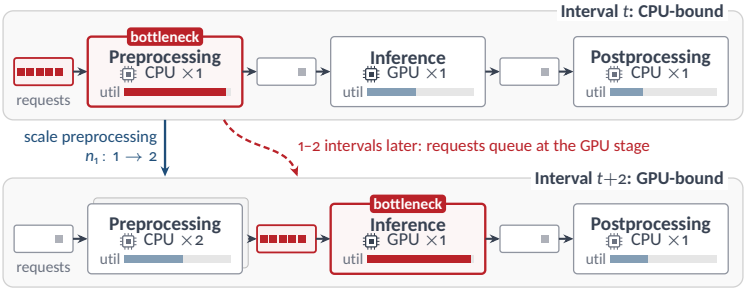
SAIR: Cost-Efficient Multi-Stage ML Pipeline Autoscaling via In-Context Reinforcement Learning

Jianchang Su[1], Yifan Zhang[1], Shengkai Lin[2], Shizhen Zhao[2],
Yusheng Zheng[3], Yiwei Yang[3], Wei Zhang[1]

[1]University of Connecticut [2]Shanghai Jiao Tong University [3]UC Santa Cruz

NeurIPS 2026

Scaling one stage moves the bottleneck to the next



- **Bottleneck migration:** the limiting stage changes with workload intensity.
- **Cross-stage coupling:** scaling preprocessing alone can cause a GPU contention spike 1 to 2 intervals later.
- **Ambiguous signals:** high GPU utilization can mean a GPU shortage or downstream congestion.

Prior autoscalers need offline data or see one stage

Approach	Offline data	Cross-stage	GPU rate
HPA / VPA	none	×	×
Threshold	none	×	×
FIRM / AWARE	10^3 episodes	✓	×
Sinan	per-app profile	partial	×
InferLine	per-pipeline profile	✓	×
SAIR	**none**	✓	✓

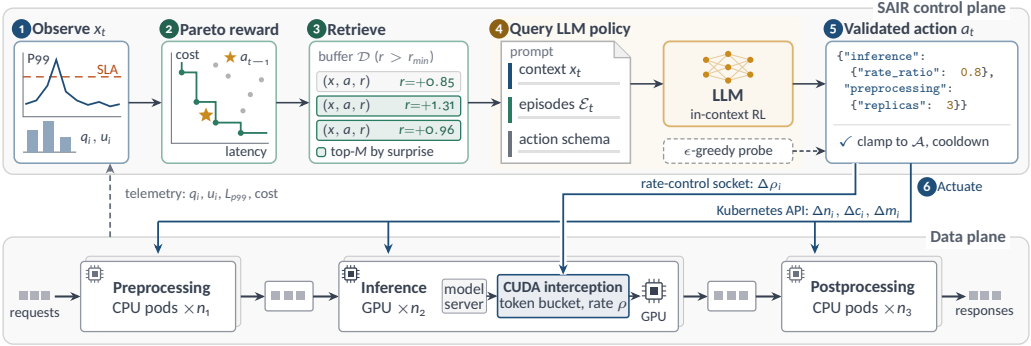
Zero-shot deployment

A new pipeline comes without traces. Learning-based controllers lack training data, and reactive controllers ignore cross-stage effects.

Our question

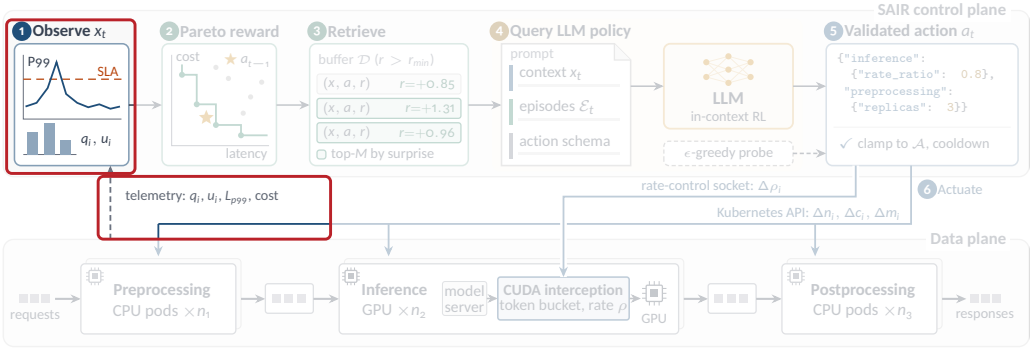
Can a controller learn a cross-stage policy online, from the first decision interval?

SAIR runs one in-context RL round per decision interval



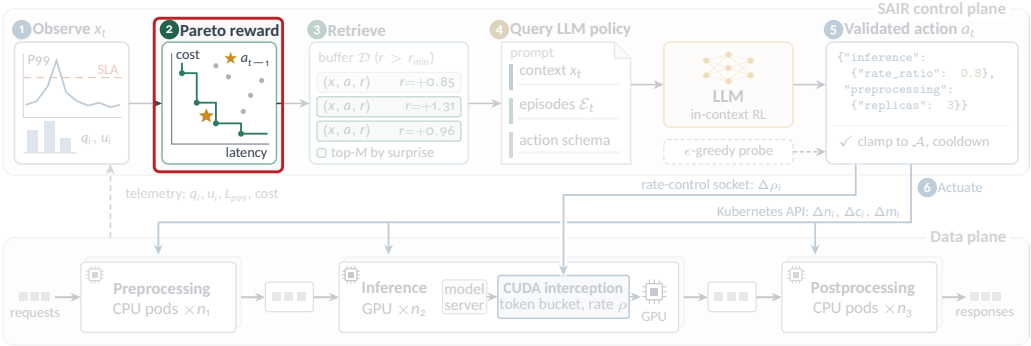
Each 30 s decision interval is one round. The LLM weights stay fixed; the policy improves through the episodes it retrieves.

SAIR runs one in-context RL round per decision interval



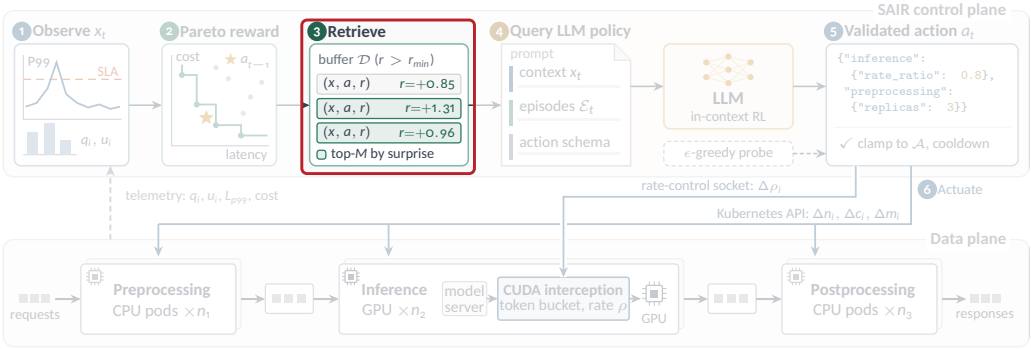
1. **Observe:** stage telemetry (queues, utilization, P99, cost) forms the context x_t.

SAIR runs one in-context RL round per decision interval



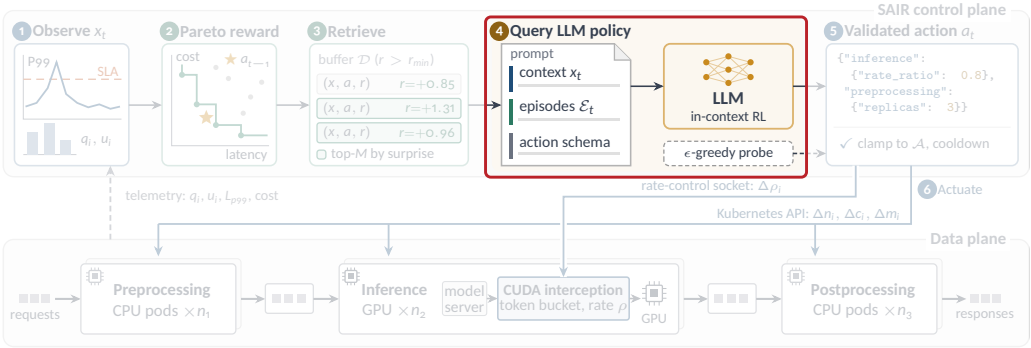
2. Score: the Pareto-shaped reward scores the previous action; positive episodes enter the buffer $\mathcal{D}$.

SAIR runs one in-context RL round per decision interval



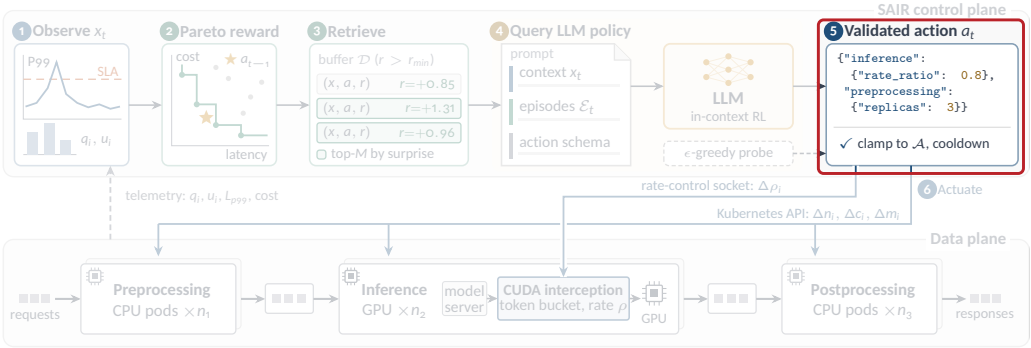
3. **Retrieve:** the top-M episodes by Bayesian surprise.

SAIR runs one in-context RL round per decision interval



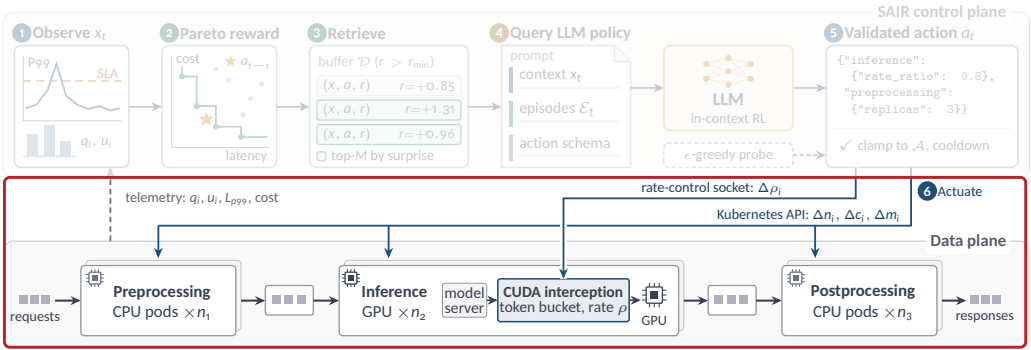
4. **Query the LLM:** context, episodes, and the action schema; ϵ-greedy probes explore.

SAIR runs one in-context RL round per decision interval



5. **Validate:** the JSON action is clamped to the grid $\mathcal{A}$, with cooldowns.

SAIR runs one in-context RL round per decision interval



6. Actuate: replicas and CPU/memory through the Kubernetes API, GPU rate through CUDA interception.

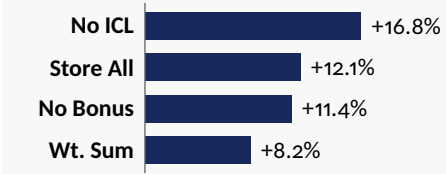
Three mechanisms make in-context RL reliable

Pareto-shaped reward
Ranks non-dominated latency-cost trade-offs above dominated ones, with a provable margin $\alpha - \beta$.

Bayesian-surprise retrieval
$\text{score}(e) = \text{sim}(x_e, x_t) \cdot |r_e - \mathbb{E}[r \mid \mathcal{D}_{-e}]|$ picks the top-M episodes from a growing buffer of positive episodes.

Continuous GPU rate control
`LD_PRELOAD` CUDA interception throttles kernel launches with a token bucket: sub-millisecond actuation, 0.6% overhead.

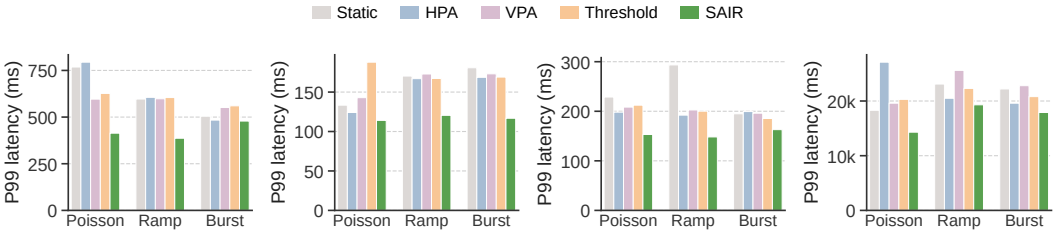
Removing a component raises P99



No ICL: without retrieval. Store All: negative episodes kept. No Bonus: without the proactive term. Wt. Sum: weighted sum in place of Pareto. Text generation, 3 seeds.

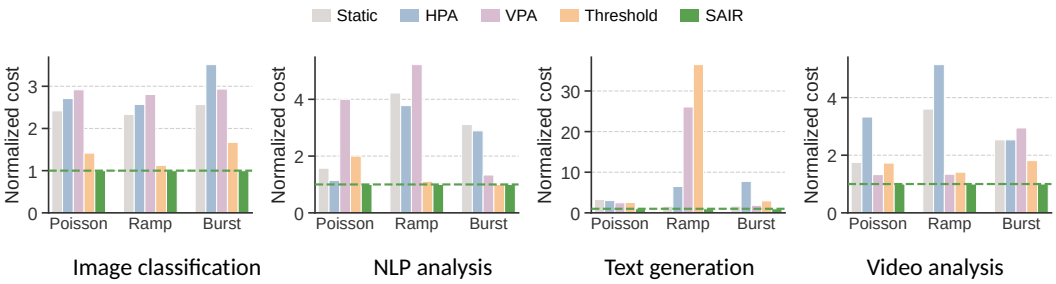
A validator clamps every action to the grid $\mathcal{A}$. The regret bound splits into retrieval and LLM-selection terms, each bounded by one of these mechanisms.

SAIR has the best or tied-best P99 in every configuration



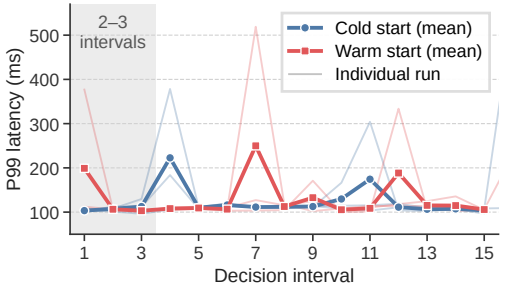
- Up to 50% lower P99 than the next-best baseline; lower is better.
- The largest margins over HPA appear on GPU-bound pipelines (NLP, text generation), where single-metric controllers scale the wrong stage.

SAIR lowers effective cost by 30–60%

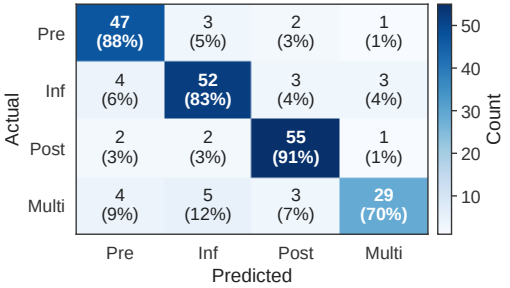


- Cost normalized to SAIR = 1, so values above 1 cost more than SAIR; 30–60% lower than the next-best baseline.
- Up to 97% lower with GPU sharing (best case), validated by co-location runs on the testbed.

SAIR catches up with a warm start in 2–3 intervals



From an empty buffer, SAIR matches warm-start P99 within 2–3 decision intervals (full-run gap 2.8%).



86% bottleneck detection accuracy on 200 injected scenarios: 88–91% single-stage, 70% multi-stage (random: 25%).

Takeaways

- An LLM with in-context experience autoscales a new multi-stage pipeline from the first interval, without offline training.
- Retrieval, the action validator, and forced probes each map to a term of the regret bound.
- Offline RL baselines trained in simulation transfer as frozen policies that hold minimal allocations.
- On a 2-node, 4-GPU cluster, SAIR serves 240 rps at P99 107 ms with unchanged prompts.

Paper

Supported in part by NSF Award #2530909 and NEUTC. Contact: wei.13.zhang@uconn.edu