COLLABORATIVE SPECIALIST KNOWLEDGE DISTILLATION FOR FEDERATED OFFLINE REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated offline reinforcement learning coordinates policies trained from decentralized offline experience, yet model-level aggregation can underutilize complementary behaviors acquired by individual clients. We propose Collaborative Specialist Knowledge Distillation (CSKD), which preserves client-specific specialization across communication rounds and transfers specialist behavior into a unified shared policy. Each client maintains a persistent residual adaptation and performs local Conservative Q-Learning, while the server composes state–policy targets contributed by multiple specialists and integrates them through policy-space distillation. On heterogeneous D4RL MuJoCo benchmarks, CSKD achieves normalized scores of 48.04 on HALFCHEETAH, 100.16 on HOPPER, and 88.54 on WALKER2D, with an average score of 78.91. Additional experiments examine repeated collaboration, client scalability, communication cost, and experience overlap, showing that specialist knowledge can be transferred effectively with substantially lower client-to-server payload than full-model exchange.

1 INTRODUCTION

1.1 MOTIVATION AND CONTRIBUTIONS

Federated offline reinforcement learning (FedORL) combines federated learning with offline reinforcement learning, enabling multiple clients to collaboratively learn a shared policy from fixed decentralized datasets without additional environment interaction (McMahan et al., 2017; Li et al., 2020; Levine et al., 2020; Kumar et al., 2020). Existing FedORL methods address heterogeneous clients through federated optimization, coverage analysis, policy regularization, and ensemble-based coordination (Qi et al., 2021; Jin et al., 2022; Woo et al., 2024; Yue et al., 2024; Rengarajan et al., 2024; Qiao et al., 2025; Qiao & Yue, 2026). These approaches improve how local models, value estimates, or policy updates are coordinated, but the behavioral capabilities independently acquired by local policies remain less explicitly utilized.

Heterogeneous local experience can produce policies with distinct behavioral capabilities. We refer to the behavior acquired from each client's partial experience as *specialist knowledge*. A client trained on a restricted portion of the offline experience may be incomplete as a global policy, yet still encode useful behavior on states represented in its local dataset. Consequently, disagreement among local policies does not necessarily indicate noisy or poorly optimized clients; it can also reflect complementary specialization induced by different local experience distributions. Model-level aggregation does not explicitly preserve such state-conditioned behavior, and useful adaptations may therefore be weakened before the shared policy can exploit them. This motivates our central question: *how can complementary specialist behaviors be preserved and integrated into a single shared policy?*

We address this problem with **Collaborative Specialist Knowledge Distillation (CSKD)**. CSKD preserves client-specific adaptations across communication rounds and transfers locally acquired behavior through state–policy knowledge. The server organizes specialist targets over their associated states, composes knowledge from multiple clients, and distills the resulting behavioral supervision into a unified shared policy. Our contributions are threefold. First, we identify a *specialist*

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

knowledge utilization problem in FedORL, where useful behavior learned from complementary local experience can be weakened by model-level collaboration. Second, we introduce CSKD, which combines persistent local specialization with joint specialist knowledge integration and policy-space distillation. Third, we evaluate CSKD under heterogeneous offline experience, repeated collaboration, increasing client populations, communication constraints, and varying experience overlap.

1.2 Related Work

Federated reinforcement learning extends federated optimization to sequential decision making (Qi et al., 2021; Jin et al., 2022). In federated offline RL, FedLCB-Q studies collaborative policy coverage (Woo et al., 2024), DRPO addresses local and global distribution shift through policy regularization (Yue et al., 2024), and FEDORA performs ensemble-directed federation under heterogeneous offline experience (Rengarajan et al., 2024). FOVA (Qiao et al., 2025) and FORLER (Qiao & Yue, 2026) further consider mixed-quality data and policy pollution. These methods mainly improve model-, value-, or update-level coordination, whereas CSKD focuses on behavioral knowledge retained by locally specialized policies.

Knowledge distillation transfers prediction- or policy-level behavior without requiring direct parameter averaging (Hinton et al., 2015; Rusu et al., 2015). Federated Distillation, FedMD, and FedDF use prediction-level knowledge for federated collaboration (Jeong et al., 2018; Li & Wang, 2019; Lin et al., 2020), while policy distillation has been used to combine multiple skills in reinforcement learning (Berseth et al., 2018; Lupu et al., 2024). SCCD (Mai et al., 2023) and FedHPD (Jiang et al., 2025) further introduce distillation-based collaboration into federated RL. CSKD differs by treating heterogeneous offline experience as a source of persistent local specialization and jointly transferring locally supported specialist behaviors into one shared policy.

2 Preliminaries

2.1 Federated Offline Reinforcement Learning

We consider the standard server–client optimization paradigm used in federated learning. At communication round t, the server broadcasts a shared model θ_s^t, and each client performs local optimization on its private dataset to obtain θ_i^{t+1}. A representative aggregation rule is FedAvg (McMahan et al., 2017),

$$\theta_s^{t+1} = \sum_{i=1}^{N} w_i \theta_i^{t+1}, \qquad \sum_{i=1}^{N} w_i = 1, \tag{1}$$

where w_i denotes the aggregation weight of client i. FedProx (Li et al., 2020) further regularizes local optimization toward the current shared model,

$$\theta_i^{t+1} \approx \arg\min_{\theta} F_i(\theta; \mathcal{D}_i) + \frac{\mu}{2} \left\| \theta - \theta_s^t \right\|_2^2. \tag{2}$$

Both formulations operate primarily in parameter space: local models or updates are first optimized independently and then coordinated by the server.

Federated offline reinforcement learning applies this paradigm to policies learned from fixed local datasets. Each client i owns an offline dataset $\mathcal{D}_i = \{(s, a, r, s')\}$ and cannot collect additional interaction data during federated training. The local learner must therefore optimize a policy while controlling extrapolation error induced by actions outside the support of $\mathcal{D}_i$. Representative offline RL methods include BCQ (Fujimoto et al., 2019), BEAR (Kumar et al., 2019), CQL (Kumar et al., 2020), TD3+BC (Fujimoto & Gu, 2021), IQL (Kostrikov et al., 2021), and Cal-QL (Nakamoto et al., 2023). In our implementation, each client uses CQL as its local offline RL backbone. At a high level, CQL augments the temporal-difference objective with a conservative regularizer,

$$\mathcal{L}_{\mathrm{CQL}} = \mathcal{L}_{\mathrm{TD}} + \alpha \mathcal{R}_{\mathrm{CQL}}, \tag{3}$$

where $\mathcal{R}_{\mathrm{CQL}}$ penalizes overestimated values on actions that are insufficiently supported by the offline data. Our contribution does not modify the underlying offline RL objective; instead, it changes how the behaviors learned by different local policies are preserved and integrated at the federation level.

2.2 POLICY-SPACE KNOWLEDGE TRANSFER

Policy distillation transfers behavioral knowledge from a teacher policy π_T to a student policy π_θ by matching their outputs over a transfer-state distribution (Hinton et al., 2015; Rusu et al., 2015). A generic distillation objective is

$$\mathcal{L}_{\mathrm{PD}}(\theta) = \mathbb{E}_{s\sim\rho}\left[\mathcal{D}_\pi\left(\pi_T(\cdot \mid s), \pi_\theta(\cdot \mid s)\right)\right], \tag{4}$$

where ρ denotes the state distribution used for knowledge transfer and $\mathcal{D}_\pi$ measures the discrepancy between teacher and student policies. Unlike parameter aggregation in Eq. (1), policy distillation operates directly on behavior and therefore does not require the teacher and student to share identical parameterizations.

Under heterogeneous offline experience, local policies can encode useful behavior over different state distributions even when their parameters are difficult to combine directly. CSKD therefore transfers policy behavior rather than relying only on parameter aggregation. In Section 3, we extend the single-teacher objective in Eq. (4) to multiple local specialists whose behavioral supervision is supported by different client-specific state distributions.

3 PROBLEM FORMULATION

3.1 PROBLEM SETTING

We consider a federated offline reinforcement learning setting with N clients solving the same underlying Markov decision process (MDP), denoted by $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, P is the transition dynamics, r is the reward function, and γ is the discount factor. Each client i owns a fixed offline dataset $\mathcal{D}_i = \{(s, a, r, s')\}$ collected before federated training. All clients share the same task and environment, but their datasets may contain different and partially overlapping portions of the available experience. No additional interaction with the environment is assumed during federated training.

The goal is to learn a single shared policy π_s that achieves high expected discounted return over the underlying task,

$$\pi_s^* = \arg\max_\pi J(\pi), \qquad J(\pi) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]. \tag{5}$$

Unlike personalized federated learning, our objective is not to maintain a separate deployment policy for each client. Instead, all decentralized experiences should ultimately contribute to improving the same shared policy.

In conventional federated offline RL, each client performs local policy optimization using only $\mathcal{D}_i$, after which the server coordinates the resulting local models or updates. However, because different clients are trained from different portions of the offline experience, their resulting policies may acquire different behavioral capabilities even though they solve the same task. The resulting problem is therefore to integrate behaviors learned from different local experience distributions into the shared policy.

3.2 LOCAL SPECIALIST POLICIES AND KNOWLEDGE INTEGRATION

Let π_i denote the policy locally optimized by client i using its offline dataset $\mathcal{D}_i$. Since $\mathcal{D}_i$ contains only part of the experience available across the federation, π_i is not assumed to be globally optimal or uniformly better than the shared policy. Instead, we regard π_i as a *local specialist policy*: its specialization refers to behavioral capabilities acquired from the particular experience available to client i. Different clients can therefore develop complementary capabilities while optimizing the same underlying task.

We treat the collection of locally learned policies $\{\pi_i\}_{i=1}^N$ as distributed sources of specialist knowledge. Extending the single-teacher formulation in Eq. (4), each specialist contributes behavioral supervision over states supported by its own local experience. Let ρ_i denote the corresponding state

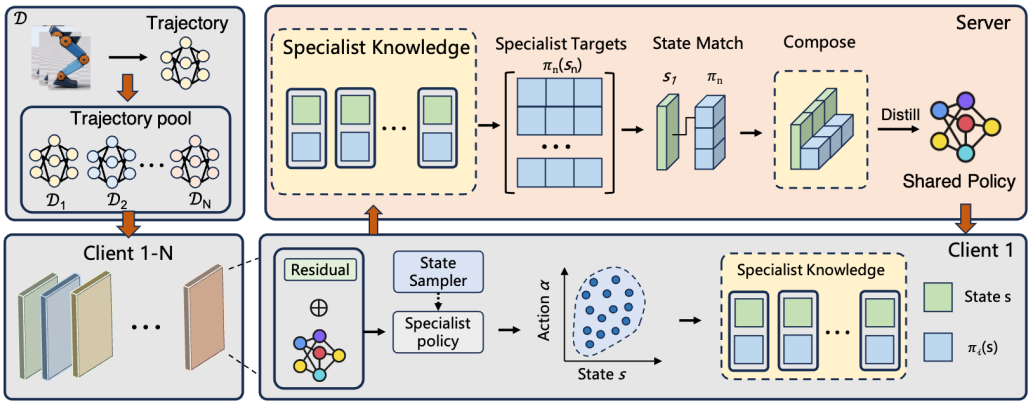

Figure 1: Overview of CSKD. Each client maintains a persistent local specialization and exports policy behavior over sampled states. The server matches states with specialist targets, composes the matched knowledge, and distills it into the shared policy for the next communication round.

distribution. The resulting specialist knowledge integration objective is

$$\pi_s^* = \arg\min_{\pi_s} \sum_{i=1}^{N} \mathbb{E}_{s\sim\rho_i} \left[\mathcal{D}_\pi \left(\pi_i(\cdot|s), \pi_s(\cdot|s)\right)\right]. \tag{6}$$

Eq. (6) expresses knowledge transfer directly in policy space. Each client contributes behavioral targets over states supported by its local experience, while the shared policy is trained to absorb these targets into a single deployable policy. This formulation does not assume that a local specialist is globally superior to the shared policy. Instead, the relevant object of transfer is the behavior that the specialist has learned on the experience available to that client.

This view separates two problems that are coupled in conventional federated optimization. The first is *specialization preservation*: a useful local adaptation must survive repeated communication rounds rather than being reset whenever a new shared model is broadcast. The second is *specialist integration*: preserving local policies alone does not improve the final shared policy unless their behavioral knowledge can be transferred to the server in a common representation. CSKD addresses these two requirements separately. Persistent local specialist learning maintains client-specific adaptations across rounds, while collaborative specialist knowledge distillation converts locally supported policy behavior into server-side training targets. Section 4 describes the resulting procedure.

4 Method

Method overview. At round t, the server broadcasts π_s^t, while each client retains a persistent local adaptation and optimizes a specialist policy π_i^t. Clients transfer state–policy knowledge $\mathcal{K}_i^t$ to the server, where the contributed targets are composed and jointly distilled into π_s^{t+1}. Figure 1 summarizes the procedure.

4.1 Persistent Local Specialist Learning

Directly replacing a local policy with the newly broadcast shared policy can erase adaptations acquired from local experience. CSKD instead decomposes client i into a shared component g_s^t and a persistent residual δ_i^t,

$$g_i^t(s) = g_s^t(s) + \delta_i^t(s), \tag{7}$$

which induces the specialist policy π_i^t. Persistence is implemented by carrying the local residual across communication rounds:

$$\delta_i^{t,0} \leftarrow \delta_i^{t-1,*}, \qquad g_i^{t,0}(s) = g_s^t(s) + \delta_i^{t,0}(s), \tag{8}$$

where $*$ denotes the residual obtained after local optimization in the previous round. Only the shared component is replaced after a server broadcast.

We use CQL (Kumar et al., 2020) for local offline optimization. The shared component is fixed during the local actor update, and only the residual is optimized:

$$\mathcal{L}^i_{\text{local}} = -\mathbb{E}_{s\sim\mathcal{D}_i}\left[\min_{j\in\{1,2\}} Q_{i,j}\left(s,\, \text{clip}\left(g^t_s(s) + \delta^t_i(s)\right)\right)\right]. \tag{9}$$

The local critics follow the standard CQL objective and remain private to the client. After optimization, δ^t_i retains the client-specific adaptation, while g^t_s provides the common policy basis shared across the federation. Additional CQL implementation details are given in Appendix A.

4.2 Collaborative Specialist Knowledge Distillation

The locally optimized policies provide the specialist knowledge used by CSKD. Rather than communicating complete actor–critic models, each client transfers policy behavior over states sampled from its local offline experience. Let $\mathcal{S}^t_i$ denote the states selected by client i at communication round t. Its specialist knowledge is represented as

$$\mathcal{K}^t_i = \left\{\left(s,\, \pi^t_i(\cdot \mid s)\right) \,\middle|\, s \in \mathcal{S}^t_i\right\}. \tag{10}$$

The pair $(s, \pi^t_i(\cdot \mid s))$ records the behavior produced by specialist i on a locally supported state.

Specialist knowledge integration. The server preserves the correspondence between each transferred state and the specialist behavior produced on that state, which we refer to as *state matching*. Knowledge from all clients is then composed into

$$\mathcal{T}^t = \bigcup_{i=1}^{N} \mathcal{K}^t_i = \left\{\left(s,\, \pi^t_i(\cdot \mid s)\right) \,\middle|\, s \in \mathcal{S}^t_i,\ i = 1, \ldots, N\right\}. \tag{11}$$

Composition operates on state–policy pairs rather than client parameters: each transferred state remains associated with the behavior produced by its source specialist. A single shared student π_θ is then optimized jointly over all specialist target sets,

$$\mathcal{L}_{\text{CSKD}} = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{|\mathcal{S}^t_i|} \sum_{s\in\mathcal{S}^t_i} \mathcal{D}_\pi\left(\pi^t_i(\cdot \mid s), \pi_\theta(\cdot \mid s)\right). \tag{12}$$

Eq. (12) is the finite-sample realization of the specialist integration objective in Eq. (6), with $\mathcal{S}^t_i$ providing samples from the transfer distribution ρ_i. The per-client normalization prevents the number of transferred states from implicitly changing a client's weight in the server objective. All specialists therefore supervise the same student while preserving the correspondence between transferred states and their source policies. After server optimization, we set $\pi^{t+1}_s \leftarrow \pi_\theta$. For continuous-control experiments, $\mathcal{D}_\pi$ is implemented as KL-based matching in the pre-squashing policy space; its implementation is detailed in Appendix A.

Communication. With K transferred states per client, state dimension d_s, and policy-output dimension d_π, CSKD transmits $C_{\text{CSKD}} = NK(d_s + d_\pi)$ scalars per round, compared with $N|\theta|$ parameters under full-model exchange. Detailed payload calculations are provided in Appendix C.

5 Experiments

Our experiments evaluate both the performance and the design of CSKD. We first compare the final shared policy with representative federated aggregation and policy-distillation baselines under heterogeneous offline experience. We then study repeated communication rounds, scalability with increasing client populations, and client-to-server communication cost. Finally, ablation and experience-overlap experiments examine the roles of persistent specialization, specialist knowledge integration, and policy-space distillation.

5.1 EXPERIMENTAL SETUP

Environments and datasets. We evaluate CSKD on three continuous-control tasks from D4RL (Fu et al., 2020): HALFCHEETAH, HOPPER, and WALKER2D, using the `medium-replay-v2` datasets. Each dataset is repartitioned across clients while preserving the same underlying task and environment. Clients optimize their policies using only local offline experience, and all methods ultimately produce a single shared policy. No additional environment interaction is used during federated training.

Baselines. We compare CSKD with **FedAvg** (McMahan et al., 2017), **FedProx** (Li et al., 2020), **DRPO** (Yue et al., 2024), **FEDORA** (Rengarajan et al., 2024), **FOVA** (Qiao et al., 2025), and **FORLER** (Qiao & Yue, 2026). We additionally include offline adaptations of the policy-distillation methods **FedHPD-style** (Jiang et al., 2025) and **SCCD-style** (Mai et al., 2023). All methods use the same data partitions, initialization, training budgets, and evaluation procedure, and results are reported as mean $\pm$ standard deviation over three seeds.

Implementation. CSKD uses CQL as the local offline RL backbone with a trainable client-specific residual policy head. Server-side specialist distillation follows Eq. (12), with $\mathcal{D}_\pi$ implemented as KL-based matching in the pre-squashing policy space. The *Residual Integration* ablation instead performs mean-squared-error matching directly in residual space.

Protocols. The main comparison uses four clients and 50 communication rounds from a common shared checkpoint. Each round contains $1{,}000$ local updates and $1{,}000$ server updates when applicable, with batch size 256. Policies are evaluated over 10 episodes without additional training-time environment interaction. A separate round-wise dynamics study is conducted on HOPPER and WALKER2D. Scalability experiments vary the number of clients, and communication is measured from the transmitted protocol payload.

5.2 MAIN RESULTS

Table 1 reports the final shared-policy performance. CSKD achieves the highest average normalized score of 78.91 among the compared federated methods, reaching 48.04 on HALFCHEETAH, 100.16 on HOPPER, and 88.54 on WALKER2D. Compared with the strongest baseline on each environment, CSKD improves the normalized score by 3.43 points on HALFCHEETAH, 11.04 points on HOPPER, and 6.59 points on WALKER2D. Its average score is 10.88 points higher than FedAvg, the strongest baseline in average performance.

The baselines cover several forms of federated coordination: FedAvg and FedProx aggregate models, DRPO and FEDORA use policy regularization or ensemble coordination, and FedHPD-style and SCCD-style transfer policy-level knowledge. CSKD differs from these approaches by preserving local specialization before explicitly transferring the resulting specialist behaviors to the shared policy. CSKD improves over the strongest compared baseline on all three evaluated environments. Section 5.5 further separates the effects of persistent specialization, specialist knowledge integration, and policy-space distillation.

5.3 MULTI-ROUND COLLABORATIVE LEARNING

We next examine the learning dynamics throughout federated training using a separate round-wise study on HOPPER and WALKER2D. Figure 2 shows a clear separation between CSKD and the two model-level federated baselines after the initial communication rounds. On HOPPER, CSKD improves rapidly and continues to increase throughout training, whereas FedAvg and FEDORA remain substantially lower. A similar pattern appears on WALKER2D: although all methods start from the same shared initialization, CSKD maintains stronger performance as local specialization and server-side distillation are repeatedly applied.

These trajectories complement the final-score comparison in Table 1. In particular, the benefit of CSKD is not restricted to a one-shot transfer from independently trained specialists. The shared policy is repeatedly redistributed to the clients, combined with their persistent local adaptations, and distilled again at the server. The sustained advantage over multiple rounds is therefore consis-

Table 1: D4RL normalized scores under the heterogeneous four-client setting (mean ± std over three seeds). Bold indicates the best federated result. FedHPD and SCCD are adapted to the offline setting.

Method	HalfCheetah	Hopper	Walker2d	Avg.
FedAvg	44.35 ± 0.40	77.78 ± 10.94	81.95 ± 5.66	68.03
FedProx	44.20 ± 0.25	63.69 ± 9.16	76.44 ± 9.14	61.44
DRPO	44.49 ± 0.14	73.23 ± 20.62	66.93 ± 12.33	61.55
FEDORA	44.24 ± 0.43	67.04 ± 18.11	66.92 ± 21.65	59.40
FOVA	41.87 ± 1.90	70.76 ± 17.57	66.70 ± 6.22	59.78
FORLER	44.61 ± 0.22	89.12 ± 2.35	59.71 ± 17.43	64.48
FedHPD-style	43.94 ± 0.58	69.06 ± 5.84	73.33 ± 7.71	62.11
SCCD-style	43.48 ± 0.96	76.46 ± 17.91	60.76 ± 31.75	60.23
CSKD	$\mathbf{48.04 \pm 0.33}$	$\mathbf{100.16 \pm 3.32}$	$\mathbf{88.54 \pm 5.31}$	**78.91**

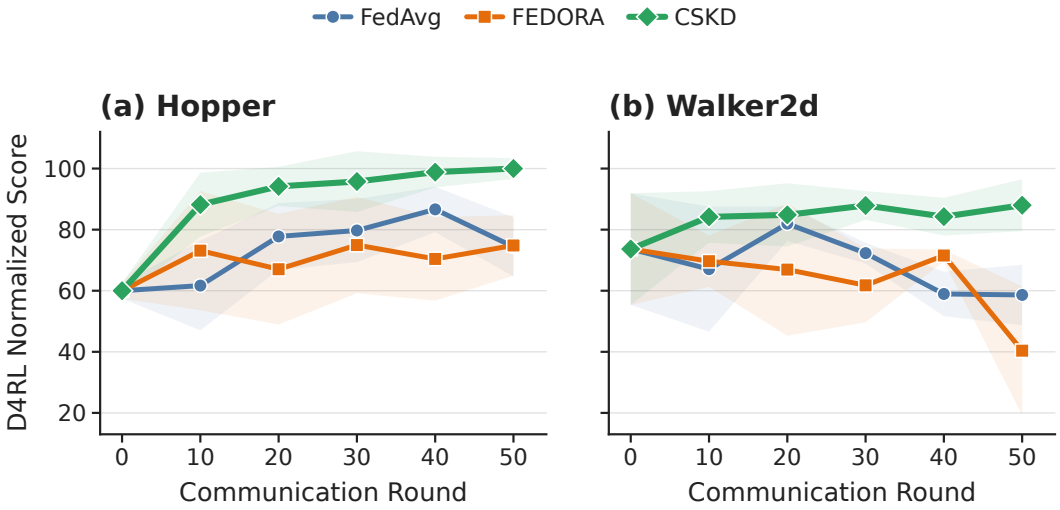

Figure 2: Round-wise dynamics of FedAvg, FEDORA, and CSKD on HOPPER and WALKER2D in the separate dynamics study. Curves show the mean D4RL normalized score over three seeds, with shaded regions indicating one standard deviation.

tent with the intended collaborative process: locally acquired specialization remains available while improvements absorbed by the shared policy are returned to all clients in subsequent rounds.

5.4 SCALABILITY AND COMMUNICATION

Figure 3 evaluates scalability and communication on HOPPER. CSKD obtains normalized scores of 83.0, 96.9, and 82.1 with 8, 16, and 32 clients, respectively. Performance therefore does not degrade immediately as more specialists are introduced and is highest at $N = 16$. The decline at $N = 32$, however, indicates that increasing the number of specialists eventually makes joint integration more difficult, since the shared policy must reconcile behavioral targets originating from a larger collection of local experience distributions.

Communication grows linearly with the number of participating clients for both full-model exchange and CSKD, but the transmitted object is substantially smaller in CSKD. Under the fixed $K = 5000$ transfer budget, CSKD communicates 2.24, 4.48, and 8.96 MB per round for 8, 16, and 32 clients, compared with 52.8, 105.5, and 211.0 MB for full-model exchange. This yields an approximately $23.6\times$ reduction across all three client populations. Thus, increasing the number of specialists enlarges the integration problem but does not require transmitting complete actor–critic models.

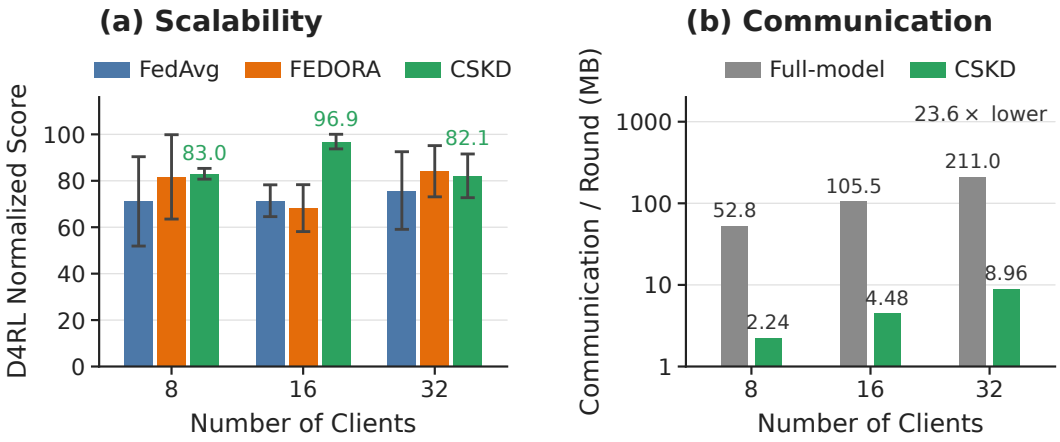

Figure 3: Scalability and communication analysis on HOPPER. **(a)** D4RL normalized score as the number of clients increases from 8 to 32. Error bars denote one standard deviation over three seeds. **(b)** Per-round client-to-server communication under the fixed $K = 5000$ specialist-knowledge transfer protocol. Full-model exchange corresponds to transmitting complete client models as in FedAvg and FEDORA.

Table 2: Ablation study under the heterogeneous protocol. Results are mean ± standard deviation over three seeds. P, K, and S denote persistent specialization, joint specialist knowledge integration, and policy-space integration.

	Components			D4RL Normalized Score			
Variant	P	K	S	HC	Hopper	Walker2d	Avg.
w/o Persistent	–	Y	Y	45.46 ± 0.23	94.83 ± 3.89	85.18 ± 6.42	75.16
Individual Distill.	Y	–	Y	45.32 ± 0.31	93.69 ± 7.85	83.91 ± 6.48	74.31
Residual Integration	Y	Y	–	46.17 ± 0.38	$\mathbf{100.72 \pm 1.34}$	84.42 ± 6.14	77.10
CSKD	Y	Y	Y	$\mathbf{48.04 \pm 0.33}$	100.16 ± 3.32	$\mathbf{88.54 \pm 5.31}$	**78.91**

5.5 ABLATION STUDY

We ablate three components of CSKD under the same heterogeneous protocol. *w/o Persistent* removes cross-round retention of client-specific adaptations. *Individual Distillation* removes joint specialist knowledge integration by distilling each client's target set separately rather than composing the multi-specialist target set $\mathcal{T}^t$. *Residual Integration* replaces policy-space integration with direct residual-space integration.

Table 2 shows that full CSKD achieves the highest average score of 78.91. Removing persistent specialization decreases the average by 3.75 points, while replacing joint integration with individual client-wise distillation yields a 4.60-point drop. Residual-space integration performs strongly on HOPPER, but its overall average remains 1.81 points below full CSKD. The results support persistent specialization, joint specialist integration, and policy-space distillation as complementary components of the final method.

5.6 EFFECT OF EXPERIENCE OVERLAP

We next vary client experience overlap while keeping the remaining evaluation protocol fixed. Figure 4 shows that the effect of overlap is non-monotonic: changing the amount of shared experience can substantially alter the final policy performance, but lower overlap does not uniformly lead to lower return. This behavior suggests that specialist collaboration depends on more than the amount of experience shared across clients.

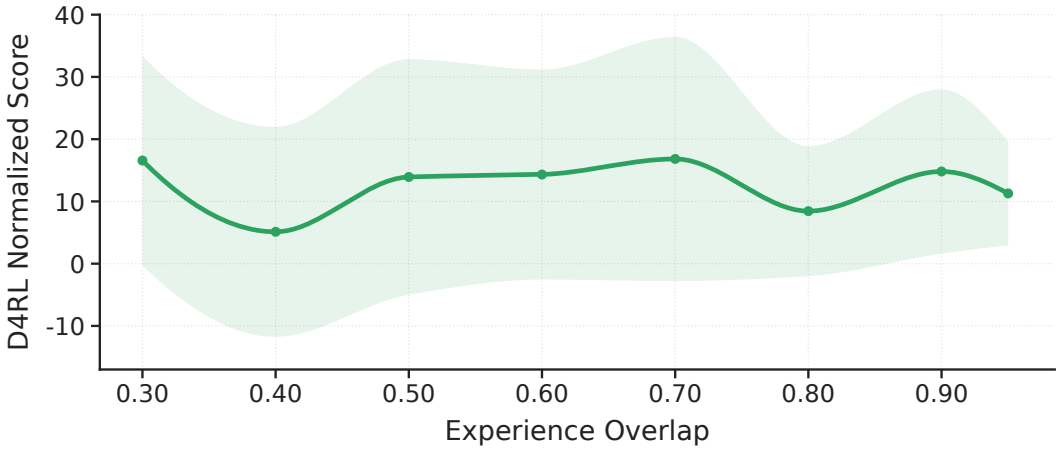

Figure 4: Effect of client experience overlap on CSKD under a fixed evaluation protocol. The curve reports the mean D4RL normalized score over three seeds, with error bars denoting one standard deviation. Performance varies non-monotonically with overlap, reflecting the interaction between specialist diversity and their integration into a single shared policy.

A useful interpretation is that overlap controls two competing properties of the local specialists. Greater separation can increase behavioral diversity because clients are optimized from more distinct experience, creating opportunities for complementary knowledge transfer. At the same time, strongly different local distributions can produce targets that are more difficult for a single shared policy to reconcile. Conversely, larger overlap provides a more common behavioral basis but may reduce the degree of specialization available for transfer. The variation in Figure 4 is therefore consistent with a trade-off between specialist diversity and integrability rather than a simple monotonic dependence on overlap. This also helps explain why increasing specialist diversity alone does not necessarily improve the shared policy.

6 Conclusion and Limitations

We introduced **Collaborative Specialist Knowledge Distillation (CSKD)** for federated offline reinforcement learning. CSKD preserves persistent local specialization and transfers distributed specialist behavior through composed state–policy targets and policy-space distillation. Across D4RL tasks, CSKD improves the final shared policy over the compared federated aggregation and distillation baselines, while additional experiments demonstrate repeated-round transfer, scalability to larger client populations, and substantially lower client-to-server payload than full-model exchange. The ablation results further show that both persistent specialization and joint specialist integration contribute to the final performance, while policy-space integration provides the strongest overall result across environments.

Our results also reveal limitations of the current formulation. Specialist integration becomes more challenging at larger client populations, while experience overlap has a non-monotonic effect on performance. The current bottleneck is therefore the capacity of a single shared policy to absorb increasingly diverse specialist targets. Improving knowledge integration under larger and more heterogeneous federations remains an important direction for future work.

AI USE STATEMENT

We used generative AI tools to assist with language polishing, literature retrieval and discovery, and research ideation and experimental planning. All AI-assisted outputs were manually reviewed, verified, and revised by the authors. We did not use generative AI to generate synthetic datasets or prove mathematical claims. The authors take full responsibility for the final content of this paper.

REPRODUCIBILITY STATEMENT

We provide the main learning objectives, federated training protocol, and evaluation procedure in the main paper. Additional implementation details for local CQL optimization and server-side policy distillation, communication-cost computation, ablation definitions, and experimental settings are provided in Appendices A–D. All reported comparisons use common data partitions, initialization, training budgets, and evaluation procedures across methods.

REFERENCES

Glen Berseth, Cheng Xie, Paul Cernek, and Michiel Van de Panne. Progressive reinforcement learning with distillation for multi-skilled motion control. *arXiv preprint arXiv:1802.04765*, 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.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Eunjeong Jeong, Seungeun Oh, Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *arXiv preprint arXiv:1811.11479*, 2018.

Wenzheng Jiang, Ji Wang, Xiongtao Zhang, Weidong Bao, Cheston Tan, and Flint Xiaofeng Fan. Fedhpd: Heterogeneous federated reinforcement learning via policy distillation. *arXiv preprint arXiv:2502.00870*, 2025.

Hao Jin, Yang Peng, Wenhao Yang, Shusen Wang, and Zhihua Zhang. Federated reinforcement learning with environment heterogeneity. In *International conference on artificial intelligence and statistics*, pp. 18–37. PMLR, 2022.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

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.

Sergey Levine, Aviral Kumar, George Tucker, Justin Fu, et al. Offline reinforcement learning: Tutorial, review. *and Perspectives on Open Problems*, 5, 2020.

Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. *Advances in neural information processing systems*, 33:2351–2363, 2020.

Andrei Lupu, Chris Lu, Jarek Liesen, Robert Lange, and Jakob Foerster. Behaviour distillation. In *International Conference on Learning Representations*, volume 2024, pp. 12668–12684, 2024.

Weiming Mai, Jiangchao Yao, Gong Chen, Ya Zhang, Yiu-Ming Cheung, and Bo Han. Server-client collaborative distillation for federated reinforcement learning. *ACM Transactions on Knowledge Discovery from Data*, 18(1):1–22, 2023.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. Pmlr, 2017.

Mitsuhiko Nakamoto, Simon Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. *Advances in Neural Information Processing Systems*, 36:62244–62269, 2023.

Jiaju Qi, Qihao Zhou, Lei Lei, and Kan Zheng. Federated reinforcement learning: Techniques, applications, and open challenges. *arXiv preprint arXiv:2108.11887*, 2021.

Nan Qiao and Sheng Yue. Forler: Federated offline reinforcement learning with q-ensemble and actor rectification. In *ICC 2026-IEEE International Conference on Communications*, pp. 1–6. IEEE, 2026.

Nan Qiao, Sheng Yue, Ju Ren, and Yaoxue Zhang. Fova: Offline federated reinforcement learning with mixed-quality data. *IEEE Transactions on Networking*, 34:2031–2046, 2025.

Desik Rengarajan, Nitin Ragothaman, Dileep Kalathil, and Srinivas Shakkottai. Federated ensemble-directed offline reinforcement learning. *Advances in Neural Information Processing Systems*, 37:6154–6179, 2024.

Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015.

Jiin Woo, Laixi Shi, Gauri Joshi, and Yuejie Chi. Federated offline reinforcement learning: Collaborative single-policy coverage suffices. *arXiv preprint arXiv:2402.05876*, 2024.

Sheng Yue, Zerui Qin, Xingyuan Hua, Yongheng Deng, and Ju Ren. Federated offline policy optimization with dual regularization. In *IEEE INFOCOM 2024-IEEE Conference on Computer Communications*, pp. 811–820. IEEE, 2024.

A Additional Implementation Details

A.1 Persistent Residual Specialist Optimization

The local specialist follows Eqs. (7)–(9). The persistent initialization follows Eq. (8).

We use two local critics $Q_{i,1}$ and $Q_{i,2}$. The critics are trained with the standard temporal-difference objective together with the conservative CQL regularizer, which penalizes high values assigned to actions outside the support of the offline dataset. The critics remain local throughout federated training and are never transmitted to the server.

A.2 Server-Side Policy Distillation

The server-side update follows Eqs. (11)–(12). Here we provide the implementation of the policy discrepancy $\mathcal{D}_\pi$ used in continuous-control experiments.

Given teacher and student action outputs $a_{\mathrm{T}}(s)$ and $a_{\mathrm{S}}(s)$, we map them to the pre-squashing space,

$$\tilde{\mu}_{\mathrm{T}}(s) = \operatorname{atanh}\left(a_{\mathrm{T}}(s)\right), \qquad \tilde{\mu}_{\mathrm{S}}(s) = \operatorname{atanh}\left(a_{\mathrm{S}}(s)\right). \tag{13}$$

Using the same detached scale $\sigma(s)$ from the base actor for teacher and student, the KL objective reduces to

$$\mathcal{D}_\pi = \frac{1}{2} \sum_d \frac{\left(\tilde{\mu}_{\mathrm{T},d}(s) - \tilde{\mu}_{\mathrm{S},d}(s)\right)^2}{\sigma_d^2(s)}. \tag{14}$$

Thus, server distillation performs variance-weighted matching in the pre-squashing policy space rather than plain action-space MSE.

B Experimental Protocol Details

B.1 Environments and Federated Training

We evaluate all methods on the `medium-replay-v2` variants of HalfCheetah, Hopper, and Walker2d from D4RL (Fu et al., 2020). Each federated experiment operates on a fixed offline dataset, and no additional environment interaction is allowed during training.

The main comparison uses four clients and 50 communication rounds. All methods start from the same shared checkpoint and use identical client partitions, training budgets, and evaluation procedures. Each communication round contains $1{,}000$ local updates and, when required by the method, $1{,}000$ server updates. The batch size is 256. Final policies are evaluated over 10 environment episodes, and all reported results use three random seeds.

The round-wise analysis on Hopper and Walker2d is conducted as a separate dynamics study. The scalability experiment varies the number of participating clients among $N \in \{8, 16, 32\}$ while keeping the remaining protocol fixed.

B.2 Heterogeneous Offline Experience

The original D4RL datasets are repartitioned across clients to create heterogeneous local offline experience while preserving the same underlying MDP. Each client is therefore trained on a different portion of the available experience, and the resulting local state distributions may only partially overlap. This construction changes the experience available to each client without introducing environment heterogeneity: all clients share the same state space, action space, transition dynamics, reward function, and task objective.

The overlap experiment further varies the amount of experience shared across clients while leaving the remaining evaluation procedure unchanged. We use this controlled variation to examine how the balance between shared experience and local specialization affects the final shared policy.

C Additional Analysis

C.1 Communication Payload

CSKD communicates specialist state–policy pairs rather than complete local actor–critic models. With N clients, K transferred states per client, state dimension d_s, and transmitted policy-output dimension d_π, the number of transmitted scalars per communication round is

$$C_{\mathrm{CSKD}} = NK\left(d_s + d_\pi\right). \tag{15}$$

Assuming 32-bit floating-point transmission, the corresponding client-to-server payload is

$$M_{\mathrm{CSKD}} = 4NK\left(d_s + d_\pi\right) \text{ bytes}. \tag{16}$$

In the scalability experiment, we fix $K = 5000$. On Hopper, this gives client-to-server communication of 2.24, 4.48, and 8.96 MB per communication round for 8, 16, and 32 clients, respectively. Full-model transmission requires 52.8, 105.5, and 211.0 MB under the same client populations. The resulting ratio remains approximately $23.6\times$ across the three settings because both protocols scale linearly with the number of participating clients.

We report only client-to-server communication associated with federated knowledge transfer. The common server-to-client broadcast of the shared policy is excluded from this comparison.

C.2 Ablation Variants

The ablations in Table 2 isolate three components of CSKD.

w/o Persistent. The client-specific adaptation is not retained across communication rounds. Each client therefore performs local specialization from the newly received shared basis without carrying its previous residual specialization forward.

Individual Distillation. The server does not construct the joint multi-specialist target set $\mathcal{T}^t$. Instead, each client's specialist targets are distilled separately. This variant preserves policy-level knowledge transfer but removes joint specialist knowledge integration.

Residual Integration. The server retains joint specialist integration but replaces policy-space distillation with direct integration in residual space. The server-side objective is therefore based on residual-space mean-squared error rather than the KL-based policy discrepancy used by full CSKD.

These variants correspond to the P, K, and S components reported in Table 2: persistent specialization, joint specialist knowledge integration, and policy-space integration, respectively.

D Hyperparameter and Reproducibility Details

Table 3 summarizes the principal settings used in the main experiments. Parameters not specific to CSKD follow the corresponding local CQL implementation and are kept fixed across compared methods whenever applicable.

Table 3: Main experimental settings used in CSKD.

Setting	Value
Offline RL backbone	CQL
Main number of clients	4
Communication rounds	50
Local updates / round	1,000
Server updates / round	1,000
Batch size	256
Evaluation episodes	10
Number of seeds	3
Dataset	D4RL `medium-replay-v2`
Main environments	HalfCheetah, Hopper, Walker2d
Specialist transfer budget K	5000 (communication study)
Server distillation	Policy-space KL
Residual Integration loss	Residual-space MSE
Scalability clients	$8, 16, 32$

For all main comparisons, the same dataset partition, shared initialization, optimization budget, and evaluation procedure are used across methods. Reported scores are the mean and standard deviation over three seeds. The main results and the full CSKD row in the ablation study use the same four-client evaluation protocol.