We thank the reviewers' valuable comments. Our responses are listed below.

To Reviewer 1: (Q1)"...mechanic of policy transfer is not given,..., transfer means a simple copying of parameters...".
A: We want to highlight that CHDRL follows the basic mechanism of transfer learning [ref1] including 3 essential points, i.e., *what, how, and when to transfer*. We handle all the 3 points in policy transfer among heterogeneous agents. The difficulties and the corresponding solutions are elaborated in Lines 136-160. "simple copying of the parameters" is only a small portion of *what to transfer*, and actually CHDRL does more than copying of the parameters as different agents have different policy structures which makes the simple copy infeasible.
(Q2)"augmenting the experience buffer with other algorithm, ...,clarify why it does (not) introduce any bias in the data."
A: Global agents are not simply replaying experiences from the global memory that augments experiences of different agents. Instead, they replay experiences from global and local memory buffers following a probability distribution, which alleviates the bias caused by replaying global memory only. Moreover, for local memory, intuitive rules are set to store local agents' experiences as elaborated in Lines 173-175. The rules guarantee that local memory only saves experiences similar to the global agent's current policy, which further reduces bias, as evidenced in [14].
(Q3)"... be replaced by a different way of' tinkering 'with a algorithm or its hyperparameters,... on-policy algorithms are here mainly for exploration, but..., with the right setting, work as good as the given complicated framework.". **A**: As evidenced by [6,14], off-policy agents suffer from bootstrapping error and extrapolation error regardless of how hyper-parameters are fine-tuned. It is also well known that Off-policy Q learning is not to converge even with linear function approximation [ref2]. Thus, to the best of our knowledge, it is extremely hard to overcome these issues by "tinkering." These points are our motivations to integrate different methods. Moreover, except for benefiting exploration from on-policy agents, we also maintain other on-policy agents' advantages, e.g., stability, in CHDRL, as shown in Lines 65-74. Verified by Section 5.3, on-policy agents do help find a better policy around the off-policy agent.
(Q4)"...the size of the experience replay buffer can be easily controlled. These settings are not explored, ...".
A: The intuition here is to enable off-policy agents to benefit from diverse local experiences so that it can make more progress [6,14]. Uniformly replaying experiences from global memory that augments all the experiences of different agents definitely fail to do so. Thus, we propose a global-local memory buffer and employ a rule to save the experiences that are similar to the off-policy agents' current policy in local memory, see Lines 172-175. The size of the local memory is fixed. We don't specifically fine-tune it in the experiment, and already achieve good improvements. The global memory is similar to the conventional RL methods' setting, i.e., SAC, and we simply follow the default configuration.
(Q5) "..,X-axis on the graphs is labeled 'time-steps'...,generate 3-times more "time-steps", ... source code is not given."
A: Time-steps are accumulated interaction steps with the environment. For a fair comparison, we use the accumulated time steps of 3 algorithms. Specifically, we sum up each agent's time steps so that the total time-steps stay consistent with the other baselines. We will publish our source code.
(Q6) "...'How to Transfer' which do not give any details on how to transfer." **A**: We elaborate on the policy transfer from *what, how, and when to transfer* in section 4. How to transfer focuses on the transfer methodology we employed, i.e., a hierarchical transfer manner: 1) off-policy agents to both on-policy agents and EAs agents, and 2) from on-policy agents to EAs agents. We guess the reviewer may question on *what to transfer*. We transfer $\mu_\phi(s)$ among different agents. Line 10-19 in Alg 1 clearly shows *what, how, and when to transfer*.
(Q7)"...No work combining different algorithms is presented...". **A**: To the best of our knowledge, this is the first work combining heterogeneous agents, including off-policy, on-policy, and Evolutionary-based RL methods.

To Reviewer 2: (Q1) "...choice of hyperparameters... the paper did not provide information on how to choose..."
A: For CHDRL, we did not specifically fine-tune the hyperparameters and already achieved good results. We believe the performance can be further boosted by fine-tuning the hyper-parameters.
(Q2)"...better to include theoretical analysis, e.g, the convergence guarantee,...".
A: We agree that theoretical analysis makes the work more solid. We will work on this in future work.

To Reviewer 3: (Q1)"Possible Computation Cost...". **A**: The computation cost of CHDRL mainly comes from the global agent, which is comparable with SAC in our case, as it keeps learning for other agents' experiences in the background. The local agents run much faster than global agents, especially the EA agent, as it is gradient-free.

To Reviewer 4 (Q1)"The results on mujoco tasks are not that impressive compared to state of the art..., these hyperparameters can provide better results, they require further optimization by the user...".
A: As shown in Figure 2, CHDRL achieves clear improvements on the first four tasks. Regarding the mean of the max average return for the four tasks, CHDRL is 4921.25, which outperforms PPO (1526.75) 3394.5, SAC (4172) 749.5, and CEM (933) 3988.25. For the 5th task where conventional RL methods fail, CHDRL still achieves comparable results with CEM, as SAC and PPO do not help, CHDRL only reflects the advantage of CEM. Moreover, we did not fine-tune the hyper-parameters. We believe CHDRL can achieve better results after carefully fine-tuning.
(Q2)"I couldn't understand if the figure 3d is missing data or not.". **A**: It is because LM/GM/CL cannot enhance the final performance in task Swimmer, and the learning curves of the three methods are mostly overlapped. In this case, the gradient-based agents fail to learn, and only gradient-free CEM agent works.

[ref1] A Survey on Transfer Learning.
[ref2] Residual Algorithms: Reinforcement Learning with Function Approximation.