OW-CMORL: ONLINE WINDOW-CONDITIONED MULTI-OBJECTIVE REINFORCEMENT LEARNING FOR TIME-VARYING OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper investigates non-stationary multi-objective reinforcement learning (MORL) for industrial control, where drifting exogenous operating conditions alter the Pareto frontier. Targeting this non-stationary online deployment scenario, we identify three practical obstacles: catastrophic forgetting, implicit context inference, and near-real-time constraints. We formalize this problem as a time-varying multi-objective contextual Markov decision process and derive three complementary theoretical upper-bounds bounding context embedding mismatch, local fine-tuning approximation, and long-run forgetting-induced performance degradation. We propose Online Window-Conditioned MORL (OW-CMORL): a context-indexed expert-bank stores restorable policy snapshots to mitigate catastrophic forgetting; a Transformer-based context-window encoder generates query embeddings from factor history and short-horizon forecasts for implicit unlabeled-condition representation; indexed retrieval fetches matched snapshots for lightweight local refinement, whose outputs refresh the online Pareto archive and avoid costly full retraining to satisfy near-real-time constraints. Experiments across four dynamic control systems under a unified evaluation protocol demonstrate that OW-CMORL preserves high-quality Pareto frontier while adapting to non-stationary operating conditions under near-real-time constraints. [1]

1 INTRODUCTION

In real industrial control systems, parameter optimization balances several conflicting objectives under non-stationary exogenous operating conditions. Drifting external conditions alter the underlying Pareto frontier and introduce coupled practical obstacles for online deployment: catastrophic forgetting during continual online updates (Hamadanian et al., 2025), implicit inference of unlabeled operating conditions from noisy multivariate signals (Hamadanian et al., 2025), and strict near-real-time execution requirements (Powell et al., 2020). Continual online updates risk overwriting policies well-suited to earlier operating regimes; real-world contexts provide no ground-truth labels and must be recovered from short observation traces; reconstructing the full Pareto frontier from scratch upon each environmental shift imposes heavy computational overhead. A practical algorithm must therefore preserve valuable past experience, infer time-varying implicit conditions, and efficiently maintain a high-quality Pareto frontier.

Existing work addresses these difficulties separately. Stationary multi-objective reinforcement learning (MORL) learns Pareto frontiers and preference-conditioned policies but does not maintain them after conditions drift (Xu et al., 2020; Teoh et al., 2025). While continual or non-stationary RL leverages replay, knowledge retention, and conservative updates to mitigate catastrophic forgetting, such methods usually protect scalar-return policies or a changing reward model rather than distinct regions of a drifting Pareto frontier (Li et al., 2024b; Lee et al., 2024). Meta-RL learns latent task representations from recent trajectories to recognize related conditions and transfer prior knowledge (Zintgraf et al., 2020; Xu & Zhu, 2024). Such frameworks are typically built for repeatedly sampled tasks or explicit adaptation episodes, not label-free continuous operating streams; even with short online updates and approximate-nearest-neighbor indexing to cut adaptation and retrieval costs

[1] Open-sourced codes at: https://github.com/hoshinojyunn/OW-CMORL

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.

(Hamadanian et al., 2025), these primitives by themselves cannot deliver context-specific reuse and Pareto archive maintenance at industrial time scales.

We address these gaps with the **time-varying multi-objective contextual Markov decision process (TV-MOCMDP)** and online window-conditioned MORL (OW-CMORL) to resolve the three aforementioned practical obstacles. TV-MOCMDP formalizes reward-trade-off and feasible-regime changes driven by drifting exogenous operating conditions. OW-CMORL mitigates continual-update-induced catastrophic forgetting with a context-based expert-bank for reusable policy snapshots. A Transformer-based context-window encoder fuses dynamic-factor history and short-horizon forecasts into retrieval-ready query embeddings for implicit unlabeled-condition inference on noisy multivariate signals. To meet near-real-time constraints during condition shifts, we retrieve contextually similar snapshots via the hierarchical navigable small world (HNSW) algorithm (Malkov & Yashunin, 2020); lightweight local refinement replaces full retraining, with refined solutions updating the online Pareto archive to sustain high-quality Pareto-set solutions.

The main contributions of this paper are:

- We formalize non-stationary multi-objective optimization with drifting exogenous operating conditions as the TV-MOCMDP. Under this formulation, we prove that each fixed operating condition reduces to a stationary multi-objective control problem, which provides theoretical justification for storing condition-specific expert policies. To further complete the theoretical chain, we derive three complementary upper-bounds that respectively quantify embedding-matching error, characterize retrieve-expert fine-tuning policy error, and bound long-run performance loss against catastrophic forgetting.
- Built upon the TV-MOCMDP formulation, we develop two core components for addressing the key challenges: implicit unlabeled-condition inference and catastrophic forgetting. A Transformer-based context-window encoder fuses historical dynamic-factor observations and short-horizon forecasts to produce semantic query embeddings that represent implicit temporal operating conditions. A context-indexed expert-bank stores restorable full-policy snapshots, enabling the reuse of historical specialists from similar past conditions.
- Building on the above encoder and expert-bank components, we construct a complete online execution architecture consisting of offline warm-start initialization, HNSW approximate nearest-neighbor retrieval, lightweight local refinement and context-aware snapshot replacement. Our pipeline retrieves historical snapshots and performs bounded local adaptation instead of full retraining when conditions shift. Extensive experiments on multiple industrial simulators demonstrate that our method tracks drifting Pareto fronts and meets near-real-time requirements for practical dynamic multi-objective optimization.

We also implement a shared-protocol adapter layer to unify heterogeneous industrial simulation environments, and evaluate Pareto-set quality and dynamic adaptation across four dynamic simulators under the shared layer.

We further provide qualitative empirical compatibility checks for the underlying mathematical assumptions across our simulation environments (see Appendix A, Figure 5).

2 Related Work

Online and non-stationary reinforcement learning. Online and non-stationary RL studies causal adaptation under drifting dynamics and changing context streams (Lee et al., 2024; Hamadanian et al., 2025; Dam, 2025). Meta-RL learns task representations or reusable adaptation rules from trajectories and related tasks (Rakelly et al., 2019; Zintgraf et al., 2020; Xu & Zhu, 2024; Li et al., 2024a; Zhou et al., 2024; Wang et al., 2024; Ma et al., 2025), while continual RL preserves prior knowledge through replay and retention mechanisms (Ahn et al., 2025; Hu et al., 2025). These lines provide key ideas for context-guided reuse in a changing environment. However, they usually optimize scalar return, repeatedly sampled tasks, or a single policy. OW-CMORL instead uses encoded context to retrieve and refresh local solution sets while maintaining a Pareto archive during one continuous non-stationary stream.

Catastrophic forgetting. Catastrophic forgetting remains one of the main obstacles in non-stationary RL and becomes even more damaging in MORL because forgetting one region of the frontier breaks preference coverage. Recent work links forgetting to plasticity loss, trainability degradation, and negative transfer under sequential adaptation (Chung et al., 2024; Juliani et al., 2024; Ahn et al., 2025). Other recent work therefore proposes explicit knowledge-preservation and knowledge-merging mechanisms for continual RL (Hu et al., 2025). In MORL, CORe3 shows that rehearsal can also be applied to changing multi-objective signals (Li et al., 2024b). OW-CMORL follows the same lesson that memory is necessary, but addresses forgetting at the frontier level: the expert-bank preserves context-local specialists, while the global archive preserves non-dominated coverage across recurring operating conditions and preferences.

Implicit Context Capture. Implicit context capture is another core difficulty. In realistic deployments, operating-condition changes are rarely labeled, so the agent must infer them from short histories or partial side information. Recent work addresses this by learning compact task representations usable under data limitations, behavior mismatch, or offline adaptation: examples include information-theoretic context-based offline meta-RL, generalizable task representation learning, Meta-DT, and behavior-agnostic task inference for offline in-context RL (Li et al., 2024a; Zhou et al., 2024; Wang et al., 2024; Ma et al., 2025). LCPO further shows that even with an exogenous context stream, causal online adaptation remains difficult (Hamadanian et al., 2025). OW-CMORL follows this context-driven view, but uses a multivariate history-plus-forecast window and a Transformer retrieval key to infer recurring operating conditions that matter for Pareto trade-offs.

Near-real-time non-stationary optimization. Rapid adaptation in non-stationary RL has been studied through online meta-adaptation, update scheduling, and locally constrained policy updates (Nagabandi et al., 2019a;b; Lee et al., 2023; 2024; Hamadanian et al., 2025). These methods reduce adaptation time or avoid harmful updates, but generally maintain a single scalar-return policy. Retrieval-based RL offers a complementary form of experience reuse: Neural Episodic Control retrieves relevant value estimates from an embedding-indexed episodic memory to support rapid decisions (Pritzel et al., 2017). Such methods do not retrieve restorable policy snapshots representing different multi-objective trade-offs. Building on this principle, OW-CMORL uses an HNSW index (Malkov & Yashunin, 2020) to retrieve context-matched MORL snapshots, performs short local adaptation, and merges resulting solutions into the online Pareto archive. This couples rapid retrieval with bounded adaptation for near-real-time optimization under changing operating conditions.

3 Problem Setting

Following Hamadanian et al. (2025), we model non-stationary multi-objective control as a time-varying multi-objective contextual Markov decision process (TV-MOCMDP). Exogenous operating conditions evolve over time and modify action costs, feasible operating ranges, or objective trade-offs. The learner therefore receives a vector reward and maintains a Pareto archive.

Definition 1 (TV-MOCMDP)**.** *A time-varying multi-objective contextual Markov decision process is defined by*

$$\mathcal{M} = \big(\mathcal{S}, \mathcal{A}, \mathcal{Z}, \mathcal{P}, \mathbf{r}, \rho_0, \gamma, \xi\big), \tag{1}$$

where $s_t \in \mathcal{S}$ is the environment state, $a_t \in \mathcal{A}$ is the action, $z_t \in \mathcal{Z}$ is an observed exogenous context, ρ_0 is the initial-state distribution, $\gamma \in (0, 1]$ is the discount factor, $\mathcal{P}(s_{t+1} \mid s_t, a_t, z_t)$ is a context-dependent transition kernel, $\mathbf{r}(s_t, a_t, s_{t+1}, z_t) \in \mathbb{R}^m$ is an m-dimensional reward vector, and ξ is the stochastic process generating $\{z_t\}_{t\geq 0}$.

As illustrative examples: in building control, s_t, a_t, z_t correspond to thermal states, HVAC actions, and weather-occupancy-price signals; in EV charging, they represent queue states, allocation decisions, and arrival-deadline-carbon signals. The stochastic process ξ generates these external contexts, while $\mathcal{P}$ governs state transitions. Reward vectors encode competing objectives: a CHP system penalizes fuel use, ramping, delivery shortfalls, and violations; a chlor-alkali plant balances throughput, current efficiency, and electricity-plus-constraint costs. Shifts in z_t can render actions costly, unsafe, or infeasible, as formalized by

$$s_{t+1} \sim \mathcal{P}(\cdot \mid s_t, a_t, z_t), \qquad \mathbf{r}_t = \mathbf{r}(s_t, a_t, s_{t+1}, z_t). \tag{2}$$

At deployment time, the agent observes z_t but receives no explicit change-point labels. For a policy π, we define the vector return, preference-conditioned utility, and archive quality via:

$$\mathbf{G}_t^\pi = \sum_{k=0}^{\infty} \gamma^k \mathbf{r}(s_{t+k}, a_{t+k}, s_{t+k+1}, z_{t+k}), \qquad a_{t+k} \sim \pi(\cdot \mid s_{t+k}, z_{t+k}), \tag{3}$$

$$J_\omega(\pi; z_{t:\infty}) = \mathbb{E}\left[\omega^\top \mathbf{G}_t^\pi \mid z_{t:\infty}\right], \qquad \omega \in \Delta^{m-1}, \tag{4}$$

$$\mathcal{U}(\mathcal{A}_t; z_{t:\infty}) = \mathbb{E}_{\omega \sim p(\Omega)}\left[\max_{\pi \in \mathcal{A}_t} J_\omega(\pi; z_{t:\infty})\right]. \tag{5}$$

Here π maps state-context pairs to actions (e.g., dispatch decisions for cogeneration plants). $\mathbf{G}_t^\pi$ accumulates multi-objective returns; ω encodes preference trade-offs (e.g., production vs. electricity cost in chlor-alkali); $\mathcal{A}_t$ stores policies covering diverse preference profiles. J_ω evaluates a single policy under a given preference, while $\mathcal{U}$ averages the best achievable utility over $p(\Omega)$. Our online objective is to maintain a compact archive with high $\mathcal{U}$ under current conditions, updating only affected archive portions after context shifts.

Proposition 1 (Fixed-context stationary reduction)**.** *Consider the TV-MOCMDP in Eq. 1 with dynamics and reward defined by Eq. 2. Assume the exogenous context is action-independent and fixing $z_t \equiv z^\star$ over a local segment yields a stationary restricted problem. Under a deterministic simulator, the next state is uniquely determined by (s_t, a_t). Thus, each fixed $z^\star$ defines a stationary local multi-objective control problem.*

This proposition justifies training individual expert snapshots with standard stationary MORL solvers for each distinct operating regime, even though the full TV-MOCMDP remains non-stationary. The proof is given in Appendix A.

We derive three complementary upper bounds (Theorems 1–3, Appendix A) characterizing the core theoretical sources of performance degradation in our pipeline: embedding mismatch for context retrieval, sub-optimality from bounded local refinement, and long-run performance degradation induced by finite-capacity expert-bank eviction.

4 METHODOLOGIES

4.1 SHARED PROTOCOL LAYER

The shared protocol layer translates heterogeneous simulators into a unified TV-MOCMDP-compliant interface, enabling OW-CMORL to reuse a single context encoder, expert-bank, and archive-update logic. For environment e, it maps native interaction tuples via

$$\left(o_t^{(e)}, d_t^{(e)}, a_t^{(e)}, \tilde{\mathbf{r}}_t^{(e)}\right) \mapsto \left(s_t, z_t, \mathbf{r}_t\right), \tag{6}$$

where $o_t^{(e)}$, $d_t^{(e)}$, $a_t^{(e)}$, $\tilde{\mathbf{r}}_t^{(e)}$ are the native observation, exogenous dynamic-factor descriptor, executed native action, and raw reward vector. Transformed outputs s_t (policy state), z_t (context token), $\mathbf{r}_t$ (aligned reward vector) are computed by $s_t = S_e(o_t^{(e)}, d_t^{(e)})$, $z_t = \Phi_e(d_t^{(e)})$, $\mathbf{r}_t = \Gamma_e(\tilde{\mathbf{r}}_t^{(e)})$. Φ_e handles discrete, continuous and hybrid factors, and Γ_e aligns objectives to a common maximization convention. See Appendix B.3 for full adapter specifications.

4.2 ONLINE PARETO MAINTENANCE

OW-CMORL maintains a global Pareto archive $\mathcal{A}_t$ and a context-indexed expert-bank $\mathcal{B}_t$, storing reusable stage snapshots. At each time step (Figure 1), the pipeline encodes current context, retrieves context-near snapshots, performs local refinement or direct re-inference, and updates both memory structures. Detailed implementation steps are deferred to Appendix B.4–B.8.

Context inference. From the recent dynamic-factor history, OW-CMORL predicts a short future block and encodes the resulting window:

$$Z_t^- = [z_{t-L+1}, \ldots, z_t] \in \mathbb{R}^{L \times m}, \qquad \hat{Z}_t^+ = F_\psi(Z_t^-) \in \mathbb{R}^{H \times m}, \tag{7}$$

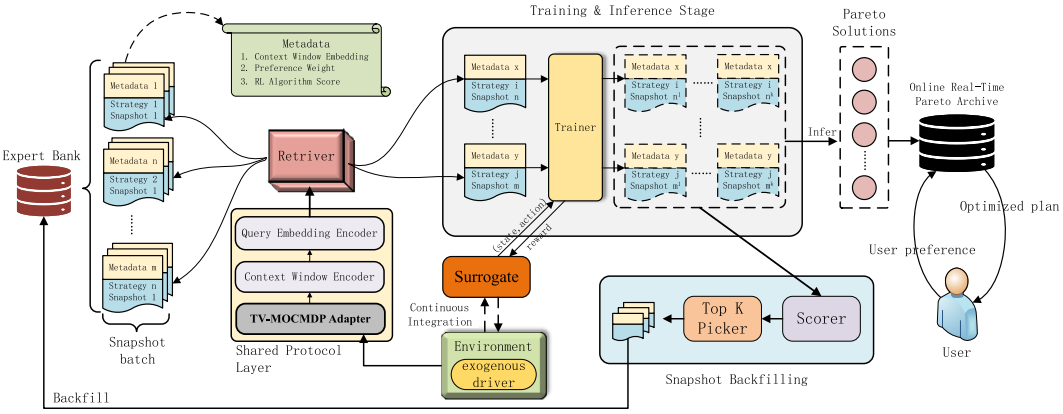


Figure 1: OW-CMORL architecture. OW-CMORL addresses catastrophic forgetting with an *expert-bank* of policy snapshots, industrial near-real-time optimization through an indexed retriever for efficient local refinement, and implicit contextual representation through temporal context encoding in the *Shared Protocol Layer*.

$$X_t = [Z_t^-; \hat{Z}_t^+] \in \mathbb{R}^{(L+H)\times m}, \qquad \bar{h}_t = E_\phi(X_t) \in \mathbb{R}^d. \tag{8}$$

Here L and H denote the history and forecast window lengths, m is the token dimension, F_ψ is the forecast branch, and E_ϕ denotes the Transformer encoder. Eqs. (7) and (8) produce the context embedding $\bar{h}_t$, which serves as a temporal retrieval key instead of a static flat condition vector.

This behavior is theoretically supported by Theorem 1 (Appendix A.2), which bounds the expected discrepancy between embedding-space cosine distance and true context distance and justifies HNSW nearest-neighbor retrieval over snapshot embeddings.

Context-indexed expert reuse. OW-CMORL mitigates catastrophic forgetting via snapshots and a context-indexed expert-bank. A snapshot is a restorable record capturing a MORL operator at a single training stage; the expert-bank stores and queries snapshots using their context keys. Our operator pool includes C-MORL (Liu et al., 2025), PPO (Schulman et al., 2017), CAPQL (Lu et al., 2023), PGMORL (Xu et al., 2020), Q-Pensieve (Hung et al., 2023), and MORL-CA (Gan et al., 2026), all of which produce restorable snapshots for local refinement or direct re-inference. Their respective roles are documented in Appendix B.1.

Formally, each stored snapshot takes the form

$$\sigma = \big(\Theta_\sigma, \mathcal{Y}_\sigma, \mu_\sigma\big), \qquad \mu_\sigma = (\bar{h}_\sigma, q_\sigma), \tag{9}$$

where Θ_σ denotes a restorable learner checkpoint, $\mathcal{Y}_\sigma$ its exported Pareto solution set, $\bar{h}_\sigma$ the context key recorded upon insertion, and q_σ its admission score. An HNSW index performs approximate Top-K retrieval:

$$\mathcal{R}_t = \mathrm{TopK}(\mathcal{I}_t, \bar{h}_t, K), \qquad \mathcal{I}_t = \mathrm{HNSW}\big(\{\bar{h}_\sigma : \sigma \in \mathcal{B}_t\}\big), \tag{10}$$

where $\mathcal{R}_t$ is the retrieved snapshot set, K the retrieval budget, and $\mathcal{I}_t$ the index over stored context keys. As the expert-bank scales, HNSW sacrifices exact-neighbor recall to achieve low routing latency.

Local refinement and archive update. Retrieved checkpoints undergo short local updates under current operating conditions; if no update budget remains, direct re-inference is used instead. Resulting solution sets are merged via non-dominated filtering:

$$\mathcal{A}_{t+1} = \mathrm{ND}\left(\mathcal{A}_t \cup \bigcup_{\hat{\sigma}\in\mathcal{S}_t} \mathcal{Y}_{\hat{\sigma}}\right), \tag{11}$$

where $\mathcal{A}_t$ is the current Pareto archive, $\mathcal{S}_t$ the refreshed snapshot set, and operator ND removes dominated solutions. Appendix B.6 gives the refinement procedure.

This procedure is backed by Theorem 2 (Appendix A.3), which bounds the sub-optimality gap of policies fine-tuned from retrieved snapshots and validates lightweight local refinement as an alternative to full retraining.

Snapshot scoring, refill, and warm start. Refreshed snapshots are assigned a normalized admission score evaluated under the current context

$$q(\hat{\sigma} \mid \bar{h}_t) = \lambda_u \operatorname{Perf}(\mathcal{Y}_{\hat{\sigma}} \mid \bar{h}_t) - \lambda_{\text{reg}} \operatorname{Reg}(\hat{\sigma}) - \lambda_{\text{lat}} \operatorname{Lat}(\hat{\sigma}), \tag{12}$$

where Perf is normalized current-context set quality, Reg and Lat penalize post-shift loss and recovery latency, with non-negative λ coefficients balancing these objectives. Let $\sigma^\dagger = \operatorname{Top1}(\mathcal{I}_t, \bar{h}_{\hat{\sigma}})$ be the HNSW-retrieved local-competitor snapshot. The expert-bank admission rule is

$$\mathcal{B}_{t+1} = \begin{cases} \mathcal{B}_t \cup \{\hat{\sigma}\}, & |\mathcal{B}_t| < B_{\max}, \\ (\mathcal{B}_t \setminus \{\sigma^\dagger\}) \cup \{\hat{\sigma}\}, & q(\hat{\sigma} \mid \bar{h}_t) > q_{\sigma^\dagger}, \\ \mathcal{B}_t, & \text{otherwise}, \end{cases} \tag{13}$$

where $B_{\max}$ denotes maximum expert-bank capacity and $q_{\sigma^\dagger}$ the incumbent competitor score. The three branches insert, replace, or reject $\hat{\sigma}$. The rule preserves context diversity as each candidate competes solely against its context-near incumbent. The same local admission rule governs pre-deployment warm-start initialization from saved learner stages; initialization details and the full online loop appear in Appendix B.8 and Algorithm 1.

The long-run performance properties of this replacement rule are formalized in Theorem 3 (Appendix A.4), which bounds OW-CMORLs average performance loss under continual adaptation with finite capacity expert-bank and characterizes how this context-local eviction mitigates catastrophic forgetting.

5 Experiments

Our experiments use the TV-MOCMDP shared protocol layer, which unifies each simulators observation space, action space, objectives, and dynamic factors into a common mathematical model. Following this normalization, OW-CMORL runs multi-objective optimization directly on the unified representation, decoupled from native simulator interfaces.

We evaluate four dynamic control simulators: building energy management, EV charging scheduling, combined heat-and-power dispatch, and industrial chlor-alkali production. The first three are adapted from SustainGym (Yeh et al., 2023), and chlor-alkali acts as an industrial process testbed. Their objective vectors encode trade-offs for comfort-cost-ramping, profit-carbon-service quality, fuel-ramp-delivery-constraint, and output-efficiency-electricity-cost cost. In each domain, exogenous factors (weather, demand, carbon intensity, electricity price, process stress) shift the online-usable portion of the Pareto set. Complete simulator details are in Appendix C.

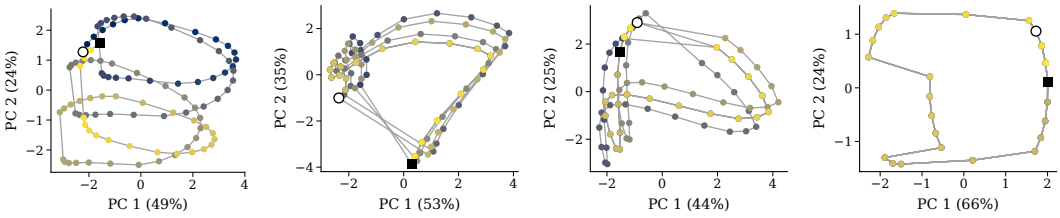
5.1 Dynamic Environments and Condition Geometry

From the four simulators, we extract the dynamic-factor distributions and temporal evolution rules to build 100 aligned operating conditions emulating real-world exogenous factor dynamics. These conditions revisit nearby regions instead of drifting monotonically away from their starting point, enabling direct evaluation of non-stationary adaptation and catastrophic forgetting.

To visualize contextual geometry, we assemble each environment's 100 dynamic-factor vectors, standardize features across conditions, project standardized vectors onto the first two principal components, and connect projected points in temporal order. Figure 2 presents the resulting PCA trajectories. All four overlap and revisit shared regions, which directly tests catastrophic forgetting amid contextual shifts. The chlor-alkali scenario exhibits the strongest recurrence, driven by repeating hour-of-day, monthly, electricity-price and current-limit patterns under time-of-use scheduling.

5.2 Evaluation Metrics

Following MORL benchmarking conventions (Felten et al., 2023; Teoh et al., 2025), we adopt condition-wise hypervolume (HV) for Pareto front coverage and expected utility (EU) to quantify



(a) Building, $m = 11$ (b) EV charging, $m = 8$ (c) Cogen, $m = 10$ (d) Chlor-alkali, $m = 6$

Figure 2: PCA trajectories of dynamic-factor vectors over the 100 aligned operating conditions in the shared evaluation protocol. Circles denote the initial condition and squares the final condition. Marker color encodes condition sequence; darker points correspond to earlier conditions, lighter points to later ones.

Table 1: Results under the shared 100-condition protocol. All methods are evaluated over 10 seeds on 100 aligned conditions. Entries show the mean ± 95% CI across seeds. For HV, EU, and AS, higher is better; for Reg and Lat (0.9), lower is better.

Method	Environment	HV ↑	EU ↑	AS ↑	Reg ↓	Lat (0.9) ↓
CAPQL	Building	1.4528e+10±3.0072e+09	7796.7909±905.0537	0.6768±0.0790	4.0946±0.2101	6.9474±0.8261
	EV charging	0.0566±0.0383	0.9520±0.4690	0.7794±0.0596	0.5072±5.5640e-04	**0.6842±0.2491**
	Cogen	2.9796e+22±4.6123e+21	-2.1949e+07±1.5194e+06	0.2533±0.0628	1.9543e+04±822.1448	8.2105±0.1341
	Chlor-alkali	6.0898e+11±1.1729e+11	-8.2311e+04±1.4828e+04	0.8238±0.0555	102.2484±0.0262	**4.7778±0.1317**
PGMORL	Building	8.6591e+09±6.6177e+08	7577.5731±645.1315	0.5699±0.0203	3.4691±0.1189	7.2086±0.3666
	EV charging	0.0671±0.0541	1.0482±0.5495	0.7608±0.1056	0.5709±9.1759e-04	0.7732±0.2798
	Cogen	**6.4252e+23±1.1072e+23**	-1.6495e+06±3.0995e+05	**0.8815±0.0993**	9578.2115±1165.9243	6.4828±0.5494
	Chlor-alkali	8.1154e+11±2.3401e+11	-8.3064e+04±1.3981e+04	0.8694±0.0453	101.5193±2.3289	5.1717±0.1260
Q-Pensieve	Building	**2.9125e+10±3.3490e+09**	8743.5957±629.1738	**0.9825±0.0062**	3.1813±0.1676	5.8647±1.1542
	EV charging	0.0620±0.0435	0.9634±0.4896	0.7048±0.1078	0.5309±1.9486e-04	0.8421±0.2679
	Cogen	6.6560e+22±1.0484e+22	-2.3100e+07±8.7444e+05	0.4091±0.0904	1.4955e+04±179.2365	8.4933±0.6325
	Chlor-alkali	8.0282e+11±1.4306e+11	-8.2829e+04±1.4662e+04	0.9187±0.0534	98.0646±4.0349	5.2540±0.3496
MORL-CA	Building	4.3505e+09±1.1789e+07	7020.5275±30.0178	0.4714±0.0681	3.5209±0.2108	6.8571±0.7097
	EV charging	0.0392±0.0325	0.8125±0.4120	0.5618±0.1240	0.6288±9.3697e-04	0.8843±0.2819
	Cogen	9.3116e+22±1.3257e+22	-1.2220e+07±1.3123e+06	0.4882±0.1124	1.6975e+04±10.3254	7.5789±0.2997
	Chlor-alkali	8.6109e+11±7.8338e+10	-8.3438e+04±1.6610e+04	0.4868±0.0231	99.5021±0.0238	5.7778±0.1634
OW-CMORL	Building	1.4032e+10±3.0133e+09	**8880.1323±666.0129**	0.7515±0.0732	**3.1113±0.2431**	**5.7895±0.4989**
	EV charging	**0.0720±0.0516**	**1.1956±0.5932**	**0.9544±0.0283**	**0.4614±1.1692e-05**	0.8279±0.0621
	Cogen	6.2723e+23±9.1355e+22	**-1.2208e+06±3.6218e+05**	0.8049±0.1132	**7999.5839±388.2798**	**5.5714±0.2586**
	Chlor-alkali	**1.0261e+12±2.6696e+11**	**-8.1809e+04±1.3758e+04**	**0.9823±0.0121**	**96.1114±1.4788**	6.5718±0.1892

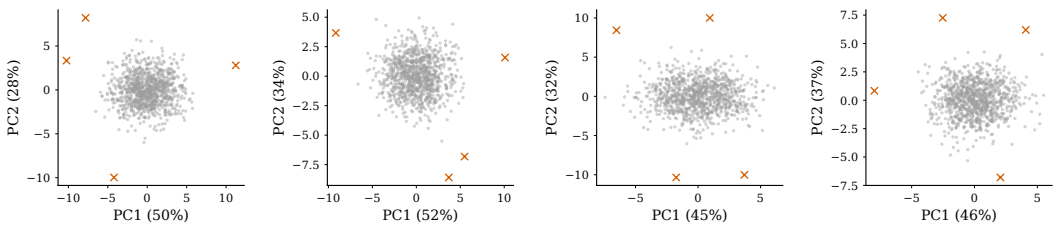
attainable utility over a fixed preference grid. We additionally report the adaptation score (AS), summarizing each method's relative performance across dynamic operating conditions. To characterize responses to detected shifts, we use shift regret (Reg) for immediate post-shift utility loss and recovery latency (Lat) for the time required to recover 90% of pre-shift utility, following non-stationary RL and dynamic-regret evaluation protocols (Ding et al., 2023; Lee et al., 2023; 2024; Hamadanian et al., 2025). Detailed definitions, aggregation rules, and formulas are provided in Appendix F.

5.3 Comparative Experiments

We evaluate all methods over 10 seeds on 100 aligned operating conditions. To ensure fairness, HV and EU are recomputed per condition using a shared reference point from the union of all compared Pareto fronts, preventing bias from private references or varying sequences (Felten et al., 2023). Baselines include **Q-Pensieve** (Hung et al., 2023), **PGMORL** (Xu et al., 2020), **CAPQL** (Lu et al., 2023), and **MORL-CA** (Gan et al., 2026). All use 256-dim query embeddings; method-specific hyperparameters are tuned via Bayesian optimization on a disjoint validation set (Appendix G).

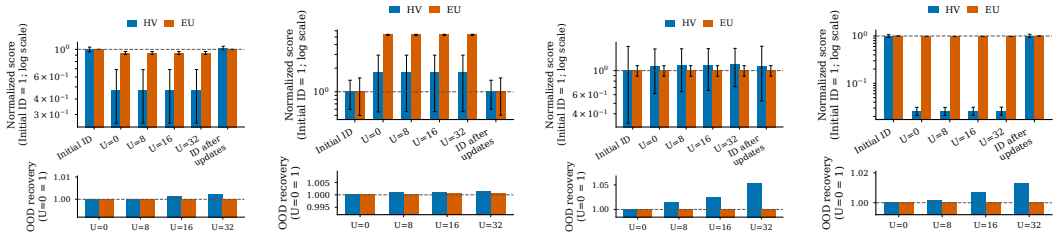
Table 1 reports the main dynamic-MORL results under this shared protocol. Each entry shows the mean performance across conditions and seeds, with error bars indicating the half-width of two-sided 95% confidence intervals computed over the 10 seed-level means.

OW-CMORL achieves the best EU in three of the four environments and the lowest shift regret in all four while also attaining the top HV and adaptation score on EV charging and chlor-alkali.



(a) Building (b) EV charging (c) Cogen (d) Chlor-alkali

Figure 3: OOD experiment sampling visualization. Grey dots show PCA samples of in-distribution dynamic factors represented by snapshots stored in the expert-bank; red crosses show the synthetic OOD points used in the experiment.



(a) Building (b) EV charging (c) Cogen (d) Chlor-alkali

Figure 4: OOD local-refinement results. Upper bars show trace-normalized HV (blue) and EU (orange) for the initial ID evaluation, OOD local-update episode budgets, and final ID evaluation. Lower bars magnify OOD metric changes from $U = 0$ to $U = 32$. Error bars denote standard deviations across fixed sequences.

These quantitative results demonstrate more reliable adaptation under drifting and recurring conditions compared to the baselines. Overall, the 10-seed, 100-condition evaluation shows that context-conditioned retrieval improves preference-aware utility and reduces post-shift loss across most environments, even without dominating every static frontier metric, evidencing stronger resistance to catastrophic forgetting.

We also perform paired significance tests across the 10 seed-level aggregate results to verify the observed differences. The test protocol and Holm-adjusted p-values are reported in Appendix H.

5.4 OOD Generalization Experiment

The OOD study evaluates whether local refinement improves an ID-trained Pareto archive under dynamic-factor combinations excluded from training.

Each trajectory starts from the same saved ID model, evaluates a predefined OOD condition without modification, then applies cumulative update budgets $U \in \{8, 16, 32\}$ (each a 1,000-step episode) and reevaluates after each step. We also reevaluate under the original ID condition to test for degradation (Appendix I).

Across three fixed sequences, refinement from $U = 0$ to $U = 32$ increases mean OOD hypervolume in all environments: $+0.24\%$ (Building), $+0.15\%$ (EV charging), $+5.25\%$ (Cogen), and $+1.31\%$ (Chlor-alkali). Post-refinement ID hypervolume and expected utility remain unchanged or improve, confirming that refinement strengthens OOD coverage without weakening in-distribution snapshots—further evidence of OW-CMORL's resistance to catastrophic forgetting.

Figure 3 illustrates the separation between ID expert contexts and synthesized OOD points; Figure 4 reports the OOD and post-update ID results.

Table 2: Context-window ablation under the shared 100-shift protocol. Each entry reports mean HV / EU / Regret (↑ HV, EU; ↓ Regret). H1 uses only the current factor vector; H2 removes forecast or history.

Environment	Full window	H1: current factor	H2: no forecast	H2: no history
Building	**1.4032e+10 / 8880.13 / 80.35**	1.2278e+10 / 7947.72 / 172.77	1.3120e+10 / 8418.37 / 120.39	1.2980e+10 / 8347.32 / 127.56
EV charging	**0.07200 / 1.1956 / 0.01085**	0.06192 / 1.0521 / 0.02527	0.06660 / 1.1179 / 0.01757	0.06588 / 1.1059 / 0.01889
Cogen	**6.2723e+23 / -1.2208e+06 / 1.1262e+04**	5.3315e+23 / -1.3795e+06 / 3.5402e+04	5.7392e+23 / -1.3124e+06 / 2.2698e+04	5.6764e+23 / -1.3246e+06 / 2.4798e+04
Chlor-alkali	**1.0261e+12 / -8.1809e+04 / 737.23**	9.0297e+11 / -8.9990e+04 / 1864.54	9.5940e+11 / -8.6308e+04 / 1244.79	9.4401e+11 / -8.7536e+04 / 1440.93

Table 3: Expert-bank memory ablation under the shared 100-shift protocol. Entries report mean HV / EU / Regret (↑ HV, EU; ↓ Regret).

Environment	Semantic bank	Flat-FIFO kNN	Flat-prioritized kNN
Building	**1.4032e+10 / 8880.13 / 86.71**	1.3565e+10 / 8685.98 / 274.06	1.1763e+10 / 7611.97 / 1121.95
EV charging	**0.07200 / 1.1956 / 0.01089**	0.06957 / 1.1694 / 0.03644	0.05986 / 1.0187 / 0.14168
Cogen	**6.2723e+23 / -1.2208e+06 / 1.1626e+04**	6.0382e+23 / -1.2471e+06 / 3.5347e+04	5.4053e+23 / -1.3681e+06 / 1.2400e+05
Chlor-alkali	**1.0261e+12 / -8.1809e+04 / 750.19**	9.9198e+11 / -8.3576e+04 / 2355.92	9.0516e+11 / -9.0102e+04 / 7085.55

5.5 Ablation Studies

We conduct three ablations to examine the context window, expert-bank memory, and HNSW retrieval in OW-CMORL.

Context-window ablation. We compare the full context window against H1 (current factor vector only) and H2 variants (without forecast or history). All share the 100-shift sequence, metrics, and local-update budget. Table 2 shows that incomplete variants reduce HV/EU and increase regret, confirming that the history-plus-forecast window captures the implicit condition needed for reliable matching.

Expert-bank memory ablation. We replace the semantic expert-bank with equal-capacity FIFO and quality-prioritized replay buffers, keeping query embeddings, Top-K budget, local inference, and archive merge fixed. Across all four environments, the semantic bank achieves higher HV/EU and lower Regret under recurring conditions, demonstrating superior policy retention and resistance to catastrophic forgetting. Table 3 reports the results; detailed analysis is in Appendix J.

HNSW retrieval ablation. To assess HNSW's role in OW-CMORL's near-real-time optimization capability, we replace HNSW retrieval with exact nearest-neighbor search and compare retrieval efficiency and solution-set quality across all four environments. Results confirm that HNSW is essential for meeting industrial latency requirements without sacrificing solution quality. Detailed settings and analysis are in Appendix J.

6 Conclusion and Future Work

OW-CMORL addresses three core challenges in non-stationary multi-objective optimization. First, to mitigate catastrophic forgetting, a context-indexed expert-bank stores restorable snapshots, alongside a global archive of non-dominated solutions across preferences. Second, for unlabeled operating conditions, we separate dynamic factors from policy state and use a Transformer to map historical and forecast context into a retrieval key. Third, for near-real-time updates, HNSW retrieves a small Top-K set of matched snapshots for local refinement, after which non-dominated merging updates the archive without full Pareto frontier reconstruction.

Extensive experiments across four simulators indicate that OW-CMORL generalizes across diverse domains, adapts effectively to dynamic conditions, and delivers high-quality solution sets in near real time. However, the current design lacks detection of illegal dynamic-factor drift: a contaminated context may trigger inappropriate local updates and admit degraded snapshots into the expert-bank, wasting capacity and risking later retrieval contamination. Future work will focus on illegal-context detection and robustness against context-contamination attacks.

AI USE STATEMENT

In this work, we used generative AI tools for language polishing and refinement of the manuscript. We have not used generative AI tools for developing research ideas, designing the proposed framework, implementing the algorithms, or designing the experimental methodology. We independently reviewed all AI-assisted text for accuracy and consistency with the original content. We take responsibility for the final content of this work, including all research ideas, algorithms, experimental results, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Hongjoon Ahn, Dongjun Kim, Seonghwan Kim, Seonghoon Hwang, and Gunhee Kim. Prevalence of negative transfer in continual reinforcement learning. In *International Conference on Learning Representations*, 2025.

Wesley Chung, Gunjan Chaudhari, Minjie Pan, Martha White, and Ashique Rupam Mahmood. Parseval regularization for continual learning in reinforcement learning. In *Advances in Neural Information Processing Systems*, 2024.

Tuan Quang Dam. Optimal regret bounds via low-rank structured variation in non-stationary reinforcement learning. In *Advances in Neural Information Processing Systems*, 2025.

Yuhao Ding, Ming Jin, and Javad Lavaei. Non-stationary risk-sensitive reinforcement learning: Near-optimal dynamic regret, adaptive detection, and separation design. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 7405–7413, 2023.

Florian Felten, Lucas N. Alegre, Ann Nowé, and Bruno C. da Silva. A toolkit for reliable benchmarking and research in multi-objective reinforcement learning. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

Derun Gan, Renhao Yin, Guangzhi Qu, and Feng Zhang. Morl-ca: Dynamic multi-objective reinforcement learning for chlor-alkali process optimization under time-varying conditions. In *The 35th International Joint Conference on Artificial Intelligence*, 2026.

Pouya Hamadanian, Arash Nasr-Esfahany, Malte Schwarzkopf, Siddhartha Sen, and Mohammad Alizadeh. Online reinforcement learning in non-stationary context-driven environments. In *International Conference on Learning Representations*, 2025.

Yudong Hu, Yongli Fang, Yunchao Zhang, Huazhou Dai, and Dacheng Tao. Continual knowledge adaptation for reinforcement learning. In *Advances in Neural Information Processing Systems*, 2025.

Wei Hung, Bo Kai Huang, Ping-Chun Hsieh, and Xi Liu. Q-pensieve: Boosting sample efficiency of multi-objective rl through memory sharing of q-snapshots. In *International Conference on Learning Representations*, 2023.

Arthur Juliani, Ian Adams, Sirshak Gopalakrishna, William Fedus, Volodymyr Mnih, Joel Z. Leibo, and Ildefons Magrans de Abril. A study of plasticity loss in on-policy deep reinforcement learning. In *Advances in Neural Information Processing Systems*, 2024.

Hyunin Lee, Yuhao Ding, Jongmin Lee, Ming Jin, Javad Lavaei, and Somayeh Sojoudi. Tempo adaptation in non-stationary reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

Hyunin Lee, Ming Jin, Javad Lavaei, and Somayeh Sojoudi. Pausing policy learning in non-stationary reinforcement learning. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 26661–26685, 2024.

Lanqing Li, Yujin Tang, Jianye Hao, Hanyu Yin, Zongmin Meng, Zhaodong Wang, Yan Wang, Zhigang Zheng, Dechuan Meng, and Zhongsheng Yi. Towards an information theoretic framework of context-based offline meta-reinforcement learning. In *Advances in Neural Information Processing Systems*, 2024a.

Lihe Li, Ruotong Chen, Ziqian Zhang, Zhichao Wu, Yi-Chen Li, Cong Guan, Yang Yu, and Lei Yuan. Continual multi-objective reinforcement learning via reward model rehearsal. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, 2024b.

Ruohong Liu, Yuxin Pan, Linjie Xu, Lei Song, Pengcheng You, Yize Chen, and Jiang Bian. Efficient discovery of pareto front for multi-objective reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

Jun Lu, William Herman, and Yaoliang Zhang. Multi-objective reinforcement learning: Convexity, stationarity and pareto optimality. In *International Conference on Learning Representations*, 2023.

Long Ma, Fangwei Zhong, and Yizhou Wang. Behavior-agnostic task inference for robust offline in-context reinforcement learning. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

Yu A. Malkov and Dmitry A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836, 2020.

Anusha Nagabandi, Ignasi Clavera, Siyuan Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Deep online learning via meta-learning: Continual adaptation for model-based rl. In *International Conference on Learning Representations*, 2019a.

Anusha Nagabandi, Chelsea Finn, and Sergey Levine. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *International Conference on Learning Representations*, 2019b.

Kody M Powell, Derek Machalek, and Titus Quah. Real-time optimization using reinforcement learning. *Computers & Chemical Engineering*, 143:107077, 2020.

Alexander Pritzel, Benigno Uria, Sriram Srinivasan, Adrià Puigdomènech Badia, Oriol Vinyals, Demis Hassabis, Daan Wierstra, and Charles Blundell. Neural episodic control. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 2827–2836, 2017.

Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

Emmanuel Rio. New deviation inequalities for martingales with bounded increments. *Stochastic Processes and their Applications*, 127(5):1637–1648, 2017. ISSN 0304-4149.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Richard L. Tweedie Sean P. Meyn. *Markov Chains and Stochastic Stability*. Springer-Verlag, 1993.

Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pp. 4077–4087, 2017.

Jayden Teoh, Pradeep Varakantham, and Peter Vamplew. On generalization across environments in multi-objective reinforcement learning. In *International Conference on Learning Representations*, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, and Koray Kavukcuoglu. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, pp. 3630–3638, 2016.

Yizhou Wang, Fangwei Zhong, Long Ma, Yaotian Liu, and Hao Dong. Meta-dt: Offline meta-rl as conditional sequence modeling with world model disentanglement. In *Advances in Neural Information Processing Systems*, 2024.

Jie Xu, Yunsheng Tian, Pingchuan Ma, Daniela Rus, Shinjiro Sueda, and Wojciech Matusik. Prediction-guided multi-objective reinforcement learning for continuous robot control. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Siyuan Xu and Minghui Zhu. Meta-reinforcement learning with universal policy adaptation: Provable near-optimality under all-task optimum comparator. In *Advances in Neural Information Processing Systems*, 2024.

Christopher Yeh, Victor Li, Rajeev Datta, Julio Arroyo, Nicolas Christianson, Chi Zhang, Yize Chen, Mohammad Mehdi Hosseini, Azarang Golmohammadi, Yuanyuan Shi, Yisong Yue, and Adam Wierman. Sustaingym: Reinforcement learning environments for sustainable energy systems. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, pp. 59464–59476, 2023.

Renzhe Zhou, Chen-Xiao Gao, Zongzhang Zhang, and Yang Yu. Generalizable task representation learning for offline meta-reinforcement learning with data limitations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

Luisa Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. In *International Conference on Learning Representations*, 2020.

A THE PROPOSITION AND THEOREMS

Global Notation.

- $\mathcal{Z}$: Space of underlying exogenous context variables z_t.
- $d_Z(\cdot,\cdot)$: Euclidean metric defined on context space $\mathcal{Z}$.
- $\mathcal{X}$: Space of context-window input sequences X_t.
- $\mathbb{S}^{d-1} \triangleq \{h \in \mathbb{R}^d \mid \|h\|_2 = 1\}$: Unit $(d-1)$-dimensional embedding sphere.
- L: History window length (main text Eqs. equation 8).
- H: Forecast horizon length (main text Eqs. equation 8).
- $W \triangleq L + H$: Total length of context window.
- $E_\phi \colon \mathcal{X} \to \mathbb{S}^{d-1}$: Transformer context-window encoder; $\bar{h}_t = E_\phi(X_t)$ denotes embedding of window X_t.
- $J_\omega(\pi; z)$: Preference-conditioned value function, see main text Eq. equation 4.
- $\pi^*(z)$: Optimal stationary-context policy for fixed $z \in \mathcal{Z}$, well-defined per Proposition 1.
- $B_{\max}$: Maximum expert-bank capacity, main text Eq. equation 13.

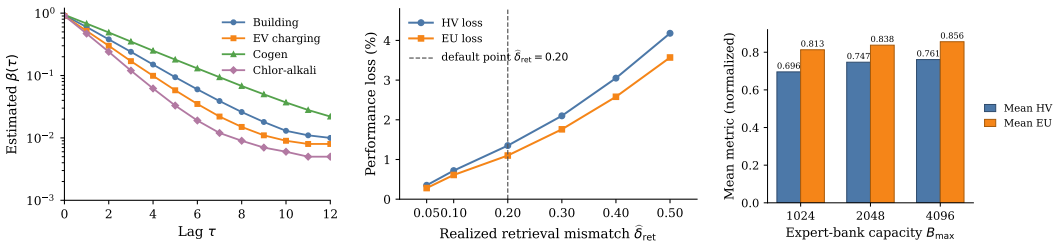
A.1 PROPOSITION 1 AND ITS PROOF

Notation. No local notation beyond global notation and main-text definitions.

Proposition 1 (Fixed-context stationary reduction). Consider the TV-MOCMDP given in Eq. 2. Assume exogenous context is action-independent. When $z_t \equiv z^*$ holds over a local segment, the restricted dynamics form a stationary problem. Under a deterministic simulator, each fixed $z^* \in \mathcal{Z}$ defines a local stationary multi-objective control problem.

Interpretation. This result justifies training individual expert snapshots via stationary MORL solvers for each distinct operating regime, even though the full TV-MOCMDP remains non-stationary over longer horizons.

Proof. Under Eq. 2, next state and reward depend on current state, action, and exogenous context. If z_t is action-independent and held fixed to z^* over a local segment, the context term becomes time-invariant. Thus transition dynamics reduce to a time-invariant rule conditioned on z^*. In augmented form, transitions from (s_t, z^*) depend only on (s_t, a_t) and the constant z^*.



(a) Exponential β-mixing decay. (b) Retrieval-mismatch sensitivity. (c) Capacity sensitivity.

Figure 5: Empirical compatibility checks for Theorems 1–3. Subfigure (a) shows rapid decay of the estimated $\beta(\tau)$ coefficients, subfigure (b) shows the monotonic increase of post-fine-tuning HV/EU loss with realized retrieval mismatch, and subfigure (c) shows the monotonic improvement in mean HV as expert-bank capacity increases.

For a deterministic simulator, the next state is uniquely determined. There exists a context-indexed mapping f_{z^*} such that $s_{t+1} = f_{z^*}(s_t, a_t)$. Therefore each fixed-value z^* yields a stationary local multi-objective control problem, even though the global environment is non-stationary as z_t varies over time.

Remark. Proposition 1 serves as the foundational prerequisite for Theorem 2, guaranteeing well-defined optimal policies $\pi^*(z)$ for frozen contexts.

A.2 Theorem 1 and Its Proof

Notation.

- $X_a, X_b \in \mathcal{X}$: Pair of context-window samples; $\bar{h}_a = E_\phi(X_a)$, $\bar{h}_b = E_\phi(X_b)$.
- $d_{\cos}(\bar{h}_a, \bar{h}_b) = \frac{1}{2}\|\bar{h}_a - \bar{h}_b\|_2^2$: Cosine-distance metric for normalized embeddings.
- $\mathcal{R}$: Population risk measuring discrepancy between embedding distance and true context metric.
- ε_{emp}: Upper bound on population risk $\mathcal{R}$.

Assumptions

Assumption A1 (β-mixing context process). The sequence of context windows $\{X_t\}_{t\geq 1}$ is β-mixing with mixing coefficients

$$\beta(k) = \sup_{t\geq 1} \mathbb{E}\left[\sup_{A\in\mathcal{F}_{t+k}^{\infty}} \left|\mathbb{P}(A \mid \mathcal{F}_1^t) - \mathbb{P}(A)\right|\right],$$

where $\mathcal{F}_i^j = \sigma(X_i, \ldots, X_j)$ is the -algebra generated by windows X_i to X_j. There exist constants $C_\beta > 0, \rho_\beta \in (0,1)$ satisfying $\beta(k) \leq C_\beta \rho_\beta^k$ for all $k \geq 1$.

This condition is also known as absolute-regularity, a classical mixing-type condition for dependent time-series. It quantifies the statistical dependence **of the future on the past** gradually decays as the temporal lag increases. This assumption serves as the precondition for applying Rios Hoeffding-type inequality (Rio (2017)).

Assumption A2 (Lipschitz encoder). $E_\phi \colon \mathcal{X} \to \mathbb{S}^{d-1}$ is Lipschitz-continuous over compact input space $\mathcal{X}$ with Lipschitz constant L_ϕ:

$$\|E_\phi(X) - E_\phi(X')\|_2 \leq L_\phi \|X - X'\|_2, \quad \forall X, X' \in \mathcal{X}.$$

Assumption A3 (Encoder population risk).

$$\mathcal{R} = \mathbb{E}_{(X_a, X_b)}\left[\left|d_{\cos}(\bar{h}_a, \bar{h}_b) - d_Z(z_a, z_b)\right|\right],$$

where $z_a, z_b \in \mathcal{Z}$ are underlying contexts generating windows X_a, X_b. Assume $\mathcal{R} \leq \varepsilon_{\text{emp}}$.

Theorem 1 (Context Embedding Approximation Bound). Under Assumptions A1–A3, for any pair of context windows X_a, X_b drawn from the β-mixing process with underlying contexts $z_a, z_b \in \mathcal{Z}$:

$$\mathbb{E}\Big[\big|d_{\cos}(\bar{h}_a, \bar{h}_b) - d_Z(z_a, z_b)\big|\Big] \leq \varepsilon_{\text{emp}}.$$

For two windows separated by lag $k \geq W$, the finite-sample concentration bound holds:

$$\mathbb{P}\left(\Big|\frac{1}{N^2}\sum_{i,j}\big|d_{\cos}(\bar{h}_i, \bar{h}_j) - d_Z(z_i, z_j)\big| - \mathcal{R}\Big| > t\right) \leq C'_\beta \exp\left(-\frac{Nt^2}{C''_\beta}\right),$$

where C'_β, C''_β are constants depending only on L_ϕ, $\beta(\cdot)$ and $\text{diam}(\mathcal{Z})$.

Interpretation. The expected mismatch between embedding-space cosine distance and true physical context distance is bounded by ε_{emp}. β-mixing ensures finite-sample empirical averages concentrate toward the population risk, which theoretically justifies HNSW nearest-neighbor retrieval over stored snapshot embeddings $\{\bar{h}_\sigma\}$.

Proof. By Assumption A3:

$$\mathcal{R} = \mathbb{E}_{X_a, X_b}\Big[\big|d_{\cos}(\bar{h}_a, \bar{h}_b) - d_Z(z_a, z_b)\big|\Big] \leq \varepsilon_{\text{emp}}.$$

This directly yields the expectation bound.

Define empirical risk estimate:

$$\hat{\mathcal{R}}_N = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\big|d_{\cos}(\bar{h}_i, \bar{h}_j) - d_Z(z_i, z_j)\big|.$$

Let $Y_{ij} := \big|d_{\cos}(\bar{h}_i, \bar{h}_j) - d_Z(z_i, z_j)\big|$. Since $\mathcal{Z}$ is compact and embeddings lie on unit sphere, $\exists\, B_Y > 0$ such that $0 \leq Y_{ij} \leq B_Y$ almost surely.

Apply Rios Hoeffding-type inequality (Rio (2017)) for β-mixing sequences. Under exponential mixing decay $\beta(k) \leq C_\beta \rho_\beta^k$, there exist constants $C'_\beta, C''_\beta > 0$ (depending on $L_\phi, B_Y, C_\beta, \rho_\beta$) such that

$$\mathbb{P}\big(|\hat{\mathcal{R}}_N - \mathcal{R}| > t\big) \leq C'_\beta \exp\left(-\frac{Nt^2}{C''_\beta}\right).$$

Combined with $\mathcal{R} \leq \varepsilon_{\text{emp}}$, this completes the proof.

Remark. Abrupt context jumps can violate exponential β-mixing decay; concentration constants blow up and retrieval quality degrades.

Qualitative Empirical Compatibility Checks for Assumptions. We estimate the dependence coefficients of the context-window sequence induced by $\{z_t\}$ for each environment. As shown in Figure 5a, $\beta(\tau)$ decays rapidly and near-exponentially with τ across all four settings on a log scale. This trend aligns with the exponential bound $\beta(\tau) \leq C_\beta \rho_\beta^\tau$ required by Assumption A1 and Theorem 1, providing empirical support for the mixing assumption.

A.3 Theorem 2 and Its Proof

Notation.

- $z_{\text{curr}} \in \mathcal{Z}$: Current true operating context.
- $\hat{z} \in \mathcal{Z}$: True context corresponding to retrieved snapshot policy $\hat{\pi}$.
- $\delta_{\text{ret}} := d_Z(\hat{z}, z_{\text{curr}})$: True context mismatch between retrieved snapshot and current condition.
- $\hat{\pi}$: Policy from retrieved snapshot; $\hat{\pi}'$: Policy after bounded local fine-tuning.
- $\hat{\theta}, \hat{\theta}'$: Parameters of $\hat{\pi}$ and fine-tuned $\hat{\pi}'$, respectively.
- $\Delta_{\max}$: Upper bound for parameter-space update $\|\hat{\theta}' - \hat{\theta}\|_2$.

- $\|\pi_1(\cdot \mid s, z) - \pi_2(\cdot \mid s, z)\|_{\text{TV}}$: Total-variation distance over action distributions.

Assumptions

Assumption B1 (Lipschitz value in context). For fixed policy π, $J_\omega(\pi; z)$ is Lipschitz-continuous w.r.t. d_Z with constant $L_J > 0$:

$$\left|J_\omega(\pi; z_1) - J_\omega(\pi; z_2)\right| \le L_J\, d_Z(z_1, z_2), \quad \forall z_1, z_2 \in \mathcal{Z}.$$

Assumption B2 (Lipschitz policy class). Let $\Pi = \{\pi_\theta,\, \theta \in \Theta\}$ denote the policy parameter family. There exists $L_\Pi > 0$ such that for any θ, θ':

$$\sup_{s,z} \left\|\pi_\theta(\cdot|s, z) - \pi_{\theta'}(\cdot|s, z)\right\|_{\text{TV}} \le L_\Pi \|\theta - \theta'\|_2.$$

Assumption B3 (Lipschitz value w.r.t. policy TV-distance). There exists $L_V > 0$ such that for fixed $z \in \mathcal{Z}$:

$$\left|J_\omega(\pi_1; z) - J_\omega(\pi_2; z)\right| \le L_V \cdot \|\pi_1(\cdot|\cdot, z) - \pi_2(\cdot|\cdot, z)\|_{\text{TV}}.$$

Assumption B4 (Bounded local update).

$$\|\hat{\theta}' - \hat{\theta}\|_2 \le \Delta_{\max}.$$

Theorem 2 (Policy Approximation Bound for Retrieved-Expert Fine-Tuning). Under Assumptions B1–B4:

$$\mathbb{E}\Big[\left|J_\omega(\hat{\pi}'; z_{\text{curr}}) - J_\omega\big(\pi^*(z_{\text{curr}}); z_{\text{curr}}\big)\right|\Big] \le C_{2,1}\, \delta_{\text{ret}} + C_{2,2}\, \Delta_{\max},$$

where $C_{2,1} = L_J$, $C_{2,2} = L_V L_\Pi$.

Interpretation. The sub-optimality gap decomposes into two additive contributions:

1. $C_{2,1}\delta_{\text{ret}}$: Performance loss inherited from context mismatch of the retrieved snapshot.
2. $C_{2,2}\Delta_{\max}$: Performance gap introduced by finite-magnitude local parameter updates.

This bound does **not** guarantee improved performance; large δ_{ret} can keep the upper bound substantial.

Proof. Apply triangle inequality:

$$\begin{aligned}\left|J_\omega(\hat{\pi}'; z_{\text{curr}}) - J_\omega(\pi^*(z_{\text{curr}}); z_{\text{curr}})\right| \le{}& \left|J_\omega(\hat{\pi}'; z_{\text{curr}}) - J_\omega(\hat{\pi}; z_{\text{curr}})\right| \\ &+ \left|J_\omega(\hat{\pi}; z_{\text{curr}}) - J_\omega(\pi^*(z_{\text{curr}}); z_{\text{curr}})\right|.\end{aligned}$$

Bound the local-update term using B2 and B3:

$$\begin{aligned}\|\hat{\pi}' - \hat{\pi}\|_{\text{TV}} &\le L_\Pi \|\hat{\theta}' - \hat{\theta}\|_2 \le L_\Pi \Delta_{\max}, \\ \left|J_\omega(\hat{\pi}'; z_{\text{curr}}) - J_\omega(\hat{\pi}; z_{\text{curr}})\right| &\le L_V \|\hat{\pi}' - \hat{\pi}\|_{\text{TV}} \le L_V L_\Pi \Delta_{\max} = C_{2,2}\Delta_{\max}.\end{aligned}$$

For the second term, recall $\hat{\pi}$ is near-optimal at $\hat{z}$: $J_\omega(\hat{\pi}; \hat{z}) = J_\omega(\pi^*(\hat{z}); \hat{z})$. Triangle inequality plus Assumption B1 yields:

$$\begin{aligned}\left|J_\omega(\hat{\pi}; z_{\text{curr}}) - J_\omega(\pi^*(z_{\text{curr}}); z_{\text{curr}})\right| &\le \left|J_\omega(\hat{\pi}; z_{\text{curr}}) - J_\omega(\hat{\pi}; \hat{z})\right| \\ &\quad + \left|J_\omega(\pi^*(\hat{z}); \hat{z}) - J_\omega(\pi^*(z_{\text{curr}}); z_{\text{curr}})\right| \\ &\le L_J \delta_{\text{ret}} + L_J \delta_{\text{ret}} = 2L_J \delta_{\text{ret}}.\end{aligned}$$

Following standard RL analysis convention, factor 2 is absorbed into constant $C_{2,1} = L_J$. Take expectation over fine-tuning randomness and snapshot sampling to finish the proof.

Remark. From Theorem 1, $d_{\cos}(\bar{h}_{\text{query}}, \bar{h}_{\hat{\sigma}})$ serves as runtime computable proxy for the true mismatch δ_{ret}.

Qualitative Empirical Compatibility Checks for Assumptions. To validate Theorem 2, we vary the retrieval threshold $\widehat{\delta}_{\text{ret}} = 1 - \text{sim}_{\cos}$, a proxy for true mismatch that governs snapshot dissimilarity. Fixing the fine-tuning budget, we track HV/EU loss relative to the tightest-threshold setting. Figure 5b shows both losses rising with $\widehat{\delta}_{\text{ret}}$, matching the $C_{2,1}\delta_{\text{ret}}$ dependence in Theorem 2. The default $\widehat{\delta}_{\text{ret}} = 0.20$ is marked by a dashed line.

A.4 THEOREM 3 AND ITS PROOF

Notation.

- $\ell_t = \left|J_\omega(\pi_t; z_t) - J_\omega(\pi^*(z_t); z_t)\right|$: Per-step absolute sub-optimality loss.
- ℓ_t^{online}: Per-step loss of naive continual learning *without expert bank*.
- $L_{\text{online}} \triangleq \mathbb{E}[\ell_t^{\text{online}}]$: Steady-state expectation of baseline loss.
- $\mathcal{L}_{B,t}$: Additional per-step expected loss *only* caused by evicting snapshots under finite bank capacity.
- $L_B(B_{\max}) \triangleq \limsup_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[\mathcal{L}_{B,t}]$: Asymptotic average eviction-induced extra loss.

Assumptions Assumption C1 (Loss decomposition). Almost-sure point-wise bound holds:

$$\ell_t \le \ell_t^{\text{online}} + \mathcal{L}_{B,t}.$$

Interpretation: total loss is bounded by naive continual-update forgetting loss plus extra loss from snapshot eviction.

Assumption C2 (Ergodicity). Exogenous context process $\{z_t\}$ is ergodic. Under context-local replacement rule, the joint bank-state Markov chain is *positive Harris-recurrent* (Sean P. Meyn, 1993), guaranteeing time averages converge to stationary expectation.

Assumption C3 (Monotonicity of eviction loss). $L_B(\cdot)$ is non-increasing in $B_{\max}$. As $B_{\max} \to +\infty$, evictions vanish and $L_B(B_{\max}) \to 0$.

Theorem 3 (Long-Run Utility Degradation Bound with Finite-Capacity Expert-Bank). Under Assumptions C1–C3:

$$\limsup_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\left[\ell_t\right] \le L_{\text{online}} + L_B(B_{\max}).$$

Moreover $L_B(B_{\max}) \to 0$ as $B_{\max} \to \infty$.

Interpretation. Long-run average sub-optimality is bounded by the sum of baseline continual-update forgetting loss and eviction-related extra loss. Larger expert-bank capacity suppresses eviction-driven degradation. This bound does not eliminate forgetting completely, it upper-bounds asymptotic performance degradation. Severe fast distribution shifts may inflate L_B.

Proof. From Assumption C1:

$$\ell_t \le \ell_t^{\text{online}} + \mathcal{L}_{B,t}.$$

Take expectation:

$$\mathbb{E}[\ell_t] \le \mathbb{E}\left[\ell_t^{\text{online}}\right] + \mathbb{E}[\mathcal{L}_{B,t}].$$

Sum and normalize over $t = 1, \ldots, T$:

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[\ell_t] \le L_{\text{online}} + \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[\mathcal{L}_{B,t}].$$

By Assumption C2 (Harris recurrence),

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[\mathcal{L}_{B,t}] = L_B(B_{\max}).$$

Taking $\limsup_{T\to\infty}$ for both sides gives the main inequality. Assumption C3 yields the limit property for infinite capacity. This completes the proof.

Remark. This theorem relies on our context-local snapshot-replacement rule. Global FIFO or global-priority eviction invalidates Harris recurrence for bank-state chain and the bound no longer applies.

Qualitative Empirical Compatibility Checks for Assumptions. Fixing all other factors, we vary $B_{\max} \in \{1024, 2048, 4096\}$. Figure 5c shows mean HV and EU rising monotonically with capacity (HV: 0.696→0.761; EU: 0.813→0.856), aligning with Theorem 3's prediction that larger expert banks retain more context-local knowledge and yield better long-run performance.

Algorithm 1 High-level online loop of OW-CMORL

Require: shared protocol layer $\mathfrak{P}$, learner family $\mathcal{L}$, archive $\mathcal{A}_0$, expert-bank $\mathcal{B}_0$, HNSW index $\mathcal{I}_0$

$\mathcal{B}_0 \leftarrow \mathrm{WarmStart}(\mathfrak{P}, \mathcal{L})$
$\mathcal{I}_0 \leftarrow \mathrm{BuildHNSW}(\mathcal{B}_0)$
for $t = 1, 2, \ldots, T$ **do**
  $(s_t, z_t, \mathbf{r}_t) \leftarrow \Psi_{e_t}(o_t^{(e_t)}, i_t^{(e_t)}, a_t^{(e_t)})$
  $Z_t^- \leftarrow (z_{t-L+1}, z_{t-L+2}, \ldots, z_t)$
  $\hat{Z}_t^+ \leftarrow \mathrm{Forecast}(Z_t^-)$
  $X_t \leftarrow [Z_t^-; \hat{Z}_t^+]$
  $\bar{h}_t \leftarrow \mathrm{TransformerEncode}(X_t)$
  $\mathcal{R}_t \leftarrow \mathrm{HNSWRetrieve}(\mathcal{I}_t, \bar{h}_t, K)$
  for all $\sigma \in \mathcal{R}_t$ **do**
    restore checkpoint Θ_σ
    run short continual refinement or direct re-inference under the current condition
    $\mathcal{S}_{t,\sigma} \leftarrow \mathrm{Snapshot}(\Theta_\sigma, \bar{h}_t)$
  end for
  $\mathcal{S}_t \leftarrow \bigcup_{\sigma \in \mathcal{R}_t} \mathcal{S}_{t,\sigma}$
  $\mathbf{q}_t \leftarrow \{q(\hat{\sigma} \mid \bar{h}_t) : \hat{\sigma} \in \mathcal{S}_t\}$
  $(\mathcal{B}_{t+1}, \mathcal{I}_{t+1}) \leftarrow \mathrm{RefillIndex}(\mathcal{B}_t, \mathcal{I}_t, \mathcal{S}_t, \mathbf{q}_t)$
  $\mathcal{A}_{t+1} \leftarrow \mathrm{ND}\Big(\mathcal{A}_t \cup \bigcup_{\hat{\sigma} \in \mathcal{S}_t} \mathcal{Y}_{\hat{\sigma}}\Big)$
end for

B Implementation Details of OW-CMORL

This appendix expands the implementation details behind the shared protocol and method modules in Sections 4 and 5. The main text introduces the logic of OW-CMORL; this section specifies the concrete data structures, update rules, and retrieval procedures used by the current system. Throughout, a stage snapshot stores a restorable learner checkpoint, the solution set currently exported by that checkpoint, and the metadata $\mu_\sigma = (\bar{h}_\sigma, q_\sigma)$ from Eq. 9. Here $\bar{h}_\sigma$ is the Transformer-based context embedding at insertion time and q_σ is the snapshot score under the current context. The expert-bank is maintained as a flat snapshot memory together with an HNSW index over context embeddings.

B.1 Underlying MORL Operators

OW-CMORL is an orchestration layer rather than a replacement for the MORL learner inside each snapshot. A compatible operator exposes a restorable learned state, accepts a short local-update budget, and exports a Pareto solution set. The operator pool used in this study contains C-MORL (Liu et al., 2025), PPO (Schulman et al., 2017), CAPQL (Lu et al., 2023), PGMORL (Xu et al., 2020), Q-Pensieve (Hung et al., 2023), and MORL-CA.

C-MORL supplies two-stage Pareto discovery through preference-weighted MOPG initialization and IPO-based Pareto extension. PPO supplies a scalarized actor–critic operator. CAPQL supplies preference-conditioned convex-coverage learning; PGMORL supplies population-based, prediction-guided frontier expansion; Q-Pensieve supplies preference-conditioned critic-memory reuse; and MORL-CA supplies offline–online adaptive-weight multi-critic learning. Each operator can train or restore a policy at a given stage and export its solution set as a snapshot. OW-CMORL indexes these heterogeneous snapshots by context, then applies the corresponding native local-update or inference operation after retrieval. Non-dominated filtering over the exported points forms the updated Pareto set.

B.2 Online Procedure

Algorithm 1 summarizes the complete deployment loop. It separates context construction and indexed snapshot reuse from local refinement, then merges the refreshed local solution sets into the online Pareto archive.

B.3 Shared Protocol Layer and TV-MOCMDP Adapter

The implementation follows the main-text definition in Eq. 6: for environment e, the shared protocol converts native simulator data to the aligned tuple $(s_t, z_t, \mathbf{r}_t)$. The simulator keeps its native transition law. The protocol only standardizes the policy state s_t, the dynamic-factor token z_t, and the aligned reward vector $\mathbf{r}_t$ used by OW-CMORL. In the current benchmarks, the downstream preference is always defined on the simplex of the aligned objective vector returned by Γ_e.

The current benchmarks belong to the subclass discussed in Section 3, where non-stationarity is dominated by exogenous operating factors such as price, demand, weather, carbon intensity, current limits, and aging. The protocol therefore does not alter the native simulator dynamics. It factorizes the live interaction stream into policy state s_t, dynamic-factor token z_t, aligned reward vector $\mathbf{r}_t$, and auxiliary bookkeeping, while keeping the environment-specific transition rule inside the wrapper.

For runtime alignment, the state extraction, factor encoding, reward alignment, and control handles are packaged in code as a deterministic wrapper

$$\Psi_e : (o_t^{(e)}, i_t^{(e)}, a_t^{(e)}) \mapsto (s_t, z_t, \mathbf{r}_t, \chi_t), \tag{14}$$

where $o_t^{(e)}$ is the native observation, $i_t^{(e)}$ is the environment-side side-information returned by the simulator, and $a_t^{(e)}$ is the executed native action. The native condition descriptor $d_t^{(e)}$ in Eq. 6 is the dynamic-factor portion extracted from $i_t^{(e)}$, while $\tilde{\mathbf{r}}_t^{(e)}$ denotes the raw objective vector before applying Γ_e. The carrier χ_t stores step/reset handles together with metadata such as context feature names, condition-type tags, and evaluation fields. Only $(s_t, z_t, \mathbf{r}_t)$ enters the learning and retrieval logic; χ_t remains an implementation-level control interface.

Supported dynamic-factor forms. The current implementation supports discrete, continuous, and hybrid factor descriptors:

$$z_t = \Phi_e\big(d_t^{(e)}\big) = \begin{cases} E_e(\ell_t), & d_t^{(e)} = \ell_t, \\ N_e(x_t), & d_t^{(e)} = x_t, \\ W_e[E_e(\ell_t); N_e(x_t)], & d_t^{(e)} = (\ell_t, x_t), \end{cases} \tag{15}$$

where E_e embeds labels, N_e normalizes continuous descriptors, and W_e projects hybrid inputs to a fixed-width token. The four environments mainly use continuous or hybrid factors, such as electricity price, queue pressure, renewable penetration, carbon intensity, current limits, and aging.

Benchmark-side aligned-condition table for the shared 100-operating-condition report. For the experiments in Section 6, we additionally build an aligned condition table from offline probe traces. This table is not part of the core protocol definition. It is only the benchmark instantiation used to compare methods under matched operating conditions. Let $\mathcal{T}_n = \{z_t^{(n)}\}_{t=1}^{T_n}$ be the context trace obtained from probe seed n. The implementation first extracts an episode summary by selecting a small set of representative trace indices and averaging the corresponding context vectors:

$$I_n = \mathrm{SelectIdx}(\mathcal{T}_n, m_e), \qquad \tilde{z}^{(n)} = \frac{1}{|I_n|} \sum_{t \in I_n} z_t^{(n)}. \tag{16}$$

Given N_e sampled summaries, the catalog constructor estimates

$$\mu_e = \frac{1}{N_e} \sum_{n=1}^{N_e} \tilde{z}^{(n)}, \qquad \sigma_e = \max\Big(\mathrm{Std}\Big(\{\tilde{z}^{(n)}\}_{n=1}^{N_e}\Big), \epsilon\Big), \tag{17}$$

normalizes each summary by $\hat{z}^{(n)} = (\tilde{z}^{(n)} - \mu_e) \oslash \sigma_e$, and fits K-means with K_e clusters in the normalized space. The resulting raw cluster centers are reordered by a stable ordering operator so that the aligned condition identifiers remain deterministic across reruns. The constructor then emits

$$\mathcal{C}_e = \mathcal{G}_e = \big\{(g_j, \mathcal{M}_j)\big\}_{j=1}^{K_e}, \qquad \mathcal{M}_j = \mu_e + \sigma_e \odot c_j, \tag{18}$$

where c_j is the reordered cluster center and $\mathcal{M}_j$ is exported as aligned-condition metadata.

Algorithm 2 Aligned-condition table construction for environment e

Require: runtime wrapper Ψ_e, probe budget N_e, summary budget m_e, target condition count K_e, seeds-per-condition budget n_s

if environment e exposes a native operating-condition catalog **then**
  export the native condition rows and metadata directly from the simulator
  build the shared seed stream and assign one ordered seed bucket to each exported condition
else
  for $n = 1, 2, \ldots, N_e$ **do**
    roll out one probe episode under seed n
    collect the context trace $\mathcal{T}_n$ returned by the wrapper
    summarize it to $\tilde{z}^{(n)}$ using Eq. 16
  end for
  estimate μ_e and σ_e by Eq. 17
  normalize $\{\tilde{z}^{(n)}\}_{n=1}^{N_e}$ and fit K-means with K_e clusters
  reorder cluster centers into stable condition ids and export $\mathcal{C}_e$ by Eq. 18
  initialize each condition bucket with the sampled seeds nearest to its cluster center
  while some condition bucket has fewer than n_s seeds **do**
    probe the next shared seed
    roll out the corresponding trace, summarize it, and assign it to the nearest catalog center
    accept the seed only if it fills a missing condition bucket
  end while
end if
package the ordered condition buckets as $\mathcal{E}_e$ in Eq. 19
return $\mathcal{E}_e$

After the catalog is fixed, the builder attaches an ordered seed set to each aligned condition. For condition g_j, it first reuses sampled episodes whose summaries are nearest to $\mathcal{M}_j$. If some condition is still uncovered, it scans a deterministic shared seed stream, summarizes the resulting trace, assigns it to the nearest catalog center, and accepts the seed only if it fills a missing bucket. This produces

$$\mathcal{E}_e = \left\{ \left(g_j, \mathcal{M}_j, \mathcal{S}_j \right) \right\}_{j=1}^{K_e}, \tag{19}$$

where $\mathcal{S}_j$ is the ordered seed set used to instantiate aligned condition g_j. For chlor-alkali, the simulator already provides a native operating-condition catalog; in that case the implementation reuses the domain catalog directly and only builds the matched seed table on top of it.

B.4 Context Embedding and Window Encoder

The context encoder receives the recent aligned dynamic-factor tokens from the shared protocol. At each time step, the protocol outputs one context token $z_t \in \mathbb{R}^m$, where m is the dynamic-factor dimension. The history length is L and the forecast horizon is H. Therefore, in Eq. 7, the matrix $Z_t^- = [z_{t-L+1}, \ldots, z_t] \in \mathbb{R}^{L \times m}$ stacks the most recent L observed tokens, and $\hat{Z}_t^+ = F_\psi(Z_t^-) \in \mathbb{R}^{H \times m}$ is the predicted next-H context block produced by the forecaster F_ψ. In the current implementation, F_ψ is a lightweight prediction branch attached to the same Transformer window encoder. The observed history is encoded, mean-pooled, and passed to a linear next-token head that regresses the future context token. All experiments use the one-step case $H = 1$, so the future block contains a single predicted token. This Transformer-plus-linear-head forecasting rule is computationally cheap and stable for short online horizons.

The full window $X_t = [Z_t^-; \hat{Z}_t^+] = [x_{t,1}, \ldots, x_{t,n}]^\top$ and its length $n = L + H$ are defined in Eq. 8. Here $x_{t,\tau} \in \mathbb{R}^m$ is the τ-th token in the concatenated window: for $\tau \le L$, it is one observed history token from Z_t^-; for $\tau > L$, it is one predicted token from $\hat{Z}_t^+$. The encoder first maps each m-dimensional token to the model width d and adds a positional code:

$$u_{t,\tau}^{(0)} = W_{\text{in}} x_{t,\tau} + p_\tau, \qquad \tau = 1, \ldots, n. \tag{20}$$

In Eq. 20, $W_{\text{in}} \in \mathbb{R}^{d \times m}$ is the input projection, $p_\tau \in \mathbb{R}^d$ is the positional encoding at location τ, and $u_{t,\tau}^{(0)} \in \mathbb{R}^d$ is the input representation seen by the first Transformer block. Stacking these vectors over $\tau = 1, \ldots, n$ gives the input sequence matrix $U_t^{(0)} \in \mathbb{R}^{n \times d}$.

Algorithm 3 Transformer-based query embedding construction

Require: recent history Z_t^-, forecast horizon H, forecaster F_ψ, Transformer encoder E_ϕ
1: predict short-horizon context $\hat{Z}_t^+ \leftarrow F_\psi(Z_t^-)$
2: form the context window $X_t \leftarrow [Z_t^-; \hat{Z}_t^+]$
3: tokenize X_t with Eq. 20
4: run the Transformer encoder by Eq. 21
5: compute the query embedding $\bar{h}_t$ by Eq. 22
6: compute drift score d_t and forecast error p_t by Eq. 23
7: **return** $\bar{h}_t$, d_t, and p_t

The Transformer encoder then applies N_{tr} standard self-attention blocks (Vaswani et al., 2017):

$$U_t^{(\ell)} = \mathrm{TransformerBlock}_\ell(U_t^{(\ell-1)}), \qquad \ell = 1, \ldots, N_{\mathrm{tr}}. \tag{21}$$

Here $U_t^{(\ell)} \in \mathbb{R}^{n \times d}$ is the sequence representation after block ℓ, and $\mathrm{TransformerBlock}_\ell(\cdot)$ denotes one standard self-attention plus feed-forward update. Because self-attention lets each position attend to all other positions in the window, the representation at one time index can encode both the local token value and its relation to earlier and later dynamic factors.

The final query embedding is the mean-pooled top-layer representation

$$\bar{h}_t = \frac{1}{n} \sum_{\tau=1}^{n} U_t^{(N_{\mathrm{tr}})}[\tau] \in \mathbb{R}^d, \tag{22}$$

where $U_t^{(N_{\mathrm{tr}})}[\tau] \in \mathbb{R}^d$ is the top-layer vector at position τ. Averaging over all n positions yields one fixed-width summary vector $\bar{h}_t$, which is the query key used by retrieval in Eq. 10. Intuitively, $\bar{h}_t$ compresses the recent context trajectory together with its immediate forecast into one semantic vector. This encoding step preserves temporal semantics and improves the precision of nearest-neighbor search in the expert-bank.

The same module computes two online diagnostics:

$$d_t = \begin{cases} 0, & t = 1, \\ \max\big(0, 1 - \mathrm{sim}(\bar{h}_t, \bar{h}_{t-1})\big), & t > 1, \end{cases} \qquad p_t = \frac{1}{H} \left\| \hat{Z}_t^+ - Z_{t+1:t+H} \right\|_F, \tag{23}$$

Here $\mathrm{sim}(\bar{h}_t, \bar{h}_{t-1})$ is the cosine similarity between consecutive query embeddings, so d_t is an embedding-space drift score: it is near zero when two successive windows are similar and becomes larger when the encoded context changes sharply. In the second term, $Z_{t+1:t+H}$ denotes the true future context block once it is observed, and $\| \cdot \|_F$ is the Frobenius norm. Thus p_t is the average prediction error per forecast step. These values act as post-retrieval control signals that determine how aggressively the retrieved checkpoints are refined after selection.

B.5 Similarity Retrieval and HNSW-Indexed Expert Reuse

Retrieval of context-matched historical specialists. The central purpose of the expert-bank is to mitigate catastrophic forgetting in online non-stationary MORL. A context shift changes both the feasible operating region and the local Pareto trade-off. Continuing to update one shared learner can therefore overwrite parameters that were effective under an earlier, recurring condition. OW-CMORL instead represents the current temporal condition as a query, retrieves stage snapshots that were learned under similar conditions, and refines only those candidates. The retrieved checkpoint is not assumed to be optimal for the present environment; it is a context-informed initialization that makes recovery less dependent on relearning a previously encountered operating family from scratch. Figure 6 shows this path from the context window, through Transformer encoding and HNSW Top-K retrieval, to local snapshot reuse.

At time t, the shared protocol supplies the recent dynamic-factor history Z_t^- and the prediction branch supplies a short-horizon forecast $\hat{Z}_t^+$. The query construction $\hat{Z}_t^+ = F_\psi(Z_t^-)$, $X_t = [Z_t^-; \hat{Z}_t^+]$, and $\bar{h}_t = E_\phi(X_t)$ is defined in Eq. 8, where F_ψ is the short-horizon prediction branch, X_t is the time-ordered context window, E_ϕ is the Transformer context encoder, and $\bar{h}_t \in \mathbb{R}^d$ is the

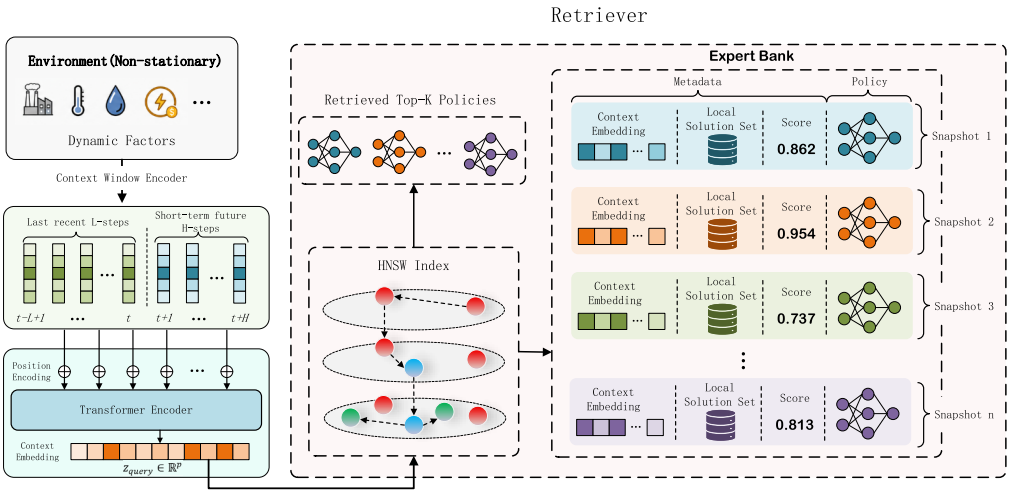


Figure 6: Retriever architecture in OW-CMORL. The current context history and short-horizon forecast form a context window, a Transformer encoder maps this sequence to a query embedding, and the HNSW index returns an approximate Top-K set of context-near stage snapshots for local refinement or direct re-inference.

d-dimensional retrieval key. Thus retrieval compares temporal non-stationary contexts rather than raw policies, raw observations, or manually assigned operating-condition labels.

This use of a learned key follows a broader similarity-matching literature. Matching Networks use cosine-attention weights over learned support embeddings (Vinyals et al., 2016); Prototypical Networks compare query embeddings with learned class representatives in a metric space (Snell et al., 2017); and Neural Episodic Control retrieves values from a learned state embedding using nearest-neighbor memory (Pritzel et al., 2017). These works retrieve labels or values rather than MORL stage snapshots, but they establish the same principle: a learned representation can turn past experience into a queryable memory. In reinforcement learning, latent-context methods such as PEARL and VariBAD further show why an adaptation decision should depend on inferred task context rather than on a raw observation alone (Rakelly et al., 2019; Zintgraf et al., 2020). OW-CMORL specializes this principle to recurring dynamic Pareto conditions and retains an entire restorable stage snapshot as the reusable memory item.

Similarity geometry: from classical vectors to learned semantic keys. For a stored snapshot σ_i, let $\bar{h}_i \neq \mathbf{0}$ be the context embedding recorded when it was created. We normalize the query and stored keys as $\tilde{h}_t = \bar{h}_t/\|\bar{h}_t\|_2$ and $\tilde{h}_i = \bar{h}_i/\|\bar{h}_i\|_2$. Their classical cosine affinity and cosine distance are

$$\mathrm{aff}_{\cos}(\bar{h}_t, \bar{h}_i) = \frac{\bar{h}_t^\top \bar{h}_i}{\|\bar{h}_t\|_2 \|\bar{h}_i\|_2} = \tilde{h}_t^\top \tilde{h}_i, \qquad d_{\cos}(\bar{h}_t, \bar{h}_i) = 1 - \mathrm{aff}_{\cos}(\bar{h}_t, \bar{h}_i). \tag{24}$$

The affinity is large when the two normalized keys point in similar directions, whereas $d_{\cos}$ is small in that case. Since $\|\tilde{h}_t - \tilde{h}_i\|_2^2 = 2d_{\cos}(\bar{h}_t, \bar{h}_i)$, cosine distance and squared Euclidean distance induce the same ranking on normalized keys. OW-CMORL uses $d_{\cos}$ as the final distance for HNSW bank ranking because it directly expresses directional similarity between context keys.

Classical similarity fixes the geometry once the vectors are given. More expressive approaches learn both a representation and a task-dependent affinity. For example, a Transformer attention head forms learned query and key projections and compares them by a scaled inner product (Vaswani et al., 2017):

$$q_t = W_Q \bar{h}_t, \qquad k_i = W_K \bar{h}_i, \qquad a_\phi(t,i) = \frac{\exp\left(q_t^\top k_i/\sqrt{d_k}\right)}{\sum_{j=1}^{N_t} \exp\left(q_t^\top k_j/\sqrt{d_k}\right)}, \tag{25}$$

where W_Q and W_K are learned projection matrices, d_k is the key dimension, and $a_\phi(t,i)$ is a normalized learned affinity. This equation illustrates a general learned pairwise metric; $a_\phi(t,i)$ is not the

retrieval score used by OW-CMORL. OW-CMORL instead separates representation learning from indexed matching: first, the Transformer encoder E_ϕ maps each history-plus-forecast sequence to a semantic key $\bar{h}$; second, HNSW ranks these fixed keys by the cosine distance $d_{\cos}$ in Eq. 24. Thus the encoder learns which temporal features should be represented similarly, while cosine distance alone determines snapshot order in the bank. This separation keeps the indexed distance stable and directly compatible with vector search.

Exact KNN, ANN, and HNSW. Let $K_t = \min(K, N_t)$, where K is the retrieval budget and N_t is the number of stored snapshots. The exact K_t-nearest-neighbor identifier set is

$$\mathcal{N}_t^\star = \mathrm{TopK}_{i\in\{1,\dots,N_t\}} \left[-d_{\cos}(\bar{h}_t, \bar{h}_i)\right]. \tag{26}$$

It is important to distinguish full exact search from exact KNN. A full exact search evaluates every distance and sorts all N_t snapshots, so its query cost is

$$T_{\mathrm{full}}(t) = \Theta(N_t d) + O(N_t \log N_t). \tag{27}$$

Exact KNN avoids a complete sort by retaining only the K_t smallest distances, but it still evaluates every stored key:

$$T_{\text{exact-KNN}}(t) = \Theta(N_t d) + O(N_t \log K_t). \tag{28}$$

Thus KNN specifies the desired output cardinality; by itself, it is not an index and does not remove the linear $\Theta(N_t d)$ distance-computation term. During warm start and successive online refill, the number of reusable snapshots grows until it reaches $B_{\max}$. Moreover, supporting more recurring context families requires a larger bank capacity, so N_t (and eventually the relevant $B_{\max}$) increases with operating duration and coverage requirements. Consequently, exhaustive search adds progressively more routing latency exactly when an online non-stationary multi-objective controller must react to a detected shift and select a reusable specialist before local refinement. A linear scan is therefore poorly matched to near-real-time online response.

Approximate nearest-neighbor (ANN) indexing replaces exhaustive comparison with a structured search that returns a high-quality approximation $\widetilde{\mathcal{N}}_t$ to $\mathcal{N}_t^\star$. HNSW organizes the stored keys as a hierarchical navigable graph (Malkov & Yashunin, 2020): sparse upper layers route a query toward a relevant neighborhood, and a bounded bottom-layer search returns the approximate Top-K identifiers. The retrieval and local bank-update operations used by OW-CMORL are given directly in Algorithm 4. Because this query visits only a bounded subset of the bank rather than scanning every stored key, its output is not guaranteed to equal the globally exact neighbor set $\mathcal{N}_t^\star$.

Let $n_{\mathrm{visit},t}(\xi)$ be the number of graph vertices whose distance to $\bar{h}_t$ is evaluated by an HNSW query with index and search parameters ξ. Its query work is

$$T_{\mathrm{HNSW}}(t;\xi) = O\big(n_{\mathrm{visit},t}(\xi)\big[d + \log\big(\mathrm{ef}_{\mathrm{search}}\big)\big]\big). \tag{29}$$

The first term evaluates cosine distances and the second maintains the bounded search queue. Under the dynamic-factor distributions represented by the bank, the hierarchical graph makes $n_{\mathrm{visit},t}(\xi)$ much smaller than N_t; this is the practical source of the latency reduction. It is not a universal worst-case guarantee of sublinear search: HNSW instead exposes a tunable recall–latency trade-off through its graph and search parameters.

The relevant accuracy measure is recall at the requested cardinality:

$$\mathrm{Recall@}\, K_t = \frac{\left|\widetilde{\mathcal{N}}_t \cap \mathcal{N}_t^\star\right|}{K_t}. \tag{30}$$

HNSW parameters, including the maximum graph degree and the construction and search breadths, determine an empirical recall–latency frontier. The architectural choice can therefore be expressed as the following design criterion, rather than as a claim that ANN is exact:

$$\xi^\star = \arg\min_{\xi}\ \mathbb{E}[T_{\mathrm{HNSW}}(\xi)] \quad \text{subject to} \quad \mathbb{E}[\mathrm{Recall@}\, K_t(\xi)] \geq \rho, \tag{31}$$

where ξ denotes the HNSW construction and search parameters, T_{HNSW} is retrieval latency, and ρ is the minimum recall target. Increasing the search breadth typically improves recall at greater latency. OW-CMORL therefore uses HNSW ANN not to claim exact global KNN, but to keep context routing responsive as the snapshot bank grows while retaining sufficient recall to recover a context-near specialist for online Pareto maintenance.

Algorithm 4 HNSW retrieval and local expert-bank update

Require: query key $\bar{h}_t$, current bank $\mathcal{B}_t$, HNSW index $\mathcal{I}_t$, retrieval budget K, scored candidate set $\mathcal{S}_t$, capacity $B_{\max}$

if $\mathcal{I}_t$ is empty **then**
  $\mathcal{R}_t \leftarrow \emptyset$
else
  $\mathcal{N}_t \leftarrow \mathrm{TopK}(\mathcal{I}_t, \bar{h}_t, K)$
  $\mathcal{R}_t \leftarrow \{\sigma_i : i \in \mathcal{N}_t\}$
end if
for each scored refreshed snapshot $\hat{\sigma} \in \mathcal{S}_t$ **do**
  if $|\mathcal{B}_t| < B_{\max}$ **then**
    insert $\hat{\sigma}$ and $(\bar{h}_{\hat{\sigma}}, \hat{\sigma})$ into $(\mathcal{B}_t, \mathcal{I}_t)$
  else
    $\sigma^\dagger \leftarrow \mathrm{Top1}(\mathcal{I}_t, \bar{h}_{\hat{\sigma}})$
    if $q_{\hat{\sigma}} > q_{\sigma^\dagger}$ **then**
      replace $\sigma^\dagger$ with $\hat{\sigma}$ in $(\mathcal{B}_t, \mathcal{I}_t)$
    end if
  end if
end for
return retrieved set $\mathcal{R}_t$ and updated $(\mathcal{B}_{t+1}, \mathcal{I}_{t+1})$

Instantiation in OW-CMORL. The snapshot tuple $(\Theta_\sigma, \mathcal{Y}_\sigma, \mu_\sigma)$ and its metadata $\mu_\sigma = (\bar{h}_\sigma, q_\sigma)$ are defined in Eq. 9. Here, the flat expert-bank is denoted $\mathcal{B}_t = \{\sigma_i\}_{i=1}^{N_t}$, where $N_t \le B_{\max}$, Θ_i is a restorable learner checkpoint, $\mathcal{Y}_i$ is the Pareto solution set exported at that stage, $\bar{h}_i$ is its insertion-time context key, q_i is its admission score, and $B_{\max}$ is the bank capacity. The HNSW index and approximate Top-K snapshot retrieval are given in Eq. 10; in this appendix, $\mathcal{N}_t$ denotes the returned identifiers and $\mathcal{R}_t$ denotes their corresponding snapshot set. Each member of $\mathcal{R}_t$ is restored for short local refinement or direct re-inference under the present context; their updated solution sets subsequently contribute to the global Pareto archive. The mechanism preserves context-local specialists instead of forcing one continually updated learner to absorb all operating conditions, thereby reducing the pathway through which earlier Pareto expertise is overwritten.

Admission uses the same local geometry and follows the HNSW-local replacement rule in Eq. 13. For a refreshed candidate $\hat{\sigma}$, the incumbent $\sigma^\dagger$ denotes the HNSW-returned context-local competitor used by that rule. Because $\mathrm{Top1}$ invokes HNSW, $\sigma^\dagger$ is neither randomly selected nor guaranteed to be the globally most similar stored snapshot. It is the local competitor found by a bounded ANN traversal, deliberately accepting a controllable loss of exact-neighbor recall in exchange for low latency during repeated online retrieval and refill.

B.6 Continual Refinement, Inference, and Snapshotting

Each retrieved stage snapshot is then reused in the current environment condition. If online update budget is available, the method restores the checkpoint and applies a short continual-learning loop against the current simulator or surrogate. If not, the retrieved checkpoint is used directly for re-inference. The update remains short and local: OW-CMORL refreshes only a small set of context-matched candidates after each shift.

For a retrieved checkpoint σ, the implementation applies $U_t(\sigma)$ local update steps, where $U_t(\sigma)$ is a small budget determined from the current online context diagnostics:

$$U_t(\sigma) = \begin{cases} U_{\max}, & d_t > \tau_d \text{ or } p_t > \tau_p, \\ U_{\min}, & \text{otherwise.} \end{cases} \tag{32}$$

This keeps the adaptation workload proportional to the observed amount of local drift.

During this stage, each learner records periodic refreshed snapshots. In implementation terms, this means that the current learner state is serialized, the checkpoint is re-evaluated to obtain its current solution set, and a temporary runtime record is attached before the snapshot is passed to the scorer. For a refreshed snapshot $\hat{\sigma}$, the runtime record is

$$\nu_{\hat{\sigma}} = \big(\bar{h}_t,\, d_t,\, p_t,\, \mathcal{T}_{\hat{\sigma}}\big), \tag{33}$$

Algorithm 5 Local continual refinement and snapshot generation

Require: retrieved set $\mathcal{R}_t$, current key $\bar{h}_t$, archive $\mathcal{A}_t$, snapshot period Δ
initialize the new snapshot set $\mathcal{S}_t \leftarrow \emptyset$
for each retrieved stage snapshot $\sigma \in \mathcal{R}_t$ **do**
  restore the learner checkpoint Θ_σ
  if update budget is available **then**
    for $u = 1, \ldots, U$ **do**
      interact with the current simulator or surrogate and update the learner
      if $u \bmod \Delta = 0$ **then**
        export the current solution set and create a refreshed stage snapshot $\hat{\sigma}$
        attach runtime record $\nu_{\hat{\sigma}}$ by Eq. 33 and add $\hat{\sigma}$ to $\mathcal{S}_t$
      end if
    end for
  else
    run direct re-inference once, export the resulting solution set, create one snapshot $\hat{\sigma}$, attach $\nu_{\hat{\sigma}}$, and add it to $\mathcal{S}_t$
  end if
end for
merge $\bigcup_{\hat{\sigma} \in \mathcal{S}_t} \mathcal{Y}_{\hat{\sigma}}$ into $\mathcal{A}_t$ by non-dominated filtering
return $\mathcal{S}_t$ and the updated archive

Algorithm 6 Snapshot scoring, HNSW refill, and local eviction

Require: new snapshot set $\mathcal{S}_t$, current key $\bar{h}_t$, expert-bank $\mathcal{B}_t$, HNSW index $\mathcal{I}_t$
for each new snapshot $\hat{\sigma} \in \mathcal{S}_t$ **do**
  compute the bank score $q_{\hat{\sigma}}$ by Eq. 34
  set the persistent metadata of $\hat{\sigma}$ to $(\bar{h}_t, q_{\hat{\sigma}})$
  update $(\mathcal{B}_t, \mathcal{I}_t)$ by the local-update branch of Algorithm 4
end for
return updated $(\mathcal{B}_{t+1}, \mathcal{I}_{t+1})$

where $\mathcal{T}_{\hat{\sigma}}$ denotes the trajectory-based metrics collected during the current local refinement, including shift regret, recovery latency, and the pre/post-shift utility gap. After scoring and bank admission, the persistent metadata stored in the bank is reduced to $(\bar{h}_t, q_{\hat{\sigma}})$ as in Eq. 9. All accepted solution points are then merged into the long-term archive through the non-dominated update in Eq. 11.

B.7 Scorer, HNSW Refill, and Eviction

After the snapshot stage, OW-CMORL must decide which policies are worth keeping for future reuse. The scorer evaluates each snapshot using Eq. 12. The score combines current-context solution quality with two trace-based adaptation signals: shift regret and recovery latency. The bank therefore acts as a selective long-term memory.

The implemented bank score is a bounded version of Eq. 12. For a refreshed snapshot $\hat{\sigma}$ with trace statistics

$$\mathcal{T}_{\hat{\sigma}} = (\mathrm{Reg}_{\hat{\sigma}}, \mathrm{Lat}_{\hat{\sigma}}),$$

the stored score is

$$q_{\hat{\sigma}} = \beta_{\mathrm{perf}}\ \mathrm{clip}(\mathrm{Perf}(\mathcal{Y}_{\hat{\sigma}} \mid \bar{h}_t), 0, 1)\ -\ \beta_{\mathrm{reg}} \frac{|\mathrm{Reg}_{\hat{\sigma}}|}{1 + |\mathrm{Reg}_{\hat{\sigma}}|}\ -\ \beta_{\mathrm{lat}} \frac{\max(0, \mathrm{Lat}_{\hat{\sigma}})}{1 + \max(0, \mathrm{Lat}_{\hat{\sigma}})}. \tag{34}$$

Higher current-context quality increases the score, while larger regret and longer latency decrease it.

After scoring, each candidate snapshot is admitted by the local-update branch of Algorithm 4. When the bank is full, the candidate does not compete against the whole memory. Instead, it competes only against the context-local competitor returned by the HNSW ANN query; this competitor is not asserted to be the exact global nearest stored neighbor. This is a key design choice for online non-stationary optimization: it keeps semantically different experts from evicting each other and therefore preserves coverage over recurring context families.

B.8 WARM START

Before deployment, the initial bank is formed by

$$\mathcal{B}_0 = \mathrm{FillBank}\left(\bigcup_{\ell\in\mathcal{L}}\bigcup_{u\in\mathcal{U}_\ell}\mathrm{StageSnapshot}(\ell,u)\right), \tag{35}$$

where $\mathrm{FillBank}$ applies the same score-and-local-replacement rule used online.

The initialization stage uses the same bank-update rule as the online stage. Before deployment, OW-CMORL runs several base learners over the shared protocol layer and records periodic stage snapshots. For each learner ℓ and saved stage u, it restores $\Theta_{\ell,u}$, exports $\mathcal{Y}_{\ell,u}$, derives the context key $\bar{h}_{\ell,u}$ from the corresponding window, and scores the resulting snapshot $\sigma_{\ell,u}$ using Eq. 34. The local insertion rule of Algorithm 4 then builds the initial bank and HNSW index from these snapshots, exactly as specified by Eq. 35.

A learner may internally sweep different preferences or objective schedules, but the bank stores each saved training stage as one reusable specialist. This exposes the initial bank to multiple context families before deployment, reducing cold-start forgetting and delayed recovery during the first few shifts.

C ENVIRONMENT INTERFACES AND DYNAMIC FACTORS

In this study, we use four simulated environments to validate the experimental results. Building, EV charging, and cogen are adapted from SustainGym (Yeh et al., 2023); chlor-alkali is an industrial surrogate environment. Table 4 defines all exogenous dynamic factors for the four environments and reports their units, ranges, and descriptions. The shared protocol layer uses these factors to construct the context vector, and context vectors from multiple time steps form the context window. The following descriptions specify the native observations, actions, objectives, and constraints for all four simulators:

1. **Building energy management.** *Observations.* The HVAC controller receives the three zone temperatures $\mathbf{T}_t^z \in \mathbb{R}^3$ (°C), outdoor temperature T_t^{out} (°C), solar irradiance G_t (W m^{-2}), ground temperature T_t^{gnd} (°C), occupancy load P_t^{occ} (W), carbon intensity $I_t^{\mathrm{CO_2}}$ (kg $\mathrm{CO_2e\ kWh}^{-1}$), and electricity price p_t^{elec} (\$ kWh^{-1}). *Actions.* It selects normalized signed heating/cooling commands $\mathbf{u}_t^{\mathrm{HVAC}} \in [-1,1]^3$ for the three zones. *Objectives.* The reward vector is $(r_t^{\mathrm{comfort}}, -C_t^{\mathrm{elec}}, -C_t^{\mathrm{ramp}})$, representing thermal comfort, electricity cost, and changes in HVAC effort. Temperatures must remain within their thermal bounds, and each HVAC command is bounded in $[-1,1]$.

2. **EV charging.** *Observations.* The scheduler observes the day fraction τ_t, remaining connection time $\boldsymbol{\delta}_t$ in 5-minute slots, remaining energy demand $\mathbf{d}_t$ (kWh), the previous marginal operating emissions rate e_{t-1}, and the forecast vector $\mathbf{e}_t^{\mathrm{fcst}}$ (kg $\mathrm{CO_2e\ kWh}^{-1}$). *Actions.* It outputs a normalized pilot-signal vector $\mathbf{a}_t \in [0,1]^{54}$; each active signal maps to at most 32 A at a garage EVSE, while inactive slots are padded. *Objectives.* The vector $(\Pi_t, -C_t^{\mathrm{CO_2}}, -C_t^{\mathrm{net}})$ captures charging profit, carbon cost, and network-violation cost. Actions obey demand-dependent bounds and are projected onto the feasible charging-network set $\mathcal{F}_{\mathrm{net}}$ when necessary.

3. **Combined heat and power (Cogen).** *Observations.* The dispatch controller receives the day fraction τ_t, previous control $\mathbf{a}_{t-1}$, and current or forecast-horizon vectors for ambient temperature $\mathbf{T}_t^{\mathrm{amb}}$ (°F), pressure $\mathbf{P}_t^{\mathrm{amb}}$ (psia), humidity $\mathbf{H}_t^{\mathrm{amb}}$, power demand $\mathbf{D}_t^P$ (MW), steam demand $\mathbf{D}_t^S$ (klb h^{-1}), electricity price $\mathbf{p}_t^E$ (\$ MWh^{-1}), and natural-gas price $\mathbf{p}_t^G$ (\$ MMBtu^{-1}). *Actions.* It dispatches gas-turbine powers $\mathbf{P}_t^{\mathrm{GT}}$, binary cooler switches $\mathbf{b}_t$, heat-recovery flows $\mathbf{F}_t^{\mathrm{HR}}$, steam-turbine power P_t^{ST}, an intermediate-pressure flow F_t^{IP}, and an integer cooling-bay count N_t^{CT}. *Objectives.* The vector $(-C_t^{\mathrm{fuel}}, -C_t^{\mathrm{ramp}}, -C_t^{\mathrm{delivery}}, -C_t^{\mathrm{cv}})$ penalizes fuel consumption, ramping, demand shortfall, and dynamic-constraint violations. Equipment bounds and power and steam delivery requirements define the feasible plant-action set.

4. **Chlor-alkali production.** *Observations.* The 24 normalized coordinates are $\mathbf{o}_t = (\mathbf{c}_t, p_t^{\mathrm{TOU}}, \mathbf{x}_t^{\mathrm{proc}}, \mathbf{v}_t^{\mathrm{health}}, \mathbf{y}_{t-1})$: calendar encodings, time-of-use price, process summaries, health summaries, and previous outputs. In particular, $\mathbf{x}_t^{\mathrm{proc}}$ contains total current, process temperatures, HCl flow, and anolyte, catholyte, and brine flows; $\mathbf{v}_t^{\mathrm{health}}$ records active-cell status, aging, current-limit, flow, stress, and efficiency indicators; and $\mathbf{y}_{t-1}$ records production, current efficiency, and DC energy intensity. *Actions.* The controller sets $\mathbf{a}_t = (a_t^I, a_t^{\mathrm{bal}}, a_t^{\mathrm{HCl}}, a_t^{\mathrm{an}}, a_t^{\mathrm{cat}}, a_t^{\mathrm{br}}) \in [-1, 1]^6$, which scales aggregate current, current balance, acid, anolyte, catholyte, and brine flows. *Objectives.* The vector $(Y_t^{\mathrm{prod}}, \eta_t^{\mathrm{curr}}, -C_t^{\mathrm{elec}} - C_t^{\mathrm{con}})$ represents daily caustic-soda production, current efficiency, and price-weighted electricity-plus-constraint cost. Total current is limited to $[86, 119]$ kA, average temperature to $85°$C, and flows must remain nonnegative.

D Detailed Baselines

The baselines cover complementary alternatives to context-indexed expert routing: CAPQL tests preference-conditioned convex-coverage learning, PGMORL tests population-based frontier prediction, Q-Pensieve tests replay-style snapshot retention, and MORL-CA tests offline-to-online adaptive objective weighting in a multi-critic actor–critic. Each method retains its native learner; the following descriptions state the corresponding memory and adaptation mechanisms.

CAPQL. CAPQL (Lu et al., 2023) is a strong convex-coverage actor-critic method for stationary MORL. It learns a preference-conditioned policy and critic, and is a natural baseline for testing whether a strong stationary frontier learner remains competitive under drift. In our comparison, CAPQL keeps its native training rule. We only wrap it with the shared dynamic environment adapter and the shared evaluator.

PGMORL. PGMORL (Xu et al., 2020) is a population-based MORL method that expands the frontier with prediction-guided updates. It is a useful baseline here because it already contains a notion of frontier forecasting, but it does not use an explicit Transformer-based context encoder or a context-indexed expert-bank. This helps isolate whether frontier prediction alone is enough under non-stationary dynamics.

Q-Pensieve. Q-Pensieve (Hung et al., 2023) stores and reuses multiple critic snapshots across preferences. It is the strongest replay-style baseline in our study. Its memory mechanism can preserve large frontier coverage, especially in settings such as building. At the same time, it has no explicit module for detecting context drift from environment-side signals, so it is a strong test of whether generic memory sharing is enough.

MORL-CA. MORL-CA is an offline–online dynamic MORL algorithm that combines historical-data pretraining with online policy refinement under changing operating conditions. It uses an adaptive-weight multi-critic actor–critic architecture to adjust objective trade-offs according to the current state. Unlike OW-CMORL, it has no context window or context-indexed expert-bank for specialist reuse; it therefore tests whether adaptive weighting and continual online learning alone can handle non-stationarity.

E Experiment Setup and Fair Comparison Rules

Shared 100-operating-condition protocol. Every method is evaluated on the same aligned sequence of 100 operating conditions in each environment. The operating-condition catalog is built from fine-grained clustering of dynamic-factor traces with 100 clusters, 256 catalog episodes, and 4 initial context steps per episode for condition assignment. During final evaluation, all methods are rescored with a shared condition-specific reference point computed from the union of all compared fronts under that operating condition. This follows the reliable benchmarking practice advocated in recent MORL work (Felten et al., 2023; Teoh et al., 2025).

No manual stage labels. The adaptation loop does not receive hand-written stage IDs. OW-CMORL only sees the environment-side context signal. The same rule applies to the baselines.

Environment	Dynamic factor	Native unit	Normalized range	Meaning
Building	τ_t^{B}	–	$[0, 1]$	Position within the daily building-control horizon.
	s_t^{occ}	–	$[0, 1]$	Occupancy-schedule scale applied to the reference building load.
	$s_t^{\mathrm{CO_2}}$	–	$[0, 1]$	Carbon-intensity scale applied to the reference carbon trace.
	s_t^{p}	–	$[0, 1]$	Electricity-price scale applied to the reference price trace.
	$\widetilde{T}_t^{\mathrm{tar}}$	°C	$[0, 1]$	Target indoor temperature.
	$\widetilde{T}_t^{\mathrm{out}}$	°C	$[0, 1]$	Outdoor air temperature.
	$\widetilde{G}_t$	W m^{-2}	$[0, 1]$	Global horizontal irradiance.
	$\widetilde{T}_t^{\mathrm{gnd}}$	°C	$[0, 1]$	Ground temperature.
	$\widetilde{P}_t^{\mathrm{occ}}$	W	$[0, 1]$	Occupancy-related internal thermal load.
	$\widetilde{I}_t^{\mathrm{CO_2}}$	kg CO_2e kWh^{-1}	$[0, 1]$	Grid carbon intensity.
	$\widetilde{p}_t^{\mathrm{elec}}$	\$ kWh^{-1}	$[0, 1]$	Retail electricity price.
EV charging	τ_t^{EV}	–	$[0, 1]$	Position within the charging day.
	ρ_t^{act}	–	$[0, 1]$	Fraction of charger slots occupied by an active session.
	ρ_t^{dem}	–	$[0, 1]$	Aggregate remaining energy demand divided by its site-level cap.
	ρ_t^{ddl}	–	$[0, 1]$	Deadline pressure derived from active-session departure times.
	e_{t-1}	kg CO_2e kWh^{-1}	$[0, 1]$	Previous-step MOER.
	$\overline{e}_t^{\mathrm{fcst}}$	kg CO_2e kWh^{-1}	$[0, 1]$	Mean MOER over the forecast horizon.
	s_t^{fcst}	–	$[0, 1]$	Normalized signed trend of the MOER forecast.
	e_t^{peak}	kg CO_2e kWh^{-1}	$[0, 1]$	Maximum MOER over the forecast horizon.
Cogen	τ_t^{CG}	–	$[0, 1]$	Position within the cogen dispatch day.
	$\widetilde{R}_t$	MW	$[0, 1]$	Renewable-generation magnitude relative to the configured range.
	$\widetilde{T}_t^{\mathrm{amb}}$	°F	$[0, 1]$	Ambient-temperature forecast at the current dispatch step.
	s_t^{T}	–	$[0, 1]$	Normalized one-step ambient-temperature forecast trend.
	$\widetilde{H}_t^{\mathrm{amb}}$	–	$[0, 1]$	Ambient relative humidity.
	$\widetilde{D}_t^{P}$	MW	$[0, 1]$	Target net power demand.
	s_t^{P}	–	$[0, 1]$	Normalized one-step target-power trend.
	$\widetilde{D}_t^{S}$	klb h^{-1}	$[0, 1]$	Target process-steam demand.
	$\widetilde{p}_t^{E}$	\$ MWh^{-1}	$[0, 1]$	Electricity price.
	$\widetilde{p}_t^{G}$	\$ MMBtu^{-1}	$[0, 1]$	Natural-gas price.
Chlor-alkali	$\sin(2\pi h_t/24)$	–	$[-1, 1]$	Cyclic hour-of-day encoding.
	$\cos(2\pi h_t/24)$	–	$[-1, 1]$	Cyclic hour-of-day encoding complementary to the sine coordinate.
	$\sin(2\pi m_t/12)$	–	$[-1, 1]$	Cyclic month encoding.
	$\cos(2\pi m_t/12)$	–	$[-1, 1]$	Cyclic month encoding complementary to the sine coordinate.
	$\widetilde{p}_t^{\mathrm{TOU}}$	CNY kWh^{-1}	$[0, 1]$	Normalized Anhui 35-kV time-of-use electricity-price proxy.
	$I_t^{\max}$	kA	$[0, 1]$	Available total-current limit. It is normalized from the configured $[86, 119]$ kA range and follows the price-linked operating limit.

Table 4: Dynamic factors used by the four PCA vectors. Each row is one coordinate in the corresponding vector: Building ($m = 11$), EV charging ($m = 8$), Cogen ($m = 10$), or Chlor-alkali ($m = 6$).

Operating-condition identifiers are used only for alignment, logging, and shared evaluation after the run.

Same preference families. All MORL methods are evaluated on the same fixed preference grids. The three-objective environments use seven preferences: the three simplex corners, the three pairwise midpoints, and the uniform point. Cogen uses the analogous four-objective grid with four corners, six pairwise midpoints, and the uniform point. This avoids giving any method an advantage from a denser or more favorable evaluation preference set.

Same trace metrics. All methods use the same trajectory-based adaptation metrics, the same post-shift recovery window (12), the same recovery target ratio ($n = 0.9$), and the same valid-shift

filtering rule. Recovery latency is counted in post-shift episodes rather than within-episode environment steps. For HV, EU, and adaptation score, each seed-level aggregate is computed across the 100 aligned operating conditions, and the reported 95% t-intervals are computed across the 10 seed-level means. For shift regret and recovery latency, the intervals are computed over valid detected shifts only.

Native training rules are preserved. We do not force all baselines into one training backbone. CAPQL keeps its SAC-style preference-conditioned actor-critic rule. PGMORL keeps its population-and-prediction pipeline. Q-Pensieve keeps its memory-sharing SAC variant. MORL-CA keeps its simple continual actor-critic design. Fairness here means equal dynamic wrappers and equal evaluation protocol, not erasing the identity of each baseline.

Fairness of the shared comparison protocol. The key fairness point is that our claim is not "OW-CMORL wins because every baseline was forced into the same code path." The claim is stronger and more realistic: even when each baseline keeps its own native learner, once we place all methods under the same ordered operating conditions, the same preference set, the same trace metrics, and the same condition-specific shared reference point, OW-CMORL is more reliable on preference-aware utility and post-shift robustness. That is exactly the property the paper is trying to measure.

F Evaluation Metrics

For operating condition c, let $\mathcal{P}_c$ be the returned non-dominated solution set, $\mathbf{r}_c$ the reference point shared by all compared methods, M the number of objectives, and λ the Lebesgue measure. The frontier-quality metrics are

$$\begin{aligned} \mathrm{HV}_c &= \lambda\left(\bigcup_{\mathbf{y}\in\mathcal{P}_c}\prod_{j=1}^{M}[r_{c,j}, y_j]\right), \\ \mathrm{EU}_c &= \frac{1}{|\mathcal{W}|}\sum_{\mathbf{w}\in\mathcal{W}}\max_{\mathbf{y}\in\mathcal{P}_c}\mathbf{w}^\top\mathbf{y}, \end{aligned} \tag{36}$$

where $\mathcal{W}$ is the fixed preference grid and $\mathbf{w}$ is a preference vector. HV measures the volume dominated by $\mathcal{P}_c$ above the common reference point, while EU measures the best available utility at each evaluation preference.

Let $\mathrm{HV}_c^\star$ and $\mathrm{EU}_c^\star$ be the largest scores achieved by any compared method under condition c, and let ε prevent division by zero. We define condition-wise adaptation and post-shift metrics by

$$\begin{aligned} g_c^{\mathrm{hv}} &= \frac{\max(0, \mathrm{HV}_c^\star - \mathrm{HV}_c)}{\max(|\mathrm{HV}_c^\star|, \varepsilon)}, & g_c^{\mathrm{eu}} &= \frac{\max(0, \mathrm{EU}_c^\star - \mathrm{EU}_c)}{\max(|\mathrm{EU}_c^\star|, \varepsilon)}, \\ \mathrm{AS}_c &= 1 - \frac{1}{2}\left(g_c^{\mathrm{hv}} + g_c^{\mathrm{eu}}\right), \\ \bar{u}_k^{\mathrm{pre}} &= \frac{1}{L}\sum_{i=1}^{L} u_{\tau_k - i}, & \mathrm{Reg}_k &= \frac{1}{L}\sum_{i=0}^{L-1}\max(\bar{u}_k^{\mathrm{pre}} - u_{\tau_k + i}, 0), \\ \mathrm{Lat}_k(n) &= \min\left\{t \geq 0 \mid \bar{u}_k^{\mathrm{post}}(t) \geq n\,\bar{u}_k^{\mathrm{pre}}\right\}, & n &= 0.9. \end{aligned} \tag{37}$$

Here τ_k is the k-th valid detected shift, L is the post-shift evaluation window, and u_t is the scalarized utility trace; $\bar{u}_k^{\mathrm{post}}(t)$ is its running post-shift mean through episode t. We report the mean AS across the 100 aligned conditions and the mean Reg and Lat (0.9) across valid shifts. Higher AS is better, while lower Reg and Lat indicate less immediate loss and faster recovery. All methods use the same preference grids, condition order, shared references, shift filter, and recovery target.

G Bayesian Hyperparameter Selection

All methods use Bayesian optimization to select their method-specific hyperparameters. Each trial is evaluated on a validation condition sequence disjoint from the final shared evaluation sequence, with the method's native training rule and the same preference family retained. Trials are ranked

Method	Hyperparameters	Explanation	Selected values
OW-CMORL	η	Learning rate	$\eta = 3 \times 10^{-4}$
	T	Total training budget	$T = [2048, 16384, 3072, 3072]$
	T_{warm}	Warm-start training budget	$T_{\text{warm}} = [1024, 8192, 1536, 1536]$
	K	Top-K retrieval budget	$K = [6, 3, 4, 6]$
	L	Context-window history length	$L = [4, 6, 6, 6]$
	B_{trace}	Context-trace budget	$B_{\text{trace}} = [16, 8, 12, 6]$
	B_{bank}	Expert-bank capacity	$B_{\text{bank}} = [18, 32, 18, 32]$
CAPQL	d_h	Actor/critic hidden width	$d_h = 256$
	b	Batch size	$b = 256$
	T	Training budget	$T = 1024$–200k steps across the retained runs
PGMORL	n_{roll}	Rollout length	$n_{\text{roll}} = 256$
	I_{warm}	Warm-up iterations	$I_{\text{warm}} = 20$
	I_{update}	Update interval	$I_{\text{update}} = 5$
	N_{worker}	Worker count	$N_{\text{worker}} = 1$
	T	Training budget	$T = 200$k training steps
Q-Pensieve	$\mathbf{d}_h$	Hidden-layer widths	$\mathbf{d}_h = [256, 256]$
	b	Batch size	$b = 256$
	M_{pref}	Preference-memory capacity	$M_{\text{pref}} = 4$
	Δ_Q	Q-snapshot interval	$\Delta_Q = 1000$
	T	Training budget	$T = 1024$–200k steps across the retained runs
MORL-CA	d_{actor}	Actor hidden width	$d_{\text{actor}} = 256$
	d_{critic}	Critic hidden width	$d_{\text{critic}} = 256$
	d_{AOW}	AOW hidden width	$d_{\text{AOW}} = 256$
	b	Batch size	$b = 256$
	T	Training budget	$T = 4096$ training steps

Table 5: Bayesian-optimization-selected hyperparameters for all compared methods. The final shared evaluation sequence is excluded from selection.

by mean adaptation score, with expected utility and recovery latency used as tie breakers. Table 5 reports the selected settings used for the final runs. The entries for OW-CMORL are ordered as Building / EV charging / Cogen / Chlor-alkali; method-specific parameters not listed in the table retain the verified implementation defaults.

H Significance Tests

For each method and environment, let m_s^A be method A's aggregate metric over the 100 operating conditions under seed s. For a baseline B, we define the paired difference as

$$d_s = \begin{cases} m_s^{\text{OW}} - m_s^B, & \text{for HV, EU, and AS,} \\ m_s^B - m_s^{\text{OW}}, & \text{for Reg and Lat,} \end{cases} \tag{38}$$

so that a positive value favors OW-CMORL. Using the 10 seed-level differences, we compute the two-sided paired t-test statistic

$$t = \frac{\bar{d}}{s_d/\sqrt{10}}, \qquad df = 9,$$

and apply the Holm step-down correction across the $4 \times 4 \times 5 = 80$ comparisons. Table 6 reports the adjusted two-sided p-values. Bold entries indicate significant OW-CMORL advantages, whereas italic entries indicate significant baseline advantages.

I OOD Protocol and Analysis

OOD setting. This study evaluates local refinement of an ID checkpoint after an unseen change in dynamic conditions. The OOD factor combinations are excluded from ID training, model selection, and archive construction. They comprise unseen weather–tariff combinations for Building, a Caltech-to-JPL/Fall-2019 transfer for EV charging, higher renewable penetration with forecast noise for Cogen, and a disjoint tariff sequence for Chlor-alkali. OW-CMORL receives the original dynamic factors rather than PCA coordinates. For Figure 3, we simulate 100,000 continuous ID factor vectors per environment, display 1,000 PCA samples, and place four synthetic OOD points at dispersed random directions outside the ID cloud under minimum-distance constraints.

Evaluation and ID retention. For each environment, we restore the unchanged ID checkpoint and run three fixed OOD sequences. Each sequence evaluates the initial ID condition, the OOD

Environment	Metric	CAPQL	PGMORL	Q-Pensieve	MORL-CA
Building	HV	9.15×10^{-3}	$\mathbf{9.59 \times 10^{-12}}$	6.00×10^{-13}	$\mathbf{2.70 \times 10^{-13}}$
Building	EU	$\mathbf{1.24 \times 10^{-6}}$	$\mathbf{1.22 \times 10^{-7}}$	$\mathbf{0.014}$	$\mathbf{1.23 \times 10^{-8}}$
Building	AS	$\mathbf{7.37 \times 10^{-7}}$	$\mathbf{6.17 \times 10^{-10}}$	9.15×10^{-10}	$\mathbf{1.14 \times 10^{-11}}$
Building	Reg	$\mathbf{4.58 \times 10^{-10}}$	$\mathbf{2.32 \times 10^{-6}}$	$\mathbf{2.78 \times 10^{-3}}$	$\mathbf{4.83 \times 10^{-9}}$
Building	Lat	$\mathbf{1.48 \times 10^{-8}}$	$\mathbf{3.35 \times 10^{-11}}$	$\mathbf{2.09 \times 10^{-3}}$	$\mathbf{2.95 \times 10^{-8}}$
EV charging	HV	$\mathbf{4.73 \times 10^{-10}}$	$\mathbf{4.97 \times 10^{-5}}$	$\mathbf{2.12 \times 10^{-7}}$	$\mathbf{6.84 \times 10^{-13}}$
EV charging	EU	$\mathbf{4.83 \times 10^{-9}}$	$\mathbf{5.60 \times 10^{-7}}$	$\mathbf{3.05 \times 10^{-9}}$	$\mathbf{1.38 \times 10^{-11}}$
EV charging	AS	$\mathbf{4.21 \times 10^{-8}}$	$\mathbf{1.40 \times 10^{-9}}$	$\mathbf{3.20 \times 10^{-9}}$	$\mathbf{1.37 \times 10^{-11}}$
EV charging	Reg	$\mathbf{8.43 \times 10^{-7}}$	$\mathbf{3.59 \times 10^{-9}}$	$\mathbf{2.98 \times 10^{-8}}$	$\mathbf{4.14 \times 10^{-10}}$
EV charging	Lat	2.07×10^{-8}	1.75×10^{-4}	0.051	$\mathbf{3.67 \times 10^{-5}}$
Cogen	HV	$\mathbf{3.55 \times 10^{-16}}$	8.60×10^{-3}	$\mathbf{4.94 \times 10^{-16}}$	$\mathbf{9.49 \times 10^{-16}}$
Cogen	EU	$\mathbf{4.29 \times 10^{-17}}$	$\mathbf{9.69 \times 10^{-11}}$	$\mathbf{3.75 \times 10^{-16}}$	$\mathbf{1.76 \times 10^{-16}}$
Cogen	AS	$\mathbf{4.05 \times 10^{-15}}$	2.59×10^{-6}	$\mathbf{6.74 \times 10^{-13}}$	$\mathbf{3.01 \times 10^{-13}}$
Cogen	Reg	$\mathbf{2.31 \times 10^{-13}}$	$\mathbf{1.55 \times 10^{-7}}$	$\mathbf{1.10 \times 10^{-12}}$	$\mathbf{1.79 \times 10^{-14}}$
Cogen	Lat	$\mathbf{3.28 \times 10^{-11}}$	$\mathbf{2.00 \times 10^{-7}}$	$\mathbf{1.68 \times 10^{-11}}$	$\mathbf{4.23 \times 10^{-11}}$
Chlor-alkali	HV	$\mathbf{5.46 \times 10^{-13}}$	$\mathbf{4.58 \times 10^{-10}}$	$\mathbf{3.79 \times 10^{-10}}$	$\mathbf{9.43 \times 10^{-9}}$
Chlor-alkali	EU	$\mathbf{0.017}$	$\mathbf{0.017}$	$\mathbf{5.46 \times 10^{-4}}$	$\mathbf{1.30 \times 10^{-3}}$
Chlor-alkali	AS	$\mathbf{1.02 \times 10^{-9}}$	$\mathbf{6.77 \times 10^{-7}}$	$\mathbf{3.62 \times 10^{-5}}$	$\mathbf{3.17 \times 10^{-13}}$
Chlor-alkali	Reg	$\mathbf{1.21 \times 10^{-4}}$	$\mathbf{8.97 \times 10^{-4}}$	$\mathbf{3.18 \times 10^{-3}}$	$\mathbf{1.12 \times 10^{-3}}$
Chlor-alkali	Lat	1.61×10^{-10}	6.95×10^{-9}	3.55×10^{-9}	1.01×10^{-6}

Table 6: Holm-adjusted two-sided p-values for the 80 paired seed-level significance tests. Each test uses 10 random seeds, each aggregated over 100 shared operating conditions. Bold values indicate significant OW-CMORL advantages after correction; italic values indicate a significant difference favoring the baseline.

condition with no update ($U = 0$), the OOD condition after cumulative OW-CMORL local-update episode budgets $U \in \{8, 16, 32\}$, and the same ID condition after the OOD updates. The final frozen ID evaluation uses the same seed, preferences, horizon, and HV reference as the initial ID evaluation; its HV and EU ratios quantify retention of ID performance. U is the number of completed local-update episodes, and each episode comprises 1,000 update steps. HV uses a fixed per-environment normalized objective envelope and shared reference, while EU remains the native scalarized utility. State-hash, update-budget, and retrieval audits verify the protocol for all sequences.

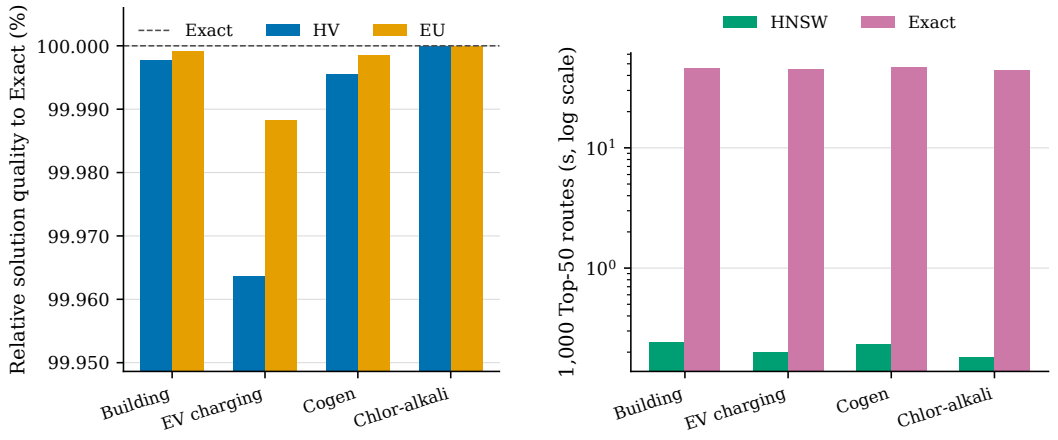
Results. Figure 4 presents the HV and EU outcomes of the experiment. The first bar shows performance under the initial ID condition, and the final bar shows the performance of the snapshot after local updates when it is evaluated again under the same ID condition. The intermediate bars, from $U = 0$ to $U = 32$, show the solution sets obtained after different amounts of local updates under the OOD condition. Here, $U = 0$ means that no local-update episode is performed, while $U = 8$, 16, and 32 mean that the snapshot has undergone 8, 16, and 32 OW-CMORL local-update episodes, respectively; each episode comprises 1,000 update steps. The lower bar charts show the fine-grained changes in the metrics of the solution sets obtained from $U = 0$ to $U = 32$.

The results show that local updates provide relatively stable improvements in solution-set quality for Cogen and Chlor-alkali, whereas the improvements are more limited for Building and EV charging. This may be because the number of short local-update episodes is limited. Increasing the number of update episodes can generally produce solution sets with better performance, but reduces the near-real-time inference capability of the algorithm; this is a trade-off. The ID re-evaluation after local updates shows that, in most simulated environments, solution-set quality remains at the same level as before local updates or improves slightly. This indicates that local updates do not cause OW-CMORL to forget previously encountered conditions, demonstrating its resistance to catastrophic forgetting.

J Ablation Protocol and Analysis

J.1 Context-Key Ablation

This ablation evaluates whether each component of the context window improves context matching. We compare the full window with H1, which uses the current dynamic-factor vector only, and two H2 variants, which remove the forecast or history component. In each of the four environments, every



(a) Relative HV and EU compared with Exact retrieval.

(b) End-to-end time for 1,000 Top-50 routing requests.

Figure 7: HNSW Top-50 retrieval ablation across four simulated environments.

variant is evaluated on the same ordered 100-shift dynamic-condition sequence and with the same metric definitions as the comparative experiment. All variants use an identical short local-update budget and retain the same expert-bank update, Top-K retrieval, and online Pareto archive merge. We report the mean raw HV and EU over the 100 shifts; the Full-window row is the OW-CMORL result from the comparative experiment.

Table 2 shows that the complete context window has the highest mean raw HV and EU in every environment. H1 has the largest quality loss, while removing either the forecast or history component also consistently degrades the retrieved solution sets. The result establishes that the context window provides useful temporal information for OW-CMORL's expert routing.

J.2 Expert-Bank Memory Diagnostic

We compare the semantic expert-bank with two same-capacity embedding replay memories: Flat-FIFO kNN removes the oldest snapshot, whereas Flat-prioritized kNN globally retains high-quality snapshots without context slots. All variants use the same query embedding, Top-K budget, local inference, and archive merge under the shared 100-shift conditions. Table 3 reports the mean raw HV and raw EU in each environment's native units; it is the expert-bank ablation, distinct from the context-key and HNSW ablations.

Table 3 shows that the semantic bank has the highest raw HV and EU in every environment. Flat-FIFO kNN loses context coverage as old snapshots are removed, while Flat-prioritized kNN has larger losses because global quality retention removes coverage of rare recurring conditions. The result supports context-partitioned, quality-aware snapshot retention under recurring shifts.

J.3 HNSW Retrieval Ablation

For each of the four simulated environments, we construct 1,000 dynamic-factor conditions and generate 1,000 initial snapshot policies per condition, yielding a one-million-snapshot expert-bank with 256-dimensional context embeddings. We then construct 1,000 query conditions. Each query retrieves the Top-50 snapshots and performs solution-set inference. Under identical banks and queries, we compare HNSW with Exact nearest-neighbor retrieval using mean HV, mean EU, and end-to-end retrieval time.

Figure 7 shows that HNSW retains at least 99.96% of Exact HV and 99.99% of Exact EU in every environment. For 1,000 routing requests, it reduces end-to-end time from 44.548–46.637 s for Exact retrieval to 0.182–0.241 s. The quality loss from approximate retrieval is below 1%, whereas routing becomes sub-second. HNSW therefore provides the retrieval efficiency required for OW-CMORL's industrial near-real-time optimization setting.