LEARNING GENERALIZABLE SKILLS FROM OFFLINE MULTI-TASK DATA FOR MULTI-AGENT COOPERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning cooperative multi-agent policy from offline multi-task data that can generalize to unseen tasks with varying numbers of agents and targets is an attractive problem in many scenarios. Although aggregating general behavior patterns among multiple tasks as skills to improve policy transfer is a promising approach, two primary challenges hinder the further advancement of skill learning in offline multi-task MARL. Firstly, extracting general cooperative behaviors from various action sequences as common skills lack bringing cooperative temporal knowledge into them. Secondly, existing works only involve common skills and can not adaptively choose independent knowledge as task-specific skills in each task for fine-grained action execution. To address these challenges, we propose an approach named **Hi**erarchical and **S**eparate **S**kill **D**iscovering (HiSSD) for generalizable offline multi-task MARL through skill learning. HiSSD leverages a hierarchical framework that jointly learns common and task-specific skills. The common skills learn cooperative temporal knowledge and enable in-sample exploration for offline multi-task MARL. The task-specific skills represent the priors of each task and achieve a task-guided fine-grained action execution. To verify the advancement of our method, we conduct experiments on multi-agent MuJoCo and SMAC benchmarks. After training policy using HiSSD on offline multi-task data, the empirical results show that HiSSD assigns effective cooperative behaviors and obtains superior performance in unseen tasks.

1 INTRODUCTION

Cooperative multi-agent reinforcement learning (MARL) has drawn great attention to many attractive problems such as games, intelligent warehouses, automated driving, and social science (Vinyals et al., 2019; Yun et al., 2022; Gronauer & Diepold, 2022). When it comes to large-scale tasks, the MARL method yields superior performance compared to the traditional control techniques. In most real-world applications, however, building high-fidelity simulators or deploying online interaction can be rather costly or even infeasible. Meanwhile, multi-agent systems are expected to perform flexibility among tasks with varying numbers of agents and targets. To address these issues, training multi-agent policies that can transfer across tasks with various numbers of agents under limited experience has become an attractive direction to tackle real-world multi-agent applications (Wu et al., 2019; Kumar et al., 2019).

Although training the multi-agent policy on a single task and fine-tuning on the target task is a simple way for policy transfer, it has the following drawbacks (Wen et al., 2022; Hu et al., 2020; Yang et al., 2022; Long et al., 2019a): (i) the fine-tuning stage still requires costly interaction. (ii) it lacks the capacity to handle tasks with various numbers of agents and targets. To overcome these issues, existing works leverage Transformer (Vaswani et al., 2017) to enable a flexible population-invariant framework (Long et al., 2019a; Wang et al., 2019). They also learn general cooperative behaviors as common skills from offline multi-task data to improve multi-agent policy transfer. ODIS (Zhang et al., 2022) conducts a two-stage offline multi-task MARL to discover generalizable multi-agent common skills. They pre-train the common skills from a global view and then optimize the policy by discovering the value-maximized skills on the multi-task data. HyGen (Zhang et al., 2024) integrates online and offline learning to ensure both multi-task generalization and trainingefficiency. These methods obtained convincing improvement by learning generalizable common skills and reduced interaction costs during policy transfer.

However, existing works only learn general and reusable cooperation behaviors by aggregating cooperative actions from multi-task data. Equipping offline multi-task MARL with skill learning to improve policy transfer remains an issue. Firstly, extracting general cooperative behaviors from various cooperative actions as common skills lack bringing cooperative temporal knowledge into them. Existing works have demonstrated the significance of learning temporal knowledge in multi-agent cooperation (Xu et al., 2022b; Song et al., 2023). Secondly, existing works in literature mainly focus on discovering task-irrelevant common knowledge. Yet, few of them consider learning task-specific knowledge which is also beneficial for policy transfer in offline multi-task MARL (Yang et al., 2023; Xu et al., 2022a; Bose et al., 2024).

In light of these issues, we propose a framework called **Hi**erarchical and **S**eparate **S**kill **D**iscovering (HiSSD) for generalizable offline multi-task MARL through skill learning. HiSSD separates the knowledge from multi-task data into common and task-specific skills and leverages a hierarchical framework that jointly learns two of them. Concretely, the common skill represents the general cooperation patterns that involve cooperative temporal knowledge and enable in-sample dynamics exploration for offline multi-task MARL. The task-specific skill represents the unique knowledge of action execution in different tasks and achieves a task-guided fine-grained action execution. Therefore, HiSSD effectively bridges offline multi-agent policy improvement and adaptive multi-task action execution with common and task-specific skills learning.

Overall, our contributions can be summarized as following points: (i) We present HiSSD, an offline multi-task MARL method that leverages the hierarchical framework and jointly learns common and task-specific skills. (ii) HiSSD is proposed to learn common skills representing cooperative behaviors among multiple tasks for offline multi-agent policy exploration and action guidance. (iii) Meanwhile, HiSSD adaptively abstracts task-specific skills for each task to achieve a task-guided fine-grained imitation. (iv) We conduct experiments on the SMAC and multi-agent MuJoCo benchmarks. After training policy using HiSSD on offline multi-task data, the empirical results show that HiSSD assigns effective cooperative behaviors and obtains superior performance in unseen tasks.

2 Preliminaries

2.1 Cooperative Multi-Agent Reinforcement Learning

Cooperative multi-agent reinforcement learning is formulated as a decentralized partially observable Markov decision process (Dec-POMDP) (Oliehoek et al., 2016), as the problem is defined by a tuple $\mathcal{G} = \langle K, \mathcal{S}, \Omega, \mathcal{A}, \mathcal{P}, O, \mathcal{R}, \gamma \rangle$. Here, $K = \{1, ..., k\}$ is the set of agents. The global observation $s \in \mathcal{S}$ is unobservable to each agent in the centralized training and decentralized execution (CTDE) pipeline. Cooperative agent k use local observation $o_k \in \Omega$ drawn from the observation function $O(s, k)$ to sample actions a_k from the actions space $\mathcal{A}$. During interaction, the joint action $\mathbf{u} = \{a_1, a_2, ..., a_k\}$ leads to a next state $s' \sim \mathcal{P}(s'|s, \mathbf{u})$ and a global reward r. The training goal is to learn a cooperative multi-agent system to maximize the cumulative reward $\mathcal{R}$. We use $\boldsymbol{\tau} = \{\tau_1, \tau_2, ..., \tau_k\}$ to denote the trajectory of each agent, and the policy evaluation used to estimate the performance of the joint policy $\boldsymbol{\pi}(\mathbf{u}|\boldsymbol{\tau})$ is normally defined by rewards in infinite-horizon tasks,

$$\mathbb{E}\left[\mathcal{R}\right] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t(s_t, \mathbf{u}_t, s_{t+1}|\boldsymbol{\pi})\right], \tag{1}$$

where $\gamma \in [0, 1)$ is the discount factor, r_t denotes the reward at time t.

2.2 Learning Generalizable Policy from Offline Multi-Task Data

While cooperative multi-agent reinforcement learning has achieved advancement in many scenarios, it is still hard to transfer the policy to unseen tasks without additional interaction (Hu et al., 2020; Iqbal et al., 2021b). An effective solution is leveraging multi-task learning, extracting generalizable knowledge across tasks to improve policy transfer. During multi-task learning, tasks in the same task set share the same type of units but have different distributions, e.g., the number of agents and targets may change among the tasks. Denote $\mathcal{M}$ as overall tasks, the whole data $\mathcal{D}^{\mathcal{M}}$ are divided into source task data $\mathcal{D}^{\text{Source}}$ (or $\mathcal{D}^{\mathcal{T}}$) and target task data $\mathcal{D}^{\text{Target}}$, where the target tasks are unseen during training. The goal is to train the multi-agent policy on the source task data that can be transferred to unseen tasks in the same task set without additional interaction.

3 Method

In this section, we propose our Hierarchical and Separable Skill Discovering (HiSSD) for offline multi-task MARL. Our main solution is leveraging the hierarchical skill learning framework and jointly learning common and task-specific skills among multiple cooperation tasks. We begin by illustrating the overall framework of our offline multi-task MARL. We then detail the high-level planner with common skills and the low-level controller with task-specific skills. Finally, we describe the overall objective and training pipeline.

3.1 Offline Multi-Task MARL with Cooperative Skill Learning

Skill is a series of latent variables representing general and reusable knowledge among tasks to guide action execution (Zhang et al., 2022; 2024). Besides the solution proposed by existing works, we give two insights into multi-task skill learning for further advancement in policy transfer. Firstly, integrating cooperative temporal knowledge into common skills helps decision-making. It gives both dynamics transition information and a global perspective into multi-agent policy. Secondly, learning task-specific skills to guide action execution is beneficial to transfer policy adaptively. It brings each task's unique knowledge into the controller and adjusts the output action distribution. In this way, we propose an offline multi-task MARL method that jointly learns common and task-specific skills to improve policy transfer. Figure 1 gives a brief illustration of our framework.

Specifically, our method can be divided into two parts: (a) The high-level planner and (b) The low-level controller. The high-level planner contains a common skills encoder π_{θ_h}, a forward predictor f_ϕ, and a value net V_ξ^{tot}. We feed local observation $o_t^{1:K}$ into π_{θ_h} to extract common skills $c_t^{1:K}$. f_ϕ receives common skills to output the predicted next global state s'_{t+1} and local information $l_{t+1}^{1:K}$. The local information can be seen as the local observation's embedding. The value net receives $o_t^{1:K}$ and $l_t^{1:K}$ to approximate the accumulated reward $\sum \gamma r_t$. The low-level controller includes a task-specific skills encoder g_ω and an action decoder π_{θ_l}. The task-specific skills encoder infers task-specific skills $z_t^{1:K}$ using local observation $o_t^{1:K}$. The action decoder utilizes local observation, common skills, and task-specific skills to generate the real actions $\{a_t'^k \sim \pi_{\theta_l}(\cdot|o_t^k, c_t^k, z_t^k)\}_{k=0}^K$.

Integrating cooperative temporal knowledge into common skills indicates that these skills must be perceivable to the global dynamics transition. The transition falls into two parts, the global state transition and the value estimation. Therefore, the common skill is trained to minimize the prediction error to the real next global state s_{t+1} and maximize the accumulated reward. This training objective enables an offline exploration and integrates the cooperation-related temporal knowledge into common skills by the local-to-global forward transition prediction. As for achieving an adaptive policy transfer, the major request is to distinguish each task's specific knowledge. We learn task-specific skills by matching them with each task's prior distribution so that it acquires the ability to guide the action execution in different tasks adaptively. Moreover, our method does not require global information during execution, which differs from previous work (Liu et al., 2021). We will describe the details of the proposed skill learning in sections 3.2 and 3.3.

3.2 Learning High-Level Planner with Common Skills

In this subsection, we present the high-level planner to learn common skills from offline multi-task MARL data. We start with the probabilistic inference in MARL and construct a training objective to integrate cooperative temporal knowledge into common skills.

Inspired by Levine (2018), we define the problem of discovering the optimal high-level planner with its common skill encoder $\pi_h^\star$ as match the trajectory $p(\tau)$ given by Eq. 2. Here, $p(\tau)$ indicates to maximize K agents' accumulated reward $\sum \gamma r_t$ in each transition $p(s_{t+1}|o_t^{1:K}, c_t^{\star 1:K})$,

$$p(\tau) = \left[p(s_1) \prod_{t=1}^{T} p(s_{t+1}|o_t^{1:K}, c_t^{1:K}) \right] \exp\left(\sum_{t=1}^{T} r(s_t, c_t^{1:K}) \right). \tag{2}$$

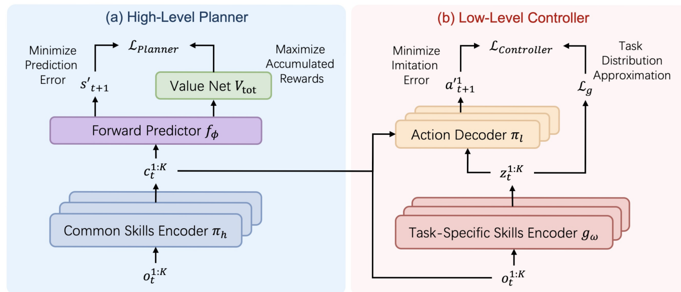

Figure 1: Overall framework of Hi-SSD. HiSSD utilizes a hierarchical framework that jointly learns common and task-specific skills from offline multi-task data to improve multi-agent policy transfer. (a) The high-level planner with common skills. HiSSD integrates cooperative temporal knowledge into common skills and enables an offline exploration. (b) The low-level controller with task-specific skills. The task-specific knowledge can guide the action execution adaptively among tasks. HiSSD uses an implicit Q-learning objective to train the value network.

where s_t denotes the global state at time step t, $o_t^{1:K}$ indicate all agents' local observation, and $c_t^{1:K}$ represent common skills generated by the encoder $c_t^{1:K} \sim \pi_h^\star$. Matching $p(\tau)$ means that the common skill encoder π_{θ_h} in our learned planner needs to generate common skills $c_t^{1:K}$ and roll out trajectories $\hat{p}(\tau)$ that minimize the KL-divergence $D_{\mathrm{KL}}(\hat{p}(\tau)\|p(\tau))$. Meanwhile, learning from offline data requires the planner to be constrained and conservative. The common skill generated by the planner must lead to the next state close to the offline dataset's distribution. Therefore we conduct the learned planner as a one-step predictor and formulate $\hat{p}(\tau)$ by,

$$\hat{p}(\tau) = \left[p(s_1)\prod_{t=1}^{T} p(s_{t+1}|o_t^{1:K}, c_t^{1:K})\right]\prod_{t=1}^{T} q(s'_{t+1}|c_t^{1:K}), \tag{3}$$

where $c_t^{1:K} \sim \pi_{\theta_h}(o_t^{1:K})$ is the common skill inferred by the learned planner. s_{t+1} represent the ground-truth global next state and $q(s'_{t+1}|c_t^{1:K})$ indicates that we use a forward predictor q to predict the next global state using all agents' common skills In this way, we could derive the KL-divergence and formulate our objective as below,

$$\mathcal{L}(\theta_h, \phi) = -D_{\mathrm{KL}}(\hat{p}(\tau)\|p(\tau)) = \mathbb{E}_{\substack{(o_t^{1:K}, s_{t+1})\sim\mathcal{D}^{\mathcal{T}},\\ c_t^{1:K}\sim\pi_{\theta_h}(o_t^{1:K})}}\left[\sum_{t=1}^{T}\underbrace{r(s_t, c_t^{1:K})}_{\text{Exploration}} - \underbrace{\log q(s'_{t+1}|c_t^{1:K})}_{\text{Prediction}}\right], \tag{4}$$

where θ_h denote the parameters of the common skill encoder π_h in the planner, $\mathcal{D}^{\mathcal{T}}$ represents the multi-task dataset. The full derivation can be found in Appendix A.1. Eq. 4 is deployed on all source tasks and aims to extract reusable cooperation knowledge as common skills $c_t^{1:K}$ among tasks. This objective divides the trajectory matching into a trade-off between exploration and prediction which fulfills the requirement of integrating cooperative temporal knowledge into common skills. The *Exploration* term guarantees a value-maximization perspective in common skills to guide the action execution. The *Prediction* term not only achieves a conservative planner in offline learning but also brings the global state information into common skills, which is different from existing works.

To approximate each term in Eq. 4, we introduce a forward predictor f_ϕ for one-step global state prediction and a value net V_ξ^{tot} for reward estimation. f_ϕ receives common skills $c_t^{1:K}$ to predict the next global state s'_{t+1} and next local information $l_{t+1}^{1:K}$. The local information can be seen as the local observation's embedding. V_ξ^{tot} uses the current observation $o_t^{1:K}$ or the local information $l^{1:K}$ to estimate the accumulated reward. The goal of Eq. 4 is to maximize the estimated reward and minimize the prediction error. To train the value net from offline dataset, we utilize the Implicit Q-learning objective (Kostrikov et al., 2021) given by,

$$\mathcal{L}_{\mathrm{IQL}}(\xi) = \mathbb{E}_{\mathcal{D}^{\mathcal{T}}}\left[L_2^\epsilon\left(r_t + \gamma\bar{V}_{\bar{\xi}}^{\text{tot}}(o_{t+1}^{1:k}) - V_\xi^{\text{tot}}(o_t^{1:k})\right)\right], \text{where } L_2^\epsilon(n) = |\epsilon - \mathbb{1}(n<0)|n^2, \tag{5}$$

where $\epsilon \in (0,1)$ and the target value function $\bar{V}_{\bar{\xi}}^{\text{tot}}$ is the momentum version of V_ξ^{tot}, $\mathcal{D}^{\mathcal{T}}$ is the multi-task dataset. This objective downweights the contributions of the TD-residual smaller than 0 while giving more weight to larger values. By substituting f_ϕ and V_ξ^{tot} into Eq. 4, we could rewrite the empirical objective for learning high-level planner with common skills as below,

$$\mathcal{L}_{\text{Planner}}(\theta_h, \phi) = \mathbb{E}_{\substack{(o_t^{1:K}, s_{t+1})\sim\mathcal{D}^{\mathcal{T}}, \\ c_t^{1:K}\sim\pi_{\theta_h}(o_t^{1:K})}} \left[V_\xi^{\text{tot}}(l_{t+1}^{1:K}) - \alpha \log f_\phi(s'_{t+1}|c_t^{1:K})\right], \tag{6}$$

where $l_{t+1}^{1:K} \sim f_\phi(\cdot|c_t^{1:K})$ is the predicted local information. The weight α serves as the trade-off between guiding to space with high-reward and space that the execution policy ought to have a correct imitation. Inspired by Xu et al. (2022a), we introduce an alternative objective that implicitly involves the behavior constraint by using the TD-residual $[r+\gamma\bar{V}_{t+1}^{\text{tot}} - V_t^{\text{tot}}]$ as the imitation weight,

$$\mathcal{L}_{\text{Planner}}(\theta_h, \phi) = \mathbb{E}_{\substack{(o_t^{1:K}, s_{t+1})\sim\mathcal{D}^{\mathcal{T}}, \\ c_t^{1:K}\sim\pi_{\theta_h}(o_t^{1:K})}} \left[\exp\left(\frac{r + \gamma\bar{V}_\xi^{\text{tot}}(l_{t+1}^{1:K}) - V_\xi^{\text{tot}}(o_t^{1:K})}{\alpha}\right) \log f_\phi(s'_{t+1}|c_t^{1:K})\right], \tag{7}$$

Following this objective, the common skill acquires a global cooperative perspective and is more likely to represent behavior patterns with high rewards from offline multi-task data.

3.3 Learning Low-Level Controller with Task-Specific Skills

After constructing the objective of common skills, we now turn to the low-level controller with task-specific skills $z_i^{1:K}$, where $i \in \mathcal{T}$ denotes the current task. Here we omit the subscript of timestep t to simplify the notation. The main function of our proposed controller is to generate real actions following the skills' guidance. Meanwhile, the job of the task-specific skill is to recognize the current task and guide the policy adaptively. To be more specific, we denote the low-level controller's components task-specific skills encoder as $g_\omega(z_i^{1:K}|o_i^{1:K})$ and the action decoder $\pi_{\theta_l}(a_i^{1:K}|z_i^{1:K}, o_i^{1:K}, c_i^{1:K})$. We adopt an objective based on β-VAE (Higgins et al., 2017) to learn action execution and utilize Self-Supervised Learning to learn task-specific representations for regularization. This learning objective is given by,

$$\mathcal{L}_{\text{Controller}}(\theta_l, \omega) = -\mathbb{E}_{\substack{\mathcal{D}^i\in\mathcal{T} \\ z_i\sim g_\omega(o_i)}} \left[\log \pi_{\theta_l}(a_i^{1:K}|o_i^{1:K}, c_i^{1:K}, z_i^{1:K}) - \beta D_{\text{KL}}(z_i^{1:K}\|p(\mathcal{D}^i))\right], \tag{8}$$

where the $\mathcal{T}$ denotes the task number, i denotes the current task, θ_l and ω represent the parameters of action decoder π_{θ_l} and task-specific skills encoder g_ω respectively, and β is the regularization coefficient. The left term requires the action decoder π_{θ_l} to maximize the likelihood of the real action $a_i^{1:K}$ from offline data, the right term minimizes the KL-divergence between the task-specific skills $z_i^{1:K}$ and the task priors $p(\mathcal{D}^i)$ as a regularization. We show the full derivation in Appendix A.2. Notably, the task-specific skill $z_i^{1:K}$ is regularized to approach the current task's unreachable prior knowledge $p(\mathcal{D}^i)$. We find that this problem can be formulated as a *contrastive learning* objective and present the lower bound in Theorem 3.1. By training the skill encoder to distinguish different tasks, it acquires the capability to integrate task-specific knowledge into $z_i^{1:K}$.

Theorem 3.1. Denote a set of N training tasks with their offline data $\mathcal{D}^{\mathcal{T}}$, $\mathcal{D}^i$ is the data of the sampled task. Let random variables x be some observations sampled from $\mathcal{D}^i$, skill $z \sim g_\omega(z|x)$, $h(x,z) = \frac{g_\omega(z|x)}{p(z)}$, $p(\mathcal{D}^i)$ is the prior distribution of current task i, then we have

$$-\mathbb{E}_{\mathcal{D}^i,z,x}\left[\log \frac{h(z,x)}{\sum_{\mathcal{D}^j\in\mathcal{D}^{\mathcal{T}}} h(z,x^j)}\right] \geq -\mathbb{E}_{x\sim\mathcal{D}^i}\left[D_{\text{KL}}(g(\cdot|x)\|p(\mathcal{D}^i)\right] \tag{9}$$

where x^j are observations sampled from task $\mathcal{D}^j$ and $i \in \{1, 2, \cdots, N\}$.

We leave the proof of Theorem 3.1 to Appendix A.3. In practice, we utilize the exponential cosine similarity $\exp(z \cdot x)$ to approximate $h(z,x)$. The sampled x must represent the current task's distribution. We find that each agent's local observation in the same task corresponds to different perspectives of the same task. Therefore, we randomly sample two agents' observations $\{o_i^m, o_i^n\}$ in the same task as the positive pairs (q, k_+) and regard trajectories from other tasks as the negative samples k_-. We introduce $g_{\bar{\omega}}$ the momentum version of g_ω to optimize the contrastive loss,

$$\mathcal{L}_g(\omega) = -\sum_{\mathcal{D}^i\in\mathcal{D}^{\mathcal{T}}} \mathbb{E}_{\substack{\{q,k_+\}\sim\mathcal{D}^i, \\ k_-\sim\mathcal{D}^{\mathcal{T}\setminus i}}} \left[\log \frac{\exp(g_\omega(q)\cdot g_{\bar{\omega}}(k_+)/\sigma)}{\exp(g_\omega(q)\cdot g_{\bar{\omega}}(k_+)/\sigma) + \sum_{\mathcal{T}\setminus i}\exp(g_\omega(q)\cdot g_{\bar{\omega}}(k_-)/\sigma)}\right], \tag{10}$$

where $\mathcal{D}^{\mathcal{T}}$ is overall source tasks, $\mathcal{D}^i$ and $\mathcal{D}^{\mathcal{T}\setminus i}$ denote the current task and other tasks in $\mathcal{D}^{\mathcal{T}}$, respectively. σ denotes the temperature and $g_{\bar{\omega}}$ is updated by the exponential moving average $\bar{\omega}' \leftarrow \eta\omega + (1-\eta)\bar{\omega}$. This training pipeline empowers the policy to embed task-specific knowledge and achieve adaptive action execution.

In summary, we integrate Eqs. 9 and 10 into Eq. 8 to obtain the empirical objective of training the low-level controller. We propose to learn the prior of the current task using Eq. 10 and replace the D_{KL} term in Eq. 8 with it. The objective is given by,

$$\mathcal{L}_{\text{Controller}}(\theta_l, \omega) = -\sum_{\mathcal{D}^i \in \mathcal{D}^{\mathcal{T}}} \mathbb{E}_{(o_i, a_i)\sim\mathcal{D}^i}\left[\log \pi_{\theta_l}(a_i^{1:K} | o_i^{1:K}, c_i^{1:K}, z_i^{1:K})\right] - \beta\mathcal{L}_g(\omega). \tag{11}$$

3.4 Training and Evaluation

HiSSD is fully trained offline and can be trained end-to-end. During training, each task's data $\mathcal{D}^i$ in the source dataset $\mathcal{D}^{\mathcal{T}}$ will be chosen for training. In every training step, we sequentially optimize Eqs. 5, 7, and 11 to train the value net, the planner, and the controller, respectively. At the test time, we only use local information to perform decentralized execution. The planner π_{θ_h} infers the common skill $c_t^{1:K}$ at each time step. The controller π_{θ_l} first generates task-specific skill $z_t^{1:K}$ and then feed $\{(o_t^k, c_t^k, z_t^k)\}_{k=0}^{K}$ into action decoder to generate the real action. Due to space limitations, we leave the pseudocode in Algorithm 1 in Appendix B.

4 Experiments

4.1 Benchmarks and Datasets

SMAC The StarCraft Multi-Agent Challenge (SMAC) (Samvelyan et al., 2019) is a popular MARL benchmark and can evaluate multi-task learning or policy transfer methods. We follow the experimental settings by Zhang et al. (2022) and use the offline dataset they collected. Similar to the D4RL benchmark (Fu et al., 2020), there are four dataset qualities labeled as *Expert*, *Medium*, *Medium-Expert*, and *Medium-Replay*. We construct task sets Marine-Easy and Marine-Hard. In each task set, units in different tasks have the same type and various numbers, all algorithms are trained on offline data from multiple source tasks and evaluated on a wide range of unseen tasks without additional data. Details are referred to the Appendix C.

MAMuJoCo Multi-Agent MuJoCo (MAMuJoCo) is a benchmark for continuous multi-agent robotic control, based on the MuJoCo environment. To fulfill the requirement of offline multi-task learning, we follow Wang et al. (2023) and collect a multi-task dataset in *HalfCheetah-v2* using HAPPO (Kuba et al., 2022a) algorithm. We partition the robotic into six agents and construct the individual task by disabling each agent. Each task's name corresponds to the joint controlled by the disabled agent. Algorithms are trained on multiple source tasks and evaluated on unseen tasks without additional data. Details of the dataset are presented in Appendix C.

4.2 Baselines

SMAC To evaluate the capacity of policy transfer using HiSSD, we introduce several comparable baselines from prior works: (i) ODIS (Zhang et al., 2022), an effective offline multi-task MARL method for cooperative skill discovery. (ii) UPDeT-m, an offline variant of UPDeT (Hu et al., 2020) by adopting the transformer-based Q mixing network. (iii) Transformer-based behavior cloning (**BC-t**) method and its variants with return-to-go information (**BC-r**). We average Hi-SSD's performance over 5 random seeds and report the best score for each task.

MAMuJoCo For the continuous robotic control task, we compare HiSSD with four recent offline MARL algorithms: (i) Behavior cloning (**BC**) method, the multi-agent version of (ii) IQL Kostrikov et al. (2021), (iii) TD3-CQL (Kumar et al., 2020), and (iv) TD3-BC (Fujimoto & Gu, 2021). Each algorithm is evaluated by using 32 independent episodes and runs with 4 seeds for training. Notably, all algorithms use the same architecture, and the details for the implementations and hyperparameters of algorithms in MAMuJoCo are shown in Appendix D.

Table 1: Average test win rates of the best policies over five random seeds in the task set *Marine-Hard* with different data qualities. For simplicity, the asymmetric task names are abbreviated. For example, the task name "5m6m" denotes the SMAC map "5m_vs_6m". Results of BC-best stands for the best test win rates between BC-t and BC-r.

Tasks	Expert				Medium			
	BC-best	UPDeT-m	ODIS	HiSSD (Ours)	BC-best	UPDeT-m	ODIS	HiSSD (Ours)
Source Tasks								
3m	97.7± 2.6	82.8±16.0	98.4± 2.7	**99.5± 0.3**	65.4±14.7	51.2± 3.4	**85.9±10.5**	62.7± 5.7
5m6m	50.4± 2.3	17.2±28.0	53.9± 5.1	**66.1± 7.0**	21.9± 3.4	6.3± 4.9	22.7± 7.1	**26.4± 3.8**
9m10m	95.3± 1.6	3.1± 5.4	80.4± 8.7	**95.5± 2.7**	63.8±10.9	28.5±10.2	**78.1± 3.8**	73.9± 2.3
Unseen Tasks								
4m	92.1± 3.5	33.0±27.1	95.3± 3.5	**99.2± 1.2**	48.8±21.1	14.1± 5.2	61.7±17.7	**77.3±10.2**
5m	87.1±10.5	33.6±40.2	89.1±10.0	**99.2± 1.2**	76.6±14.1	67.2±21.3	85.9±11.8	**88.4± 8.4**
10m	90.5± 3.8	54.7±44.4	93.8± 2.2	**98.4± 0.8**	56.2±20.6	32.9±11.3	61.3±11.3	**98.0± 0.3**
12m	70.8±15.2	17.2±28.0	58.6±11.8	**75.5±19.7**	24.0±10.5	3.2± 3.8	35.9± 8.1	**86.4± 6.0**
7m8m	18.8± 3.1	0.0± 0.0	25.0±15.1	**35.3± 9.8**	1.6± 1.6	0.0± 0.0	**28.1±22.0**	14.2±10.1
8m9m	15.8± 3.3	0.0± 0.0	19.6± 6.0	**47.0± 6.2**	3.1± 3.8	2.3± 2.6	4.7± 2.7	**15.3± 2.8**
10m11m	45.3±11.1	0.0± 0.0	42.4± 7.2	**86.3±14.6**	19.7± 8.9	4.0± 3.4	29.7±15.4	**43.6± 4.6**
10m12m	1.0± 1.5	0.0± 0.0	1.6± 1.6	**14.5± 9.1**	0.0± 0.0	0.0± 0.0	**1.6± 1.6**	0.6± 0.5
13m15m	0.0± 0.0	0.0± 0.0	**2.3± 2.6**	1.3± 2.5	0.6± 1.3	0.0± 0.0	**1.6± 1.6**	1.4± 2.4
	Medium-Expert				Medium-Replay			
Source Tasks								
3m	67.7±23.7	85.2±17.9	73.6±22.0	**86.6± 3.7**	81.1± 8.8	41.4±20.1	**83.6±14.0**	78.8± 5.3
5m6m	31.3± 6.3	1.6± 1.6	9.4± 2.2	**41.9± 9.7**	25.0± 3.1	0.8± 1.4	16.6± 4.7	**25.3±10.3**
9m10m	26.0±13.9	24.3±18.7	31.3±14.5	**83.6± 6.9**	33.4±13.1	0.8± 1.4	34.4± 8.0	**45.8± 3.9**
Unseen Tasks								
4m	81.3±18.9	43.9±39.0	82.8±13.5	**91.1± 6.1**	61.5± 9.0	35.9±12.6	55.6±14.5	**77.3± 1.9**
5m	74.0± 2.9	33.6±40.2	82.8±17.7	**98.3± 1.8**	75.0±24.2	61.7±20.3	**96.1± 4.1**	88.1±13.4
10m	78.1± 6.7	32.8±38.1	82.8±16.8	**96.4± 2.1**	82.4± 8.2	11.0± 7.8	84.4±15.1	**94.7± 2.6**
12m	64.8±24.3	9.4± 8.6	81.3±20.6	**88.4±11.8**	83.4± 4.5	2.3± 2.6	84.4± 6.6	**90.3± 3.6**
7m8m	13.3± 4.5	2.3± 4.1	15.6± 4.4	**30.5±10.4**	7.3± 6.4	1.6± 2.7	9.4± 2.2	**21.7± 4.7**
8m9m	10.2± 4.6	9.5± 8.6	10.9± 4.7	**35.2±18.3**	11.5± 3.9	0.8± 1.4	11.7± 8.7	**14.5± 4.0**
10m11m	26.6± 4.7	11.8± 8.1	33.6± 8.9	**54.7± 6.8**	**46.8± 6.6**	0.8± 1.4	35.9± 5.2	42.5± 4.4
10m12m	0.0± 0.0	0.0± 0.0	1.6± 1.6	**2.5± 1.0**	1.6± 2.7	0.0± 0.0	**2.3± 1.4**	0.5± 0.3
13m15m	0.8± 1.4	0.0± 0.0	2.3± 2.6	**5.2± 3.7**	1.6± 1.6	0.0± 0.0	2.4± 1.4	**3.6± 2.1**

4.3 Main Results

SMAC We evaluate Hi-SSD and baselines on SMAC and present the average test win rates in the Marine-Hard task set in Table 1. BC-best represents the highest test win rates between BC-t and BC-r. Below are the key results: (i) HiSSD achieves top performance on over half of the tasks. This indicates that using skills to guide action execution benefits policy transfer. (ii) Compared to ODIS, which only discovers common skills from offline multi-task data, our method outperforms it in medium and near-optimal data qualities, showing advancement in learning task-specific skills. Due to space limitation, we leave results on other task sets in appendix E.

MAMuJoCo Table 2 shows the mean and standard deviation of average returns for the offline multi-task learning on our HalfCheetah-v2 task set. The results show that HiSSD outperforms all baselines and achieves state-of-the-art performance in most tasks. Compared to baselines with only behavior cloning (BC) or conservative Q-learning (TD3-CQL), HiSSD outperforms in a wide range. We also find that compared to TD3-BC which only constrains the action execution, HiSSD transfers to the unseen tasks better. It indicates that learning cooperative temporal knowledge and task-specific skills is beneficial for learning generalizable multi-agent policy from offline multi-task data.

4.4 Ablation Study

In this section, we present more empirical results for deep analysis of HiSSD. We first show the effectiveness of our proposed skills learning which integrates cooperative temporal and task-specific knowledge. We then investigate what is the essential factor of training the task-specific skills. Finally, we visualize the learned skills for a more clear analysis.

Table 2: Average scores on HalfCheetah-v2 multi-task datasets in MAMuJoCo.

Tasks	BC	IQL	ITD3-CQL	ITD3-BC	HiSSD(ours)
Source Tasks					
complete	3188.16±566.68	4384.23±198.55	-582.41± 55.23	4365.10± 72.92	**4450.57±126.36**
back thigh	3324.22± 58.49	3675.91± 18.99	-476.23± 20.83	3685.82± 40.84	**3698.38± 13.98**
back foot	3079.01±355.66	3989.12±211.42	-487.19± 15.68	**4119.48± 61.00**	3197.83± 6.99
front thigh	1861.53±415.80	**2744.63±329.00**	-471.99± 49.16	2700.17±407.46	1948.74± 81.24
front shin	1819.94±273.96	4048.43±363.79	-491.41± 7.21	**4155.15±180.99**	3468.32±290.72
Unseen Tasks					
back shin	1964.55±268.24	1974.62±314.33	-492.87±27.02	1690.40±251.77	**3472.12± 91.95**
front foot	3468.40±369.40	3948.17±381.80	-463.22±52.85	3683.16±419.42	**4175.29±338.96**

Table 3: Ablation studies on HiSSD. We report average test win rates of the best policies over five random seeds in the task set *Marine-Hard* with different data qualities.

Data Qualities	w/o Planning	w/o Predicting	Half-Negative	L2-Loss	BC-best	HiSSD
Source Tasks						
Expert	80.7±22.4	85.9±16.4	**87.0±15.3**	80.8±21.2	81.1±21.8	84.5±17.1
Medium	52.7±25.5	53.1±20.0	54.3±20.6	42.3±21.8	50.3±20.1	**55.9±19.2**
Medium-Expert	56.0±22.4	63.5±26.1	**70.7±21.3**	57.8±27.1	41.7±18.5	64.2±23.5
Medium-Replay	51.1±28.2	47.7±25.1	50.0±22.9	44.7±28.1	46.5±26.7	**51.9±24.0**
Target Tasks						
Expert	45.8±39.0	55.0±37.4	60.9±37.1	53.2±38.9	46.8±36.8	**61.2±37.4**
Medium	38.4±37.5	37.6±34.2	46.7±38.0	42.4±38.6	25.6±26.8	**48.6±40.1**
Medium-Expert	47.2±36.9	51.2±34.3	55.8±37.6	51.0±38.9	38.8±33.0	**57.6±37.5**
Medium-Replay	45.0±37.8	42.8±36.2	48.1±37.6	47.9±40.3	41.2±33.7	**48.7±39.0**

Common Skills Analysis We conduct experiments on SMAC's Marine-Hard task set to demonstrate the effects of our proposed skills learning paradigm in HiSSD and present results in Table 3. We implement two variants of HiSSD to show the impact of three factors on HiSSD's skills learning, respectively: (i) *w/o Planning*. HiSSD only learns task-specific skills among tasks. (ii) *w/o Prediction*. HiSSD trains the planner without the next global state prediction. The results indicate that learning common skills improves policy transfer, and learning to predict the next global state acquires further advancement.

Task-Specific Skills Analysis To investigate what is the essential factor of learning task-specific skills, we conduct experiments on SMAC's Marine-Hard task set with three variants of HiSSD: (i) *Half-Negative*. We reduce to half of the negative samples during contrastive learning. (ii) *L2-Loss*. The objective is replaced with one similar to Grill et al. (2020) which does not require negative samples. According to the results in 3, learning to distinguish the difference between tasks plays a key role in learning task-specific skills. Meanwhile, increasing the number of negative samples further improves the capacity of multi-agent policy transfer.

4.5 Visualization of Learned Skills

To investigate the effectiveness of our proposed method more clearly, we evaluate HiSSD on multiple tasks in SMAC and visualize the learned skills using t-SNE (Hinton & Roweis, 2002).

Common Skills In Figure 2, we collect trajectories in multiple tasks and visualize the learned common skills. Neighboring points in the same distribution represent similar common skills. We partition these trajectories into four sub-parts by timestep to show the skill flow. According to the plots, common skills are mapped into several clusters, and each cluster contains skills from different tasks. Points with the same color in each distribution represent the collaboration between agents in the corresponding task. The results demonstrate that HiSSD acquires the capability to learn task-irrelevant common skills.

Task-Specific Skills Figure 3 represented the chosen task-specific skills during evaluation. We collect trajectories on multiple tasks in SMAC using HiSSD and partition these trajectories into

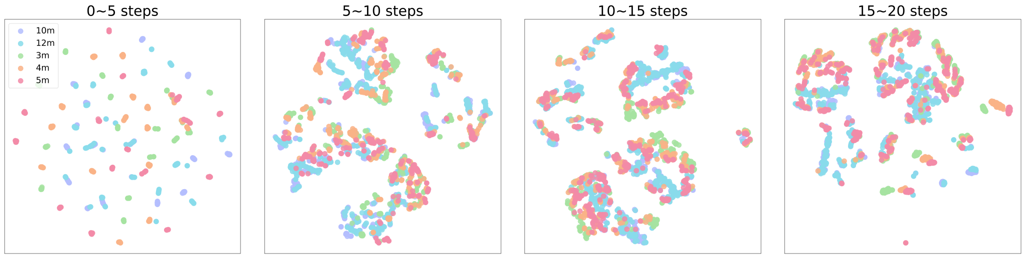

Figure 2: Visualization of learned common skills. We use HiSSD to collect trajectories on five tasks in SMAC and partition these trajectories into four time windows. Plots in each figure represent the distribution of chosen common skills. We use multiple time windows to indicate the task flow.

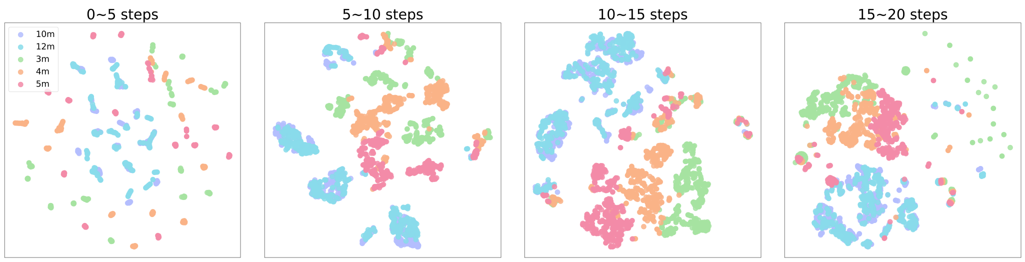

Figure 3: Visualization of learned task-specific skills. We use HiSSD to collect trajectories on five tasks in SMAC and partition these trajectories into four time windows. Plots represent the distribution of chosen task-specific skills. We use multiple time windows to indicate the task flow.

four sub-parts by timestep. From the plots, task-specific skills chosen in small-scale tasks (i.e., *3m*, *4m*, and *5m*) are mapped into different distributions. Large-scale tasks (i.e., *10m* and *12m*) will overlap with each other. For the small-scale tasks that are similar to the source tasks, the discrepancy between these tasks can be learned more efficiently, and policy can be easily generalized to the small-scale target tasks. As for the large-scale tasks (i.e., 10m and 12m), the policy is unable to capture significant discrepancies and can only infer knowledge related to the source tasks, therefore causing the distribution overlap. Moreover, with steps going on, the distribution distance is getting closer, indicating that our method can adaptively capture the dynamic transitions in different tasks. As timestep goes on, points from different tasks will become closer. This is because agents will be killed as the timestep goes on, tasks with numerous agents will degrade to various sub-tasks at the mid-term of each episode. The results indicate that our task-specific skills effectively integrate task-related knowledge, and the policy trained by HiSSD can adaptively distinguish different tasks.

5 Related Work

5.1 Offline MARL

Training policies from offline experience without interaction effectively reduce the trial and error costs when implementing RL in real-world scenarios (Levine et al., 2020; Zhang et al., 2021). Due to the distribution shift in offline learning (Fujimoto et al., 2019), training a policy from static datasets faces unexpected extrapolation errors when estimating unseen data (Wu et al., 2019; Kumar et al., 2019). Therefore, previous works consider learning behavior-constrained policies (Kumar et al., 2020; Kostrikov et al., 2021), which can be extended to the MARL paradigm. They aim at adopting sufficient conservatism to current online MARL methods (Yang et al., 2021; Jiang & Lu, 2021; Pan et al., 2022), training policies by value function decomposition (Sunehag et al., 2018; Rashid et al., 2020; Yang et al., 2020b; Wang et al., 2020), or multi-agent policy gradient algorithms (Lowe et al., 2017; Foerster et al., 2018; Iqbal & Sha, 2019; Yu et al., 2022; Kuba et al., 2022b). Another effective way for offline learning is leveraging the powerful transformer-based model (Chen et al., 2021; Lee et al., 2022; Meng et al., 2023; Liu et al., 2023) or diffusion model (Janner et al., 2022; Pearce et al., 2022; He et al., 2024). Yet the problem of combining generative models with the policy improvement paradigm of reinforcement learning remains unsolved (Zheng et al., 2022; Wang et al., 2022; Kang et al., 2024).

5.2 Multi-Task MARL

Multi-task learning plays a key role in improving data-efficiency and generalization in MARL. It highlights knowledge reuse (Da Silva & Costa, 2016; Shen et al., 2021; Sodhani et al., 2021), which is beneficial for transferable multi-agent collaboration. This paradigm requires the policy to hold a flexible structure for deploying agents across tasks with varying input dimensions (Agarwal et al., 2020; Wang et al., 2019; Hu et al., 2020; Zhou et al., 2021). Recent works consider multiple ways to realize multi-task adaptations such as policy representations learning (Grover et al., 2018), evolutionary-based curriculum learning (Long et al., 2019b), randomized entity-wise factorization (Iqbal et al., 2021a), high-level cooperation strategy reusing (Liu et al., 2021), and training transformer-based population-invariant policies (Hu et al., 2020; Wen et al., 2022; Liu et al., 2023). Although these methods relieve the need for learning from scratch during transferrin, generalizing policies without simultaneous learning or fine-tuning remains challenging Zhang et al. (2022).

5.3 MARL with Skill Learning

Hierarchical MARL with skill learning is a practical way to solve complex decision-making tasks. This paradigm embeds behavior patterns in a skill space, promoting the exploration of cooperative multi-agents with state empowerment from information theory (Barto & Mahadevan, 2003; Eysenbach et al., 2019; Yang et al., 2024; He et al., 2020; Liu et al., 2022). MASD (He et al., 2020) introduces an information bottleneck for cooperation patterns discovery. HSD (Yang et al., 2020a) and HSL (Liu et al., 2022) utilize hierarchical architectures to discover diverse behaviors. HMASD (Yang et al., 2024) treats skill discrimination as a sequential modeling problem. However, their framework requires global information during execution. VO-MASD (Chen et al., 2024) discovers hierarchy-like cooperation skills in a pre-training stage to speed up the online learning. ODIS (Zhang et al., 2022) combines offline multi-task learning with hierarchical MARL to learn a generalizable multi-agent policy. Although they discover skills following the value function decomposition, they only consider the common skill among tasks and ignore the tasks-specific knowledge. HyGen (Zhang et al., 2024) follows ODIS and integrates online exploration to further improve transfer capability, especially in middle data qualities. In this article, we propose a hierarchical policy that jointly learns common and task-specific skills from offline multi-task data, further enhancing the capacity of multi-agent policy's generalization.

6 Conclusion

In this paper, we propose a new hierarchical multi-agent policy that jointly learns common and task-specific skills from offline multi-task data, further improving the capacity of policy transfer in offline multi-task MARL. We analyze the primary challenges in current offline multi-task MARL methods and propose novel objectives to overcome these challenges. We compare HiSSD to SOTA methods on popular MARL benchmarks and certify that it acquires promising improvement. One limitation of Hi-SSD is its training stability, and we consider it as our future work. Hopefully, our proposed skill learning pipeline can lead to a new branch for offline multi-task learning in MARL.

7 Reproducibility Statement

The source code of our method is provided in the supplementary materials. The full derivation of our proposed objective and theorem is presented in Appendix A. The pseudocode is presented in Appendix B. We leave the detailed description of the used benchmarks and datasets in Appendix C and present the implementation details in Appendix D.

References

Akshat Agarwal, Sumit Kumar, Katia Sycara, and Michael Lewis. Learning transferable cooperative behavior in multi-agent teams. In *Proceedings of the 19th International Conference on*

Autonomous Agents and MultiAgent Systems, pp. 1741–1743, 2020.

Andrew G Barto and Sridhar Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete event dynamic systems*, 13:341–379, 2003.

Avinandan Bose, Simon Shaolei Du, and Maryam Fazel. Offline multi-task transfer rl with representational penalization. *arXiv preprint arXiv:2402.12570*, 2024.

Jiayu Chen, Bhargav Ganguly, Tian Lan, and Vaneet Aggarwal. Variational offline multi-agent skill discovery. *arXiv preprint arXiv:2405.16386*, 2024.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Felipe Leno Da Silva and Anna Helena Reali Costa. Transfer learning for multiagent reinforcement learning systems. In *Proceedings of the twenty-fifth international joint conference on artificial intelligence*, pp. 3982–3983, 2016.

Benjamin Eysenbach, Julian Ibarz, Abhishek Gupta, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pp. 2052–2062. PMLR, 2019.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Sven Gronauer and Klaus Diepold. Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review*, 55(2):895–943, 2022.

Aditya Grover, Maruan Al-Shedivat, Jayesh Gupta, Yuri Burda, and Harrison Edwards. Learning policy representations in multiagent systems. In *International conference on machine learning*, pp. 1802–1811. PMLR, 2018.

Haoran He, Chenjia Bai, Kang Xu, Zhuoran Yang, Weinan Zhang, Dong Wang, Bin Zhao, and Xuelong Li. Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning. *Advances in neural information processing systems*, 36, 2024.

Shuncheng He, Jianzhun Shao, and Xiangyang Ji. Skill discovery of coordination in multi-agent reinforcement learning. *arXiv preprint arXiv:2006.04021*, 2020.

Irina Higgins, Loic Matthey, Arka Pal, Christopher P Burgess, Xavier Glorot, Matthew M Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *ICLR (Poster)*, 3, 2017.

Geoffrey E Hinton and Sam Roweis. Stochastic neighbor embedding. *Advances in neural information processing systems*, 15, 2002.

Siyi Hu, Fengda Zhu, Xiaojun Chang, and Xiaodan Liang. Updet: Universal multi-agent rl via policy decoupling with transformers. In *International Conference on Learning Representations*, 2020.

Shariq Iqbal and Fei Sha. Actor-attention-critic for multi-agent reinforcement learning. In *International conference on machine learning*, pp. 2961–2970. PMLR, 2019.

Shariq Iqbal, Christian A Schroeder De Witt, Bei Peng, Wendelin Böhmer, Shimon Whiteson, and Fei Sha. Randomized entity-wise factorization for multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 4596–4606. PMLR, 2021a.

Shariq Iqbal, Christian A Schroeder De Witt, Bei Peng, Wendelin Böhmer, Shimon Whiteson, and Fei Sha. Randomized entity-wise factorization for multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 4596–4606. PMLR, 2021b.

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pp. 9902–9915. PMLR, 2022.

Jiechuan Jiang and Zongqing Lu. Offline decentralized multi-agent reinforcement learning. *arXiv preprint arXiv:2108.01832*, 2021.

Bingyi Kang, Xiao Ma, Chao Du, Tianyu Pang, and Shuicheng Yan. Efficient diffusion policies for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *Deep RL Workshop NeurIPS 2021*, 2021.

JG Kuba, R Chen, M Wen, Y Wen, F Sun, J Wang, and Y Yang. Trust region policy optimisation in multi-agent reinforcement learning. In *ICLR 2022-10th International Conference on Learning Representations*, pp. 1046. The International Conference on Learning Representations (ICLR), 2022a.

JG Kuba, R Chen, M Wen, Y Wen, F Sun, J Wang, and Y Yang. Trust region policy optimisation in multi-agent reinforcement learning. In *ICLR 2022-10th International Conference on Learning Representations*, pp. 1046. The International Conference on Learning Representations (ICLR), 2022b.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

Kuang-Huei Lee, Ofir Nachum, Mengjiao Sherry Yang, Lisa Lee, Daniel Freeman, Sergio Guadarrama, Ian Fischer, Winnie Xu, Eric Jang, Henryk Michalewski, et al. Multi-game decision transformers. *Advances in Neural Information Processing Systems*, 35:27921–27936, 2022.

Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Bo Liu, Qiang Liu, Peter Stone, Animesh Garg, Yuke Zhu, and Anima Anandkumar. Coach-player multi-agent reinforcement learning for dynamic team composition. In *International Conference on Machine Learning*, pp. 6860–6870. PMLR, 2021.

Jie Liu, Yinmin Zhang, Chuming Li, Chao Yang, Yaodong Yang, Yu Liu, and Wanli Ouyang. Masked pretraining for multi-agent decision making. *arXiv preprint arXiv:2310.11846*, 2023.

Yuntao Liu, Yuan Li, Xinhai Xu, Yong Dou, and Donghong Liu. Heterogeneous skill learning for multi-agent tasks. *Advances in Neural Information Processing Systems*, 35:37011–37023, 2022.

Qian Long, Zihan Zhou, Abhinav Gupta, Fei Fang, Yi Wu, and Xiaolong Wang. Evolutionary population curriculum for scaling multi-agent reinforcement learning. In *International Conference on Learning Representations*, 2019a.

Qian Long, Zihan Zhou, Abhinav Gupta, Fei Fang, Yi Wu, and Xiaolong Wang. Evolutionary population curriculum for scaling multi-agent reinforcement learning. In *International Conference on Learning Representations*, 2019b.

Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

Linghui Meng, Muning Wen, Chenyang Le, Xiyun Li, Dengpeng Xing, Weinan Zhang, Ying Wen, Haifeng Zhang, Jun Wang, Yaodong Yang, et al. Offline pre-trained multi-agent decision transformer. *Machine Intelligence Research*, 20(2):233–248, 2023.

Frans A Oliehoek, Christopher Amato, et al. *A concise introduction to decentralized POMDPs*, volume 1. Springer, 2016.

Ling Pan, Longbo Huang, Tengyu Ma, and Huazhe Xu. Plan better amid conservatism: Offline multi-agent reinforcement learning with actor rectification. In *International conference on machine learning*, pp. 17221–17237. PMLR, 2022.

Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, et al. Imitating human behaviour with diffusion models. In *The Eleventh International Conference on Learning Representations*, 2022.

Bei Peng, Tabish Rashid, Christian Schroeder de Witt, Pierre-Alexandre Kamienny, Philip Torr, Wendelin Böhmer, and Shimon Whiteson. Facmac: Factored multi-agent centralised policy gradients. *Advances in Neural Information Processing Systems*, 34:12208–12221, 2021.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178):1–51, 2020.

Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2186–2188, 2019.

Jiayi Shen, Xiantong Zhen, Marcel Worring, and Ling Shao. Variational multi-task learning with gumbel-softmax priors. *Advances in Neural Information Processing Systems*, 34:21031–21042, 2021.

Shagun Sodhani, Amy Zhang, and Joelle Pineau. Multi-task reinforcement learning with context-based representations. In *International Conference on Machine Learning*, pp. 9767–9779. PMLR, 2021.

Haolin Song, Mingxiao Feng, Wengang Zhou, and Houqiang Li. Ma2cl: masked attentive contrastive learning for multi-agent reinforcement learning. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pp. 4226–4234, 2023.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2085–2087, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Micha"el Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. Qplex: Duplex dueling multi-agent q-learning. In *International Conference on Learning Representations*, 2020.

Weixun Wang, Tianpei Yang, Yong Liu, Jianye Hao, Xiaotian Hao, Yujing Hu, Yingfeng Chen, Changjie Fan, and Yang Gao. Action semantics network: Considering the effects of actions in multiagent systems. In *International Conference on Learning Representations*, 2019.

Xiangsen Wang, Haoran Xu, Yinan Zheng, and Xianyuan Zhan. Offline multi-agent reinforcement learning with implicit global-to-local value regularization. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL `https://openreview.net/forum?id=BXQtgwA2n0`.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2022.

Muning Wen, Jakub Kuba, Runji Lin, Weinan Zhang, Ying Wen, Jun Wang, and Yaodong Yang. Multi-agent reinforcement learning is a sequence modeling problem. *Advances in Neural Information Processing Systems*, 35:16509–16521, 2022.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Haoran Xu, Li Jiang, Li Jianxiong, and Xianyuan Zhan. A policy-guided imitation approach for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:4085–4098, 2022a.

Zhiwei Xu, Bin Zhang, Yuan Zhan, Yunpeng Baiia, Guoliang Fan, et al. Mingling foresight with imagination: Model-based cooperative multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 35:11327–11340, 2022b.

Jiachen Yang, Igor Borovikov, and Hongyuan Zha. Hierarchical cooperative multi-agent reinforcement learning with skill discovery. In *International Conference on Autonomous Agents and Multi-Agent Systems*, 2020a.

Mingyu Yang, Yaodong Yang, Zhenbo Lu, Wengang Zhou, and Houqiang Li. Hierarchical multi-agent skill discovery. *Advances in Neural Information Processing Systems*, 36, 2024.

Yaodong Yang, Jianye Hao, Ben Liao, Kun Shao, Guangyong Chen, Wulong Liu, and Hongyao Tang. Qatten: A general framework for cooperative multiagent reinforcement learning. *arXiv preprint arXiv:2002.03939*, 2020b.

Yaodong Yang, Guangyong Chen, Weixun Wang, Xiaotian Hao, Jianye Hao, and Pheng-Ann Heng. Transformer-based working memory for multiagent reinforcement learning with action parsing. *Advances in Neural Information Processing Systems*, 35:34874–34886, 2022.

Yiqin Yang, Xiaoteng Ma, Chenghao Li, Zewu Zheng, Qiyuan Zhang, Gao Huang, Jun Yang, and Qianchuan Zhao. Believe what you see: Implicit constraint approach for offline multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34:10299–10312, 2021.

Yucheng Yang, Tianyi Zhou, Qiang He, Lei Han, Mykola Pechenizkiy, and Meng Fang. Task adaptation from skills: Information geometry, disentanglement, and new objectives for unsupervised reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2023.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems*, 35:24611–24624, 2022.

Won Joon Yun, Soohyun Park, Joongheon Kim, MyungJae Shin, Soyi Jung, David A Mohaisen, and Jae-Hyun Kim. Cooperative multiagent deep reinforcement learning for reliable surveillance via autonomous multi-uav control. *IEEE Transactions on Industrial Informatics*, 18(10):7086–7096, 2022.

Fuxiang Zhang, Chengxing Jia, Yi-Chen Li, Lei Yuan, Yang Yu, and Zongzhang Zhang. Discovering generalizable multi-agent coordination skills from multi-task offline data. In *The Eleventh International Conference on Learning Representations*, 2022.

Kaiqing Zhang, Zhuoran Yang, Han Liu, Tong Zhang, and Tamer Başar. Finite-sample analysis for decentralized batch multiagent reinforcement learning with networked agents. *IEEE Transactions on Automatic Control*, 66(12):5925–5940, 2021.

Mingliang Zhang, Sichang Su, Chengyang He, and Guillaume Sartoretti. Hybrid training for enhanced multi-task generalization in multi-agent reinforcement learning. *arXiv preprint arXiv:2408.13567*, 2024.

Qinqing Zheng, Amy Zhang, and Aditya Grover. Online decision transformer. In *international conference on machine learning*, pp. 27042–27059. PMLR, 2022.

Tianze Zhou, Fubiao Zhang, Kun Shao, Kai Li, Wenhan Huang, Jun Luo, Weixun Wang, Yaodong Yang, Hangyu Mao, Bin Wang, et al. Cooperative multi-agent transfer learning with level-adaptive credit assignment. *arXiv preprint arXiv:2106.00517*, 2021.

A ADDITIONAL DERIVATION

A.1 OBJECTIVE FOR HIGH-LEVEL PLANNER

In this section, we adopt the probabilistic inference to formulate our objective based on the Levine (2018). We assume that the training goal for behavior cloning is to minimize the distance between the optimal trajectory $p(\tau)$ formulated in Eq. 2 and the rollout trajectory $\hat{p}(\tau)$. $\hat{p}(\tau)$ is required to predict the next global state s'_{t+1} that is not far from the offline dataset. Therefore, we could formulate $\hat{p}(\tau)$ as below,

$$\hat{p}(\tau) = \left[p(s_1)\prod_{t=1}^{T} p(s_{t+1}|o_t^{1:K}, c_t^{1:K})\right]\prod_{t=1}^{T} q(s'_{t+1}|c_t^{1:K}), \tag{12}$$

where $q(\cdot)$ is a predefined predictor, $c_t^{1:K}$ is the learned common skill. A practical way to match this objective is by adopting an optimization-based approximate inference approach. In this article, we minimize the KL-divergence between the approximated trajectory and the optimal trajectory to achieve this objective,

$$D_{\mathrm{KL}}(\hat{p}(\tau)\|p(\tau)) = -\mathbb{E}_{\tau\sim\hat{p}(\tau)}\left[\log p(\tau) - \log\hat{p}(\tau)\right]. \tag{13}$$

In this way, we derive Eq. 13 and obtain the planner's objective,

$$\begin{aligned}
-D_{\mathrm{KL}}(\hat{p}(\tau)\|p(\tau)) &= \mathbb{E}_{\tau\sim\mathcal{D}}\left[\log p(\tau) - \log\hat{p}(\tau)\right] \\
&= \mathbb{E}_{\tau\sim\mathcal{D}}\Bigg[\log p(s_1) + \sum_{t=1}^{T}(\log p(s_{t+1}|o_t^{1:K}, c_t^{1:K}) + r(s_t, c_t^{1:K})) - \\
&\qquad \log p(s_1) - \sum_{t=1}^{T}(\log p(s_{t+1}|o_t^{1:K}, c_t^{1:K}) + \log q(s'_{t+1}|o_t^{1:K}, c_t^{1:K}))\Bigg] \\
&= \mathbb{E}_{\tau\sim\mathcal{D}}\left[\sum_{t=1}^{T} r(s_t, c_t^{1:K}) - \log q(s'_{t+1}|o_t^{1:K}, c_t^{1:K})\right] \\
&= \sum_{t=1}^{T}\mathbb{E}_{\tau\sim\mathcal{D}}\left[r(s_t, c_t^{1:K}) - \log q(s'_{t+1}|o_t^{1:K}, c_t^{1:K})\right].
\end{aligned} \tag{14}$$

A.2 OBJECTIVE FOR LOW-LEVEL CONTROLLER

To formulate the objective for our fine-grained action controller, we perform the controller as a generative model and introduce the variational inference. With the help of Jensen's inequality, we formulate the ELBO as below,

$$\begin{aligned}
\log p(a) = \log\int_z p(a,z)\frac{q(z|o)}{q(z|o)} &= \log\mathbb{E}_{q(z|o)}\left[\frac{p(a,z)}{q(z|o)}\right] \\
&\geq \mathbb{E}_{q(z|o)}\left[\log p(a,z)\right] - \mathbb{E}_{q(z|o)}\left[\log q(z|o)\right] \\
&= \mathbb{E}_{q(z|o)}\left[p(a|z)\cdot p(z)\right] - \mathbb{E}_{q(z|o)}\left[\log q(z|o)\right] \rightarrow \text{ELBO},
\end{aligned} \tag{15}$$

where $q(\cdot)$ is a predefined encoder, o denotes the local observation of each agent, and z denotes the task embedding. Practically, we introduce a task-guided controller π_{θ_l} as $p(a|z)$ and a task discriminator g_ω as $q(z|o)$. The prior distribution $p(z)$ is unreachable to each task during training and we use $p(\mathcal{D}^i)$ to represent the prior distribution of skill z in each task. We also embed the common skill c_t and local observation o_t into π_{θ_l}. Thus, we rewrite Eq. 15 and formulate the objective of our low-level controller in a decentralized manner,

$$\begin{aligned}
\mathcal{L}_{\text{Controller}}(\theta_c,\omega) &= -\mathcal{L}_{\text{ELBO}} \\
&= -\mathbb{E}_{\mathcal{D}^i\sim\mathcal{D}}\left[\mathbb{E}_{g_\omega(z_t^i|o_t^i)}\left[\log\pi_{\theta_l}(a_t^i|z_t^i, o_t^i, c_t^i)\right] + \mathbb{E}_{g_\omega(z_t^i|o_t^i)}\left[\log\frac{p(\mathcal{D}^i)}{g_\omega(z_t^i|o_t^i)}\right]\right] \\
&= -\mathbb{E}_{\mathcal{D}^i\sim\mathcal{D}}\left[\mathbb{E}_{g_\omega(z_t^i|o_t^i)}\left[\log\pi_{\theta_l}(a_t^i|z_t^i, o_t^i, c_t^i)\right] - D_{\mathrm{KL}}(z_t^i\|p(\mathcal{D}^i)))\right],
\end{aligned} \tag{16}$$

where i denotes the current task and $\mathcal{D}$ indecates the offline d ataset.

A.3 Bridging KL-Divergence and Contrastive Learning

In this section, we illustrate how to bridge the contrastive loss to approximate the KL-divergence in Eq. 8 We first introduce a lemma. Then we give a proof of Theorem 3.1.

Lemma A.1. Given $\mathcal{D}^i \sim \mathcal{D}^{\mathcal{T}}$ as the the current task's data distribution. Denote x as the local observations sampled from the offline data $x \sim \mathcal{D}^i$, g_ω is the task-specific skill encoder, skill $z \sim g_\omega(z|x)$. Then we have,

$$\frac{p(z|\mathcal{D}^i)}{p(z)} = \int \frac{p(x|\mathcal{D}^i)p(z|x,\mathcal{D}^i)}{p(z)}\mathrm{d}x = \int \frac{p(x)g_\omega(z|x)}{p(z)}\mathrm{d}x = \mathbb{E}_x\left[\frac{g_\omega(z|x)}{p(z)}\right]. \tag{17}$$

Theorem 3.1. Denote a set of N training tasks with their offline data $\mathcal{D}^{\mathcal{T}}$, $\mathcal{D}^i$ is the data of the sampled task. Let random variables x be some observations sampled from $\mathcal{D}^i$, skill $z \sim g_\omega(z|x)$, $h(x,z) = \frac{g_\omega(z|x)}{p(z)}$, $p(\mathcal{D}^i)$ is the prior distribution of current task i, then we have

$$-\mathbb{E}_{\mathcal{D}^i,z,x}\left[\log \frac{h(z,x)}{\sum_{\mathcal{D}^j \in \mathcal{D}^{\mathcal{T}}} h(z,x^j)}\right] \geq -\mathbb{E}_{x\sim\mathcal{D}^i}\left[D_{\mathrm{KL}}(g(\cdot|x)\|p(\mathcal{D}^i)\right] \tag{9}$$

where x^j are observations sampled from task $\mathcal{D}^j$ and $i \in \{1,2,\cdots,N\}$.

Proof. We introduce the mutual information $\mathcal{I}(z,\mathcal{D}^i)$ between the learned skill z and the current task's data distribution $\mathcal{D}^i$, and rewrite the inequality as below,

$$-\mathbb{E}_{\mathcal{D}^i,z,x}\left[\log \frac{h(z,x)}{\sum_{\mathcal{D}^* \in \mathcal{D}^{\mathcal{T}}} h(z,x^*)}\right] \geq \mathcal{I}(z;\mathcal{D}^i) \geq -\mathbb{E}_{x\sim\mathcal{D}^i}\left[D_{\mathrm{KL}}(g(\cdot|x)\|p(\mathcal{D}^i)\right] \tag{18}$$

For the left side:

$$\begin{aligned}
\mathcal{I}(z;\mathcal{D}^i) &= \mathbb{E}_{\mathcal{D}^i,z}\left[\log \frac{p(z|\mathcal{D}^i)}{p(z)}\right] \\
&\overset{(a)}{=} \mathbb{E}_{\mathcal{D}^i,z}\left[\log \mathbb{E}_x\left[\frac{g_\omega(z|x)}{p(z)}\right]\right] \\
&\geq \mathbb{E}_{\mathcal{D}^i,z,x}\left[\log \frac{g_\omega(z|x)}{p(z)}\right] \\
&= -\mathbb{E}_{\mathcal{D}^i,z,x}\left[\log\left(\frac{p(z)}{g_\omega(z|x)}N\right)\right] + \log N \\
&\geq -\mathbb{E}_{\mathcal{D}^i,z,x}\left[\log\left(1 + \frac{p(z)}{g_\omega(z|x)}(N-1)\right)\right] + \log N \\
&\geq -\mathbb{E}_{\mathcal{D}^i,z,x}\left[\log\left(1 + \frac{p(z)}{g_\omega(z|x)}(N-1)\mathop{\mathbb{E}}_{\mathcal{D}^* \in \mathcal{D}^{\mathcal{T}}\setminus i}\frac{g_\omega(z|x^*)}{p(z)}\right)\right] + \log N \\
&= -\mathbb{E}_{\mathcal{D}^i,z,x}\left[\log\left(1 + \frac{p(z)}{g_\omega(z|x)}\sum_{\mathcal{D}^* \in \mathcal{D}^{\mathcal{T}}\setminus i}\frac{g_\omega(z|x^*)}{p(z)}\right)\right] + \log N \\
&= \mathbb{E}_{\mathcal{D}^i,z,x}\left[\log\left(\frac{\frac{g_\omega(z|x)}{p(z)}}{\frac{g_\omega(z|x)}{p(x)} + \sum_{\mathcal{D}^* \in \mathcal{D}^{\mathcal{T}}\setminus i}\frac{g_\omega(z|x^*)}{p(z)}}\right)\right] + \log N \\
&\geq \mathbb{E}_{\mathcal{D}^i,z,x}\left[\log\left(\frac{\frac{g_\omega(z|x)}{p(z)}}{\sum_{\mathcal{D}^* \in \mathcal{D}^{\mathcal{T}}}\frac{g_\omega(z|x^*)}{p(z)}}\right)\right] \\
&= \mathbb{E}_{\mathcal{D}^i,z,x}\left[\log \frac{h(z,x)}{\sum_{\mathcal{D}^* \in \mathcal{D}^{\mathcal{T}}} h(z,x^*)}\right],
\end{aligned} \tag{19}$$

where N is the number of the task set $\mathcal{D}^{\mathcal{T}}$. Here, (a) is derived from Lemma A.1.

For the right side:

$$
\begin{aligned}
\mathcal{I}(z;\mathcal{D}^i) &= \int\int g_\omega(z,x)\log\frac{g_\omega(z|x)}{g_\omega(z)}\mathrm{d}z\mathrm{d}x \\
&= \int\int g_\omega(z,x)\log g_\omega(z|x)\mathrm{d}z\mathrm{d}x - \int g_\omega(z)\log g_\omega(z)\mathrm{d}z \\
&\overset{(b)}{\leq} \int\int g_\omega(z,x)\log g_\omega(z|a)\mathrm{d}z\mathrm{d}x - \int g_\omega(z)\log p(\mathcal{D}^i)\mathrm{d}z \\
&= \int\int g_\omega(x)g_\omega(z|x)\log\frac{g_\omega(z|x)}{p(\mathcal{D}^i)}\mathrm{d}z\mathrm{d}x \\
&= \mathbb{E}_{x\sim\mathcal{D}^i}\left[\int g_\omega(z|x)\log\frac{g_\omega(z|x)}{p(\mathcal{D}^i)}\mathrm{d}z\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}^i}\left[D_{\mathrm{KL}}(g_\omega(\cdot|x)\|p(\mathcal{D}^i))\right]
\end{aligned}
\tag{20}
$$

The inequality at (b) is derived from $D_{\mathrm{KL}}(g_\omega(\cdot)\|p(\mathcal{D}^i)\geq 0$. □

B PSEUDOCODE FOR HISSD

Algorithm 1 HiSSD for Offline Multi-Task MARL

```
Inputs:
High-Level Planner π_{θ_h}, Low-Level Controller π_{θ_l}, Value Net V_ξ^tot and V̄_ξ̄^tot, Forward Predictor f_φ, Task-Specifc Skill Encoder g_ω, Training Steps T, Task Numbers N, Offline Multi-Task Dataset D^𝒯, Current Training Task i and its data D^i ~ D^𝒯, Agent Numbers {K^i}_{i=1}^𝒯, Batch Size B, learning rate δ, target update rate τ.
Training:
for each timestep n in 1..N do
    D^i ~ D^𝒯        # Sample one task data from the offline dataset.
    {s_t, r_t, {o_t, a_t, o_{t+1}}_{k=1}^K}_{j=1}^B ~ D^i
    c_t^{1:k} ← π_{θ_h}(o_t^{1:k})        # Infer common skills.
    Compute V_ξ^tot(o_t^{1:k}), V̄_ξ̄^tot(o_{t+1}^{1:k}), and V̄_ξ̄^tot(f_φ(c_t^{1:k})) with s_t
    s'_{t+1} ← f_φ(c_t^{1:k})
    z_t^{1:k} ← g_ω(o_t^{1:k})        # Infer task-specific skills.
    a'_t^{1:k} ← π_{θ_l}(z_t^{1:k}, o_t^{1:k}, c_t^{1:k})
    Optimizing Eq. 5 with V_ξ^tot(o_t^{1:k}), V̄_ξ̄^tot(o_{t+1}^{1:k}), and r_t
    Optimizing Eq. 7 with V_ξ^tot(o_t^{1:k}), V̄_ξ̄^tot(f_φ(c_t^{1:k})), r_t, and s'_{t+1}
    Optimizing Eq. 11 with c_t^{1:k}, z_t^{1:k}, a'_t^{1:k}, and samples from other tasks
end for
Execution:
for each timestep t in current environment do
    c_t^{1:k} ← π_{θ_h}(o_t^{1:k})        # Infer common skills.
    z_t ← g_ω(o_t^{1:k})        # Infer task-specific skills.
    a_t^{1:k} ← π_{θ_l}(z_t^{1:k}, o_t^{1:k}, c_t^{1:k})
end for
```

C BENCHMARKS AND DATASETS

C.1 SMAC

The StarCraft Multi-Agent Challenge (SMAC) (Samvelyan et al., 2019) is a widely used cooperative multi-agent testbed that contains diverse StarCraft micromanagement scenarios. In this paper, we utilize three distinct SMAC task sets defined by Zhang et al. (2022): *Marine-Hard* and *Marine-Easy* to evaluate the capacity of transferring policy to unseen tasks. The Marine-Hard and Marine-Easy task sets include various marine battle scenarios, and the trained multi-agent policy needs to control groups of allied marines to confront equivalent or superior numbers of built-in-AI enemy marines.

It aims to test the capacity of policy transfer in heterogeneous scenarios. Detailed attributes of these task sets are enumerated in Tables 4 and 5.

Table 4: Descriptions of tasks in the Marine-Easy task set.

Task type	Task	Ally units	Enemy units	Properties
Source tasks	3m	3 Marines	3 Marines	homogeneous & symmetric
	5m	5 Marines	5 Marines	homogeneous & symmetric
	10m	10 Marines	10 Marines	homogeneous & symmetric
Unseen tasks	4m	4 Marines	4 Marines	homogeneous & symmetric
	6m	6 Marines	6 Marines	homogeneous & symmetric
	7m	7 Marines	7 Marines	homogeneous & symmetric
	8m	8 Marines	8 Marines	homogeneous & symmetric
	9m	9 Marines	9 Marines	homogeneous & symmetric
	11m	11 Marines	11 Marines	homogeneous & symmetric
	12m	12 Marines	12 Marines	homogeneous & symmetric

Table 5: Descriptions of tasks in the Marine-Hard task set.

Task type	Task	Ally units	Enemy units	Properties
Source tasks	3m	3 Marines	3 Marines	homogeneous & symmetric
	5m_vs_6m	5 Marines	6 Marines	homogeneous & asymmetric
	9m_vs_10m	9 Marines	10 Marines	homogeneous & asymmetric
Unseen tasks	4m	4 Marines	4 Marines	homogeneous & symmetric
	5m	5 Marines	5 Marines	homogeneous & symmetric
	10m	10 Marines	10 Marines	homogeneous & symmetric
	12m	12 Marines	12 Marines	homogeneous & symmetric
	7m_vs_8m	7 Marines	8 Marines	homogeneous & asymmetric
	8m_vs_9m	8 Marines	9 Marines	homogeneous & asymmetric
	10m_vs_11m	10 Marines	11 Marines	homogeneous & asymmetric
	10m_vs_12m	10 Marines	12 Marines	homogeneous & asymmetric
	13m_vs_15m	13 Marines	15 Marines	homogeneous & asymmetric

As stated in the experiments section, we utilize the same offline multi-task dataset as Zhang et al. (2022) to maintain a fair comparison. Definitions of these four qualities are listed below:

- The **expert** dataset contains trajectory data collected by a QMIX policy trained with $2,000,000$ steps of environment interactions. The test win rate of the trained QMIX policy (as the expert policy) is recorded for constructing medium datasets.
- The **medium** dataset contains trajectory data collected by a QMIX policy (as the medium policy) whose test win rate is half of the expert QMIX policy.
- The **medium-expert** dataset mixes data from the expert dataset and the medium dataset to acquire a more diverse dataset.
- The **medium-replay** dataset is the replay buffer of the medium policy, containing trajectory data with lower qualities.

The Properties of offline datasets with different qualities are detailed in Table 6.

C.2 MAMuJoCo

Multi-Agent MuJoCo (Peng et al., 2021) is a benchmark developed for assessing and comparing the effectiveness of algorithms in continuous multi-agent robotic control. In MAMuJoCo, a robotic

Table 6: Properties of offline datasets in SMAC with different qualities.

Tasks	Quality	Trajectories	Average Return	Average Win Rate
3m	expert	2000	19.8929	0.9910
	medium	2000	13.9869	0.5402
	medium-expert	4000	16.9399	0.7656
	medium-replay	3603	N/A	N/A
5m	expert	2000	19.9380	0.9937
	medium	2000	17.3288	0.7411
	medium-expert	4000	18.6334	0.8674
	medium-replay	711	N/A	N/A
10m	expert	2000	19.9438	0.9922
	medium	2000	16.6297	0.5413
	medium-expert	4000	18.2595	0.7626
	medium-replay	571	N/A	N/A
5m_vs_6m	expert	2000	17.3424	0.7185
	medium	2000	12.6408	0.2751
	medium-expert	4000	14.9916	0.4968
	medium-replay	32607	N/A	N/A
9m_vs_10m	expert	2000	19.6140	0.9431
	medium	2000	15.5049	0.4146
	medium-expert	4000	17.5594	0.6789
	medium-replay	13731	N/A	N/A

system is partitioned into independent agents, each tasked with controlling a specific set of joints to accomplish shared objectives. To conduct offline multi-task learning, we choose *HalfCheetah*-v2 as our base scenarios and partition the robotic system into six agents. Besides the complete HalfCheetah robot, we design six new tasks by disabling each agent. Therefore, our MAMuJoCo multi-task data comprises seven tasks. Each task's name corresponds to the joint controlled by the disabled agent. We follow Wang et al. (2023) and generate offline data using the policy trained by HAPPO (Kuba et al., 2022a). The hyperparameter `env_args.agent_obsk` (determines up to which connection distance agents will be able to form observations) is set to 1. We list the average return of our datasets in Table 7.

Table 7: Properties of offline datasets on HalfCheetah-v2 in MAMuJoCo.

Tasks	Trajectories	Average Return
complete	100	2881.63
back thigh	100	2764.65
back foot	100	2880.78
front thigh	100	2011.00
front shin	100	3048.01

D Implementation Details

D.1 Details of HiSSD in SMAC

We follow prior works and decompose the observation to process the varying observation sizes among tasks. The local observation o_i is decomposed into several portions, the one including agent i's own information o_i^{own}, and the other contains other entities' information $\{o_{i,j}^{\text{entity}}\}$. To embed these portions, we utilize the transformer model which can parallel process multiple tensors. Each portion is embedded by a separate fully connected layer with an output dimension of 64, and the

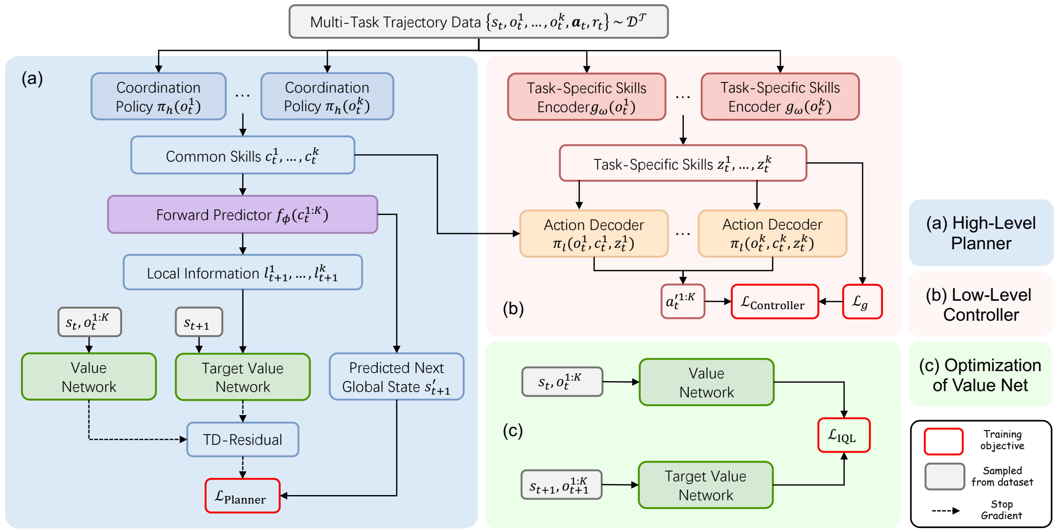

Figure 4: Overall framework of Hi-SSD in SMAC. HiSSD utilizes a hierarchical framework that jointly learns common and task-specific skills from offline multi-task data to improve multi-agent policy transfer. (a) The high-level planner with common skills. HiSSD integrates cooperative temporal knowledge into common skills and enables an offline exploration. (b) The low-level controller with task-specific skills. The task-specific knowledge can guide the action execution adaptively among tasks. (c) HiSSD uses an implicit Q-learning objective to train the value network.

transformer processes these embedding according to the attention mechanism,

$$
\begin{gathered}
Q = W^Q([o_i^{\text{own}}, o_{i,1}^{\text{entity}}, \ldots]), K = W^K([o_i^{\text{own}}, o_{i,1}^{\text{entity}}, \ldots]), V = W^V([o_i^{\text{own}}, o_{i,1}^{\text{entity}}, \ldots]), \\
[e_i^{\text{own}}, e_{i,1}^{\text{entity}}, \ldots] = \text{softmax}\left(\frac{QK^{\mathrm{T}}}{\sqrt{d_K}}\right) V, \\
\text{where } [o_i^{\text{own}}, o_{i,1}^{\text{entity}}, \ldots] = \text{Decompose}(o_i), \quad d_K = \dim(K).
\end{gathered} \tag{21}
$$

The common skill encoder, the task-specific skill encoder, the individual value network, and the action decoder are represented by a 1-layer transformer with 64 units' hidden layer to process the decomposed observation. The mixing value network is represented by the attention block following Zhang et al. (2022). The forward predictor is represented by two 1-layer transformers. To tackle the partial observability, we append the history hidden state h_{t-1}^i in each agent's common skill encoder, value net, and action decoder when applying self-attention and thus get the output of h_t^i.

The common skills are inferred by the common skill encoder and then fed into the forward predictor and action decoder for predicting the next global state and executing real action. We use a transformer to merge other enemies' information and then utilize another transformer to process all allies' and enemies' information to predict the next global state. We concatenate the decomposed information outputted by the action decoder with the task-specific skills and feed them into an MLP to get the real actions. Figure 4 illustrates the overall framework of HiSSD in SMAC. The hyperparameters used in SMAC are listed in Table 8.

Table 8: Hyperparameters of HiSSD for offline multi-task SMAC.

Hyperparameter	**Setting**
Hidden layer dimension	64
Hidden units in MLP	128
Attention dimension	64
Skill dimension per token	64
Discount factor γ	0.99
α	10
β	0.05
ϵ	0.9
Trajectories per batch	32
Training steps	30000
Optimizer	Adam
Learning rate	0.0001
Weight decay	0.0001

D.2 Details of HiSSD in MAMuJoCo

The observation shapes are consistent among tasks and we do not apply the observation decomposition deployed in SMAC. The common skill encoder, the task-specific skill encoder, the value network, the forward predictor, and the action decoder are represented by a 1-layer transformer with 64 units' hidden layer to process the decomposed observation. The common skills are inferred by the common skill encoder and then fed into the forward predictor and action decoder for predicting the next global state and executing real action. We concatenate the decomposed information outputted by the action decoder with the task-specific skills and feed them into an MLP to get the real actions. The hyperparameters used in MAMuJoCo are listed in Table 9.

Table 9: Hyperparameters of HiSSD for offline multi-task MAMuJoCo.

Hyperparameter	**Setting**
Hidden layer dimension	256
Hidden units in MLP	256
Attention dimension	256
Skill dimension per token	256
Discount factor γ	0.99
Target update rate	0.005
α	10.0
β	2.0
ϵ	0.9
Batch size	128
Training steps	1000000
Optimizer	Adam
Learning rate	0.0005

D.3 Training Costs

The training process of HiSSD with an NVIDIA GeForce RTX 3090 GPU and a 32-core CPU costs 12-14 hours typically. Our released implementation of HiSSD follows Apache License 2.0, the same as the PyMARL framework.

E Additional Results

E.1 Results on Other Task Set in SMAC.

We conduct experiments in two offline multi-task task sets: Marine-Easy and Marine-Hard. In addition to the results in the Marine-Hard task set presented in Table 1, we report the results in the Marine-Easy in Table 10. HiSSD gains convincing performance in most source and unseen tasks compared to other baselines. Despite all that, HiSSD still obtains a similar performance compared to other baselines, indicating the validity of the Hi-SSD algorithm.

Table 10: Average test win rates in the *Marine-Easy* task set with different data qualities. Results of BC-best stands for the best test win rates between BC-t and BC-r.

Tasks	Expert				Medium			
	BC-best	UPDeT-m	ODIS	Hi-SSD (Ours)	BC-best	UPDeT-m	ODIS	Hi-SSD (Ours)
Source Tasks								
3m	94.5± 4.6	83.6±12.6	97.7± 2.6	**99.5± 8.1**	67.2± 4.7	60.2±29.9	57.8± 9.2	**74.7±14.6**
5m	94.4± 7.6	74.8±22.9	95.3± 5.2	**99.9± 0.0**	79.2± 5.9	67.8± 5.9	**82.8± 5.2**	81.6±10.8
10m	86.1±22.7	83.6±19.2	88.3±20.3	**95.2± 8.4**	63.1± 7.2	48.8± 7.9	71.9± 6.6	**84.8± 8.6**
Unseen Tasks								
4m	91.2± 1.6	53.0±32.3	90.6± 7.0	**94.4± 2.9**	62.5±11.6	41.7±17.4	63.3±16.1	**74.5±15.5**
6m	75.3±22.6	37.9± 8.6	79.7±17.5	**99.7± 0.3**	86.0± 4.7	75.8±22.7	**89.8±17.6**	88.0±10.0
7m	70.3±11.0	44.2±13.2	72.7±16.9	**99.1± 0.7**	**99.9± 0.0**	65.2±25.2	96.1± 1.4	97.3± 2.3
8m	74.7±16.5	51.7±26.2	80.9±14.4	**99.8± 0.3**	96.9± 2.2	88.4±13.7	**97.7± 2.6**	93.8± 5.2
9m	97.7± 2.6	76.3±13.4	99.2± 1.4	**99.9± 0.0**	78.9±11.8	64.8±35.6	**87.5± 2.2**	75.2±15.5
11m	83.3±11.8	53.6±22.4	83.6±12.4	**99.2± 0.8**	42.2± 4.7	23.4±11.8	**64.7± 3.1**	62.0±21.8
12m	56.7±30.0	44.3±22.8	70.3±30.2	**99.7± 1.1**	29.7±23.4	13.5±11.7	41.4± 6.0	**55.5±25.7**
	Medium-Expert				Medium-Replay			
Source Tasks								
3m	81.3±18.8	48.4±36.8	89.8± 9.7	**90.9± 5.9**	77.8± 3.2	29.7±10.0	79.7± 4.7	**87.7± 2.9**
5m	74.0± 2.9	64.1±17.9	**83.7±16.0**	79.4± 6.9	5.5± 5.6	6.2±10.8	3.1± 5.4	**87.5± 2.9**
10m	90.6± 3.1	68.8±23.8	**93.8± 4.4**	60.2±21.1	0.0± 0.0	0.0± 0.0	0.0± 0.0	**84.2±4.9**
Unseen Tasks								
4m	35.2±38.0	43.7±25.0	57.8±18.8	**70.9± 9.1**	67.2± 4.7	25.0±22.6	25.0± 5.4	**71.6± 4.1**
6m	42.2± 1.6	47.7±30.0	**76.0± 6.0**	70.6± 6.1	7.8±10.2	0.0± 0.0	3.1± 5.4	**99.8± 0.3**
7m	65.6±16.4	57.8±32.9	66.4±14.6	**85.0±11.7**	0.8± 1.4	0.0± 0.0	0.0± 0.0	**99.8± 0.3**
8m	40.3±42.6	40.6±19.3	43.8±11.5	**72.8± 9.5**	0.8± 1.4	0.0± 0.0	1.6± 1.6	**96.7± 0.3**
9m	70.8±16.6	47.7±24.8	73.4±16.2	**80.0±14.6**	0.8± 1.4	0.0± 0.0	0.0± 0.0	**88.8± 1.3**
11m	55.5±12.4	**85.9±14.2**	68.8±20.3	70.9± 5.9	0.0± 0.0	0.0± 0.0	0.0± 0.0	**45.6± 4.5**
12m	29.7±29.8	46.1±15.5	62.5± 8.0	**62.7± 7.8**	0.0± 0.0	0.0± 0.0	0.0± 0.0	**38.0± 3.7**

E.2 Sensitivity Analysis of Hyper-Parameters.

To investigate the model's sensitivity analysis to β in Eq. 11, we conduct experiments on Marine-Hard task set in SMAC with expert data qualities. We list the results in Table 11. An appropriate β can ensure that the task-specific skills achieve a fair trade-off between distinguishing different tasks and extracting useful information for action execution. Empirical results indicate that HiSSD is quite robust to different value of β. When β is very small, the performance of the HiSSD becomes poor due to the lack of task-informed regularization.

F Additional Results for Rebuttal

F.1 Compared to HyGen

HyGen (Zhang et al., 2024) is a recent work that focuses on integrating offline pertaining and online exploration to speed up multi-task MARL for policy transfer. HyGen first pre-trains the skill space with an action decoder using the global information, then implements online exploration to maintain a hybrid replay buffer using offline and online data, improving the high-level policy and refining the action decoder simultaneously. Compared to HyGen, our method requires no online exploration

Table 11: Average test win rates on *Marine-Hard* task set in SMAC with expert data qualities for the model's sensitivity analysis to β. β controls the margin of regularization in Eq. (14).

Tasks	Expert				
	$\beta = 0.001$	$\beta = 0.01$	$\beta = 0.1$	$\beta = 1$	$\beta = 0.05$ (**Ours**)
Source Tasks					
3m	99.8± 0.3	**99.9± 0.0**	96.8± 4.5	99.5± 0.2	99.5± 0.3
5m6m	65.3± 1.1	64.7± 2.4	64.5± 4.9	65.9± 2.7	**66.1± 7.0**
9m10m	95.4± 2.6	95.3± 0.3	94.3± 0.6	92.0± 2.4	**95.5± 2.7**
Unseen Tasks					
4m	98.3± 1.9	97.7± 0.6	97.5± 2.2	98.8± 0.3	**99.2± 1.2**
5m	99.0± 0.8	98.8± 0.3	99.0± 0.9	99.1± 0.8	**99.2± 1.2**
10m	97.5± 1.4	98.6± 0.3	96.7± 3.4	97.3± 1.3	**98.4± 0.8**
12m	64.8±14.9	66.9±22.0	68.1±30.3	75.2±20.3	**75.5±17.9**
7m8m	34.0± 5.5	35.4±12.0	36.5±13.0	**40.6± 5.3**	35.3± 9.8
8m9m	50.8±13.3	40.2± 2.8	**58.1±13.0**	41.3± 7.2	47.0± 6.2
10m11m	79.4± 2.3	79.6± 7.4	80.6±17.4	76.7±15.8	**86.3±14.6**
10m12m	7.7± 1.6	11.2±10.6	7.9± 3.1	14.4± 8.5	**14.5± 9.1**
13m15m	1.0± 0.8	1.7± 1.6	1.0± 1.1	**2.1± 1.8**	1.3± 2.5

and pretraining steps. HyGen only conducts common skills learning while our method leverages task-specific skills to complement the common skills discovery. We compare our method to HyGen in the SMAC's Marine hard task set and propose the empirical results in Table 12. When the data quality is near-optimal (i.e., *Expert*), HiSSD outperforms HyGen on most tasks, and when the data quality is medium, HyGen benefits from online exploration and further improves policy.

Table 12: Average test win rates of the best policies over five random seeds in the task set *Marine-Hard* with *Expert* and *Medium* data qualities. Results of BC-best stands for the best test win rates between BC-t and BC-r.

Tasks	Expert			Medium		
	BC-best	HyGen	HiSSD (Ours)	BC-best	HyGen	HiSSD (Ours)
Source Tasks						
3m	97.7± 2.6	99.1± 1.0	**99.5± 0.3**	65.4±14.7	**91.5±11.0**	62.7± 5.7
5m6m	50.4± 2.3	61.2± 8.0	**66.1± 7.0**	21.9± 3.4	**31.6± 7.0**	26.4± 3.8
9m10m	95.3± 1.6	**96.4± 3.0**	95.5± 2.7	63.8±10.9	**79.2± 4.0**	73.9± 2.3
Unseen Tasks						
4m	92.1± 3.5	95.8± 4.0	**99.2± 1.2**	48.8±21.1	**91.4± 8.0**	77.3±10.2
5m	87.1±10.5	99.5± 1.0	**99.2± 1.2**	76.6±14.1	**96.5± 6.0**	88.4± 8.4
10m	90.5± 3.8	93.5± 5.0	**98.4± 0.8**	56.2±20.6	96.4± 3.0	**98.0± 0.3**
12m	70.8±15.2	**85.2± 6.0**	75.5±19.7	24.0±10.5	81.5±14.0	**86.4± 6.0**
7m8m	18.8± 3.1	28.9±12.0	**35.3± 9.8**	1.6± 1.6	**24.5± 9.0**	14.2±10.1
8m9m	15.8± 3.3	25.7± 9.0	**47.0± 6.2**	3.1± 3.8	**24.5± 9.0**	15.3± 2.8
10m11m	45.3±11.1	57.2±13.0	**86.3±14.6**	19.7± 8.9	47.2±13.0	43.6± 4.6
10m12m	1.0± 1.5	13.8± 4.0	**14.5± 9.1**	0.0± 0.0	**5.2± 2.0**	0.6± 0.5
13m15m	0.0± 0.0	**9.5± 5.0**	1.3± 2.5	0.6± 1.3	**9.3± 6.0**	1.4± 2.4

F.2 Extended Results in SMAC

We follow the multi-task learning settings in Zhang et al. (2022) and conduct experiments on *Stalker-Zealot* task set. The results are presented in Tables 13 and 14. Compared to ODIS, we find that HiSSD obtains competitive performance and surpasses ODIS when the dataset is generated by near-optimal policy (i.e., *Expert* and *Medium-Expert*). Moreover, both of the skill-learning-based methods fail to outperform BC-based methods in some tasks. We suspect this is due to the different task properties between *Marine* and *Stalker-Zealot* task set. The controlled items in *Marine* task set are homogeneous while in *Stalker-Zealot* are heterogeneous. Therefore, it is more difficult for the policy to learn cooperative patterns from the limited dataset in the Stalker-Zealot task set and obtain convincing performance in unseen tasks.

Table 13: Average test win rates in the *Stalker-Zealot* task set with different data qualities.

Data Qualities	BC-best	UPDeT-m	ODIS	HiSSD
Source Tasks				
Expert	87.9± 5.6	30.2±26.6	82.0± 6.3	**89.3± 4.0**
Medium	28.6±10.1	26.5±19.2	**37.0± 9.1**	24.6± 7.3
Medium-Expert	52.9±17.9	48.5±25.1	47.1±13.9	**55.9± 9.7**
Medium-Replay	13.2± 5.4	**15.6±13.2**	14.1± 8.8	10.8± 1.6
Target Tasks				
Expert	60.9±19.0	13.8±13.2	62.8± 7.5	**69.2± 8.7**
Medium	19.8±14.1	10.1±11.8	**24.7± 8.4**	16.6± 7.4
Medium-Expert	37.5±17.4	18.6±14.1	38.2±11.9	**40.3±12.8**
Medium-Replay	13.6±7.6	10.7±11.8	11.0±10.8	**13.6± 4.8**

Table 14: Average test win rates in the *Stalker-Zealot* task set with different data qualities. **Bold** represents the best score in each task and red indicates HiSSD performs better than ODIS.

Tasks	Expert				Medium			
	BC-best	UPDeT-m	ODIS	Hi-SSD (Ours)	BC-best	UPDeT-m	ODIS	Hi-SSD (Ours)
Source Tasks								
2s3z	93.1± 4.6	50.0±33.4	**97.7± 2.6**	95.2± 1.0	48.8± 9.8	35.0±23.0	**49.2± 8.4**	32.3±11.7
2s4z	78.1± 8.1	23.4±26.6	60.9± 6.8	**79.8± 6.0**	12.5± 8.1	18.8±10.3	**32.8±12.2**	17.0± 2.2
3s5z	92.5±4.2	17.2±19.8	87.5± 9.6	**92.8± 5.0**	24.4±12.4	25.6±24.2	**28.9± 6.8**	24.4± 7.9
Unseen Tasks								
1s3z	45.6±23.8	1.6± 1.6	76.6± 3.5	**81.6±15.2**	21.9±37.6	3.8± 5.0	41.4±18.8	**44.2± 9.9**
1s4z	**60.0±32.3**	26.6±19.3	17.2±10.5	42.0±26.1	6.2± 7.7	2.5± 3.6	**50.7± 7.5**	18.1±11.0
1s5z	**45.6±26.9**	29.7±26.4	2.5 ± 2.3	16.7±12.3	3.1± 2.6	5.0± 4.2	**14.1± 8.4**	2.5± 2.2
2s5z	75.6±11.9	23.4±22.2	27.3± 6.0	**79.7± 2.2**	14.4± 9.0	16.9±14.1	**32.0± 4.6**	11.3± 3.7
3s3z	80.6± 9.1	20.3±10.9	**89.1± 5.2**	88.0± 4.5	**45.6±14.6**	24.4±28.6	23.4± 9.2	21.9±10.7
3s4z	92.5± 5.1	12.5±19.9	**96.9± 2.2**	88.1± 9.0	40.0±19.0	28.8±31.6	**50.8±15.5**	17.2± 4.5
4s3z	67.5±19.8	6.2± 4.4	64.1±13.0	**88.6± 4.1**	28.8±26.4	11.2±18.0	13.3± 7.5	**31.9±23.2**
4s4z	53.1±18.4	7.8±13.5	**79.7±10.9**	73.4± 5.2	**20.0±12.0**	1.2± 1.5	12.5± 7.0	13.2± 6.5
4s5z	40.6±19.1	5.5± 7.8	**86.7±12.6**	65.6± 3.7	**14.4± 8.5**	5.6± 8.5	7.0± 4.1	4.5± 1.3
4s6z	48.1±23.8	4.7± 6.4	**88.3± 8.4**	68.4± 4.9	**3.8± 3.6**	1.9± 2.5	1.6± 1.6	0.9± 0.9
	Medium-Expert				Medium-Replay			
Source Tasks								
2s3z	57.5±25.1	57.5±27.1	58.6±15.5	**68.1± 8.1**	3.1± 2.6	14.4±13.2	**15.6±18.2**	9.0± 1.5
2s4z	37.5±15.3	**53.1±24.6**	41.4± 7.8	41.9±10.2	5.2± 7.4	**12.5± 9.7**	7.8± 5.2	6.0± 1.2
3s5z	**63.1±13.3**	35.0±23.5	41.4±18.5	57.8±10.7	31.3± 6.3	**20.0±16.6**	18.8± 3.1	17.5± 2.0
Unseen Tasks								
1s3z	55.6±37.7	4.4± 8.8	72.7±12.2	**73.0±10.2**	24.0±15.4	0.0± 0.0	21.1±20.4	**36.3± 7.1**
1s4z	25.0±30.7	11.9± 9.8	**44.5±20.3**	32.3±30.5	2.1± 2.9	7.5±10.0	6.2± 7.7	**24.8± 9.1**
1s5z	14.4±19.4	3.8± 4.6	**42.2±31.4**	9.4± 9.5	7.3± 6.4	**11.9± 9.6**	7.8± 6.4	4.4± 2.2
2s5z	26.9±20.2	37.5±22.5	**43.0±10.7**	25.6± 7.8	12.5±15.5	**20.0±16.8**	14.1± 8.1	16.5± 2.8
3s3z	35.6±18.0	33.8±15.0	50.0±13.3	**56.6±25.6**	**35.4±12.1**	17.5±12.3	25.0±20.1	9.6± 3.3
3s4z	**74.4±16.3**	43.1±20.7	52.3± 9.5	71.7± 9.7	20.8± 9.0	15.6±11.2	19.5±16.6	**22.5±10.6**
4s3z	**69.8± 7.8**	23.8±21.0	17.2± 7.2	60.5±15.1	**17.7± 5.3**	11.2±15.0	8.6±14.9	11.0±10.4
4s4z	**41.9±14.9**	10.6±13.8	20.3± 6.8	37.3± 9.4	**15.6± 6.8**	5.6± 9.8	4.7± 8.1	9.4± 1.8
4s5z	17.3± 5.3	11.9±16.1	**21.9± 2.2**	17.0± 4.1	1.0± 1.5	**10.6±19.7**	0.8± 1.4	0.8± 0.8
4s6z	13.8± 3.2	5.0± 8.5	18.0± 5.1	**19.7± 5.9**	0.0± 0.0	**6.9±13.8**	2.3± 4.1	0.4± 0.3

F.3 Extended Results in MAMuJoCo

It's worth noting that the official code released by ODIS didn't conduct experiments on MAMuJoCo and we didn't use ODIS as a baseline. Here we reproduce ODIS on MAMuJoCo environment and report the results in Table 17. We use the same architecture and hyperparameters as HiSSD for a fair comparison. The results demonstrate that our method outperforms ODIS in discrete and continuous control tasks when generalized to unseen tasks.

Table 15: Average test win rates on *Marine-Hard* task set in SMAC with expert data qualities for the model's sensitivity analysis to ϵ. ϵ balances the imitation and exploration in offline value estimation.

Tasks	Expert				
	$\epsilon = 0.1$	$\epsilon = 0.3$	$\epsilon = 0.5$	$\epsilon = 0.7$	$\epsilon = 0.9$ (**Ours**)
Source Tasks					
3m	99.6± 0.6	**99.9± 0.0**	99.4± 0.5	99.4± 0.5	99.5± 0.3
5m6m	73.5± 5.1	71.3± 3.3	70.8± 2.4	**74.4± 1.5**	66.1± 7.0
9m10m	98.3± 0.3	97.1± 0.3	**99.0± 0.3**	96.7± 0.6	95.5± 2.7
Unseen Tasks					
4m	97.7± 1.6	99.0± 0.3	99.0± 0.8	98.4± 0.5	**99.2± 1.2**
5m	**99.8± 0.3**	**99.8± 0.3**	99.2± 0.3	99.1± 1.2	99.2± 1.2
10m	99.2± 0.8	98.5± 1.6	99.2± 0.3	**99.2± 0.3**	98.4± 0.8
12m	69.0±33.7	67.9±17.0	71.0±26.0	**87.5± 7.6**	75.5±17.9
7m8m	29.2± 2.8	23.1± 4.9	32.7± 3.8	25.4± 2.9	**35.3± 9.8**
8m9m	46.0±16.6	47.5± 9.2	45.4±14.8	44.6±11.4	**47.0± 6.2**
10m11m	82.1± 4.3	81.4± 9.7	80.4± 3.1	72.3±13.8	**86.3±14.6**
10m12m	11.5± 6.9	9.4± 5.9	8.1± 2.2	7.3± 5.5	**14.5± 9.1**
13m15m	0.2± 0.3	0.4± 0.6	0.6± 0.5	0.4± 0.6	**1.3± 2.5**

Table 16: Average test win rates on *Marine-Hard* task set in SMAC with expert data qualities for the model's sensitivity analysis to α. α weights the TD-residual Eq. 7.

Tasks	Expert			
	$\alpha = 1$	$\alpha = 5$	$\alpha = 20$	$\alpha = 10$ (**Ours**)
Source Tasks				
3m	99.2± 0.6	**99.9± 0.0**	99.8± 0.3	99.5± 0.3
5m6m	70.8± 4.7	**71.0± 2.9**	66.5± 1.1	66.1± 7.0
9m10m	97.1± 1.8	**97.9± 2.1**	93.4± 1.6	95.5± 2.7
Unseen Tasks				
4m	98.4± 0.0	98.1± 0.5	99.0± 0.6	**99.2± 1.2**
5m	**99.8± 0.3**	**99.8± 0.3**	98.8± 1.8	99.2± 1.2
10m	99.4± 0.5	99.4± 0.9	99.0± 0.3	**98.4± 0.8**
12m	59.2±36.2	63.8±33.6	66.5±26.5	**75.5±17.9**
7m8m	34.5± 5.0	34.8±10.3	34.6± 2.8	**35.3± 9.8**
8m9m	46.3± 2.5	**59.2±13.3**	53.3± 3.6	47.0± 6.2
10m11m	83.3± 8.2	77.9±13.7	77.3± 6.8	**86.3±14.6**
10m12m	10.2± 4.1	8.3± 8.3	11.9± 6.7	**14.5± 9.1**
13m15m	**2.5± 3.5**	0.2± 0.3	1.3± 1.8	1.3± 2.5

Table 17: Average scores on HalfCheetah-v2 multi-task datasets in MAMuJoCo. The ODIS method is reproduced by ourselves.

Tasks	BC	ODIS	HiSSD(ours)
Source Tasks			
complete	3188.16±566.68	3677.66±174.82	**4450.57±126.36**
back thigh	3324.22± 58.49	2381.62±198.59	**3698.38± 13.98**
back foot	3079.01±355.66	2713.40±195.63	**3197.83± 6.99**
front thigh	1861.53±415.80	**2684.99±249.70**	1948.74± 81.24
front shin	1819.94±273.96	3944.78±219.72	**3468.32±290.72**
Unseen Tasks			
back shin	1964.55±268.24	3217.59±184.15	**3472.12± 91.95**
front foot	3468.40±369.40	3930.82±342.16	**4165.29±338.96**

F.4 Additional Ablation Study

We conduct experiments to analyze the model's sensitivity of ϵ in Eq. 5 and α in Eq. 7. The results are presented in Tables 15 and 16, respectively. Although our method leverages additional components and objectives compared to previous methods, the empirical results indicate that our method is quite robust under different hyperparameter settings.

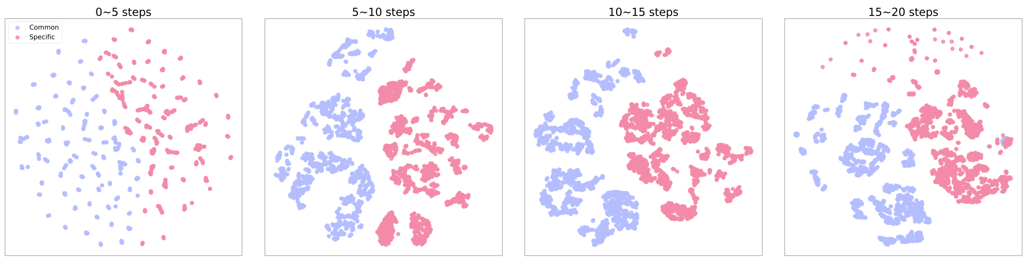

Figure 5: Visualization of learned common and task-specific skills. We use HiSSD to collect trajectories on five tasks in SMAC and partition these trajectories into four time windows. Plots in each figure represent the distribution of skills. We use multiple time windows to indicate the skill flow.

To demonstrate the effectiveness of learning task-specific skills, we conduct a variant of HiSSD named *w/o Specific* and present the empirical results in Table 18. We find that policy with only common skills significantly outperforms the BC-based method, learning both skills further improves the transfer capability of the policy.

Table 18: Additional ablation studies on HiSSD. We report average test win rates of the best policies over five random seeds in the task set *Marine-Hard* with different data qualities.

Data Qualities	w/o Specific	BC-best	HiSSD
Source Tasks			
Expert	**85.9±16.4**	81.1±21.8	84.5±17.1
Medium	53.1±20.0	50.3±20.1	**55.9±19.2**
Medium-Expert	63.5±26.1	41.7±18.5	**64.2±23.5**
Medium-Replay	47.7±25.1	46.5±26.7	**51.9±24.0**
Target Tasks			
Expert	55.0±37.4	46.8±36.8	**61.2±37.4**
Medium	37.6±34.2	25.6±26.8	**48.6±40.1**
Medium-Expert	51.2±34.3	38.8±33.0	**57.6±37.5**
Medium-Replay	42.8±36.2	41.2±33.7	**48.7±39.0**

To verify the distinction between common and task-specific skills, we visualize both skills and plot the results in Figure 5. The empirical results indicate that the learned policy can distinguish different kinds of skills at each stage of the episode, and no obvious overlap between these skills is observed.