	**1.00**	0.00	0.00	0.00

Table 7: Final performance of DHERL, Option-critic(Bacon and Precup 2017), PPO(Schulman et al. 2017), State Novelty(Şimşek and Barto 2004) and Transition Novelty(Pathak et al. 2017) on OverCooked of 6 settings.

reward-level / goal-type	*1 / any*	*1 / fix*	*1 / random*	*2 / any*	*2 / fix*	*2 / random*
DHERL	0.92	0.72	**0.71**	**0.51**	**0.43**	**0.13**
Option-critic(Bacon and Precup 2017)	0.92	**0.82**	0.00	0.00	0.00	0.00
PPO(Schulman et al. 2017)	0.81	0.57	0.06	0.00	0.00	0.00
State Novelty(Şimşek and Barto 2004)	**0.96**	0.64	0.12	0.00	0.00	0.00
Transition Novelty(Pathak et al. 2017)	0.95	0.77	0.42	0.00	0.00	0.00

Table 8: Learning Speed of DHERL, Option-critic(Bacon and Precup 2017), PPO(Schulman et al. 2017), State Novelty(Şimşek and Barto 2004) and Transition Novelty(Pathak et al. 2017) on OverCooked of 6 settings.

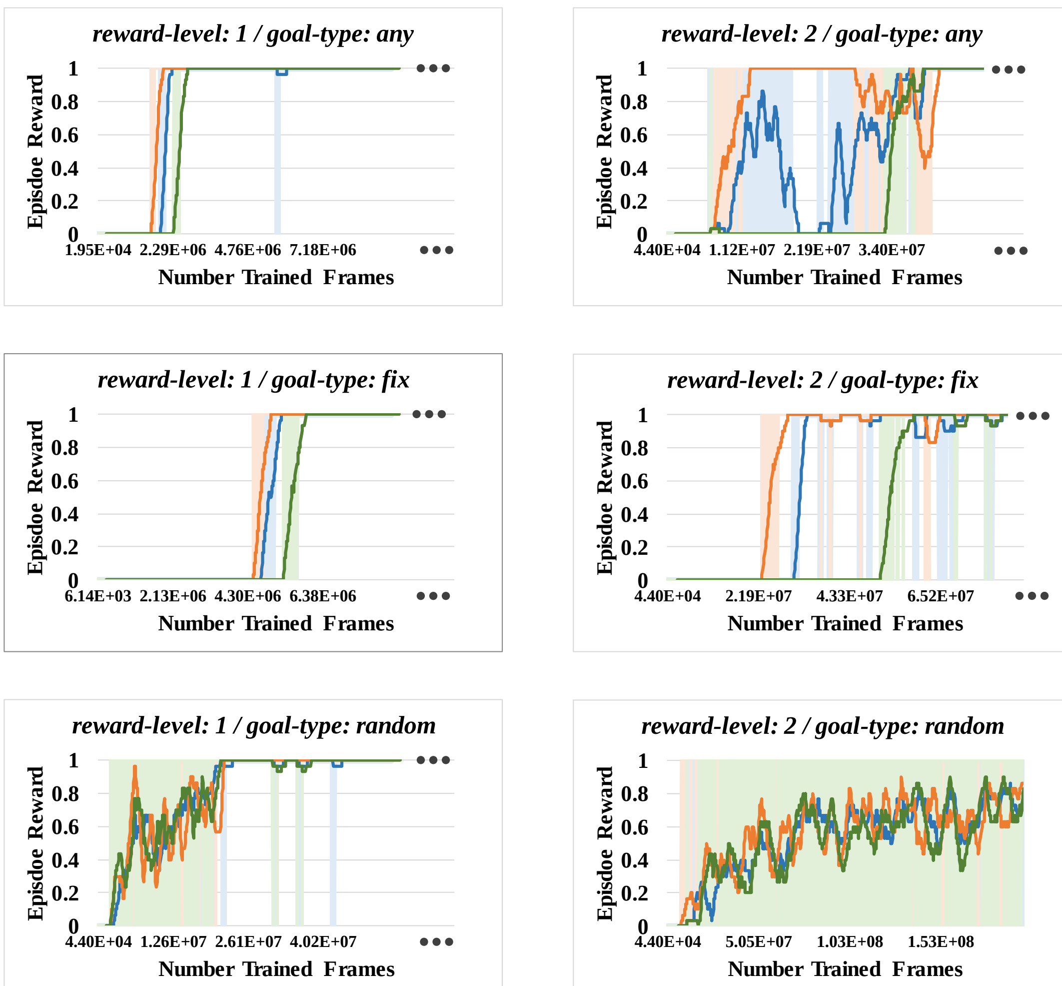
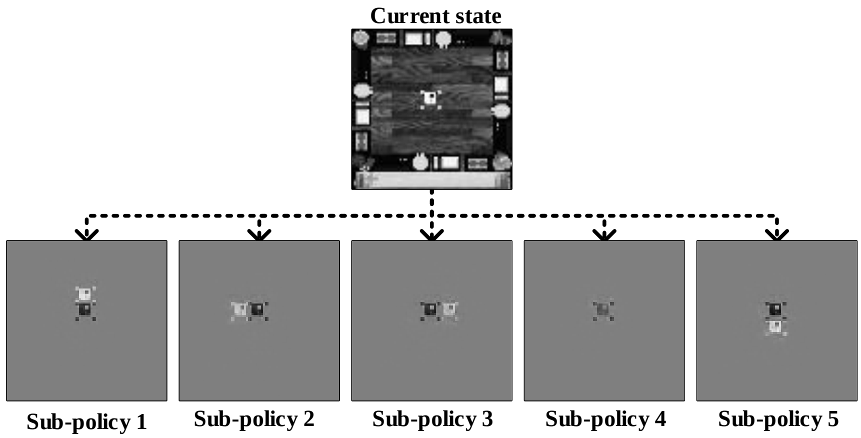
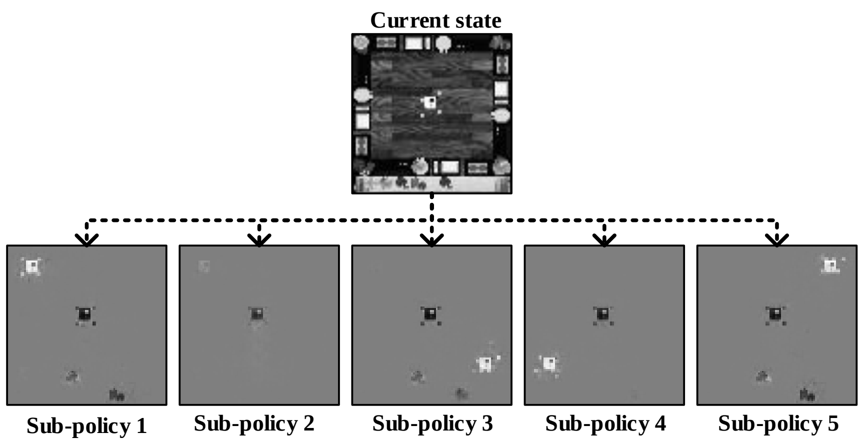


Figure 1: Episode reward curve of DEHRL on all 6 settings of OverCooked. Different colors indicate runs with different training seeds. Shallower color indicates the original curve and the darker color indicates the filtered curve.

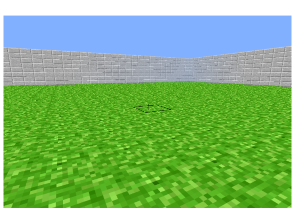


(a) Predicted states by *predictor* at level 1

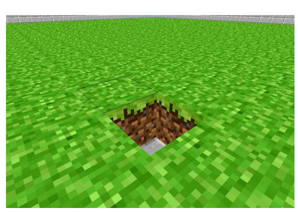


(b) Predicted states by *predictor* at level 2

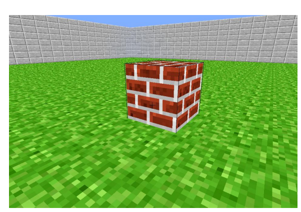
Figure 2: Predicted states by predictor at each level on OverCooked.



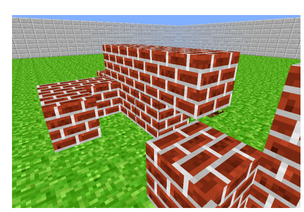
(a) Start state



(b) Break a block



(c) Build a block



(d) Jump on a block

Figure 3: Example states in MineCraft.

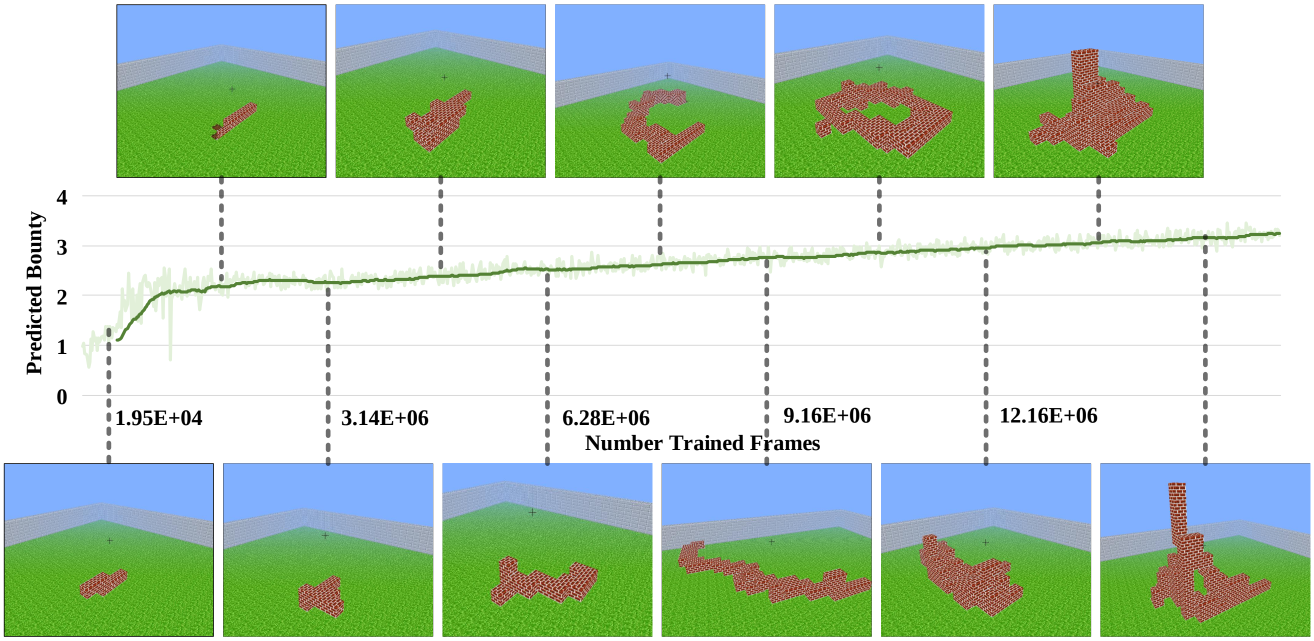


Figure 4: Predicted bounty and the world built by DEHRL.

Besides, there is an interesting question to answer for SNN (Florensa, Duan, and Abbeel 2017): If SNN is guaranteed to learn different subpolicies, will it learn the 5 useful ones provided with enough subpolicy models (set $\mathbb{A}^1 = 625$)? We train the above settings for 200M steps with 3 trials of different training seeds. Surprisingly, the best trial learns 2 useful subpolicies. The reason is that setting $\mathbb{A}^1 = 625$ makes the estimation of the mutual information easily inaccurate, since the mutual information is estimated for every a_t^1 in $\mathbb{A}^1$.

Figure 1 shows the learning curves of DEHRL with three different training seeds.

Figure 2 shows the predicted states of the *predictor* at each level in DEHRL. Since the size of observations and the predictions is 84 and they are gray scaled, it would be hard to have a clear visualization of the predictions. Thus, just for better visualization, current observation is subtracted from the predictions to remove the unchanged parts in Figure 2.

Performance on MineCraft

Figure 3 shows the example states observed by the *agent* in MineCraft. Since the predicted bounty ($\hat{b}_t^l$) is a good indication of the diversity of current sub-policies, we plot the averaged $\hat{b}_t^l$ over all levels and visualize the world built by DEHRL at that time in Figure 4.

References

Bacon, Pierre-Luc, H. J., and Precup, D. 2017. The option-critic architecture. *AAAI*.

Florensa, C.; Duan, Y.; and Abbeel, P. 2017. Stochastic neural networks for hierarchical reinforcement learning. *ICLR*.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Maas, A. L.; Hannun, A. Y.; and Ng, A. Y. 2013. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, 3.

Oh, J.; Guo, X.; Lee, H.; Lewis, R. L.; and Singh, S. 2015. Action-conditional video prediction using deep networks in atari games. In *NIPS*.

Pathak, D.; Agrawal, P.; Efros, A. A.; and Darrell, T. 2017. Curiosity-driven exploration by self-supervised prediction. In *ICML*.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Şimşek, Ö., and Barto, A. G. 2004. Using relative novelty to identify useful temporal abstractions in reinforcement learning. In *ICML*.

Zeiler, M. D.; Taylor, G. W.; and Fergus, R. 2011. Adaptive deconvolutional networks for mid and high level feature learning. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2018–2025. IEEE.