Independent Skill Transfer for Deep Reinforcement Learning

Qiangxing Tian[1,2] , **Guanchu Wang**[2] , **Jinxin Liu**[1,2] , **Donglin Wang**[2*] and **Yachen Kang**[2]
[1]Zhejiang University, Hangzhou, China
[2]School of Engineering, Westlake University, Hangzhou, China
{tianqiangxing, liujinxin, wangdonglin, kangyachen}@westlake.edu.cn, hegsns@gmail.com

Abstract

Recently, diverse primitive skills have been learned by adopting the entropy as intrinsic reward, which further shows that new practical skills can be produced by combining a variety of primitive skills. This is essentially skill transfer, very useful for learning high-level skills but quite challenging due to the low efficiency of transferring primitive skills. In this paper, we propose a novel efficient skill transfer method, where we learn independent skills and only independent components of skills are transferred instead of the whole set of skills. More concretely, independent components of skills are obtained through independent component analysis (ICA), which always have a smaller amount (or lower dimension) compared with their mixtures. With a lower dimension, independent skill transfer (IST) exhibits a higher efficiency on learning a given task. Extensive experiments including three robotic tasks demonstrate the effectiveness and high efficiency of our proposed IST method in comparison to direct primitive-skill transfer and conventional reinforcement learning.

1 Introduction

Deep reinforcement learning (DRL) has wide applications in various challenging fields, such as real-world visual navigation [Zhu *et al.*, 2017], playing games [Silver *et al.*, 2016] and robotic controls [Schulman *et al.*, 2015]. However, conventional algorithms are incapable to deal with complex environments with quite difficult tasks and extremely sparse reward [Kulkarni *et al.*, 2016]. Inspired by the human intelligence that can explore the environment by themselves and learn various skills to significantly improve their ability and accomplish tasks, multi-skill DRL has been proposed as a potential solution to handle tasks in complex environments [Eysenbach *et al.*, 2018]. A significant breakthrough of muti-skill learning is the development of autonomous skill discovery, which can acquire multiple skills autonomously without extrinsic reward by maximizing an information theoretic objective [Gregor *et al.*, 2016] [Singh *et al.*, 2019]. By transferring pre-trained skills, the learning process in a new environment can be greatly shorten and the efficiency is thus enhanced.

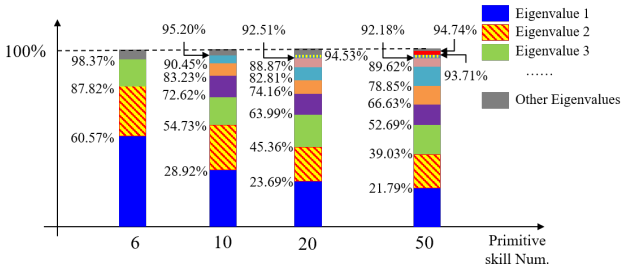


Figure 1: PCA of primitive skills: $6, 10, 20$ and 50 primitive skills are generated by existing skill discovery algorithms. PCA on actions shows that there is strong correlation between these primitive skills, where the normalized eigenvalue of cross-correlation denotes the percentage of each independent component.

In this paper, skill transfer is to learn primitive skills in a source environment without extrinsic reward and reutilize them in a target environment, which can acquire a higher learning efficiency than learning from scratch. Skill transfer can greatly accelerate the learning process in a target environment because the learned primitive skills provides reusable abstraction of the source environment [Sahni *et al.*, 2017]. However, with the existing direct skill transfer, there are two problems resulting in the low learning efficiency:

- **Strong correlation:** We analyze the statistical characteristics of primitive skills that are generated from the existing skill discovery methods [Eysenbach *et al.*, 2018] [Sharma *et al.*, 2019]. By using principal components analysis (PCA) [Bryant and Yarnold, 1995] on actions of primitive skills, Figure 1 shows the eigenvalue and the corresponding percentage of principal components in primitive skills. It is observed that the actions generated by distinct primitive skills are **strong correlated** with each other, which indicates that the amount of skills can be reduced by eliminating this correlation so that the skill dimension is thus decreased.
- **Unbalance of skill discovery and transfer:** In the skill discovery, the agent learns each of primitive skills **separately** in a source environment. However, in the skill transfer, all primitive skills are **combined** to instruct the

*Corresponding author

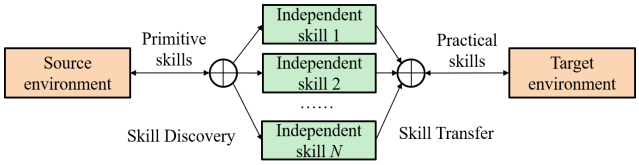


Figure 2: Framework of Independent Skill Transfer (IST).

agent in a target environment [Peng *et al.*, 2019]. As noticed, the primitive skill in skill discovery and skill transfer play a distinct role. A more balanced scheme might improve the performance of skill transfer.

To overcome these two problems above, we propose to learn independent skills for efficient skill transfer, where the learned primitive skills with strong correlation are decomposed into independent skills. More concretely, Figure 2 shows the process of independent skill transfer (IST). Diverse primitives skills are firstly cultivated in a source environment. By taking actions to represent such primitive skills, the agent then decomposes these primitive skills to acquire independent skills, where independent component analysis (ICA) [Hyvärinen and Oja, 2000] is employed on those primitive skill's actions. Finally, the agent transfers independent skills into new practical skills in a target environment.

On the one hand, our proposed IST method is able to reduce the dimension of skills and enhance the efficiency of skill transfer. On the other hand, each of primitive skills is the combination of all independent skills, which is balanced with the combined practical skill in Figure 2. Therefore, the advantage of our proposed IST can be listed as follows:

- The correlation between skills can be largely deducted and a lower dimension is thus obtained to enhance the efficiency of skill transfer.
- Combination of independent skills take effects in both skill discovery and transfer, where transferring independent skills is more essential.
- Independent skills are task-independent, which can be transferred to a variety of practical skills in a target environment.

The contribution of this work is summarized as follow. First, we propose to learn low-dimension independent skills from primitive skills. Secondly, we propose to transfer independent skills to practical skills in target environments, which exhibits a higher efficiency and better generalization than conventional methods. Finally, various experiments are conducted to demonstrate the effectiveness of IST method.

2 Related Work

In recent years, skill discovery and transfer has gained more and more attention from researchers who work in DRL.

Skill Discovery. Information theory has been widely applied in the development of multiple skills in DRL. In order to enable the agent to own skills to explore the environment, the VIM [Mohamed and Rezende, 2015] adds the relative entropy between states and actions into the reward function to broaden the observation of the agent. Soft actor-critic (SAC) [Haarnoja *et al.*, 2018a] discovers a policy function with maximum entropy to gather useful skills for the agent. Moreover, an important work, i.e. the DIAYN [Eysenbach *et al.*, 2018], also maximizes the mutual information between skills and states to ensure that states are used to distinguish skills.

Skill transfer. A popular method of skill transfer is to learn a hierarchical policy for skill reutilization. In the pre-train stage, a collection of primitive skills are learned based on skill discovery methods, where each skill is encouraged to specialize distinct observations [Coros *et al.*, 2009] [Frans *et al.*, 2017]. While in transferring the primitive skills, a meta controller is trained to select primitive skills depending on the specific high-level task [Hausknecht and Stone, 2015]. Even though existing methods on skill transfer are elaborative and remarkable, several drawbacks remain to be solved. One of significant problems is the low transfer efficiency since the integration of primitive skills fails to consider their dependencies on each other [Peters *et al.*, 2017].

Independent component analysis (ICA). ICA attempts to decompose a multivariate signal into independent non-Gaussian signals. Preprocessing steps for ICA involves centering and dimensionality reduction, which can be achieved by PCA. Existing works on ICA include kernel-independent component analysis [Bach and Jordan, 2002], infomax [Bell and Sejnowski, 1995] and FastICA [Oja and Yuan, 2006]. In this work, we consider to employ FastICA to decompose primitive skills into independent skills, which allows much easier generalization and transfer [Bengio, 2017], and can greatly enhance the transfer efficiency.

3 Preliminary

3.1 Soft Actor-critic

A standard reinforcement learning framework is characterized by observation space $\mathcal{S}$, action space $\mathcal{A}$, reward function r, transition function $\mathbb{T}$ and discount factor γ, which formulates a five-tuple $< \mathcal{S}, \mathcal{A}, r, \mathbb{T}, \gamma >$. At each step t, an agent observes the current state $\boldsymbol{s}_t \in \mathcal{S}$, and executes an action $\boldsymbol{a} \in \mathcal{A}$ according to the policy function $\varpi_\phi(\boldsymbol{a}|\boldsymbol{s}) : \mathcal{S} \rightarrow \mathcal{A}$. In this paper, we use the SAC [Haarnoja *et al.*, 2018b] algorithm to update the value function $Q(\boldsymbol{s}, \boldsymbol{a})$ and the policy function. $\varpi_\phi(\boldsymbol{a}|\boldsymbol{s})$ is updated to maximize the following cumulative reward $J(\phi)$ (i.e. $\phi^* = \arg\max J(\phi)$)

$$J(\phi) = \mathbb{E}_{\boldsymbol{s}\in\mathcal{S},\epsilon\sim\mathcal{N}}\big[Q(\boldsymbol{s}, f(\epsilon, \boldsymbol{s}_t)) - \beta \log \varpi_\phi(f(\epsilon, \boldsymbol{s}_t)|\boldsymbol{s})\big], \quad (1)$$

where $\mathcal{N}$ denotes the spherical Gaussian distribution; $f(\epsilon, \boldsymbol{s}_t)$ can be obtained via reparameterization trick; the entropy term $-\mathbb{E}_{\boldsymbol{a}_t\sim\varpi_\phi} \log \varpi_\phi(\boldsymbol{a}|\boldsymbol{s})$ multiplied by temperature parameter β is introduced to control the encouragement of exploration.

3.2 Skill Discovery

The skill space $\mathcal{Z}$ is introduced to the framework of standard reinforcement learning to diversify the learning process of agents. Different skills $\pi(\boldsymbol{a}|\boldsymbol{s}, z) : \mathcal{S} \times \mathcal{Z} \rightarrow \mathcal{A}$ can alter the current observation by the agent in a consistent manner.

In the recent work DIAYN [Eysenbach *et al.*, 2018], the authors propose to use intrinsic reward for skill discovery. In

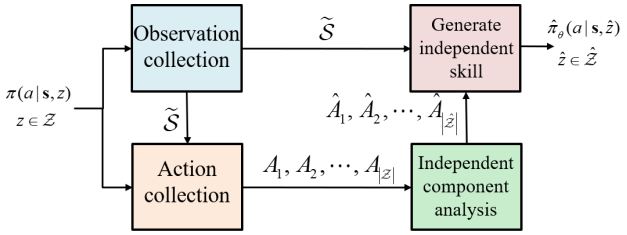


Figure 3: Framework of Learning Independent Skills (LIS).

each step t, $Q(\boldsymbol{s}_t, z, \boldsymbol{a}_t) : \mathcal{S}\times\mathcal{Z}\times\mathcal{A} \to \mathcal{R}$ and $\pi(\boldsymbol{a}_t|\boldsymbol{s}_t, z)$ are updated based on SAC, and a discriminator network $q(z|\boldsymbol{s}_t)$ is employed to learn the posterior distribution of each skill given $\boldsymbol{s}_t$, where its parameters are updated via stochastic gradient descent (SGD). To generate a variety of different skills, DIAYN employs the reward function given by

$$r(\boldsymbol{s}_t, \boldsymbol{a}_t) = \log q(z|\boldsymbol{s}_{t+1}) - \log p(z), \tag{2}$$

where the prior $p(z)$ is often selected as uniform distribution.

4 Learn Independent Skills (LIS)

In this paper, we use policy networks $\hat{\pi}_\theta(\boldsymbol{a}|\boldsymbol{s}, \hat{z}), \hat{z} \in \hat{\mathcal{Z}}$ to denote independent skills. For the convenience of specification, we define the actions generated by independent skill as **independent action**, and those produced by primitive skill as **primitive action**. Furthermore, our proposed LIS is based on the following three assumptions:

- Primitive actions are the linear combination of independent actions.
- On the same observation, the actions generated by distinct independent skill are independent with each other, which reflects the independence between independent skills.
- Without loss of generality, the number of independent skill $|\hat{\mathcal{Z}}|$ is no more than the number of primitive skills $|\mathcal{Z}|$.

In brief, the aim of LIS is to learn finite $\hat{\pi}_\theta(\boldsymbol{a}|\boldsymbol{s}, \hat{z}), \hat{z} \in \hat{\mathcal{Z}}$, such that $\forall \boldsymbol{s} \in \mathcal{S}, \boldsymbol{a} \sim \pi(\boldsymbol{a}|\boldsymbol{s}, z)$ is the linear combination of $\hat{\boldsymbol{a}} \sim \hat{\pi}_\theta(\boldsymbol{a}|\boldsymbol{s}, \hat{z})$. As shown in Figure 3, we first sample subset $\tilde{\mathcal{S}}$ from the observation space $\mathcal{S}$. Secondly, we sample action $\boldsymbol{A}_z$ based on primitive skill $\pi(\boldsymbol{a}|\boldsymbol{s}, z)$ for $z \in \mathcal{Z}$. We then convert the primitive actions $\boldsymbol{A}_z$ to independent actions $\hat{\boldsymbol{A}}_z$ through ICA. Finally, we utilize $\tilde{\mathcal{S}}$ and $\hat{\boldsymbol{A}}_z$ to learn independent skills $\hat{\pi}_\theta(\boldsymbol{a}|\boldsymbol{s}, \hat{z})$ for $\hat{z} \in \hat{\mathcal{Z}}$ via supervised learning.

4.1 Collection of Observation and Action

It is inappropriate to sample observations from the whole observation space $\mathcal{S}$ that is extremely large and continuous. Hence, a strategy for collecting observation and action is proposed in this paper, where we sample a key subset $\tilde{\mathcal{S}} \subset \mathcal{S}$ from the observation space.

Different skills enable an agent to have multiple, stable and finite state transitions, which are considered as key subset of observation space $\mathcal{S}$ in this paper. Specifically, for each primitive skill $\pi(\boldsymbol{a}|\boldsymbol{s}, z)$, we sample L trajectories $\tau_{z,1}, \tau_{z,2}, ..., \tau_{z,L}$ independently, where each $\tau_{z,i} = \{\boldsymbol{s}_{z,i,1}, \boldsymbol{a}_{z,i,1}, \boldsymbol{s}_{z,i,2}, \boldsymbol{a}_{z,i,2}, ..., \boldsymbol{s}_{z,i,T_i}\}$ for $1 \le i \le L$ denotes the trajectory in each episode. The observation set of skill z consists of all the states in $\tau_{z,i}$ for $1 \le i \le L$, given by

$$\tilde{\mathcal{S}}_z = \{\boldsymbol{s}_{z,i,j} | 1 \le i \le L, 1 \le j \le T_i\}. \tag{3}$$

Thus, the key subset $\tilde{\mathcal{S}}$ of observation space can be expressed as $\tilde{\mathcal{S}} = \tilde{\mathcal{S}}_1 \cup \tilde{\mathcal{S}}_2 \cup ... \cup \tilde{\mathcal{S}}_{|\mathcal{Z}|}$.

In the action collection, for each skill $\pi(\boldsymbol{a}|\boldsymbol{s}, z)$, we sample K actions on each of the observations in $\tilde{\mathcal{S}}$, and reserve all the actions in the following matrix $\boldsymbol{A}_z$

$$\boldsymbol{A}_z = \big[\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_K \sim \pi(\boldsymbol{a}|\boldsymbol{s}, z) | \boldsymbol{s} \in \tilde{\mathcal{S}}\big], \tag{4}$$

where it is worth noting that partial actions of $\pi(\boldsymbol{a}|\boldsymbol{s}, z)$ on $\tilde{\mathcal{S}}_z$ can be directly obtained from trajectories $\tau_{z,1}, \tau_{z,2}, ..., \tau_{z,L}$.

As the mixed signal for ICA, matrix $\boldsymbol{A}_z$ has to be flattened[1], i.e. $A_z = \text{flatten}(\boldsymbol{A}_z)$. After the processing of ICA, we get independent components $\hat{A}_{\hat{z}}$ from A_z. Then we reconstruct $\hat{A}_{\hat{z}}$ to matrix $\hat{\boldsymbol{A}}_z$ by adopting the inverse operation.

Algorithm 1 Learn independent skills (LIS)

Input: $\pi(\boldsymbol{a}|\boldsymbol{s}, z), z \in \mathcal{Z}$
Output: $\hat{\pi}_\theta(\boldsymbol{a}|\boldsymbol{s}, \hat{z}), \hat{z} \in \hat{\mathcal{Z}}$

Sample $\tilde{\mathcal{S}}_1, \tilde{\mathcal{S}}_2, ..., \tilde{\mathcal{S}}_{|\mathcal{Z}|}$ via Eq. (3).
Sample $\boldsymbol{A}_1, \boldsymbol{A}_2, ..., \boldsymbol{A}_{|\mathcal{Z}|}$ via Eq. (4), and transform to $A_1, A_2, \cdots, A_{|\mathcal{Z}|}$.
//———— ICA ————//
Calculate $\boldsymbol{W}_P$ via PCA, and whiten $A = [A_1, A_2, \cdots, A_{|\mathcal{Z}|}]$.
while not converged **do**
  Calculate $\boldsymbol{W}_I$ according to Eq. (6) and (7).
end while
Calculate $\hat{A}_1, \hat{A}_2, ..., \hat{A}_{|\hat{\mathcal{Z}}|}$ via Eq. (5), and reconstruct to $\hat{\boldsymbol{A}}_1, \hat{\boldsymbol{A}}_2, ..., \hat{\boldsymbol{A}}_{|\hat{\mathcal{Z}}|}$.
//———— End of ICA ————//
for $\boldsymbol{s} \in \tilde{\mathcal{S}}, \hat{z} \in \hat{\mathcal{Z}}$ **do**
  Calculate $\boldsymbol{\mu}_{\boldsymbol{s},\hat{z}}$ and $\boldsymbol{\sigma}_{\boldsymbol{s},\hat{z}}$.
  $\min_\theta \big\{D_{KL}\big[\hat{\pi}_\theta(\hat{\boldsymbol{a}}|\boldsymbol{s}, \hat{z}) || \hat{p}(\hat{\boldsymbol{a}}|\boldsymbol{\mu}_{\boldsymbol{s},\hat{z}}, \boldsymbol{\sigma}_{\boldsymbol{s},\hat{z}})\big]\big\}$.
end for
return $\hat{\pi}_\theta(\boldsymbol{a}|\boldsymbol{s}, \hat{z}), \hat{z} \in \hat{\mathcal{Z}}$

4.2 Generation of Independent Actions

In this subsection, the primitive action $A_z, z \in \mathcal{Z}$ is regarded as mixed signal to generate independent actions $\hat{A}_{\hat{z}}, \hat{z} \in \hat{\mathcal{Z}}$. Specifically, we aim to find a $|\hat{\mathcal{Z}}| \times |\mathcal{Z}|$ matrix $\boldsymbol{W}_P$ and $|\hat{\mathcal{Z}}| \times |\hat{\mathcal{Z}}|$ full-rank matrix $\boldsymbol{W}_I$, such that

$$\begin{bmatrix} \hat{A}_1^T \\ \hat{A}_2^T \\ \vdots \\ \hat{A}_{|\hat{\mathcal{Z}}|}^T \end{bmatrix} = \boldsymbol{W}_I \boldsymbol{W}_P \begin{bmatrix} A_1^T \\ A_2^T \\ \vdots \\ A_{|\mathcal{Z}|}^T \end{bmatrix}, \tag{5}$$

[1] A general flattening like column-based or row-based conversion works for the transformation of $n \times K|\tilde{\mathcal{S}}|$ matrix $\boldsymbol{A}_z$ into $nK|\tilde{\mathcal{S}}| \times 1$ vector A_z, where n is the dimension of $\boldsymbol{a}$. Here, column-based transformation is applied.

where $\hat{A}_1, \hat{A}_2, ..., \hat{A}_{|\hat{\mathcal{Z}}|}$ are independent with each other.

In this paper, we use PCA [Wold *et al.*, 1987] to estimate $\boldsymbol{W}_P$, where the number of independent skills $|\hat{\mathcal{Z}}|$ can be achieved by clustering the eigenvalues of cross-correlation matrix of A_z. We take the eigenvalues in Figure 1 as an example: for the case of 6 primitive skills, $|\hat{\mathcal{Z}}| = 3$ is reasonable since more than 98% component of primitive actions can be represented by three independent components.

To estimate $\boldsymbol{W}_I$, we employ the FastICA [Hyvärinen and Oja, 2000], where the independent signals are actually independent actions. The details are given as follows,

- First of all, the primitive actions $A = [A_1, \cdots, A_{|\mathcal{Z}|}]$ has to be whitened so that $\mathbb{E}\big[AA^T\big] = \boldsymbol{I}$.
- Secondly, we estimate $\boldsymbol{W}_I = [\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_{|\mathcal{Z}|}]$ by calculating each of its columns, separately. In each iteration of ICA, we maximize the independence of output signal via updating $\boldsymbol{w}_i$ by

$$\boldsymbol{w}_i \leftarrow \mathbb{E}\big[A\mathrm{g}(\boldsymbol{w}_i A)\big] - \mathbb{E}\big[\mathrm{g}'(\boldsymbol{w}_i A)\big]\boldsymbol{w}_i, \tag{6}$$

  for $1 \le i \le |\hat{\mathcal{Z}}|$, where $\mathrm{g}(x) = \frac{1}{1+e^{-x}}$, $\mathrm{g}'(x) = \frac{-e^{-x}}{1+e^{-x}}$, and we orthogonalize $\boldsymbol{W}_I$ by

$$\boldsymbol{W}_I \leftarrow (\boldsymbol{W}_I \boldsymbol{W}_I^T)^{-\frac{1}{2}} \boldsymbol{W}_I. \tag{7}$$

After the algorithm converges, the independent actions $\hat{A}_{\hat{z}}$ for $\hat{z} \in \hat{\mathcal{Z}}$ can be obtained according to Eq. (5).

4.3 Generation of Independent Skills

After reconstructing $\hat{A}_1, \hat{A}_2, ..., \hat{A}_{|\hat{\mathcal{Z}}|}$ into a vector form $\hat{\boldsymbol{A}}_1, \hat{\boldsymbol{A}}_2, ..., \hat{\boldsymbol{A}}_{|\hat{\mathcal{Z}}|}$, we assume that

$$\hat{\boldsymbol{A}}_{\hat{z}} = \big[\hat{\boldsymbol{a}}_1, \hat{\boldsymbol{a}}_2, ..., \hat{\boldsymbol{a}}_K \sim \hat{\pi}_\theta(\hat{\boldsymbol{a}}|\boldsymbol{s}, \hat{z}) | \boldsymbol{s} \in \tilde{\mathcal{S}}\big], \tag{8}$$

and calculate the expectation $\boldsymbol{\mu}_{\boldsymbol{s},\hat{z}}$ as well as standard deviation $\boldsymbol{\sigma}_{\boldsymbol{s},\hat{z}}$ of $\hat{\boldsymbol{a}}_1, \hat{\boldsymbol{a}}_2, ..., \hat{\boldsymbol{a}}_K$ for each observation $\boldsymbol{s}$. Then, we use neural networks to learn independent skills $\hat{\pi}_\theta(\hat{\boldsymbol{a}}|\boldsymbol{s}, \hat{z})$ by minimizing the KL divergence between $\hat{\pi}_\theta(\hat{\boldsymbol{a}}|\boldsymbol{s}, \hat{z})$ and the empirical distribution $\hat{p}(\hat{\boldsymbol{a}}|\boldsymbol{\mu}_{\boldsymbol{s},\hat{z}}, \boldsymbol{\sigma}_{\boldsymbol{s},\hat{z}})$:

$$\min_\theta \big\{ D_{KL}\big[\hat{\pi}_\theta(\hat{\boldsymbol{a}}|\boldsymbol{s}, \hat{z}) || \hat{p}(\hat{\boldsymbol{a}}|\boldsymbol{\mu}_{\boldsymbol{s},\hat{z}}, \boldsymbol{\sigma}_{\boldsymbol{s},\hat{z}})\big]\big\}, \tag{9}$$

where we employ Gaussian distribution to characterize $\hat{\pi}_\theta(\hat{\boldsymbol{a}}|\boldsymbol{s}, \hat{z})$ and $\hat{p}(\hat{\boldsymbol{a}}|\boldsymbol{\mu}_{\boldsymbol{s},\hat{z}}, \boldsymbol{\sigma}_{\boldsymbol{s},\hat{z}})$. For more details about LIS, please refer to Algorithm 1.

5 Independent Skill Transfer (IST)

In this section, we propose to transfer independent skills to practical skills in a target environment by learning a transfer policy $\varpi_\phi([\boldsymbol{\alpha}, b]|\boldsymbol{s})$ as shown in Figure 4. Since primitive actions are linear combination of independent actions, the transfer policy $\varpi_\phi([\boldsymbol{\alpha}, b]|\boldsymbol{s})$ produces a weight $\boldsymbol{\alpha}_t$ and bias b_t in each time step t based on observation $\boldsymbol{s}_t$, so the composite action can be calculated by

$$\boldsymbol{a}_t = \hat{\boldsymbol{a}}_1 \alpha_{t,1} + \hat{\boldsymbol{a}}_2 \alpha_{t,2} + ... + \hat{\boldsymbol{a}}_{|\hat{\mathcal{Z}}|} \alpha_{t,|\hat{\mathcal{Z}}|} + \boldsymbol{1} \otimes b_t, \tag{10}$$

where independent actions $\hat{\boldsymbol{a}}_1 \sim \hat{\pi}_\theta(\hat{\boldsymbol{a}}|\boldsymbol{s}_t, \hat{z}_1)$, $\hat{\boldsymbol{a}}_2 \sim \hat{\pi}_\theta(\hat{\boldsymbol{a}}|\boldsymbol{s}_t, \hat{z}_2)$, $\cdots$, respectively, $\otimes$ denotes the kronecker product, and b_t denotes the bias.

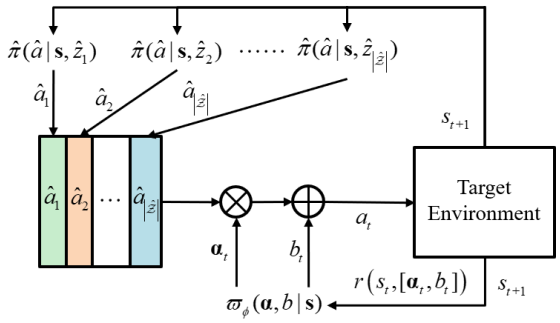


Figure 4: Process of Independent Skill Transfer (IST).

After executing the composite action $\boldsymbol{a}_t$, the agent can achieve the observation $\boldsymbol{s}_{t+1} \sim \mathbb{T}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t)$ and reward $r(\boldsymbol{s}_t, \boldsymbol{\alpha}_t)$ from the target environment. Then, we employ SAC to update the transfer policy $\varpi_\phi([\boldsymbol{\alpha}, b]|\boldsymbol{s})$.

6 Experiment

In this section, we evaluate the behavior of independent skills in skill transfer[2], where our experiments are conducted to answer the following three questions:

- Is our proposed collection of observation effective?
- Does our proposed IST outperform state-of-the-art?
- How is the performance of IST on difficult tasks?

In the pre-training stage, we employ DIAYN to learn 6 primitive skills ($|\mathcal{Z}| = 6$), which can be used for both IST and primitive skill transfer (PST). According to the source environment *HalfCheetah-v3* [3], we set up target environments by considering extra key elements as shown in Figure 5, including HalfCheetah-Hurdle (HCH), HalfCheetah-Ascending (HCA) and HalfCheetah-Upstair (HCU).

6.1 Collection of Observation

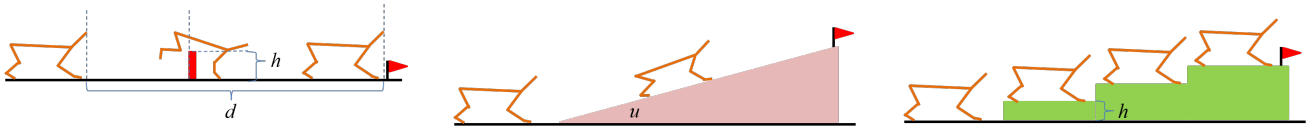
Figure 6 plots the loss of independent skills $\hat{\pi}_\theta(\hat{\boldsymbol{a}}|\boldsymbol{s}, \hat{z})$ on training dataset and validation dataset. It is observed that the proposed strategy sampling from the key subset $\tilde{S}$ performs better, where the loss converges to 10^{-8} on both training and validation sets, which indicates the validity of observation and action collection in characterizing the primitive skills $\pi(\boldsymbol{a}|\boldsymbol{s}, z)$. Furthermore, as the trajectory length T increases from 150 to 250, a notable improvement can be obtained. Hence, we keep $T = 250$ in the following experiments. This answers the first question above.

6.2 Performance of IST

We compare IST with state-of-the-art to show the effectiveness and high efficiency of our proposed method.

[2] https://github.com/qxtian/Learning-Independent-SKills

[3] https://gym.openai.com/envs/



(a) HCH: The agent requires to stride over a hurdle and walk to the destination, which becomes harder as d and h increase.

(b) HCA: The agent has to climb up to a hill to reach the destination, where it gets harder as the inclination angle u increases.

(c) HCU: The agent goes upstairs to reach the destination, where the task turns to be harder as the height of stair h increases.

Figure 5: Complex tasks: We regard *HalfCheetah-v2* as the source environment, and construct 3 target environments: HCH, HCA and HCU by adding obstruction with adjustable size in the environment, where the difficulty-level depends on the size of obstacles.

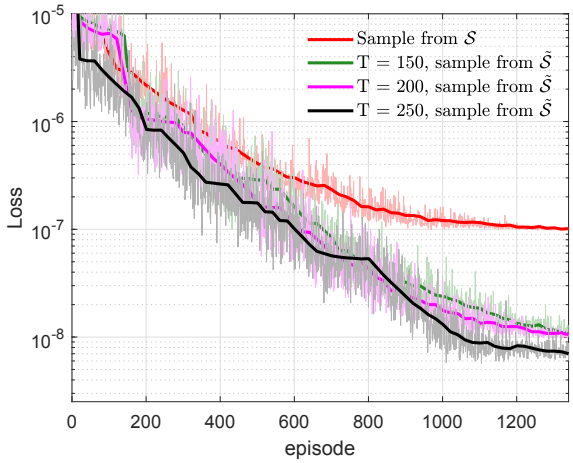


Figure 6: Convergence of loss, where the dark line and light color mark the loss on training set and validation set, respectively, and the sampling size from $\mathcal{S}$ (red) is identical with that of $T = 250$ (black).

Compared with PST. Regarding PST [Peng *et al.*, 2019], the composite action is thought of as the linear combination of primitive actions and SAC is adopted to learn the combined weights. While for IST, PCA is employed for skill actions, where the percentages[4] are 60.57%, 27.25%, 10.55%, 1.46%, 0.15% and 0.02%, respectively, as given in Figure 1. As seen, the last 3 eigenvalues are negligible. Hence, we consider to learn 3 independent skills ($|\hat{\mathcal{Z}}| = 3$) for the generation of practical skills. In such case, more than 98% component of primitive skills can be reserved in the conversion to independent skills. Furthermore, due to the independence between skills and lower dimension of weight $\boldsymbol{\alpha}$ decreased from $\mathbb{R}^{|\mathcal{Z}|}$ to $\mathbb{R}^{|\hat{\mathcal{Z}}|}$, IST allows more efficient skill transfer. Consequently, as in Figure 7 (a)-(c), IST shows a higher efficiency of reward collection than PST for all cases. Particularly, from Figure 7 (f) where $h = 0.3$, the average return of IST converges in less than 150 episodes. In contrast, the average return of PST fails to converge, indicating the failure of skill transfer in finite episodes.

Compared with primitive skill selection (PSS). In PSS [Sharma *et al.*, 2019], a network is trained to select an optimal skill from diverse pre-training primitive skills to deal with a specific task. In each step, a single primitive skill is selected and thus activated for transfer, but it is hardly as amenable as the combination of independent skills for practical skills. In fact, selecting a single skill can be a special case of linear combination of independent skills with weights $\boldsymbol{\alpha}_t = [0, \cdots, 1, \cdots, 0]$ and bias $b_t = 0$. Therefore, it is observed from Figure 7 (a)-(c) that IST collects reward more efficiently and exhibits stronger generalization ability than PSS.

Compared with conventional RL. We compare the performance of IST with the existing reinforcement learning: SAC. Empirically, prior information of source environment has been reserved in primitive skills and then refined to independent skills as mentioned above. In contrast, SAC learning from scratch has no reutilization of primitive skills or prior environment. Hence, it is observed from Figure 7 (a)-(c) that, IST enables a higher learning efficiency than SAC learning from scratch by incorporating diverse prior experiences. Conclusively, the proposed method manages to reserve and refine sufficient prior information from the source environment for reutilization in a target environment.

6.3 Skill Transfer on Difficult Tasks

In this subsection, we evaluate the performance of skill transfer when the difficulty level gets increased. As shown in Table 1, when the given task gets harder (e.g. HCH $h : 0.3 \rightarrow 0.35$ etc.), all skill transfer methods suffer from a degradation of performance in terms of success rate.

In such case, the proposed IST achieves the least degradation and the best performance compared with others (PST has nearly the same degradation, but less success rate), indicating that IST is less sensitive to the difficulty level of tasks. Furthermore, IST shows the best performance over most of tasks except for the negligible inferiority than SAC in HCA with $u = 13.76°$ and $19.5°$. Hence, the proposed IST exhibits its capability on difficult tasks, and we show in Figure 8 the frames of learned different tasks completed by IST.

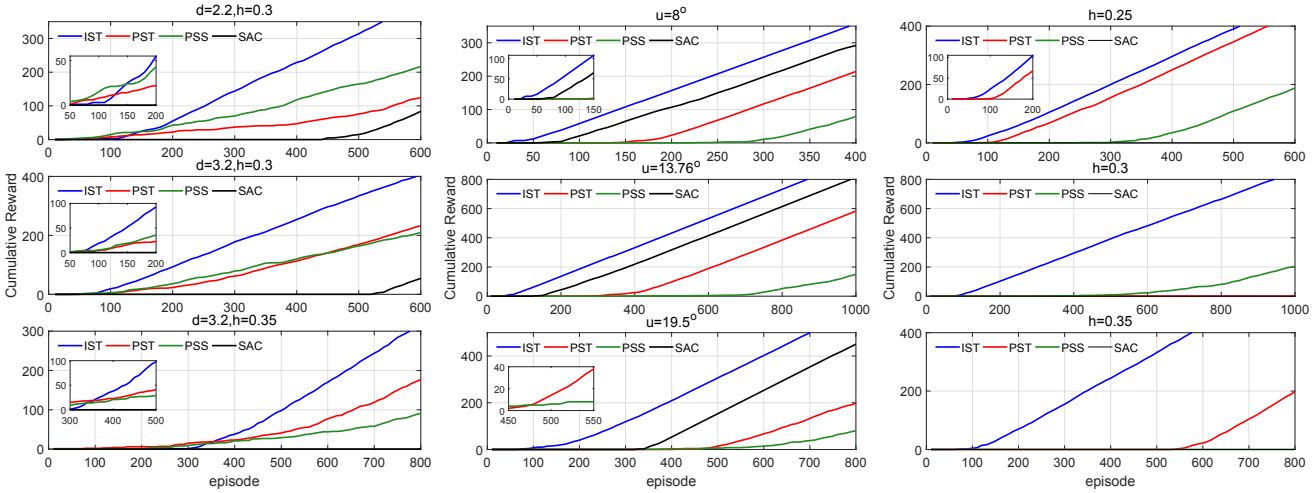
7 Conclusion

In this work , we propose to learn independent skills from primitive skills and further transfer them to high-level complex tasks. Effective observation collection and independent skills guarantee the success of low-dimension skill transfer. Experiment results show a higher learning efficiency and stronger generalization ability of our proposed method.

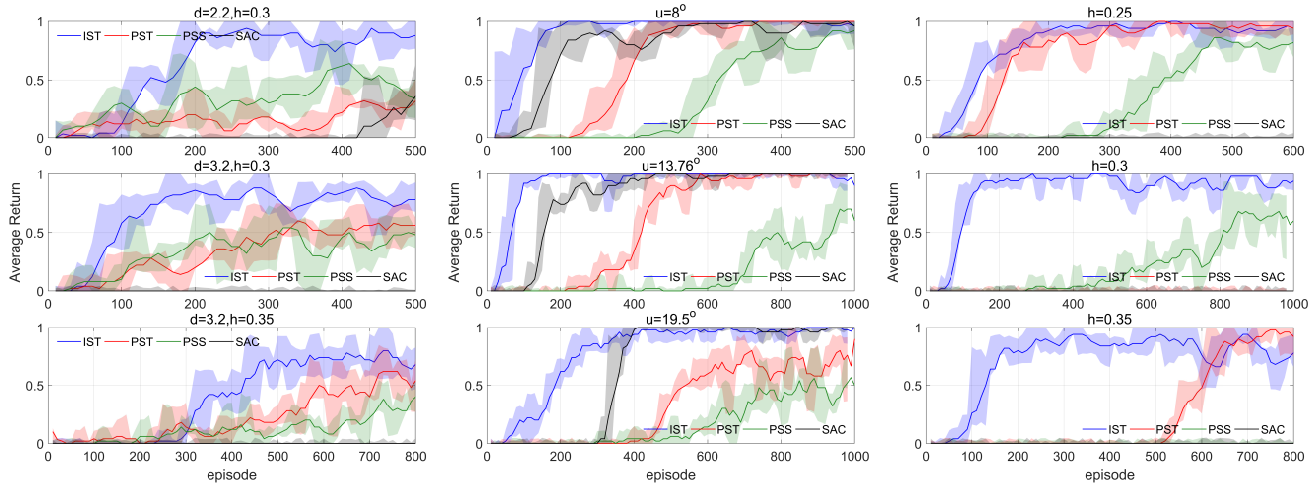
Acknowledgements

We thank the funding support from the Westlake University and Bright Dream Joint Institute for Intelligent Robotics.

[4]These percentages can be changed based on the real situation.



(a) Cumulative reward in HCH. (b) Cumulative reward in HCA. (c) Cumulative reward in HCU.

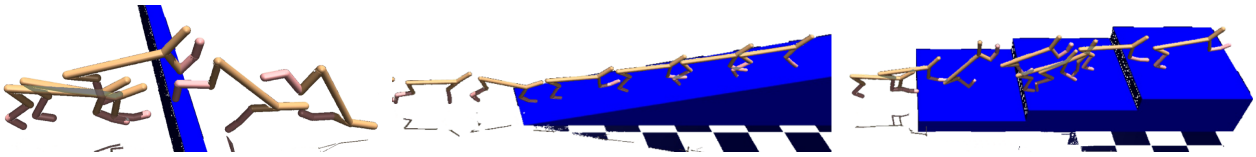


(d) Average return in HCH. (e) Average return in HCA. (f) Average return in HCU.

Figure 7: Reward collection of IST, PST, PSS and SAC on various tasks.

Environment	HCH			HCA			HCU		
Difficulty-level	$d=2.2$ $h=0.3$	$d=3.2$ $h=0.3$	$d=3.2$ $h=0.35$	$u=8°$	$u=13.76°$	$u=19.5°$	$h=0.25$	$h=0.3$	$h=0.35$
IST	**93.2 %**	**84.9 %**	**83.8%**	**100%**	99.3%	97.2%	**98.8%**	**97.4%**	**95.2%**
PST	73.5%	69%	64.7%	99.9%	99.1%	97.2%	97.3%	97.1%	94.8%
PSS	50.1%	32.4%	37.2%	95%	58.8%	45.1%	75.8%	72.3%	–
SAC	80.5%	75.4%	–	99.2%	**99.7%**	**98.1%**	–	–	–

Table 1: Success rate of IST, PST, PSS and SAC over HCH, HCA and HCU within 1000 episodes, where '–' denotes failure.



(a) HCH. (b) HCA. (c) HCU.

Figure 8: For IST, we collect and show one trajectory in each of complex tasks.

References

[Bach and Jordan, 2002] Francis R Bach and Michael I Jordan. Kernel independent component analysis. *Journal of machine learning research*, 3(Jul):1–48, 2002.

[Bell and Sejnowski, 1995] Anthony J Bell and Terrence J Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural computation*, 7(6):1129–1159, 1995.

[Bengio, 2017] Yoshua Bengio. The consciousness prior. *arXiv preprint arXiv:1709.08568*, 2017.

[Bryant and Yarnold, 1995] Fred B Bryant and Paul R Yarnold. Principal-components analysis and exploratory and confirmatory factor analysis. 1995.

[Coros *et al.*, 2009] Stelian Coros, Philippe Beaudoin, and Michiel Van de Panne. Robust task-based control policies for physics-based characters. In *ACM Transactions on Graphics (TOG)*, volume 28, page 170. ACM, 2009.

[Eysenbach *et al.*, 2018] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018.

[Frans *et al.*, 2017] Kevin Frans, Jonathan Ho, Xi Chen, Pieter Abbeel, and John Schulman. Meta learning shared hierarchies. *arXiv preprint arXiv:1710.09767*, 2017.

[Gregor *et al.*, 2016] Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*, 2016.

[Haarnoja *et al.*, 2018a] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

[Haarnoja *et al.*, 2018b] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

[Hausknecht and Stone, 2015] Matthew Hausknecht and Peter Stone. Deep reinforcement learning in parameterized action space. *arXiv preprint arXiv:1511.04143*, 2015.

[Hyvärinen and Oja, 2000] Aapo Hyvärinen and Erkki Oja. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430, 2000.

[Kulkarni *et al.*, 2016] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Advances in neural information processing systems*, pages 3675–3683, 2016.

[Mohamed and Rezende, 2015] Shakir Mohamed and Danilo Jimenez Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *Advances in neural information processing systems*, pages 2125–2133, 2015.

[Oja and Yuan, 2006] Erkki Oja and Zhijian Yuan. The fastica algorithm revisited: Convergence analysis. *IEEE transactions on neural networks*, 17:1370–81, 12 2006.

[Peng *et al.*, 2019] Xue Bin Peng, Michael Chang, Grace Zhang, Pieter Abbeel, and Sergey Levine. Mcp: Learning composable hierarchical control with multiplicative compositional policies. *arXiv preprint arXiv:1905.09808*, 2019.

[Peters *et al.*, 2017] Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. MIT press, 2017.

[Sahni *et al.*, 2017] Himanshu Sahni, Saurabh Kumar, Farhan Tejani, and Charles Isbell. Learning to compose skills. *arXiv preprint arXiv:1711.11289*, 2017.

[Schulman *et al.*, 2015] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[Sharma *et al.*, 2019] Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. *arXiv preprint arXiv:1907.01657*, 2019.

[Silver *et al.*, 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.

[Singh *et al.*, 2019] Avi Singh, Larry Yang, Kristian Hartikainen, Chelsea Finn, and Sergey Levine. End-to-end robotic reinforcement learning without reward engineering. *arXiv preprint arXiv:1904.07854*, 2019.

[Wold *et al.*, 1987] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[Zhu *et al.*, 2017] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE, 2017.