Appendix A Implementation Details.

Environments. In our experiments, except for Mujoco style tasks, we also incorporate MetaWorld tasks and AntMaze navigation tasks. The MetaWorld benchmark is paramount in multi-task reinforcement learning. Antmaze tasks from D4RL are widely used in offline RL and offline-to-online RL settings. Although the Antmaze environment does not involve reward changes, it presents significant challenges in terms of exploration and sparse rewards. We list the choice of $M_i$ and $M_j$ in the experimental section below.

<table>
<thead>
<tr>
<th>Environments</th>
<th>$M_i$ (offline task)</th>
<th>$M_j$ (online task)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quadruped</td>
<td>walk</td>
<td>roll</td>
</tr>
<tr>
<td>Walker</td>
<td>run</td>
<td>jump</td>
</tr>
<tr>
<td>top left</td>
<td>bottom left</td>
<td></td>
</tr>
<tr>
<td>top right</td>
<td>bottom right</td>
<td></td>
</tr>
<tr>
<td>bottom right</td>
<td>bottom left</td>
<td></td>
</tr>
<tr>
<td>MetaWorld</td>
<td>drawer-close-v2</td>
<td>drawer-open-v2</td>
</tr>
<tr>
<td>window-open-v2</td>
<td>window-close-v2</td>
<td></td>
</tr>
<tr>
<td>Antmaze</td>
<td>medium</td>
<td>medium</td>
</tr>
<tr>
<td>large</td>
<td>large</td>
<td></td>
</tr>
</tbody>
</table>

Datasets. The offline datasets in our experiments for the Quadruped, Walker, and Reach environments are obtained from UTDS [53]. UTDS employs the TD3 [48] algorithm to collect data at different levels. The dataset consists of four types: medium-replay, medium, expert, and replay. The medium-replay data encompasses all experiences collected during the training of a medium-level TD3 agent, while the replay data includes all experiences obtained from training an expert-level TD3 agent. The medium and expert data denote narrower data distributions, where the medium data is generated by a medium-level TD3 agent and the expert data is generated by an expert-level TD3 agent. The AntMaze dataset is obtained from D4RL [55], a widely used resource in offline RL. The offline dataset of MetaWorld tasks is obtained through interactions with the officially provided scripted policy.

Implementation of our method and offline RL methods. For Quadruped, Walker, and Reach tasks, we use the Jax implementations of baseline methods from jrlzoo\(^1\). We build the implementation of ESR-O2O on top of rlpd\(^2\). For Antmaze tasks, we cite the reported scores from CORL\(^3\). Agents are pre-trained for 1M steps and then fine-tuned for 250k steps. Each method on each task has been run across 5 random seeds. In Antmaze environments, since behavior cloning is commonly employed in other offline-to-online learning methods, we incorporate it into actor learning as well.

Implementations of offline-to-online RL methods. In our work, we compare ESR-O2O with many offline-to-online RL algorithms, including AWAC, Off2On, PEX, and PROTO, though they are designed for online fine-tuning without reward gaps. For Off2On, PEX, and PROTO, we build on top of their official codes, with light modifications to be suitable for our tasks. We use their default architectures and hyper-parameters. Agents are pre-trained for 500k timesteps during the offline stage and then fine-tuned for 250k timesteps in the new environment.

---

1) https://github.com/fuyw/jrlzoo
2) https://github.com/ikostrikov/rlpd
3) https://github.com/tinkoff-ai/CORL
Appendix B

Appendix B.1 Comparisons with offline RL methods.

When the reward gap is large, all offline RL methods cannot obtain effective policies during fine-tuning, as shown in Figure B1. In scenarios with small reward gaps, it is possible to leverage pre-trained policies or value functions for the new task. IQL and AWAC, which employ the same policy extraction method, demonstrate robustness in task switching. Conversely, CQL and TD3-BC, which extract policies by maximizing the Q-functions, are more sensitive to the distributional shift, resulting in instability. Figure B2 illustrates that ESR-O2O significantly outperforms other methods when provided with medium’, medium-replay’, and ‘replay’ data, while still maintaining comparable performance with the top-performing baseline method, IQL. In certain tasks such as Walker, ESR-O2O even surpasses the performance of IQL.

Figure B1  Fine-tuning performance in scenarios with large reward gaps. While baseline methods struggle to perform well across all tasks during the fine-tuning stage, ESR-O2O demonstrates significant improvements in the fine-tuned policies, regardless of the quality of the offline datasets.

Figure B2  Fine-tuning performance in scenarios with large reward gaps. While baseline methods struggle to perform well across all tasks during the fine-tuning stage, ESR-O2O demonstrates significant improvements in the fine-tuned policies, regardless of the quality of the offline datasets.
Appendix B.2 Results in the MetaWorld benchmark.

We have conducted experiments on the MetaWorld benchmark, which contains a variety of challenging tasks and serves as an excellent platform for evaluating generalization capabilities in the presence of reward gaps. As shown in Table A1, we consider four specific tasks, including the transition from 'drawer-close-v2' to 'drawer-open-v2' and the transition from 'window-open-v2' to 'window-close-v2'. Although the offline and online tasks share the same dynamics function, they have distinct reward functions. These manipulation tasks are notably more challenging compared to the DMC Mujoco locomotion tasks. In addition to the tabular comparison provided in Table 1, we also present average return curves across five random seeds in Figure B3. These results clearly demonstrate the superior performance of our method during fine-tuning when compared to PEX and PROTO, underscoring our method’s exceptional generalization ability.

![Figure B3](image.png)

**Figure B3** Fine-tuning performance in MetaWorld tasks. Since these manipulation tasks are more challenging than Mujoco locomotion tasks, current offline-to-online RL methods cannot perform well during fine-tuning like the results in Figure 8. In contrast, ESR-O2O is robust to reward gaps and successfully adapts to the new environment.

Appendix C More explanations about ablation studies.

Whether using ensembles: As illustrated in Figure 9, eliminating the ensemble of SR significantly impairs the fine-tuning performance. In scenarios with a large reward gap, such as the reach task, we observe that useful policies cannot be learned without the ensemble of SR. This effect is even more pronounced when the available data is generated from narrow data distributions. For scenarios with a smaller reward gap, such as the quadruped task, the performance drop is relatively small when eliminating the ensemble of SR alone, but becomes substantial when both the ensemble of SR and Q networks are discarded. In summary, the ensemble of SR is crucial for addressing scenarios with a large reward gap. Additionally, the ensemble of Q networks plays a significant role when dealing with scenarios with a small reward gap or when the offline data is abundant.

The quantity of ensembles: In our extended experiments, we systematically varied the ensemble’s network count from 2 to 10, and the results are presented in Figure 10 of the revised manuscript. Our results revealed that when we use more than 6 ensemble networks, the performance is consistently stable. Meanwhile, the stability and performance of offline-to-online adaptation improve as the number of networks increases. In contrast, reducing the number of networks to 4 led to a substantial decrease in performance in specific tasks. Notably, when the network count reduces to 2, the networks struggled to learn effective representations through online fine-tuning. Furthermore, it is worth noting that we implemented a network parallelization technique in our code, similar to the approach described in [1]. This parallelization method substantially mitigates the computation time overhead associated with increasing the number of networks.

Appendix D More experimental results in Reach, Quadruped, and Walker.

Additional experimental results about the fine-tuning performance are presented on Figures D1, D2, and D3.
Figure D1  Performance comparisons on reach tasks.
Figure D2 Performance comparisons on quadruped tasks.
Figure D3  Performance comparisons on walker tasks.