Appendix: Cooperative Exploration for Multi-Agent Deep Reinforcement Learning

In this appendix we first provide the proofs for Claim 1 and Claim 2 in Sec. A and Sec. B. We then provide information regarding the MPE and SMAC environments (Sec. C, Sec. D), implementation details (Sec. E), and the absolute metric (Sec. F). Next, we provide additional results on MPE tasks (Sec. G), additional results of baselines (Sec. H) and training curves (Sec. I).

A. Proof of Claim 1

Claim 1. *Consider the 2-player l-action matrix game in Example 1. Let $m = l^2$ denote the total number of action configurations. Let T_m^{share} and $T_m^{non\text{-}share}$ denote the number of steps needed to see all m action configurations at least once for exploration with shared goal and for exploration without shared goal respectively. Then we have $\mathbb{E}[T_m^{share}] = m$ and $\mathbb{E}[T_m^{non\text{-}share}] = m\sum_{i=1}^m \frac{1}{i} = \Theta(m \ln m)$.*[2]

Proof. When exploring without shared goal, the agents don't coordinate their behavior. It is equivalent to uniformly picking one action configuration from the m configurations. We aim to show after $T_m^{\text{non-share}}$ time steps, the agents tried all m distinct action configurations. Let T_i be the number of steps to observe the i-th distinct action configuration after seeing $i-1$ distinct configurations. Then

$$\mathbb{E}[T_m^{\text{non-share}}] = \mathbb{E}[T_1] + \cdots + \mathbb{E}[T_m]. \tag{8}$$

In addition, let $P(i)$ denotes the probability of observing the i-th distinct action configuration after observing $i-1$ distinct configurations. We have

$$P(i) = 1 - \frac{i-1}{m} = \frac{m-i+1}{m}. \tag{9}$$

Note that T_i follows a geometric distribution with success probability $P(i) = \frac{m-i+1}{m}$. Then the expected number of timesteps to see the i-th distinct configuration after seeing $i-1$ distinct configurations is

$$\mathbb{E}[T_i] = \frac{m}{m-i+1}. \tag{10}$$

Hence, we obtain

$$\begin{aligned}\mathbb{E}[T_m^{\text{non-share}}] &= \mathbb{E}[T_1] + \cdots + \mathbb{E}[T_m] \\ &= \sum_{i=1}^m \frac{m}{m-i+1} \\ &= m\sum_{i=1}^m \frac{1}{i}.\end{aligned} \tag{11}$$

From calculus, $\int_1^m \frac{1}{x}dx = \ln m$. Hence we obtain the following inequality

$$\sum_{i=1}^m \frac{1}{i+1} \le \int_1^m \frac{1}{x}dx = \ln m \le \sum_{i=1}^m \frac{1}{i}. \tag{12}$$

From Eq. (12), we obtain $\sum_{i=1}^m \frac{1}{i} = \mathcal{O}(\ln m)$[3] and $\sum_{i=1}^m \frac{1}{i} = \Omega(\ln m)$[4], which implies

$$\sum_{i=1}^m \frac{1}{i} = \Theta(\ln m). \tag{13}$$

Combining Eq. (11) and Eq. (13), we get $\mathbb{E}[T_m^{\text{non-share}}] = \Theta(m \ln m)$.

When performing exploration with shared-goal, the least visited action configuration will be chosen as the shared goal. The two agents coordinate to choose the actions that achieve the goal at each step. Hence, at each time step, the agents are able to visit a new action configuration. Therefore, exploration with shared goal needs m timesteps to visit all m action configurations, *i.e.*, $T_m^{\text{share}} = m$, which completes the proof. □

B. Proof of Claim 2

Claim 2. *Consider a special case of Example 1 where the payoff matrix depends only on one agent's action. Let T^{sub} denote the number of steps needed to discover the maximal reward when exploring the action space of agent one and agent two independently. Let T^{full} denote the number of steps needed to discover the maximal reward when the full action space is explored. Then, we have $T^{sub} = \mathcal{O}(l)$ and $T^{full} = \mathcal{O}(l^2)$.*

Proof. When we explore the action spaces of agent one and agent two independently, there are $2l$ distinct action configurations (l action configurations for each agent) to explore. Since the reward function depends only on one agent's action, one of these $2l$ action configurations must lead to the maximal reward. Therefore, by checking distinct action configurations at each time step, we need at most $2l$ steps to receive the maximal reward, *i.e.*, $\mathbb{E}[T^{\text{sub}}] = \mathcal{O}(l)$.

In contrast, when we explore the joint action space of agent one and agent two. There are l^2 distinct action configurations. Because the reward function depends only on one agent's action, l of these l^2 action configurations must lead to the maximal reward. In the worst case, we choose the $l^2 - l$ action configurations that don't result in maximal reward in the first $l^2 - l$ steps and receive maximal reward at the $l^2 - l + 1$ step. Therefore, we have

[2] $\Theta(g)$ means asymptotically bounded above and below by g.

[3] $\mathcal{O}(g)$ means asymptotically bounded above by g.

[4] $\Omega(g)$ means asymptotically bounded below by g.

$\mathbb{E}[T^{\text{full}}] = \mathcal{O}(l^2 - l + 1) = \mathcal{O}(l^2)$, which concludes the proof. □

C. Details Regarding MPE Environments

In this section we provide details regarding the sparse-reward and dense-reward version of MPE tasks. We first present the sparse-reward version of MPE:

- *Pass-sparse*: Two agents operate within two rooms of a 30×30 grid. There is one switch in each room, the rooms are separated by a door and agents start in the same room. The door will open only when one of the switches is occupied. The agents see collective positive reward and the episode terminates only when both agents changed to the other room. The task is considered solved if both agents are in the right room.
- *Secret-Room-sparse*: *Secret-Room-sparse* extends *Pass-sparse*. There are two agents and four rooms. One large room on the left and three small rooms on the right. There is one door between each small room and the large room. The switch in the large room controls all three doors. The switch in each small room only controls the room's door. All agents need to navigate to one of the three small rooms, *i.e.*, target room, to receive positive reward. The grid size is 25×25. The task is considered solved if both agents are in the target room.
- *Push-Box-sparse*: There are two agents and one box in a 15×15 grid. Agents need to push the box to the wall to receive positive reward. The box is heavy, so both agents need to push the box in the same direction at the same time to move the box. The task is considered solved if the box is pushed to the wall.
- *Island-sparse*: Two agents and a wolf operate in a 10×10 grid. Agents get a collective reward of 300 when crushing the wolf. The wolf and agents have maximum energy of eight and five respectively. The energy will decrease by one when being attacked. Therefore, one agent cannot crush the wolf. The agents need to collaborate to complete the task. The task is considered solved if the wolf's health reaches zero.

To study the performance of CMAE and baselines in a dense-reward setting, we add 'checkpoints' to guide the learning of the agents. Specifically, to add checkpoints, we draw concentric circles around a landmark, *e.g.*, a switch, a door, a box. Each circle is a checkpoint region. Then, the first time an agent steps in each of the checkpoint regions, the agent receive an additional checkpoint reward of $+0.1$.

- *Pass-dense*: Similar to *Pass-sparse*, but the agents see dense checkpoint rewards when they move toward the switches and the door. Specifically, when the door is open, agents receive up to ten checkpoint rewards when they move toward the door and the switch in the right room.
- *Secret-Room-dense*: Similar to *Secret-Room-sparse*, but the checkpoint rewards based on the agents' distance to the door and the target room's switch are added. Specifically, when the door is open, agents receive up to ten checkpoint rewards when they move toward the door and the switch in the target room.
- *Push-Box-dense*: Similar to *Push-Box-sparse*, but the checkpoint rewards based on the ball's distance to the wall is added. Specifically, agents receive up to six checkpoint rewards when they push the box toward the wall.
- *Island-dense*: Similar to *Island-sparse*, but the agent receives $+1$ reward when the wolf's energy decrease.

D. Details of SMAC environments

In this section, we present details for the sparse-reward and dense-reward versions of the SMAC tasks. We first discuss the sparse-reward version of the SMAC tasks.

- *3m-sparse*: There are three marines in each team. Agents need to collaboratively take care of the three marines on the other team. Agents only see a reward of $+1$ when all enemies are taken care of.
- *2m_vs_1z-sparse:* There are two marines on our team and one Zealot on the opposing team. In *2m_vs_1z-dense*, Zealots are stronger than marines. To take care of the Zealot, the marines need to learn to fire alternatingly so as to confuse the Zealot. Agents only see a reward of $+1$ when all enemies are taken care of.
- *3s_vs_5z-sparse*: There are three Stalkers on our team and five Zealots on the opposing team. Because Zealots counter Stalkers, the Stalkers have to learn to force the enemies to scatter around the map and attend to them one by one. Agents only see a reward of $+1$ when all enemies are attended to.

The details of the dense-reward version of the SMAC tasks are as follows.

- *3m-dense*: This task is similar to *3m-sparse*, but the reward is dense. An agent sees a reward of $+1$ when it causes damage to an enemy's health. A reward of -1 is received when its health decreases. All the rewards are collective. A reward of $+200$ is obtained when all enemies are taken care of.

	CMAE with QMIX	QMIX + bonus Weighted QMIX + bouns
Batch size	32	32
Discounted factor	0.99	0.99
Critic learning rate	0.0005	0.0005
Agent learning rate	0.0005	0.0005
Optimizer	RMSProp	RMSProp
Replay buffer size	5000	5000
Epsilon anneal step	50000	{50000, 1M}
Exploration bonus coefficient	N.A.	{1, 10, 50}
Goal bonus ($\hat{r}$)	{0.01, 0.1, 1}	N.A.

Table 3. Hyper-parameters of CMAE and baselines for SMAC tasks.

- *2m_vs_1z-dense*: Similar to *2m_vs_1z-sparse*, but the reward is dense. The reward function is similar to *3m-dense*.

- *3s_vs_5z-dense*: Similar to *3s_vs_5z-sparse*, but the reward is dense. The reward function follows the one in the *3m-dense* task.

Note that for all SMAC experiments we used StarCraft version SC2.4.6.2.69232. The results for different versions are not directly comparable since the underlying dynamics differ. Please see Samvelyan et al. (2019)[5] for more details regarding the SMAC environment.

E. Implementation Details

E.1. Normalized Entropy Estimation

As discussed in Sec. 3, we use Eq. (3) to compute the normalized entropy for a restricted space $\mathcal{S}_k$, *i.e.*,

$$\eta_k = H_k / H_{\max,k} = -\left(\sum_{s\in\mathcal{S}_k} p_k(s)\log p_k(s)\right) / \log(|\mathcal{S}_k|).$$

Note that $|\mathcal{S}_k|$ is typically unavailable even in discrete state spaces. Therefore, we use the number of current observed distinct outcomes $|\hat{\mathcal{S}_k}|$ to estimate $|\mathcal{S}_k|$. For instance, suppose $\mathcal{S}_k$ is a one-dimensional restricted state space and we observe $\mathcal{S}_k$ takes values $-1, 0, 1$. Then $|\hat{\mathcal{S}_k}| = 3$ is used to estimate $|\mathcal{S}_k|$ in Eq. (3). $|\hat{\mathcal{S}_k}|$ typically gradually increases during exploration. In addition, for $|\hat{\mathcal{S}_k}| = 1$, *i.e.*, for a constant restricted space, the normalized entropy will be set to infinity.

E.2. Architecture and Hyper-Parameters

We present the details of architectures and hyper-parameters of CMAE and baselines next.

MPE environments: We combine CMAE with Q-learning. For *Pass*, *Secret-room*, and *Push-box*, the Q value function is represented via a table. The Q-table is initialized to zero. The update step size for exploration policies and target policies are 0.1 and 0.05 respectively. For *Island* we use a DQN (Mnih et al., 2013; 2015). The Q-function is parameterized by a three-layer perceptron (MLP) with 64 hidden units per layer and ReLU activation function. The learning rate is 0.0001 and the replay buffer size is $1M$. In all MPE tasks, the bonus $\hat{r}$ for reaching a goal is 1, and the discount factor γ is 0.95.

For the baseline EITI and EDTI (Wang et al., 2020), we use their default architecture and hyper-parameters. The main reason that EITI and EDTI need a lot of environment steps for convergence according to our observations: a long rollout (512 steps × 32 processes) between model updates is used. In an attempt to optimize the data efficiency of baselines, we also study shorter rollout length, *i.e.*, {128, 256}, for both EITI and EDTI. However, we didn't observe an improvement over the default setting. Specifically, after more than 500M environment steps of training on *Secret-Room*, EITI with 128 and 256 rollout length achieves 0.0% and 54.8% success rate. EDTI with 128 and 256 rollout length achieves 0.0% and 59.6% success rate, which is much lower than the success rate of 80% achieved by using the default setting.

SMAC environment: We combine CMAE with QMIX (Rashid et al., 2018). Following their default setting, for both exploration and target policies, the agent is a DRQN (Hausknecht & Stone, 2015) with a GRU (Chung et al., 2014) recurrent layer with a 64-dimensional hidden state. Before and after the GRU layer is a fully-connected layer of 64 units. The mix network has 32 units. The discount factor γ is 0.99. The replay memory stores the latest 5000 episodes, and the batch size is 32. RMSProp is used with a learning rate of $5 \cdot 10^{-4}$. The target network is updated every 100 episodes. For goal bonus $\hat{r}$ (Alg. 2), we studied $\{0.01, 0.1, 1\}$ and found

[5] https://github.com/oxwhirl/smac

		CMAE (Ours)	Q-learning	Q-learning + Bonus	EITI	EDTI
Pass-sparse	Final	**1.00±0.00**	0.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00
	Absolute	**1.00±0.00**	0.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00
Secret-Room-sparse	Final	**1.00±0.00**	0.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00
	Absolute	**1.00±0.00**	0.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00
Push-Box-sparse	Final	**1.00±0.00**	0.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00
	Absolute	**1.00±0.00**	0.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00
Island-sparse	Final	**0.55±0.30**	0.00±0.00	0.00±0.00	0.00±0.00	0.00±0.00
	Absolute	**0.61±0.23**	0.00±0.00	0.01±0.01	0.00±0.00	0.00±0.00
Pass-dense	Final	**5.00±0.00**	1.25±0.02	1.42±0.14	0.00±0.00	0.18±0.01
	Absolute	**5.00±0.00**	1.30±0.03	1.46±0.08	0.00±0.00	0.20±0.01
Secret-Room-dense	Final	**4.00±0.57**	1.62±0.16	1.53±0.04	0.00±0.00	0.00±0.00
	Absolute	**4.00±0.57**	1.63±0.03	1.57±0.06	0.00±0.00	0.00±0.00
Push-Box-dense	Final	1.38±0.21	**1.58±0.14**	1.55±0.04	0.10±0.01	0.05±0.03
	Absolute	1.38±0.21	**1.59±0.04**	1.55±0.04	0.00±0.00	0.18±0.01
Island-dense	Final	**138.00±74.70**	87.03±65.80	110.36±71.99	11.18±0.62	10.45±0.61
	Absolute	163.25±68.50	141.60±92.53	**170.14±62.10**	16.84±0.65	16.42±0.86

Table 4. Final metric and *absolute metric* of CMAE and baselines on sparse-reward and dense-reward MPE tasks.

		CMAE (Ours)	Weighted QMIX	Weighted QMIX + Bonus	QMIX	QMIX + Bonus
3m-sparse	Final	**47.7±35.1**	2.7±5.1	11.5±8.6	0.0±0.0	11.7±16.9
	Absolute	**62.0±41.0**	8.1±4.5	15.6±7.3	0.0±0.0	22.8±18.4
2m_vs_1z-sparse	Final	**44.3±20.8**	0.0±0.0	19.4±18.1	0.0±0.0	19.8±14.1
	Absolute	**47.7±35.1**	0.0±0.0	23.9±16.7	0.0±0.0	30.3±26.7
3s_vs_5z-sparse	Final	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
	Absolute	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
3m-dense	Final	98.7±1.7	98.3±2.5	**98.9±1.7**	97.9±3.6	97.3±3.0
	Absolute	99.3±1.8	98.8±0.3	99.0±0.3	**99.4±2.1**	98.5±1.2
2m_vs_1z-dense	Final	98.2±0.1	**98.5±0.1**	96.0±1.8	97.1±2.4	95.8±1.7
	Absolute	98.7±0.4	98.6±1.6	99.1±0.9	**99.1±0.6**	96.0±1.6
3s_vs_5z-dense	Final	81.3±16.1	92.2±6.6	**95.3±2.2**	75.0±17.6	78.1±24.4
	Absolute	85.4±22.6	**95.4±4.4**	95.4±3.2	76.5±24.3	79.1±14.2

Table 5. Final metric and *absolute metric* of success rate (%) of CMAE and baselines on sparse-reward and dense-reward SMAC tasks.

0.1 to work well in most tasks. Therefore, we use $\hat{r} = 0.1$ for all SMAC tasks. The hyper-parameters of CMAE with QMIX and baselines are summarized in Tab. 3.

F. Absolute Metric and Final Metric

In addition to the final metric reported in Tab. 1 and Tab. 2, following Henderson et al. (2017); Colas et al. (2018), we also report the *absolute metric*. Absolute metric is the best policies' average episode reward over 100 evaluation episodes. The final metric and absolute metric of CMAE and baselines on MPE and SMAC tasks are summarized in Tab. 4 and Tab. 5.

G. Additional Results on MPE Task: Island

In addition to the MPE tasks considered in Sec. 4, we consider one more challenging MPE task: Island. The details of both sparse-reward and dense-reward version of Island, *i.e.*, *Island-sparse* and *Island-dense* are presented in Sec. C. We compare CMAE to Q-learning, Q-learning with count-based exploration, EITI, and EDTI on both *Island-sparse* and *Island-dense*. The results are summarized in Tab. 4. As Tab. 4 shows, in the sparse-reward setting, CMAE is able to achieve higher than 50% success rate. In contrast, baselines struggle to solve the task. In the dense-reward setting, CMAE performs similar to baselines. The training curves are shown in Fig. 5 and Fig. 6.

Task (target success rate)	CMAE (Ours)	EITI	EDTI
Pass-sparse (80%)	**2.43M±0.10M**	384M±1.2M	381M±2.8M
Secret-Room-sparse (80%)	**2.35M±0.05M**	448M±10.0M	382M±9.4M
Push-Box-sparse (10%)	**0.47M±0.04M**	307M±2.3M	160M±12.1M
Push-Box-sparse (80%)	**2.26M±0.02M**	307M±3.9M	160M±8.2M
Island-sparse (20%)	**7.50M±0.12M**	480M±5.2M	322M±1.4M
Island-sparse (50%)	**13.9M±0.21M**	> 500M	> 500M

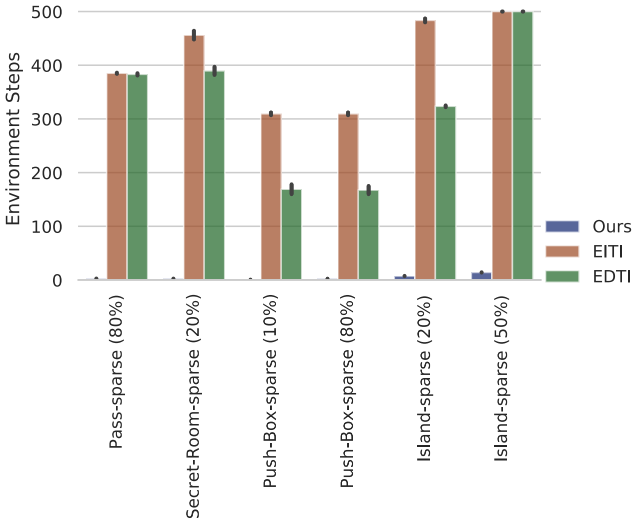

Table 6. Environment steps required to achieve the indicated target success rate on *Pass-sparse*, *Secret-Room-sparse*, *Push-Box-sparse*, and *Island-sparse* environments.

H. Additional Results of Baselines

Following the setting of EITI and EDTI (Wang et al., 2020), we train both baselines for 500M environment steps. On *Pass-sparse*, *Secret-Room-sparse*, and *Push-Box-sparse*, we observe that EITI and EDTI (Wang et al., 2020) need more than 300M steps to achieve an 80% success rate. In contrast, CMAE achieves a 100% success rate within 3M environment steps. On *Island-sparse*, EITI and EDTI need more than 3M environment steps to achieve a 20% success rate while CMAE needs less than 8M environment steps to achieve the same success rate. The results are summarized in Tab. 6.

I. Additional Training Curves

The training curves of CMAE and baselines on both sparse-reward and dense-reward MPE tasks are shown in Fig. 5 and Fig. 6. The training curves of CMAE and baselines on both sparse-reward and dense-reward SMAC tasks are shown in Fig. 7 and Fig. 8. As shown in Fig. 5, Fig. 6, Fig. 7, and Fig. 8, in challenging sparse-reward tasks, CMAE consistently achieves higher success rate than baselines. In dense-reward tasks, CMAE has similar performance to baselines.

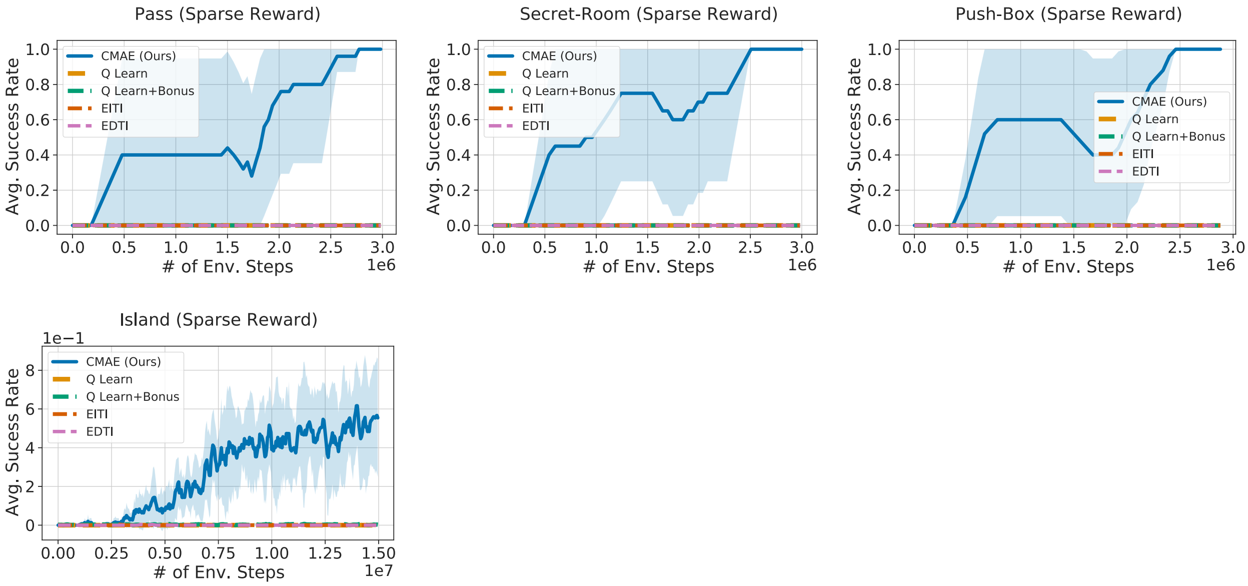

Figure 5. Training curves on sparse-reward MPE tasks.

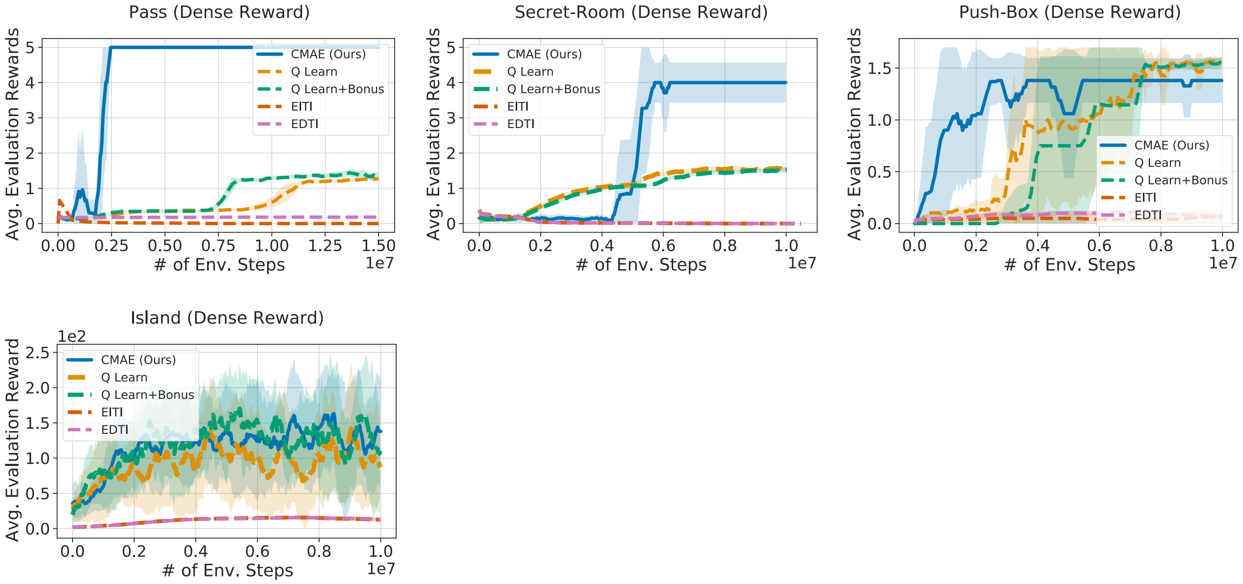

Figure 6. Training curves on dense-reward MPE tasks.

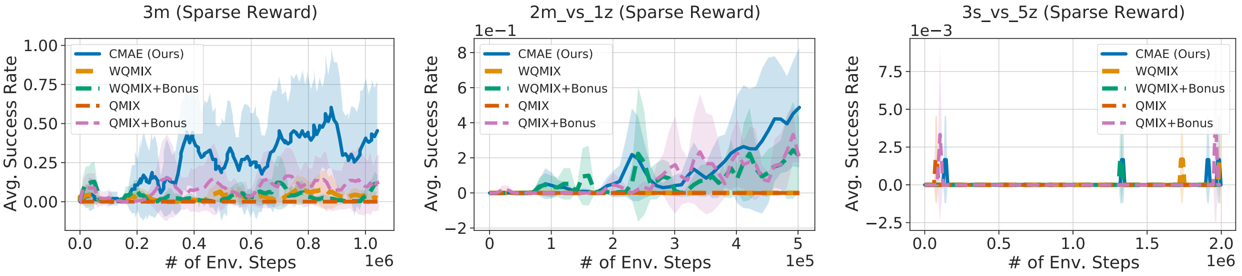

Figure 7. Training curves on sparse-reward SMAC tasks.

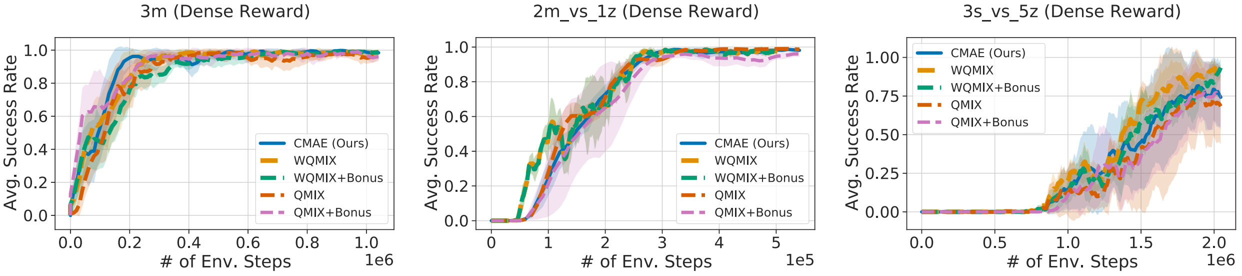

Figure 8. Training curves on dense-reward SMAC tasks.