Capacity-Aware Federated Offline Reinforcement Learning for Resource-Heterogeneous Edge Clients

Anonymous Authors[1]

Abstract

Federated offline reinforcement learning (FORL) is a promising abstraction for distributed edge decision making, particularly in resource-heterogeneous wireless environments with decentralized operational logs. Existing FORL methods, however, typically assume that every client can train the same full policy model, which is unrealistic when memory, compute, and energy budgets vary substantially across clients. We study FORL under client capacity heterogeneity, where high-capacity clients train a full policy model while low-capacity clients can train only masked constrained models. We propose a capacity-aware framework that combines a high-capacity warm start, aggregation-compatible masking, masked parameter aggregation, and server-side progressive distillation to transfer knowledge from an evolving full model to a constrained model that remains trainable by low-capacity clients. To isolate this algorithmic question, we evaluate Decision Transformer on standard D4RL locomotion tasks rather than domain-specific wireless simulators. Results provide preliminary evidence that low-capacity clients can be incorporated without collapsing full-model performance, while improving constrained-model performance over uniformly low-capacity federated training.

1. Introduction

Resource-heterogeneous edge systems provide a natural motivation for federated offline reinforcement learning (FORL). In particular, next-generation wireless environments generate decentralized operational logs across base stations, access points, user devices, spectrum sensors, and edge controllers. In such settings, offline reinforcement learning (offline RL) is appealing because policies can be learned from historical logs without risky online exploration, while federated learning allows multiple clients to collaborate without exposing raw local data (McMahan et al., 2017). This makes FORL a promising framework for privacy-preserving decision making in distributed edge environments (Zhou et al., 2024; Rengarajan et al., 2024; Woo et al., 2024).

A major obstacle is *client capacity heterogeneity*. Some clients can train large policy models, whereas many low-end devices are constrained by memory, compute, or energy. Existing FORL methods largely assume homogeneous model capacity. This creates a participation–performance trade-off: excluding low-capacity clients wastes their private offline logs, while forcing all clients to use the same small model sacrifices the performance available to stronger clients.

This trade-off is different from ordinary model compression. In a centralized setting, one may first train a large model and then compress it for deployment. In FORL with heterogeneous clients, however, low-capacity devices are not merely final deployment targets; they are also data holders that must participate during training if their local logs are to influence the learned policy. A useful capacity-aware FORL method should therefore satisfy two requirements simultaneously: it should preserve a strong full model for high-capacity clients, and it should provide an aggregation-compatible constrained model through which low-capacity clients can continue to train locally.

Model-heterogeneous federated learning addresses related issues in supervised learning by training sub-models or masks of a global model (Diao et al., 2021; Alam et al., 2022; Li et al., 2021). Directly applying these ideas to offline RL is nontrivial because offline RL is sensitive to extrapolation error and unstable policy or value updates (Kumar et al., 2020). Knowledge distillation can reduce the gap between large and small policies (Rusu et al., 2016; Schmitt et al., 2018), but prior distillation-based RL does not address iterative federated aggregation with heterogeneous client capacities.

We propose a capacity-aware FORL framework via progressive distillation. The server first obtains a full-model warm start from high-capacity clients, extracts masked assistant

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

teachers from the evolving full model, and progressively distills a constrained model using a small server-side auxiliary dataset. During subsequent rounds, high-capacity clients update the full model, low-capacity clients update the distilled constrained model, and the server combines their updates through masked aggregation.

This workshop paper focuses on the core algorithmic feasibility question: *can low-capacity clients participate in federated offline reinforcement learning without reducing all clients to the smallest model?* While this setting is motivated by resource-heterogeneous wireless edge deployments, our experiments use standard D4RL locomotion benchmarks to isolate the capacity-heterogeneity problem itself rather than provide wireless-specific system validation. Our contributions are: (i) a FORL formulation with heterogeneous trainable parameter budgets, (ii) a unified training loop combining masking, masked aggregation, and progressive distillation, and (iii) a compact empirical study with Decision Transformer on D4RL locomotion tasks.

2. Problem Setting

We consider K clients and a central server. Client k owns a private offline dataset $\mathcal{D}_k = \{(s_t, a_t, r_t, s_{t+1})\}$ collected by an unknown behavior policy. Raw transitions are never shared. The goal is to learn policies that maximize expected return over the union of client data $\mathcal{D} = \cup_k \mathcal{D}_k$ through parameter exchange only.

Clients are partitioned into high- and low-capacity sets, $\mathcal{K}_{\mathrm{H}}$ and $\mathcal{K}_{\mathrm{L}}$. High-capacity clients train the full model $f(\cdot;\theta)$ with parameter vector $\theta \in \mathbb{R}^{|\theta|}$. Low-capacity clients train a constrained sub-model defined by a binary mask $m \in \{0,1\}^{|\theta|}$:

$$\theta^{\mathrm{con}} = \theta \odot m. \tag{1}$$

They update and transmit only the coordinates for which $m_i = 1$.

The server also maintains a small auxiliary dataset $\mathcal{D}_{\mathrm{aux}}$ used only for server-side distillation. It is disjoint from all private client datasets, remains on the server, and is not used in client-side offline RL updates. This proxy-data assumption is common in distillation-based federated learning for cross-model knowledge transfer (Lin et al., 2020; Chen & Chao, 2021; Xie et al., 2024).

Capacity-aware objective. The objective is not to produce a single model that every client must use. Instead, the system maintains two compatible deployment models: a full model for high-capacity clients and a constrained model for low-capacity clients. The constrained model should be small enough to satisfy the low-capacity parameter budget, but it should remain aligned with the full model so that its local updates can be integrated into the global training process.

Table 1. Operational boundary of the server-side auxiliary dataset.

Constraint	Use in this work
Server-only	never transmitted to clients
Disjoint	$\mathcal{D}_{\mathrm{aux}} \cap \mathcal{D}_k = \emptyset$ for all k
Small scale	about one client split, i.e., $\approx 10\%$ of data
Distillation-only	used only in $\mathcal{L}_{\mathrm{distill}}$
No aggregation	no gradient or parameter averaging contribution

This alignment requirement motivates the use of masks over a shared parameter vector rather than independent small architectures.

Auxiliary-data boundary. The auxiliary set is not a surrogate centralized training set. It serves as a small server-side support set on which the server can query teacher and student policies under common inputs. Client-side offline RL losses, local updates, and masked aggregation are computed only from the private client datasets $\{\mathcal{D}_k\}_{k=1}^{K}$. Thus, $\mathcal{D}_{\mathrm{aux}}$ affects low-capacity clients only through the distilled initialization broadcast by the server. These operational constraints are summarized in Table 1. In experiments, $\mathcal{D}_{\mathrm{aux}}$ is reserved only for methods that instantiate server-side distillation; for baselines without distillation, the corresponding trajectories remain in the client partitions. Accordingly, $\mathcal{D}_{\mathrm{aux}}$ should not be interpreted as extra benchmark data available only to our method.

3. Method

The framework proceeds in repeated heterogeneous training cycles. Figure 1 summarizes the overall workflow, and Algorithm 1 gives the corresponding round-level procedure. A high-capacity warm start first constructs a full-model teacher. The server then extracts masked sub-models from the full model, uses them as assistant teachers for progressive distillation, and broadcasts a distilled constrained model to low-capacity clients. Subsequent rounds combine full-model updates and constrained-model updates through masked aggregation.

High-capacity warm start. Round 1 uses only high-capacity clients to construct a stable full-model teacher. Each $k \in \mathcal{K}_{\mathrm{H}}$ runs local offline RL training from $\theta^{(0)}$ on $\mathcal{D}_k$, producing $\theta_k^{(1)}$. The server computes

$$\theta^{(1)} = \sum_{k \in \mathcal{K}_{\mathrm{H}}} \frac{n_k}{\sum_{j \in \mathcal{K}_{\mathrm{H}}} n_j} \theta_k^{(1)}, \tag{2}$$

where n_k is the local data size. The resulting full model initializes progressive distillation for low-capacity clients. This warm-start stage avoids asking constrained clients to begin heterogeneous training from a weak randomly initialized sub-model.

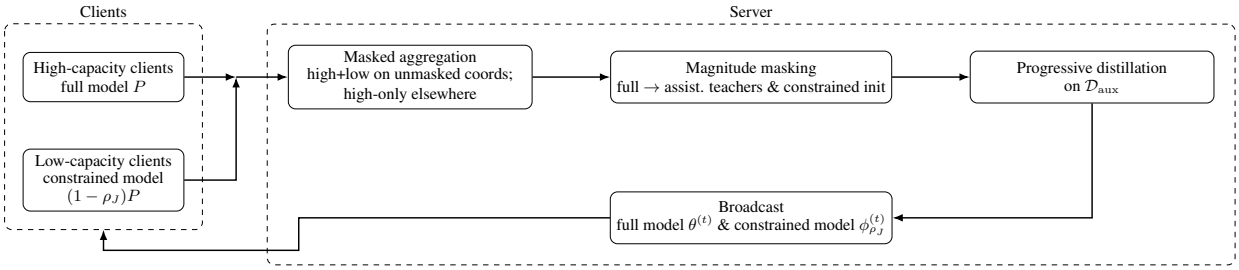

Figure 1. Conceptual view of capacity-aware federated offline RL. High-capacity clients train the full model, whereas low-capacity clients train only a constrained masked model. The server aggregates compatible coordinates, extracts masked assistant teachers, performs progressive distillation on a small auxiliary dataset, and broadcasts capacity-appropriate models for the next round.

Masked sub-model extraction. At communication round t, the server constructs magnitude masks at ordered sparsity ratios $\rho_1 < \cdots < \rho_J$. A mask $m_\rho^{(t)}$ keeps the largest $(1-\rho)$ fraction of parameters in $|\theta^{(t)}|$:

$$m_\rho^{(t)}[i] = \mathbf{1}\{|\theta_i^{(t)}| \geq \tau_\rho^{(t)}\}. \tag{3}$$

The masked model $\bar{\theta}_\rho^{(t)} = \theta^{(t)} \odot m_\rho^{(t)}$ is used either as an assistant teacher or as the initialization of the deployed constrained model. In the experiments, the deployed constrained model uses $\rho_J = 0.75$, retaining 25% of the full-model parameters.

The mask also defines which coordinates low-capacity clients are allowed to update and transmit. This is important for aggregation compatibility: a low-capacity client does not maintain an independent architecture, but rather a masked parameter vector embedded in the same coordinate system as the full model. As a result, its local updates can be merged with high-capacity updates on the unmasked coordinates.

Progressive distillation. The server initializes the low-capacity model from $\bar{\theta}_{\rho_J}^{(t)}$ and refines the same student through increasingly stronger teachers: $\bar{\theta}_{\rho_{J-1}}^{(t)}, \ldots, \bar{\theta}_{\rho_1}^{(t)}$, followed by the full model $\theta^{(t)}$. For a teacher T and student S, the Decision Transformer distillation objective is

$$\begin{aligned}\mathcal{L}_{\text{distill}}(S;T) =& \lambda_{\text{KD}}\, \mathbb{E}_{(s,\cdot)\sim\mathcal{D}_{\text{aux}}} \|\pi_T(s) - \pi_S(s)\|_2^2 \\ &+ (1-\lambda_{\text{KD}}) \mathbb{E}_{(s,a)\sim\mathcal{D}_{\text{aux}}} \|a - \pi_S(s)\|_2^2.\end{aligned} \tag{4}$$

The final distilled constrained model is denoted $\phi_{\rho_J}^{(t)}$ and is broadcast to low-capacity clients.

Progressive distillation is used as a server-side bridge between the full model and the constrained model. The assistant teachers are not deployed to clients; they only provide intermediate targets during server-side transfer. This keeps the client interface simple: high-capacity clients receive the full model, while low-capacity clients receive only the final constrained model.

Joint heterogeneous training. For $t \geq 1$, high-capacity clients receive $\theta^{(t)}$ and low-capacity clients receive $\phi_{\rho_J}^{(t)}$. After local offline RL updates, the server aggregates high-capacity models $\{\theta_k^{(t+1)}\}$ and constrained models $\{\phi_k^{(t+1)}\}$ by

$$\begin{aligned}\theta^{(t+1)} = m_{\rho_J}^{(t)} \odot \Big(\sum_{k\in\mathcal{K}_{\text{H}}} w_k \theta_k^{(t+1)} + \sum_{k\in\mathcal{K}_{\text{L}}} w_k \phi_k^{(t+1)} \Big) \\ + (1 - m_{\rho_J}^{(t)}) \odot \Big(\sum_{k\in\mathcal{K}_{\text{H}}} \tilde{w}_k \theta_k^{(t+1)} \Big),\end{aligned} \tag{5}$$

where $w_k = n_k / \sum_{j\in\mathcal{K}_{\text{H}}\cup\mathcal{K}_{\text{L}}} n_j$ and $\tilde{w}_k = n_k / \sum_{j\in\mathcal{K}_{\text{H}}} n_j$. Thus, low-capacity clients contribute only on trainable coordinates, preserving aggregation compatibility with the full model.

The two terms in Equation (5) separate the parameter space into unmasked and masked coordinates. On unmasked coordinates, both high- and low-capacity clients contribute because both groups have updated those parameters. On masked coordinates, only high-capacity clients contribute because low-capacity clients never store or update those entries. This coordinate-wise rule prevents missing parameters from being treated as zero-valued low-capacity updates.

Communication. We count the dominant transmitted parameter payload and exclude implementation-dependent overhead such as optimizer states, protocol headers, and optional sparse-index transmission. Let P be the number of full-model parameters. A high-capacity client uploads and downloads P parameters per round, whereas a low-capacity client communicates only the unmasked coordinates of the deployed constrained model. For a general deployed sparsity ρ_J, the one-way payload at rounds $t \geq 2$ is

$$C_{\text{1way}} = (H + (1-\rho_J)L)P, \tag{6}$$

and the round-trip communication is

$$C_{\text{rt}} = 2(H + (1-\rho_J)L)P. \tag{7}$$

With $\rho_J = 0.75$, this reduces to $C_{\text{rt}} = 2(H+0.25L)P$. The homogeneous full-model reference communicates $2KP =$

Table 2. Per-round communication for $K = 10$ and $\rho_J = 0.75$. Savings are measured relative to Full Fed, which requires $20P$ round-trip parameters.

Setting	1-way	Ours RT	High-Only RT	Saving
8H–2L	$8.5P$	$17P$	$16P$	15%
6H–4L	$7.0P$	$14P$	$12P$	30%
4H–6L	$5.5P$	$11P$	$8P$	45%
2H–8L	$4.0P$	$8P$	$4P$	60%
All-Low Fed	$2.5P$	$5P$	—	75%

Algorithm 1 Capacity-Aware FORL via Progressive Distillation

Input: initial full model $\theta^{(0)}$, client datasets $\{\mathcal{D}_k\}$, auxiliary data $\mathcal{D}_{\text{aux}}$, sparsities $\rho_1 < \cdots < \rho_J$
Warm start: train high-capacity clients from $\theta^{(0)}$ and aggregate them to obtain $\theta^{(1)}$
Construct masks from $\theta^{(1)}$ and progressively distill $\phi_{\rho_J}^{(1)}$
for $t = 1, \ldots, T-1$ **do**
  Broadcast $\theta^{(t)}$ to high-capacity clients and $\phi_{\rho_J}^{(t)}$ to low-capacity clients
  Clients perform local offline RL on their private datasets
  Server applies masked aggregation in Equation (5)
  Server extracts masked assistant teachers from $\theta^{(t+1)}$
  Server distills $\phi_{\rho_J}^{(t+1)}$ on $\mathcal{D}_{\text{aux}}$
end for
Output: full model $\theta^{(T)}$ and constrained model $\phi_{\rho_J}^{(T)}$

$20P$ for $K = 10$, while High-Only Fed communicates $2HP$ but discards the L low-capacity clients. Thus, the proposed method adds only $2(1-\rho_J)LP = 0.5LP$ round-trip payload over High-Only Fed to incorporate low-capacity client logs. Progressive distillation is server-side and adds computation but no client communication.

Table 2 should be read as a participation-communication trade-off rather than a pure compression table. High-Only Fed has lower communication than our method because it ignores low-capacity clients. All-Low Fed is even cheaper because every client communicates only the constrained model. The proposed method occupies the intermediate regime: it preserves a full model for high-capacity clients while adding only masked low-capacity payloads to include otherwise unused local logs.

4. Experiments

Setup. We evaluate Decision Transformer (Chen et al., 2021) on D4RL MuJoCo expert datasets: HalfCheetah, Hopper, and Walker2d (Fu et al., 2020). We use $K = 10$ clients with four capacity compositions: 8H-2L, 6H-4L, 4H-6L, and 2H-8L. Data partitioning depends on whether a method uses server-side distillation. For the proposed method, we first reserve a disjoint server auxiliary set $\mathcal{D}_{\text{aux}}$ of about one client split, i.e., approximately 10% of the total trajectories, and evenly partition the remaining trajectories across the K clients. For baselines without distillation, no auxiliary split is reserved; instead, the full dataset is evenly partitioned across clients. Thus, $\mathcal{D}_{\text{aux}}$ should not be interpreted as extra data available only to our method; rather, it is a method-specific server split used only when server-side distillation is instantiated, while baselines without distillation keep those trajectories in their client partitions. We fix $\rho_J = 0.75$ and $\lambda_{\text{KD}} = 0.2$, and report normalized D4RL scores averaged over 10 random seeds.

Baselines. We compare against three reference settings. *Centralized Full* trains a full model on all benchmark trajectories and is used only as an upper-bound reference. *High-Only Fed* excludes low-capacity clients and trains only on the H high-capacity clients. *All-Low Fed* forces all clients to use the constrained model. We also report Full Fed in the communication analysis as a homogeneous full-model communication reference.

For *High-Only Fed* and *All-Low Fed*, no auxiliary split is reserved: the trajectories that form $\mathcal{D}_{\text{aux}}$ in our method remain in the client partitions because these baselines do not use server-side distillation. Accordingly, *All-Low Fed* serves as the matched-data no-distillation baseline, whereas *High-Only Fed* intentionally uses fewer client logs because it discards low-capacity clients by design.

These baselines correspond to the main deployment alternatives under capacity heterogeneity. *High-Only Fed* prioritizes full-model quality but discards low-capacity logs. *All-Low Fed* maximizes client participation but removes the full model from the training process. *Centralized Full* is not a privacy-preserving federated baseline; it is included only to indicate the approximate performance level of a full model with the largest data access.

D4RL locomotion. Figure 2 shows results on HalfCheetah, Hopper, and Walker2d. Across client ratios and environments, the proposed method improves the constrained model relative to All-Low Fed. The improvement is especially clear in Hopper and Walker2d, where uniformly constrained federated training is unstable. Full-model performance remains close to High-Only Fed, indicating that incorporating low-capacity clients does not significantly harm high-capacity clients. As the proportion of low-capacity clients increases, the proposed method remains stable, whereas All-Low Fed degrades more severely.

Full versus constrained performance. Because the framework maintains two deployable models, the evaluation has two complementary goals. The full model should remain competitive with High-Only Fed, since high-capacity clients should not lose their strong solution merely because lower-capacity clients are admitted. The constrained

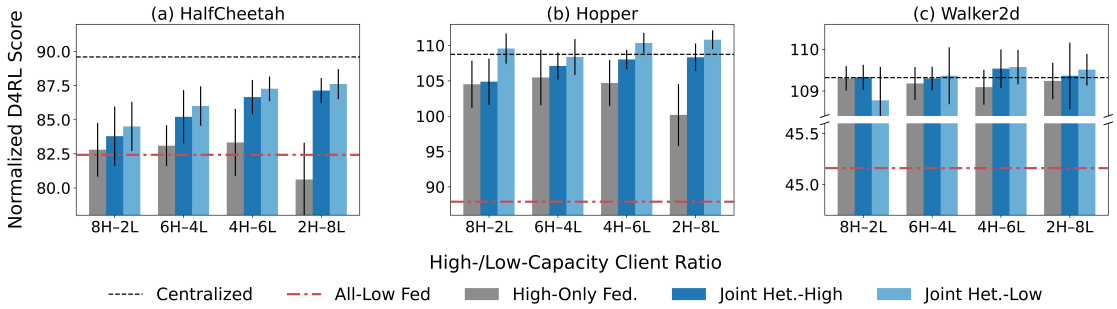

Figure 2. Decision Transformer performance on D4RL expert locomotion tasks under heterogeneous client capacities. Bars show mean ± standard deviation over 10 random seeds. The Walker2d panel uses a broken y-axis because All-Low Fed is substantially lower than the other methods. The proposed method improves the low-capacity model over All-Low Fed while keeping the high-capacity model competitive with High-Only Fed.

model should improve over All-Low Fed, since low-capacity clients should receive a stronger initialization and continue contributing their local logs. The D4RL results satisfy both criteria: the full model stays near the high-capacity-only reference, while the constrained model is substantially stronger than uniformly constrained training.

Communication–performance interpretation. The communication analysis should be interpreted together with the performance results. Relative to Full Fed, the proposed method reduces round-trip parameter traffic by 15–60% depending on the client composition. Relative to High-Only Fed, it adds only the masked payloads from low-capacity clients. The empirical question is therefore whether this additional low-capacity payload recovers useful information that High-Only Fed discards. The locomotion results suggest that it does: low-capacity clients can be incorporated while preserving a competitive full model and improving the constrained model over the all-low-capacity alternative.

What this experiment does not claim. The experiment is designed as a compact feasibility study rather than a complete empirical characterization. It tests whether the proposed training loop can occupy the middle ground between two extremes: excluding low-capacity clients and reducing every client to the constrained model. It does not claim that the selected sparsity ratio, mask type, auxiliary-data size, or distillation schedule is optimal. Those choices define one concrete instantiation of the framework.

Scope of evaluation. This workshop paper focuses on the core feasibility question: whether capacity-constrained clients can be incorporated into federated offline RL without collapsing the full-model solution. We therefore evaluate the full framework on Decision Transformer locomotion tasks and compare against the two deployment extremes: excluding low-capacity clients and forcing all clients to use the constrained model. We do not attempt to isolate every component of the framework in this short version; component-level ablations of the distillation schedule, auxiliary-data usage, and masking design are left for broader validation.

5. Discussion and Limitations

The results suggest that capacity heterogeneity is not merely a systems constraint: in FORL, it determines which clients can participate and which policy coordinates are updated during aggregation. The masking design keeps parameter indices aligned, while progressive distillation reduces the optimization gap faced by the constrained model before low-capacity deployment. This combination lets low-capacity clients contribute local logs without requiring every client to train the smallest model.

The framework should also be viewed as capacity-aware federated training, not post-hoc compression. A post-hoc compressed policy may be deployable on a low-capacity device, but it does not allow that device to contribute local offline RL updates during federated training. In contrast, the constrained model in our framework is repeatedly broadcast, locally updated, and re-aggregated on compatible coordinates. This distinction is central to the participation goal of the paper.

The main limitation is the server-side auxiliary dataset. We make this assumption explicit: $\mathcal{D}_{\text{aux}}$ is small, disjoint, server-only, and never used for client updates or aggregation, but it still provides support points for teacher–student transfer. Its role is weaker than centralized offline RL training but stronger than a purely data-free compression assumption. Reducing this dependence through synthetic proxy states, client-side sketches, or auxiliary-data-free alignment is an important next step.

This short study also uses a fixed sparsity ratio and magnitude masks. Adaptive capacity assignment, structured masks, or learned masks may further improve the performance–communication trade-off. Finally, we evaluate

only Decision Transformer locomotion tasks in the main body to isolate the core capacity-heterogeneity mechanism. Broader sparse-reward settings, data-quality heterogeneity, actor-critic backbones, and component-level ablations are important directions for future work.

6. Conclusion

We studied federated offline reinforcement learning under heterogeneous client capacities. By combining high-capacity warm start, masked aggregation, and server-side progressive distillation, the proposed framework lets low-capacity clients participate without reducing all clients to the smallest model. Decision Transformer experiments on D4RL locomotion tasks provide preliminary evidence that constrained-model performance can be improved while keeping the full model competitive with high-capacity-only federation.

Impact Statement

This work aims to improve privacy-preserving reinforcement learning for heterogeneous edge devices. Potential risks include deployment of learned policies in safety-critical environments; such use should require domain-specific safety validation beyond offline benchmark evaluation.

References

Alam, S., Liu, L., Yan, M., and Zhang, M. FedRolex: Model-heterogeneous federated learning with rolling sub-model extraction. In *Advances in Neural Information Processing Systems*, volume 35, pp. 29677–29690, 2022.

Chen, H.-Y. and Chao, W.-L. FedBE: Making bayesian model ensemble applicable to federated learning. In *International Conference on Learning Representations*, 2021.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision Transformer: Reinforcement learning via sequence modeling. In *Advances in Neural Information Processing Systems*, volume 34, pp. 15084–15097, 2021.

Diao, E., Ding, J., and Tarokh, V. HeteroFL: Computation and communication efficient federated learning for heterogeneous clients. In *International Conference on Learning Representations*, 2021.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4RL: Datasets for deep data-driven reinforcement learning, 2020.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1179–1191, 2020.

Li, A., Sun, J., Zeng, X., Zhang, M., Li, H., and Chen, Y. FedMask: Joint computation and communication-efficient personalized federated learning via heterogeneous masking. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*, pp. 42–55, 2021. doi: 10.1145/3485730.3485929.

Lin, T., Kong, L., Stich, S. U., and Jaggi, M. Ensemble distillation for robust model fusion in federated learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 2351–2363, 2020.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and Agüera y Arcas, B. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 2017.

Rengarajan, D., Ragothaman, N., Kalathil, D., and Shakkottai, S. Federated ensemble-directed offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 37, pp. 6154–6179, 2024.

Rusu, A. A., Colmenarejo, S. G., Gulcehre, C., Desjardins, G., Kirkpatrick, J., Pascanu, R., Mnih, V., Kavukcuoglu, K., and Hadsell, R. Policy distillation. In *International Conference on Learning Representations*, 2016.

Schmitt, S., Hudson, J. J., Zidek, A., Osindero, S., Doersch, C., Czarnecki, W. M., Leibo, J. Z., Kuttler, H., Zisserman, A., Simonyan, K., and Eslami, S. M. A. Kickstarting deep reinforcement learning, 2018.

Woo, J., Shi, L., Joshi, G., and Chi, Y. Federated offline reinforcement learning: Collaborative single-policy coverage suffices. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53165–53201. PMLR, 2024.

Xie, C., Huang, D.-A., Chu, W., Xu, D., Xiao, C., Li, B., and Anandkumar, A. PerAda: Parameter-efficient federated learning personalization with generalization guarantees. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23838–23848, 2024.

Zhou, D., Zhang, Y., Sonabend-W, A., Wang, Z., Lu, J., and Cai, T. Federated offline reinforcement learning. *Journal of the American Statistical Association*, 119(548):3152–3163, 2024. doi: 10.1080/01621459.2024.2310287.