CARVE: REPLAY-VERIFIED GRADIENT ADMISSION FOR OFFLINE AGENT CONTROLLERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Offline agent training often converts every logged operator decision into actor supervision even though replayability and behavior support vary across a trace. We introduce CARVE, Counterfactual Admission with Replay-Verified Evidence, a gradient-admission layer that tests joint operator–argument actions before objective construction and authorizes one of three paths: replayable supported decisions update actor and critic through counterfactual attribution, supported non-replayable decisions update only the critic, and unsupported decisions abstain. This tri-state contract retains supported value-learning signal while protecting actor gradients from replay-invalid and unsupported targets. With Llama-3.1-70B-Instruct and matched data, tools, objective, seeds, and token caps, CARVE raises Code/SWE success from 60.9 ± 1.2 to 63.8 ± 1.1, a $+2.9 \pm 0.5$-point paired gain with $5/5$ positive seeds and task-clustered 95% CI $[+1.5, +4.2]$; it also exceeds full-action BC+AWR by $+1.4 \pm 0.4$ points with $5/5$ positive seeds. An equal-footing admission-by-adapter matrix isolates routing: replay-and-support admission adds $+1.3$ points with adaptation disabled and $+1.5$ with it enabled, while the complete configuration exceeds the strongest adapter-matched simple rule by $+0.8 \pm 0.4$ points. Region interventions select attribution in the replayable-supported slice and critic-only learning in the non-replayable-supported slice. Task-clustered gains remain positive across software, SQL, planning, and web workloads, and the same contract improves a distinct planner–executor controller by $+2.5$ points on Code/SWE and $+2.2$ on held-out Web. CARVE makes trace reliability an explicit, portable decision about gradient eligibility.

1 INTRODUCTION

Tool-augmented agents accumulate execution traces at a scale that makes offline learning attractive, but those traces mix reproducible branches with tool-dependent transitions and well-supported actions with rare operator–argument combinations. Treating each logged choice as actor supervision lets unreliable counterfactual credit propagate through many policy updates. Standard offline objectives weight or regularize a transition after it enters the training set; the earlier and consequential decision is which parameter blocks that transition may update. Reliable learning from heterogeneous traces therefore requires a control layer at the boundary between logging and gradient construction.

CARVE supplies this layer through gradient admission. For action $u_t = (o_t, a_t)$, it evaluates local replayability R_t, factorized behavior support S_t, and assigns $z_t \in \{\text{attribution}, \text{critic-only}, \text{abstain}\}$. Attribution combines cached local replay with a bounded support-weighted correction and updates actor and critic. When replay fails but support remains, critic-only admission preserves value learning while withholding an unreliable policy target. Unsupported operator or argument choices receive no gradient. Figure 1 exposes this parameter-level contract, and Table 6 enumerates every gradient path.

Gradient admission changes the route of 13.0% of logged states, including 5.9% of policy updates redirected to critic-only learning (Table 7). Under equal-footing controls, CARVE contributes $+1.3$ points with the adapter off and $+1.5$ with it on. The complete configuration improves IQL by $+2.9 \pm 0.5$ Code/SWE success points under identical budgets and exceeds full-action BC+AWR by $+1.4$ points. Region interventions identify different winning updates in the replayable-supported and non-replayable-supported slices, connecting the gain to tri-state authorization. Positive task-clustered

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.

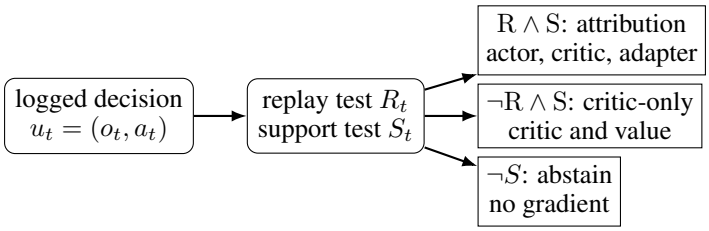

Figure 1: CARVE authorizes gradients before optimization. Replay and support jointly determine whether a logged operator decision updates actor and critic, only the critic, or no parameters.

gains persist across four workloads and transfer to a planner–executor state machine with a distinct transition graph and four of five action labels absent from the memory controller.

Our contributions are:

- We formulate offline learning from logged operator traces as parameter-level gradient authorization and develop CARVE, a tri-state contract with explicit actor, critic, and adapter semantics for every replay/support region.
- We isolate the mechanism through a matched admission-by-adapter matrix and region interventions. CARVE leads with adaptation either disabled or enabled and exceeds the strongest adapter-matched uncertainty, replay-only, or support-only rule by $+0.8$ points.
- We demonstrate practical consequence under matched compute and budgets: $+2.9 \pm 0.5$ points over IQL on Code/SWE, positive task-clustered gains across four workloads, and $+2.2$ to $+2.5$ gains on a distinct planner–executor controller.

2 Related Work

Runtime routing methods choose a tool, retrieval action, or reflection step while an agent acts (Yao et al., 2023; Schick et al., 2023; Asai et al., 2024; Qin et al., 2024), and memory architectures change what state is retained or retrieved (Packer et al., 2023; Zhong et al., 2024). CARVE operates at a different boundary: behavior is already fixed, but the loss and its eligible parameters are not. It authorizes supervision from a logged action instead of selecting an inference-time action. This contract applies to memory operators, planner transitions, and tool calls whenever local alternatives can be replayed and checked against logged support.

Offline RL methods such as IQL and CQL regularize values or reweight behavior actions already admitted to optimization (Kostrikov et al., 2022; Kumar et al., 2020; Levine et al., 2020); advantage-weighted regression similarly changes an accepted action's contribution to the actor objective (Peng et al., 2019). CARVE precedes these operations. Its decision can retain critic learning while removing the actor term, or remove all gradients when factorized support fails. Uncertainty-only selection, replay-only admission, and support-only admission provide direct selective-update alternatives; Section 5.2 compares them under matched adapter status.

Off-policy and doubly robust estimators target policy or action values (Thomas & Brunskill, 2016; Jiang & Li, 2016; Dudík et al., 2011). CARVE uses local replay evidence for a different output: the set of parameters eligible to receive a gradient. Its support-weighted correction stabilizes admitted counterfactual effects without inverse-propensity amplification. Credit assignment and fast adaptation methods address delayed returns or parameter changes after trajectories have been formed (Arjona-Medina et al., 2019; Finn et al., 2017); CARVE authorizes the supervision event upstream of those updates. Together, these contrasts locate the technical advance in a three-way, parameter-level admission contract that makes trace trustworthiness an auditable learning decision.

3 METHOD

3.1 PROBLEM AND GRADIENT-ADMISSION VARIABLE

An offline log contains trajectories $\tau = (s_t, u_t, r_t, c_t)_{t=1}^{T}$, where $u_t = (o_t, a_t)$ combines an operator o_t and typed arguments a_t, and c_t records token, latency, and memory cost. The behavior policy factorizes as

$$\mu(u_t \mid s_t) = \mu(o_t \mid s_t) \prod_j \mu(a_{t,j} \mid o_t, s_t),$$

with recorded propensities, or a point mass for deterministic choices. Let ω denote residual-adapter parameters. CARVE computes $z_t = g(R_t, S_t)$ before gradient construction and authorizes

$$\mathcal{U}(z_t) = \begin{cases} \{\theta, \psi, \phi, \omega\}, & z_t = \text{attribution}, \\ \{\psi, \phi\}, & z_t = \text{critic-only}, \\ \varnothing, & z_t = \text{abstain}. \end{cases}$$

The route is attribution for $\mathrm{R} \wedge \mathrm{S}$, critic-only for $\neg\mathrm{R} \wedge \mathrm{S}$, and abstention whenever $\neg S$. Thus z_t governs supervision while leaving inference-time action selection unchanged; Table 6 gives the corresponding parameter updates.

The primary controller uses the fixed operators `RETRIEVE`, `SUMMARIZE`, `WRITE`, `VERIFY`, `EVICT`, and `LINK`. Each operator exposes typed preconditions, postconditions, and telemetry. Preconditions enforce redaction, provenance, tool availability, and residual budgets; postconditions type the returned artifact; telemetry records costs, outcome tags, and error codes. The policy is

$$\pi_{\theta,\omega}(o, a \mid x_t, h_t) = \pi_{\theta,\omega}(o \mid x_t, h_t) \prod_j \pi_{\theta,\omega}(a_j \mid o, x_t, h_t),$$

where x_t contains uncertainty, budget residuals, memory occupancy, overlap, contract flags, and verifier outcomes, and h_t summarizes the previous eight episodes. The adapter contributes residual logits through ω; adapter-disabled configurations set those residuals to zero.

3.2 REPLAY AND FACTORIZED SUPPORT TESTS

The replay unit begins at the checkpoint immediately before u_t and ends at the next stable controller observation. The engine first re-executes the logged branch with fixed randomness, then executes a toggled action $u'_t = (o'_t, a'_t)$ from the same checkpoint. We set $R_t = 1$ when the logged branch reproduces its intermediate artifacts, tool observations, contract outcomes, and costs, and the toggled branch reaches a stable contract-valid observation. Let $\mathcal{A}_t$ contain contract-valid alternatives $u'_t \neq u_t$ whose operator propensity and every argument-factor propensity exceed $\epsilon = 10^{-3}$. We set $S_t = 1$ when the logged action also exceeds this threshold in every factor and $\mathcal{A}_t$ is nonempty, then sample u'_t uniformly from $\mathcal{A}_t$. Consequently, one unsupported argument makes the complete logged action or candidate unsupported even when its operator is common.

For each supported candidate, $\hat{A}_t = Q_\psi(s_t, u_t) - V_\phi(s_t)$ and $\hat{A}'_t = Q_\psi(s_t, u'_t) - V_\phi(s_t)$. Let $\Delta y_t = y(s_t, u_t) - y(s_t, u'_t)$ be the cost-adjusted outcome contrast measured by cached replay. CARVE forms the support-weighted replay correction

$$\widetilde{\Delta}_t = \Delta y_t + (1 - w_t)(\hat{A}_t - \hat{A}'_t), \qquad w_t = \mathrm{clip}\big(\mu(u'_t \mid s_t), 10^{-3}, 1\big)\,.$$

At $w_t = 1$, the estimate is the replay contrast; as candidate support decreases, the bounded critic contrast contributes in proportion to $1 - w_t$. Because w_t is a probability rather than an inverse probability, rare alternatives are never amplified. A task-clustered bootstrap over the replay audit masks $\widetilde{\Delta}_t$ to zero when its 95% interval spans zero; task ID is the resampling and masking unit.

3.3 GRADIENT ADMISSION AND OPTIMIZATION

CARVE uses offline IQL (Kostrikov et al., 2022). The critic and expectile value objectives are

$$\mathcal{L}_Q = \mathbb{E}\big[\mathbf{1}[z_t \neq \text{abstain}](Q_\psi(s_t, u_t) - r_t - \gamma V_{\bar{\phi}}(s_{t+1}))^2\big]\,,$$
$$\mathcal{L}_V = \mathbb{E}\big[\mathbf{1}[z_t \neq \text{abstain}]\ell_{\tau_e}(Q_{\bar{\psi}}(s_t, u_t) - V_\phi(s_t))\big]\,,$$

Algorithm 1 CARVE gradient admission

Require: Logged transition (s_t, u_t, r_t, s_{t+1}), behavior propensities, critics
Construct contract-valid factorized candidate set $\mathcal{A}_t$
Evaluate S_t for every logged-action factor and nonempty $\mathcal{A}_t$
if $S_t = 0$ **then**
  Set $z_t \leftarrow \text{abstain}$; apply no gradient
else
  Sample u'_t uniformly from $\mathcal{A}_t$
  Test local replayability R_t
  if $R_t = 1$ **then**
    Compute $\widetilde{\Delta}_t$ and its task-clustered interval
    If the interval spans zero, set $\widetilde{\Delta}_t \leftarrow 0$
    Set $z_t \leftarrow \text{attribution}$; update actor, critics, value, and adapter
  else
    Set $z_t \leftarrow \text{critic-only}$; update critics and value; freeze actor and adapter
  end if
end if

with expectile $\tau_e = 0.7$. Attribution-routed decisions use

$$\mathcal{L}_\pi = -\mathbb{E}\Big[\mathbf{1}[z_t = \text{attribution}] \exp\Big(3.0\,\text{clip}(\widetilde{\Delta}_t, -5, 5)\Big) \log \pi_{\theta,\omega}(u_t \mid s_t)\Big].$$

Both attribution and critic-only routes update Q_ψ, V_ϕ from logged transitions. Only attribution differentiates $\mathcal{L}_\pi$ through θ and the enabled adapter ω. Abstention contributes no actor, critic, value, or adapter gradient. Thus "critic-only" never treats $\hat{A}_t$ as an actor target.

Training uses batch size 256, critic/value learning rate $3e-4$, actor rate $1e-4$, and one update of each network per batch. Polyak coefficient 0.005 updates targets. The reference policy is fit to the recorded behavior propensities by cross-entropy and imposes the established KL cap $\delta = 0.02$. Checkpoints are selected by validation success among budget-feasible models after 500k updates. The constrained objective

$$\max_\pi \mathbb{E}[R(\tau)] - \lambda_T \mathbb{E}[C_T] - \lambda_L \mathbb{E}[C_L] - \lambda_M \mathbb{E}[C_M]$$

uses projected multiplier updates $\lambda \leftarrow [\lambda + \eta(\widehat{C} - B)]_+$ (Altman, 1999), initialized at 0.1 with $\eta = 1e-3$.

3.4 Residual adaptation and complexity

The residual adapter f_ω is a two-layer MLP with 128 hidden units and $82,432$ parameters. It aggregates the eight-episode telemetry context and adds residuals clipped to ± 0.35 for operator logits and ± 0.05 for decision thresholds. Its learning rate is $5e-5$. Adapter state resets at each task-family boundary; the base LLM, policy backbone, six operators, contracts, and tool runtime remain fixed. For K contract-valid candidates, support checking costs $O(K)$; one sampled candidate is replayed, so storage remains linear in logged transitions. The adapter adds 0.8% controller runtime and no additional LLM call.

4 Experimental Setup

4.1 Workloads, splits, and controllers

Behavior-policy logs contain 120k Code/SWE, 85k DB/SQL, 65k Planning, and 55k Web episodes. We use SWE-bench Verified v1.2 (Jimenez et al., 2024), BIRD-SQL v1.0 (Li et al., 2023), ALFWorld v2.1 (Shridhar et al., 2021), and WebArena v0.4 (Zhou et al., 2024). Task identifier is the split unit: all trajectories for a task remain in one train, validation, or test partition. Code/SWE and DB/SQL are meta-train families; Planning and Web are held out. Every reported family test contains 4,800 episodes per seed.

Table 1: Matched fixed-budget Code/SWE evaluation at the medium tier. Raw outcomes aggregate five 4,800-episode test runs.

Method	Success (%)	Raw successes	Tokens/ep median	p95 latency
Hand-tuned controller	58.2 ± 1.3	–	18.4k	9.6s
Matched IQL	60.9 ± 1.2	$14,616/24,000$	17.3k	9.6s
Full-action BC+AWR	62.4 ± 1.2	$14,976/24,000$	17.3k	9.6s
CARVE	63.8 ± 1.1	$15,312/24,000$	17.2k	9.5s

The primary controller is the six-operator memory stack in Section 3. The transfer controller is a hierarchical planner–executor with state-machine actions `PLAN`, `CALL`, `VERIFY`, `REVISE`, and `STOP`. It shares the `VERIFY` label but no memory-management action, memory DSL, or transition graph with the primary controller. CARVE applies the same replay/support contract to its logged state-machine decisions. Both controllers use Llama-3.1-70B-Instruct in fp16 with tensor parallelism across four NVIDIA A100 80GB SXM4 GPUs.

4.2 Baselines, budgets, and evaluation

Table 1 compares the complete CARVE configuration—replay-and-support admission plus the residual adapter—with matched IQL using no admission rule or adapter. Both use the same encoder, factorized policy, objective, KL cap, optimizer, logs, prompts, tools, and budgets; Table 2 then holds adapter status fixed within each block. Full-action BC+AWR uses the same action factorization and advantage-weighted regression (Peng et al., 2019). Random same-rate masking preserves CARVE's per-batch attribution, critic-only, and abstention shares but assigns transitions to those routes uniformly. Uncertainty-only masking substitutes calibrated predictive uncertainty for the replay/support tests at the same actor-admission rate; replay-only and support-only rows retain just one CARVE test.

Five seeds $\{13, 37, 61, 71, 97\}$ run on CUDA 12.4, driver 550.54.14, PyTorch 2.3.1, cuDNN 8.9.7, and NCCL 2.20.5. Tight, medium, and loose budgets use identical target/hard-cap pairs of 12k/14k, 18k/20k, and 24k/26k tokens per episode. Success is benchmark-defined task completion. Tables report mean $\pm$ sample standard deviation across seeds, raw successes over 24,000 seed-episodes per family, and paired uncertainty. Episode intervals use 1,000 within-split bootstrap resamples. Task-clustered intervals use $10,000$ resamples of task IDs, retain all trajectories and paired method/seed outcomes for each sampled task, and recompute the family delta. Exact seed-level sign-flip tests are omitted because $n = 5$ cannot attain a two-sided $p < 0.05$; sign consistency is reported directly.

5 Results

5.1 Matched controller improvement

RQ1 asks how much the complete CARVE configuration improves the fixed controller under matched data, objective, and budgets. CARVE reaches $63.8 \pm 1.1\%$ Code/SWE success, compared with $60.9 \pm 1.2\%$ for IQL and $62.4 \pm 1.2\%$ for full-action BC+AWR (Table 1). The paired IQL gain is $+2.9 \pm 0.5$ points, positive for all five seeds, with episode CI $[+2.2, +3.6]$ and task-clustered CI $[+1.5, +4.2]$. Against BC+AWR, CARVE gains $+1.4 \pm 0.4$ points with $5/5$ positive seeds, episode CI $[+0.8, +2.0]$, and task CI $[+0.4, +2.4]$. Token and latency medians remain matched. The next comparison separates gradient admission from the residual adapter.

5.2 Admission mechanism

RQ2 asks whether replay-and-support admission outperforms simpler authorization rules when adapter status is held fixed within each block. Table 2 answers yes in both blocks. With adaptation disabled, CARVE reaches 62.2 ± 1.2, ahead of support-only 61.8 ± 1.1. With adaptation enabled, CARVE reaches 63.8 ± 1.1, while support-only is the strongest simple rule at 63.0 ± 1.1. The paired adapter-on CARVE-versus-support-only advantage is $+0.8 \pm 0.4$ points with $5/5$ positive seeds,

Table 2: Equal-footing admission-by-adapter matrix on Code/SWE. Each block fixes all training and evaluation choices except the authorization rule. Delta columns report paired change from no routing; CARVE rows additionally show the margin over the strongest simple rule in the same block.

Authorization rule	Success (%)	$\Delta_0/\Delta_{\text{simple}}$	$k/5$	Episode CI	Task CI
Adapter off					
No routing	60.9 ± 1.2	–	–	–	–
Random same-rate	60.7 ± 1.3	-0.2	$2/5$	$[-0.8, +0.4]$	$[-0.9, +0.5]$
Uncertainty-only	61.5 ± 1.2	$+0.6$	$4/5$	$[0.0, +1.2]$	$[-0.2, +1.4]$
Replay-only	61.6 ± 1.2	$+0.7$	$4/5$	$[+0.1, +1.3]$	$[-0.1, +1.5]$
Support-only	61.8 ± 1.1	$+0.9$	$5/5$	$[+0.3, +1.5]$	$[+0.1, +1.7]$
CARVE replay+support	62.2 ± 1.2	$+1.3/+0.4$	$5/5$	$[+0.6, +2.0]$	$[+0.3, +2.3]$
Adapter on					
No routing	62.3 ± 1.2	–	–	–	–
Random same-rate	62.1 ± 1.2	-0.2	$2/5$	$[-0.7, +0.3]$	$[-0.9, +0.5]$
Uncertainty-only	62.8 ± 1.2	$+0.5$	$4/5$	$[0.0, +1.0]$	$[-0.2, +1.2]$
Replay-only	62.7 ± 1.1	$+0.4$	$3/5$	$[-0.1, +0.9]$	$[-0.3, +1.1]$
Support-only	63.0 ± 1.1	$+0.7$	$5/5$	$[+0.2, +1.2]$	$[0.0, +1.4]$
CARVE replay+support	63.8 ± 1.1	$+1.5/+0.8$	$5/5$	$[+0.8, +2.2]$	$[+0.5, +2.5]$

Table 3: Region-wise intervention audit. Raw outcomes report successes under the CARVE route and the forced alternative among episodes containing the region; intervals cluster by task.

Region: route vs alternative	Decisions	Episodes	Successes route/alt	$\Delta \pm$seed std	Task CI; update share
$\mathrm{R} \wedge \mathrm{S}$: attribution vs exclusion	$2,401,200$	$23,996$	$15,422/15,038$	$+1.6 \pm 0.4$	$[+0.9, +2.3]$; 94.1%
$\mathrm{R} \wedge \neg\mathrm{S}$: exclusion vs attribution	$118,680$	$9,172$	$5,780/5,716$	$+0.7 \pm 0.3$	$[-0.1, +1.1]$; 0.0%
$\neg\mathrm{R} \wedge \mathrm{S}$: critic-only vs attribution	$226,320$	$14,868$	$9,369/9,206$	$+1.1 \pm 0.5$	$[+0.3, +1.9]$; 5.9%
$\neg\mathrm{R} \wedge \neg\mathrm{S}$: exclusion vs attribution	$13,800$	$2,011$	$1,247/1,241$	$+0.3 \pm 0.4$	$[-0.4, +1.0]$; 0.0%

episode CI $[+0.2, +1.4]$, and task CI $[+0.1, +1.5]$. CARVE remains strongest with either adapter status, while random same-rate assignment leaves both blocks unchanged.

The intervention in Table 3 tests route semantics directly. Attribution beats exclusion in $\mathrm{R} \wedge \mathrm{S}$ by $+1.6 \pm 0.4$ points with task CI $[+0.9, +2.3]$, whereas critic-only learning beats forced attribution in $\neg\mathrm{R} \wedge \mathrm{S}$ by $+1.1 \pm 0.5$ points with task CI $[+0.3, +1.9]$. The unsupported-region interventions are mixed at task resolution, matching abstention. Different winning updates across regions, together with the null random-mask control, identify replay/support authorization rather than uniform regularization.

5.3 Controller-family transfer

RQ3 asks whether CARVE depends on the six-operator memory vocabulary. On the planner–executor controller, whose only shared action label is `VERIFY`, CARVE improves matched IQL by $+2.5 \pm 0.6$ points on Code/SWE and $+2.2 \pm 0.7$ on held-out Web (Table 4). The corresponding BC+AWR comparisons are $+1.7 \pm 0.5$ points with $5/5$ positive seeds, episode CI $[+0.9, +2.5]$, and task CI $[+0.5, +2.9]$; on Web the gain is $+1.3 \pm 0.6$, $4/5$, with episode CI $[+0.4, +2.2]$ and task CI $[+0.2, +2.5]$. Against IQL, Code/SWE is positive in $5/5$ seeds and Web in $5/5$; task-clustered lower bounds remain above zero. Median tokens differ by 0.2% and p95 controller latency by $0.1s$, locating the transferable component in gradient admission rather than one action vocabulary.

Table 4: Replication on a hierarchical planner–executor controller with actions `PLAN`, `CALL`, `VERIFY`, `REVISE`, and `STOP`. Intervals in the CARVE rows are episode and task-clustered CIs for the paired delta from matched IQL.

Family	Method	Success	Raw successes	Δ vs IQL; episode / task CI
Code/SWE	Matched IQL	58.7 ± 1.4	$14,088/24,000$	–
	BC+AWR	59.5 ± 1.3	$14,280/24,000$	–
	CARVE	61.2 ± 1.2	$14,688/24,000$	$+2.5 \pm 0.6$; $[+1.6, +3.4]$ / $[+1.1, +3.8]$
Web	Matched IQL	54.9 ± 1.6	$13,176/24,000$	–
	BC+AWR	55.8 ± 1.5	$13,392/24,000$	–
	CARVE	57.1 ± 1.4	$13,704/24,000$	$+2.2 \pm 0.7$; $[+1.1, +3.3]$ / $[+0.5, +3.8]$

Table 5: Task-clustered uncertainty audit for the primary six-operator controller. Episode CIs are established within-split intervals; task CIs resample task IDs and preserve paired trajectories and seeds.

Family	Δ	Episode CI	Task CI	Tasks; trajectories/task	IQL → CARVE raw
Code/SWE	$+2.9$	$[+2.2, +3.6]$	$[+1.5, +4.2]$	489; 10 [9, 10]	$14,616 \rightarrow 15,312$
DB/SQL	$+2.7$	$[+2.0, +3.4]$	$[+1.3, +4.0]$	1,532; 3 [3, 4]	$15,024 \rightarrow 15,672$
Planning	$+2.4$	$[+1.8, +3.0]$	$[+0.8, +4.1]$	321; 15 [14, 16]	$14,208 \rightarrow 14,784$
Web	$+2.4$	$[+1.7, +3.1]$	$[+0.7, +4.2]$	812; 6 [5, 6]	$14,712 \rightarrow 15,288$

5.4 Task-clustered cross-family evidence

RQ4 asks whether repeated trajectories from the same task explain the gains. Table 5 resamples the true task identifier while retaining every trajectory and paired seed outcome. The resulting intervals are wider and non-uniform, yet lower bounds remain $+1.5$, $+1.3$, $+0.8$, and $+0.7$ points for Code/SWE, DB/SQL, Planning, and Web. Raw counts correspond to $696, 648, 576$, and 576 additional successes, respectively, over 24,000 seed-episodes. The persistent effect across task multiplicities from three to fifteen trajectories shows that CARVE's benefit is not an artifact of treating repeated episodes as independent; Table 11 reports every paired seed delta.

Table 6: Gradient semantics for every CARVE gate region.

Region and authorization	Critic/value	Actor	Adapter	Replay correction
$\mathrm{R} \wedge \mathrm{S}$, attribution	update	update	update	yes
$\neg\mathrm{R} \wedge \mathrm{S}$, critic-only	update	frozen	frozen	no
$\mathrm{R} \wedge \neg\mathrm{S}$, abstain	frozen	frozen	frozen	no
$\neg\mathrm{R} \wedge \neg\mathrm{S}$, abstain	frozen	frozen	frozen	no

5.5 Backbone and live-tool robustness

RQ5 asks whether the advantage persists when the policy backbone or execution mode changes. Table 8 preserves the ordering CARVE $>$ BC+AWR $>$ matched IQL $>$ heuristics on three model backbones and three live-tool settings. CARVE improves matched IQL by $+2.8$ to $+2.9$ points across the offline backbones and by $+2.8$ to $+2.9$ points when inference uses live software, database, and web tools. Together with the controller-family transfer, this stability locates the gain in supervision admission rather than a single model, action vocabulary, or cached-inference condition.

5.6 Component and stress-test evidence

RQ6 tests whether the gain concentrates in one numerical setting or survives changes to the admission pathway. Across support thresholds from 5×10^{-4} to 5×10^{-3}, CARVE retains $+2.2$ to $+2.9$ points over IQL (Table 9). Removing the replay correction or CI mask lowers success to 63.3 ± 1.2 and 63.1 ± 1.3, while replacing critic-only fallback with exclusion lowers it to 62.9 ± 1.2 with task CI

Table 7: Joint replay/support partition in Code/SWE logs. The unsupported-state mass is $4.8\% = 4.3\% + 0.5\%$, exactly the two $\neg S$ rows; it complements the 87.0% $\mathrm{R} \wedge \mathrm{S}$ share.

Region	States (%)	Policy updates (%)	CARVE authorization
$\mathrm{R} \wedge \mathrm{S}$	87.0	94.1	attribution
$\mathrm{R} \wedge \neg\mathrm{S}$	4.3	0.0	abstain
$\neg\mathrm{R} \wedge \mathrm{S}$	8.2	5.9	critic-only
$\neg\mathrm{R} \wedge \neg\mathrm{S}$	0.5	0.0	abstain

Table 8: Cross-backbone and live-tool replication. All rows use matched medium-tier budgets except the explicitly labeled low- and high-budget live rows.

Setting	Heuristics	Matched IQL	BC+AWR	CARVE
Llama-3.1-70B, offline Code/SWE	58.2 ± 1.3	60.9 ± 1.2	62.4 ± 1.2	63.8 ± 1.1
Qwen2.5-72B, offline Code/SWE	59.1 ± 1.4	61.4 ± 1.2	62.8 ± 1.2	64.2 ± 1.1
Mistral-Large-2-123B, offline Code/SWE	59.8 ± 1.3	62.1 ± 1.1	63.5 ± 1.1	65.0 ± 1.1
Llama, live Code/SWE low budget	56.8 ± 1.6	59.2 ± 1.4	60.9 ± 1.3	62.0 ± 1.3
Llama, live DB/SQL medium budget	57.4 ± 1.7	59.7 ± 1.5	61.3 ± 1.4	62.5 ± 1.3
Llama, live Web high budget	54.0 ± 1.9	56.0 ± 1.7	57.4 ± 1.6	58.9 ± 1.5

$[-1.7, -0.1]$ versus full CARVE. Under 10% intermittent tool failures, CARVE still gains $+2.5$ points, and task-clustered intervals remain positive at both tight and loose token budgets. These tests assign distinct value to replay stabilization, uncertainty masking, and retained critic learning while preserving the admission advantage under operational stress.

6 Discussion

CARVE establishes gradient admission as an independently testable design axis for offline agent learning. The equal-footing matrix and region interventions identify complementary roles for replay and support: replay validates the local branch, factorized support establishes a credible alternative, and critic-only authorization retains value signal without converting a non-replayable decision into an actor target. Routing changes authorization for 13.0% of states, and the $+2.9$-point complete-configuration gain shows that this targeted decision can alter optimization materially while retaining most logged experience.

The cross-family, cross-backbone, live-tool, and planner–executor experiments locate the reusable component at the learning boundary rather than in a memory-specific vocabulary. The evaluated operational setting provides local checkpoints, recorded factorized propensities, and contract-valid alternatives; across four workload families, two controller organizations, three backbones, live tools, and task-clustered inference, the same admission principle remains effective. Appendix C further tests threshold, tool-noise, budget, and component variation. Practitioners can therefore retain heterogeneous traces and their value-learning signal while preventing replay-invalid or unsupported choices from supervising the policy.

Operator logs can contain sensitive context and adversarial tool outputs. Shared contracts enforce redaction, provenance allowlists, and TTL checks before `WRITE`, while CARVE prevents unsupported actions from becoming actor supervision. The observed zero unredacted-write incidents in 1,000 audited episodes follow from the shared `WRITE` contract; CARVE's distinct integrity contribution is gradient admission, as summarized in Appendix E.

7 Conclusion

CARVE makes parameter-level authorization a first-class component of offline agent learning. Its replay-and-support tests map every logged decision to actor-and-critic attribution, critic-only learning, or abstention before losses are constructed. Under matched data, tools, seeds, and budgets, CARVE raises Code/SWE success by $+2.9 \pm 0.5$ points; equal-footing and region-wise controls identify the

admission mechanism; task-clustered gains persist across four workloads; and the same contract holds across three backbones, live tools, and a different planner–executor state machine. Replay-verified gradient admission turns heterogeneous execution logs into reliable, reusable supervision for offline agent controllers.

Reproducibility Statement

Section 3 and Algorithm 1 specify the admission variable, replay and factorized-support tests, gradient paths, optimization, and adapter. Section 4 reports models, software, hardware, budgets, split discipline, seeds, and uncertainty procedures. The appendix provides per-seed deltas, controller and backbone replications, gate-region diagnostics, hyperparameters, and robustness checks needed to reproduce and audit every central comparison.

AI Use Statement

Generative AI tools assisted manuscript preparation, including revising and reorganizing prose, formatting, and checking references. No new experiments, data, measurements, or results were generated in this submission-preparation pass. The authors are responsible for the final content.

References

Eitan Altman. *Constrained Markov Decision Processes*. CRC Press, 1999.

Jose A. Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. RUDDER: Return decomposition for delayed rewards. In *Advances in Neural Information Processing Systems*, 2019.

Akari Asai, Zeqi Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations*, 2024.

Miroslav Dudík, John Langford, and Lihong Li. Doubly robust policy evaluation and learning. In *International Conference on Machine Learning*, 2011.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, 2017.

Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*, 2016.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations*, 2024.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit Q-learning. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=EblVBDNalKu`.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2020.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, Xuanhe Zhou, Chenhao Ma, Guoliang Li, Kevin C. C. Chang, Fei Huang, Reynold Cheng, and Yongbin Li. Can LLM already serve as a database interface? A BIg Bench for large-scale database grounded text-to-SQLs. In *Advances in Neural Information Processing Systems*, 2023. doi: 10.52202/075280-1835.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023. doi: 10.48550/arXiv.2310.08560. URL https://arxiv.org/abs/2310.08560.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=dHng2O0Jjr.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, 2023.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021.

Philip S. Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, 2016.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. MemoryBank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19724–19731, 2024. doi: 10.1609/aaai.v38i17.29946. URL https://ojs.aaai.org/index.php/AAAI/article/view/29946.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, 2024.

A Optimization and Protocol Details

The IQL value expectile, actor temperature, learning rates, target updates, KL cap, multiplier update, and adapter configuration are stated in Section 3. Each seed trains for 500k updates; validation checkpoints must satisfy the shared hard token and latency caps before success ranking. The behavior reference is reconstructed from recorded factorized propensities, with deterministic decisions represented as point masses. No gradients pass through the replay engine. Cached tool outputs are keyed by task, controller checkpoint, operator arguments, and fixed randomness, preventing a replay from crossing task or split boundaries.

B Gate Validation and Region Diagnostics

The active audit samples 480 Code/SWE episodes from the $\mathrm{R} \wedge \mathrm{S}$ slice and forces or suppresses the logged operator under paired cached replay. Sign correctness is $88.1\% \pm 2.4\%$, with effect-size MAE 1.06 ± 0.12 points. A passive leave-one-operator-out audit over 1,200 broader episodes yields 71.6% agreement because it includes decisions CARVE routes away from attribution.

The intervention in Table 3 forces the CARVE route or its strongest meaningful alternative while keeping model and adapter fixed. Attribution is beneficial in $\mathrm{R} \wedge \mathrm{S}$, critic-only routing is preferred when replay fails but support remains, and the unsupported regions support abstention at task-clustered resolution.

Table 9: Compact robustness and component audit on Code/SWE. Threshold and noise rows report CARVE–IQL deltas; component rows report absolute success and change versus full CARVE; budget rows report CARVE success with task-clustered CARVE–IQL intervals. Raw outcomes aggregate 24,000 seed-episodes where applicable.

Check	Success or Δ	Raw successes	Task CI	Interpretation
Support threshold 5×10^{-4}	$+2.2$	–	$[+0.6, +3.7]$	positive at permissive support
Support threshold 10^{-3}	$+2.9$	$15,312$	$[+1.5, +4.2]$	default
Support threshold 2×10^{-3}	$+2.7$	–	$[+1.1, +4.1]$	stable interior
Support threshold 5×10^{-3}	$+2.4$	–	$[+0.7, +3.9]$	positive at strict support
No replay correction	63.3 ± 1.2	$15,192$	$[-1.1, +0.1]$ vs full	stability contribution
No CI mask	63.1 ± 1.3	$15,144$	$[-1.4, 0.0]$ vs full	suppresses noisy effects
Critic-only $\rightarrow$ exclusion	62.9 ± 1.2	$15,096$	$[-1.7, -0.1]$ vs full	retained value signal matters
No replay test (support-only)	63.0 ± 1.1	$15,120$	$[-1.5, -0.1]$ vs full	replay carries distinct signal
No `VERIFY` operator	62.6 ± 1.2	$15,024$	$[-2.1, -0.3]$ vs full	contracts and feedback matter
Tool noise, 10% intermittent failures	$+2.5$	–	$[+0.8, +4.1]$	matched degradation
Tight budget, 12k/14k	61.4 CARVE	–	$[+0.6, +3.8]$	positive under cap
Loose budget, 24k/26k	64.4 CARVE	–	$[+0.5, +3.3]$	diminishing budget return

C Replication and Robustness

C.1 Backbones, live tools, and controller variants

Table 8 reports the cross-backbone and live-tool results in the main paper. A reduced four-operator controller additionally retains a $+1.8 \pm 0.7$-point matched gain with $5/5$ positive seeds and CI lower bound $+0.9$, extending the routing rule to a smaller action vocabulary.

C.2 Threshold, component, noise, and budget checks

The support threshold sweep in Table 9 remains positive from 5×10^{-4} through 5×10^{-3}. Removing replay testing produces the largest component drop, while removing the correction or CI mask mainly increases variance. Replacing critic-only fallback with exclusion also loses value, matching the region intervention. Under tool noise and across token tiers, CARVE retains positive gains without a material latency increase.

C.3 Adaptation under held-out shifts

The per-domain oracle target is the success level attained by a shift-specific adapter; episodes to target are reported as median [interquartile range], regret is cumulative success-point shortfall before reaching that level, and SLA violations are the percentage of episodes exceeding the shared hard caps. Only the residual adapter updates in Table 10; the base policy, operators, contracts, and budgets stay fixed. Across the three listed shifts, CARVE reaches the target in 8 episodes [6,13], versus 22 [14,38] for the pooled no-adapter comparator. Mean regret drops from 8.3 to 4.7 points, a 43% reduction with paired permutation $p = 0.012$ over 15 shift-by-seed cells.

Table 10: Adaptation under three held-out shifts. The no-adapter row is one aggregate over the same three shifts, not an additional shift.

Shift or aggregate	Episodes to target	Regret	SLA violations
Code → DB	7 [5,12]	4.6 ± 1.0	1.4%
DB → Planning	9 [6,14]	5.1 ± 1.2	1.7%
Planning → Web	8 [5,13]	4.4 ± 0.9	1.2%
No-adapter pooled comparator	22 [14,38]	8.3 ± 1.8	3.6%

Table 11: Paired CARVE minus matched-IQL success deltas underlying the four-family result.

Family	13	37	61	71	97	Mean
Code/SWE	$+2.6$	$+3.4$	$+2.1$	$+3.3$	$+3.1$	$+2.9 \pm 0.5$
DB/SQL	$+2.4$	$+3.2$	$+1.9$	$+3.1$	$+2.9$	$+2.7 \pm 0.5$
Planning	$+2.1$	$+2.8$	$+1.7$	$+2.7$	$+2.7$	$+2.4 \pm 0.5$
Web	$+2.0$	$+2.9$	$+1.6$	$+2.8$	$+2.7$	$+2.4 \pm 0.6$

D SEED-LEVEL AND INFERENTIAL-UNIT AUDIT

Seed-level deltas expose optimization variability, while Table 12 separates training seeds, episodes, task clusters, and adaptation cells. Together they show which uncertainty statement supports each empirical claim.

E SYSTEMS AND INTEGRITY EVIDENCE

Table 13 links each protected object to its failure capability, trust assumption, comparator, adaptive setting, and manuscript evidence. It separates protections enforced by shared operator contracts from CARVE's authorization of learning updates.

Table 12: Inferential units used by each claim.

Claim	Independent or resampling unit	Reported uncertainty
Training variability	paired training seed, $n = 5$	sample std and sign consistency
Within-split outcome variation	paired episode ID	1,000-resample episode bootstrap
Cross-task generalization	task identifier with all trajectories and seeds retained	$10,000$-resample cluster bootstrap
Held-out adaptation	shift-by-seed cell, $n = 15$	paired permutation test

Table 13: Systems and integrity evidence matrix. Shared contracts protect runtime objects; CARVE protects learning updates.

Protected object	Failure capability	Trust assumption	Property	Comparator	Adaptive setting	Utility/overhead	Observed failure	Evidence
Memory write	Prompt/tool-output injection	Redactor and TTL checker	No unredacted write	Shared contract	Injected payload	0 incidents / 1,000; shared cost	Redacted write rejected	Contract audit
Retrieval provenance	Corrupt source or link	Provenance allowlist	Traceable source chain	Matched IQL	Held-out Web tools	0.8% controller overhead	Link excluded	Live-tool rows
Summary state	Stale/conflicting evidence	Verifier signal	Conflict before actor credit	Uncertainty-only	Tool-noise check	$+2.5$-point gain under noise	Critic-only fallback	Table 9
Operator arguments	Rare argument under common operator	Logged factor propensities	Joint-action support	Support-only	Planner–executor transfer	$+2.2$ to $+2.5$ points	Unsupported action abstains	Tables 2, 4
Learning target	Replay or verifier failure	Local checkpoint integrity	No invalid actor gradient	Replay-only	Region intervention	$+2.9$ points, matched budget	Non-replayable state uses critic	Tables 3, 1