LINC: LABEL-FREE IN-CONTEXT NETWORK RECONFIGURATION UNDER UNOBSERVED FAILURE-MODE SWITCHES

Anonymous authors
Paper under double-blind review

ABSTRACT

Policies trained in a single operating regime often fail when the environment changes without warning—the analogue of a robot that cannot recover after a broken leg, or a network that keeps repairing hubs after the attack has moved to a region. We study this problem as in-context meta-reinforcement learning in a hidden-parameter (contextual) MDP whose latent parameter—the failure mode—is unobserved and may switch inside an episode. We propose LINC, which infers a label-free context from a short window of transition tokens by self-supervised next-step prediction with variance–covariance regularization, and conditions a PPO policy on the detached context together with a size-weighted bridging prior. At test time the agent adapts entirely in-context, without parameter updates or access to task labels or switch signals. On synthetic network-reconfiguration tasks, LINC matches strong contextual baselines when the mode is held fixed, recovers faster and with a shallower transient after unobserved switches, and degrades most gracefully under out-of-distribution mixtures, intensities, and scales.

1 INTRODUCTION

Deep RL can produce highly specialized controllers that perform well in the regime they were trained on. The same specialization becomes a liability as soon as the test environment departs from that regime: a locomotion policy fails after a broken leg or a change of terrain (Nagabandi et al., 2019), and a network-repair policy that has locked onto hub restoration keeps executing the pre-switch strategy after the failure process has silently moved to a regional outage (Figure 1). In both cases the issue is not that a good policy does not exist, but that the agent cannot identify the new dynamics *online* and retarget its behavior before the transient becomes a collapse.

We study this online-adaptation problem on adaptive network reconfiguration. Learning to control graph structure has been studied for power grids (Yoon et al., 2021) and for steering dynamical processes on graphs (Song et al., 2024), but existing learning-based reconfiguration methods assume that the failure regime is fixed within an episode or known to the agent. In deployment, hub attacks, regional failures, and random losses can switch at any time without a label. The agent then has to identify the new dynamics and retarget its repair policy, both quickly.

Gradient-based meta-RL (Finn et al., 2017; Nagabandi et al., 2019) can adapt, but only after collecting and differentiating through a new trajectory—too slow to track a switch that lasts tens of steps. Context-based methods instead decouple inference from control (Rakelly et al., 2019; Zintgraf et al., 2020; Luo et al., 2022): a context encoder compresses recent experience into a latent z_t, and the policy acts conditioned on z_t. In-context RL takes the further step of keeping all parameters frozen at test time, so that the policy improves through its input history alone (Laskin et al., 2023; Grigsby et al., 2024). What remains open is how z_t should be learned. Encoders trained to reconstruct task identities, or shaped by the training task set, become unreliable once the deployed dynamics leave that support (Wang et al., 2024; Li et al., 2024; Zhou et al., 2025), and most assume the latent stays fixed within an episode; methods that do allow intra-episode changes model the switching process explicitly (Ren et al., 2022; Chen et al., 2022).

We propose Label-free In-context Network reConfiguration (LINC). The failure mode is treated as a hidden parameter of a family of MDPs (Doshi-Velez & Konidaris, 2016; Zhang et al., 2021; Fu

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

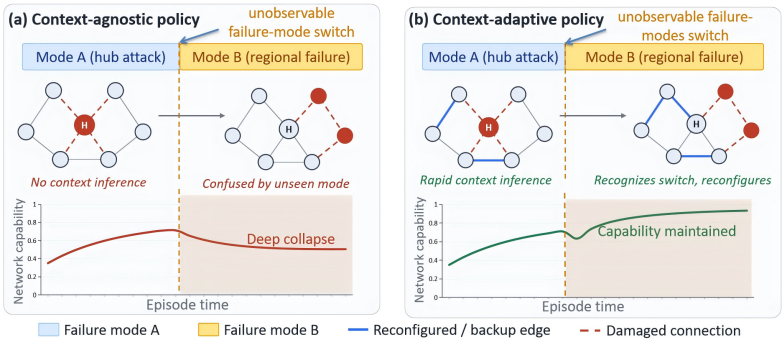

Figure 1: Under an unobservable failure-mode switch, a context-agnostic policy keeps the pre-switch repair strategy and suffers a deep capability drop (a), whereas a context-adaptive policy recognizes the switch from recent structural dynamics and maintains capability (b). Trajectories are conceptual rather than measured, in the same role as a Bayes-adaptive schematic.

et al., 2023) and is never observed. LINC infers it from how the graph responds to interventions, using three design choices: (1) a mechanism-aware graph encoder that organizes recovery-relevant structure; (2) a label-free context encoder trained only by next-token dynamics prediction with VIC-style anti-collapse, so z_t cannot latch onto training task IDs; (3) a truncated history window of length H, following the history truncation of ESCP (Luo et al., 2022), with stop-gradient into the policy, so that H steps after a switch the window contains only post-switch transitions. At test time LINC adapts purely in-context, with no gradient update, task label, or switch flag.

2 Related Work

In-context RL and latent-context meta-RL. Recurrent and sequence-model agents treat a distribution of tasks as an implicit POMDP and adapt by accumulating experience in memory, from RL^2 and attentive meta-learners through Algorithm Distillation and AMAGO (Duan et al., 2016; Mishra et al., 2018; Laskin et al., 2023; Grigsby et al., 2024). LINC is in the same in-context line (parameters stay frozen at test time) but uses a *short* window: when the mode switches inside an episode, pre-switch tokens are noise rather than memory, so the context should turn over within H steps. On the latent-context side, PEARL infers a permutation-invariant posterior over tasks (Rakelly et al., 2019) and VariBAD maintains a variational belief for Bayes-adaptive control (Zintgraf et al., 2020); ESCP accelerates encodings under sudden changes with a history-truncated recurrent encoder (Luo et al., 2022). Closest to our setting, Ren et al. (2022) treat the context as a discrete Markov chain that jumps during an episode and infer it with a Bayesian nonparametric model, and SeCBAD (Chen et al., 2022) assumes a piecewise-stable context and infers segment boundaries online; both model the switching process explicitly, whereas LINC relies on window turnover and a label-free encoder. Gradient-based meta-RL (Finn et al., 2017; Nagabandi et al., 2019) adapts by inner-loop updates, which we treat as too slow for within-episode tracking. LCPO addresses non-stationarity when context is observed and parameters keep moving (Hamadanian et al., 2025).

Hidden-parameter MDPs and network intervention. HiP-MDPs generate a family of tasks by varying a latent parameter that modulates dynamics and/or rewards (Doshi-Velez & Konidaris, 2016; Fu et al., 2023); HiP-BMDPs extend the same idea to richer observations (Zhang et al., 2021). We take the failure mode as that hidden parameter and use it only to condition the policy (policy transfer in the sense of Fu et al. (2023)), never as an input. Topology-changing control has been learned before: Yoon et al. (2021) avoid catastrophic grid states under operational constraints, and Song et al. (2024) intervene on dynamic graphs to steer event flows. Darvariu et al. (2021) learn which edge to add so as to maximize robustness to random failures or targeted attacks, with a GNN policy that transfers to larger graphs; the failure model there is fixed by the chosen objective. Learning-based reconfiguration in power and complex networks (Oh et al., 2020; Si et al., 2024; Shang et al., 2024; Zhu et al., 2026) likewise fixes the failure regime inside an episode. LINC keeps the action as "add an edge", but the object of learning is online identification of an unobserved, switching p_m.

Self-supervised representations for RL. Decoupling representation learning from policy optimization improves transfer (Stooke et al., 2021); predictive world models supply task-agnostic signals (Ha & Schmidhuber, 2018); variance-based regularization prevents collapse without negative pairs (Bardes et al., 2022). Closest to our context objective, CaDM learns a context vector from recent transitions by predicting the next state (Lee et al., 2020), DOMINO learns decomposed contexts by mutual-information maximization over transitions (Mu et al., 2022), and MoSS trains a self-supervised task representation for non-stationary adaptation (Wang et al., 2023); all target continuous-control benchmarks with fixed-size states. Predictive context learning itself is therefore not our claim. The contribution of LINC is its graph-specific integration: a scale-invariant transition token over graph summaries, a truncated window sized for intra-episode switches (after ESCP), and a structural bridging prior that the learned policy can override.

3 Problem Formulation

Following the HiP-MDP / contextual-MDP view (Doshi-Velez & Konidaris, 2016; Hallak et al., 2015; Zhang et al., 2021; Fu et al., 2023; Zintgraf et al., 2020), we consider a family of tasks indexed by a hidden parameter $m \sim p(\mathcal{M})$. Each m induces an MDP $(S, A, p_m, r, \gamma, \rho_0)$. Tasks share the state space, action space, and reward, and differ only in the transition kernel $p_m(s_{t+1} \mid s_t, a_t)$. The parameter m is never observed.

Concretely, the state s_t is a graph $G_t = (V, E_t)$ evolving under alternating damage and repair. At each step a failure process removes edges according to m—random failures, hub-targeted attacks (Albert et al., 2000), or regional outages—while the agent activates one candidate edge. Network capability $\Phi(G_t) \in [0, 1]$ (normalized algebraic connectivity and giant-component size; details in Section 5) is what the agent must keep high throughout the episode, not only at its end, much as a grid controller must avoid catastrophic operating points along the way (Yoon et al., 2021). Within a step the agent first adds an edge and the failure process then strikes, yielding G_{t+1}; the reward is the resulting increment $r_t = \Phi(G_{t+1}) - \Phi(G_t)$.

A contextual policy $\pi(a_t \mid s_t, z_t)$ conditions on a latent context z_t inferred from the recent history $\tau_{t-H:t}$, as in Bayes-adaptive control except that z_t is a point embedding rather than an explicit belief over m (Zintgraf et al., 2020). In *stationary-failure* episodes m is fixed; in *switching-failure* episodes p_m may jump to $p_{m'}$ at an unobserved time. Training samples $m \sim p(\mathcal{M})$ and keeps it fixed inside each episode. At test time parameters are frozen and the agent must adapt in-context to held-out stationary tasks, in-episode switches, and shifts in intensity, mixture, and scale. The objective is $\max_\pi \mathbb{E}_{m\sim p(\mathcal{M})}[\sum_t \gamma^t r(s_t, a_t)]$ with $a_t \sim \pi(\cdot \mid s_t, z_t)$; evaluation emphasizes the recovery *process*, not the terminal state alone.

4 Method

4.1 Overview

LINC has four modules (Figure 2): a graph encoder f_θ that embeds the current topology, a context encoder q_ϕ that infers a latent context from recent transitions, a readout head h_ψ that provides a self-supervised training signal for the context encoder, and a context-conditioned actor-critic trained with PPO (Schulman et al., 2017). At each step t, the graph encoder maps G_t to node embeddings $\{h_v\}$, a graph-level embedding g_t, and candidate-edge embeddings $\{e_{uv}\}$; a detached summary $\bar{g}_t = \mathrm{sg}(g_t)$ is written into the transition history, where $\mathrm{sg}(\cdot)$ denotes stop-gradient. The context encoder consumes the recent history $\tau_{t-H:t}$ and outputs a latent context z_t, and the policy selects a candidate edge conditioned on $(\{e_{uv}\}, \mathrm{sg}(z_t))$ together with a structural action prior. The two learning signals are kept separate: the context encoder and readout head are updated only by a self-supervised prediction loss, and the policy and value networks only by the PPO loss, with gradients stopped at z_t. Separating the two stabilizes representation learning (Stooke et al., 2021) and keeps the policy gradient from rewriting the task-inference representation. The sections below describe the graph encoder, the context encoder and its readout, the policy with its structural prior, and finally training and inference.

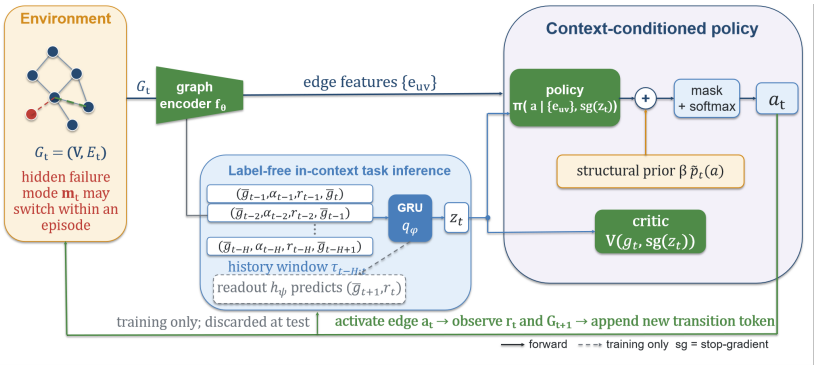

Figure 2: The architecture of LINC. The graph encoder embeds the damaged network; the GRU context encoder infers z_t from a sliding window of label-free transition tokens, trained only by self-supervised prediction; the policy scores candidate edges conditioned on $\mathrm{sg}(z_t)$ and the bridging prior, trained with PPO together with the critic.

4.2 Graph Encoder with Mechanism-Aware Features

The graph encoder adopts the graph representation used for robust network design (Zhu et al., 2026). Each node is described by mechanism-aware structural features—normalized degree, betweenness centrality, local clustering coefficient, and the size of the connected component to which it belongs—chosen because hubs and bridging nodes dominate how failures fragment a network (Albert et al., 2000). A virtual node with zero features is added and connected to every real node, and the augmented graph is processed by stacked GATv2 attention layers (Brody et al., 2022), yielding node embeddings $\{h_v\}$ together with a graph-level embedding g_t taken as the embedding of the virtual node. The embedding of each candidate edge (u, v), with endpoints in the fixed order $u < v$, combines the difference and sum of its endpoint embeddings with the global summary, $e_{uv} = [h_u - h_v;\ h_u + h_v;\ g_t]$, and candidate edges already present in G_t are masked out of the action set. The summary $\bar{g}_t$ fed into the context pathway is detached from the graph-encoder parameters, so context learning does not distort the representation used for control.

4.3 Context Encoder and Regularized Readout

Since the failure mode is never observed, LINC infers it implicitly from how the network responds to interventions. We define the transition token at step k as $x_k = (\bar{g}_k, \alpha_k, r_k, \bar{g}_{k+1})$, where α_k is a linear projection of the detached embedding of the selected edge (scale-invariant across N; a per-index lookup table is a fixed-N alternative), and the context input as the truncated history $\tau_{t-H:t} = (x_{t-H}, \ldots, x_{t-1})$, so the decision at step t uses only transitions completed before t. The token contains no failure-type label, task index, or switch indicator. A GRU (Cho et al., 2014) encodes the history into the latent context

$$z_t = q_\phi(\tau_{t-H:t}) \in \mathbb{R}^{d_z}. \tag{1}$$

We keep only the most recent H transitions because, after a switch, pre-switch tokens are stale and delay recognition of the new mode (Luo et al., 2022); long-sequence in-context agents that keep an entire rollout (Grigsby et al., 2024) have no such turnover.

So that z_t captures the transition dynamics that distinguish failure modes, we attach a readout head h_ψ during training, which predicts the next graph summary and the immediate reward from the current context, summary, and action embedding:

$$(\hat{\bar{g}}_{t+1}, \hat{r}_t) = h_\psi(z_t, \bar{g}_t, \alpha_t), \tag{2}$$

with the prediction loss over a batch B

$$\mathcal{L}_{\mathrm{dyn}} = \mathbb{E}_B\left[\|\hat{\bar{g}}_{t+1} - \bar{g}_{t+1}\|_2^2 + (\hat{r}_t - r_t)^2\right]. \tag{3}$$

Because this prediction loss is the only learning signal for q_ϕ, the context is prone to collapsing to low-variance solutions that ignore the differences between failure modes. Following VICReg

(Bardes et al., 2022), we therefore regularize each batch of contexts with a variance term that keeps every dimension of z_t above a threshold and a covariance term that decorrelates different dimensions, denoted $\mathcal{L}_{\mathrm{VC}}(z)$. The overall context objective is

$$\mathcal{L}_{\mathrm{CE}} = \mathcal{L}_{\mathrm{dyn}} + \lambda_{\mathrm{VC}}\mathcal{L}_{\mathrm{VC}}(z), \tag{4}$$

which updates only the parameters ϕ and ψ. The readout head is used exclusively during training and is discarded at inference.

4.4 Context-Conditioned Policy with Structural Prior

The actor and the critic are both conditioned on the inferred context. For each unmasked candidate edge a, the policy concatenates the edge embedding with the context and scores it with an MLP, producing learned logits $\ell_t^{\mathrm{RL}}(a) = \mathrm{MLP}_\pi([e_a; \mathrm{sg}(z_t)])$; the value head estimates $V_t = \mathrm{MLP}_V([g_t; \mathrm{sg}(z_t)])$. On top of the learned scores we add a size-weighted bridging prior taken from network science. Let $\mathrm{cid}(u)$ denote the connected component containing node u in G_t and $s(\mathrm{cid}(u))$ its size. Each candidate edge $a = (u, v)$ is scored by

$$p_t(a) = \begin{cases} \dfrac{2\, s(\mathrm{cid}(u))\, s(\mathrm{cid}(v))}{|V|^2}, & \mathrm{cid}(u) \neq \mathrm{cid}(v), \\ 0, & \text{otherwise}, \end{cases} \tag{5}$$

so that edges reconnecting two large, currently separated components receive the highest scores. The prior is standardized over the unmasked candidates to $\tilde{p}_t(a)$ (zero mean, unit variance, with a small ϵ in the denominator so that ties yield $\tilde{p}_t \equiv 0$) and added to the learned logits:

$$\ell_t(a) = \ell_t^{\mathrm{RL}}(a) + \beta\, \tilde{p}_t(a), \tag{6}$$

after which masked entries are set to $-\infty$ and the action distribution is obtained by a softmax. The prior steers exploration in the $O(|V|^2)$ action set toward edges that reconnect large components; the learned term can override it, and $\beta = 0$ gives a purely learned policy, which we ablate. The stop-gradient on z_t in both heads keeps task inference driven by the self-supervised objective in Eq. (4) rather than by the return.

4.5 Training and Inference

Training follows the protocol in the problem formulation: at the beginning of each training episode, a failure mode $m \sim p(\mathcal{M})$ is sampled and kept fixed within the episode. The agent collects on-policy rollouts in which every transition is stored together with the history snapshot used to compute its context, and the per-step reward is $r_t = \Phi(G_{t+1}) - \Phi(G_t)$, the incremental change of the network capability (instantiated in the experimental setup). After each rollout, we first update the context encoder and the readout head with $\mathcal{L}_{\mathrm{CE}}$, then recompute the contexts with the updated encoder, estimate advantages with GAE (Schulman et al., 2016), and update the policy and value networks with the clipped PPO objective with entropy regularization (Schulman et al., 2017). The behavior log-probabilities in the PPO ratio are those recorded at collection time, so the small context shift induced by the intervening $\mathcal{L}_{\mathrm{CE}}$ update is absorbed by the clipped ratio like any other off-policy drift. The whole training procedure is summarized in Appendix A.

At test time, the readout head is discarded and the policy improves entirely in-context, without updating network parameters (Laskin et al., 2023): no gradient step, task label, or switch signal is required, in contrast to gradient-based meta-RL that fine-tunes on new tasks (Finn et al., 2017; Nagabandi et al., 2019). When the failure mode switches within an episode, post-switch transitions slide into the history window, which is free of pre-switch tokens after H steps. This bounds the input to q_ϕ, not the convergence of z_t itself; how fast z_t actually reorganizes is measured empirically in Section 5.

5 Experiments

Our experiments answer four questions. **Q1:** does LINC lose less capability and recover faster than contextual and recurrent baselines after an unobserved within-episode switch? **Q2:** does this advantage persist under test-time shifts in failure composition, failure intensity, and network scale? **Q3:** how much does each component of LINC contribute? **Q4:** does the label-free context actually encode failure-mode information?

5.1 Experimental Setup

Environments and tasks. We construct synthetic networks with $N = 20$–80 nodes drawn from Barabási–Albert (Barabási & Albert, 1999), Erdős–Rényi (Erdős & Rényi, 1960), and Watts–Strogatz small-world (Watts & Strogatz, 1998) generators. These generators reproduce the heavy-tailed degree distributions and small-world clustering measured in real infrastructure—the western US power grid is an example in both original studies (Barabási & Albert, 1999; Watts & Strogatz, 1998)—and, unlike a fixed real topology, let the hidden mode, switch time, and intensity be controlled exactly, which a study of intra-episode switches requires. The network capability is measured by $\Phi_t = \frac{1}{2}\big(\lambda_2(L_t)/\lambda_2(L_{\text{healthy}}) + |\text{GCC}(G_t)|/|V|\big) \in [0, 1]$, the average of the algebraic connectivity normalized by the healthy reference network (the ratio is clipped at 1, so repairs that densify beyond the reference cannot exceed the ceiling) and the fraction of nodes in the giant connected component. Three failure modes are considered: hub attack removes edges incident to the currently highest-degree node, regional failure removes edges touching a small neighborhood around a center node, and random failure removes edges uniformly at random, each with a Poisson intensity (expected number of removed edges per step). During training, each episode samples one failure mode, its intensity, and the initial topology, with the mode fixed within the episode; the fraction of removed edges never exceeds 0.25 (Appendix C). At test time, all policy parameters are frozen, and we evaluate both stationary episodes and episodes in which the failure mode switches at time steps that are not revealed to the agent.

Evaluation metrics. AUC@50 is the average capability over the first 50 evaluation steps and measures early-stage recovery efficiency. The maximum capability drop $\Delta\Phi = \bar{\Phi}_{\text{pre}} - \min_{t_s \le t \le t_s + K} \Phi_t$ with $K = 40$ measures the transient loss caused by a switch at t_s. The recovery time t_{rec} is the number of steps required to recover 95% of the pre-switch capability; samples that have not recovered within 80 steps are right-censored at 80. We report the *restricted mean* recovery time (censored samples counted at the limit), which lower-bounds the true mean for heavily censored methods and thus understates their gap to LINC.

Baselines. We compare LINC with five baselines. ESCP (Luo et al., 2022) learns a context encoder optimized for rapid response to environment changes. PEARL (Rakelly et al., 2019) infers a probabilistic task variable; since its original form estimates the posterior once per episode, we re-estimate it from a sliding window of recent transitions at every step. PPO-GRU runs the same GRU over the same transition tokens (including reward and next graph summary) but trains it end-to-end by the PPO loss alone, without any auxiliary context objective. Prior-only greedily selects the action that maximizes the structural prior $\tilde{p}_t(a)$ without any training. Random samples uniformly among unmasked candidate edges. For fairness, all learned baselines share the same graph encoder and edge-scoring head as LINC and differ only in the adaptation mechanism; all methods use identical candidate edge sets, action masks, training tasks, interaction budgets, and evaluation protocols.

Protocol. Main comparisons report the mean over 10 training seeds, each evaluated on the same 20 pre-generated test episodes per failure mode; shaded regions denote 95% confidence intervals and error bars denote standard deviations across seeds. The stationary, switching, and distribution-shift evaluations use separately generated episode sets, so absolute values are not comparable across tables and figures. Switching test episodes last 240 steps with unobserved switches at steps $t = 80$ and $t = 160$. All learned methods are trained by sampling training episodes consecutively under identical interaction budgets: main comparisons use 500 PPO iterations (4.8×10^5 environment steps) per method. Ablation and history-length curves (Figure 5) use the same configuration but a longer budget of 1.5×10^6 environment steps to expose asymptotic behavior, and report the mean $\pm$ standard deviation over 8 seeds. Unless otherwise stated, LINC uses a history window of $H = 20$, chosen from the sweep in Section 5.4, and a prior weight of $\beta = 0.2$, chosen from a handful of pilot values on training episodes and not tuned further.

5.2 Main Results

Table 1 reports results on in-distribution stationary tasks. The three contextual methods perform comparably: ESCP is strongest under hub attacks and PEARL under regional failures, while LINC is strongest under random failures and attains the best macro-averaged Φ_T and AUC@50; the per-

Method	Hub Attack		Regional Failure		Random Failure		Macro Average	
	$\Phi_T \uparrow$	AUC@50 $\uparrow$	$\Phi_T \uparrow$	AUC@50 $\uparrow$	$\Phi_T \uparrow$	AUC@50 $\uparrow$	$\Phi_T \uparrow$	AUC@50 $\uparrow$
LINC (Ours)	0.868 ± 0.016	0.836 ± 0.018	0.861 ± 0.017	0.830 ± 0.019	$\mathbf{0.859 \pm 0.016}$	$\mathbf{0.832 \pm 0.018}$	$\mathbf{0.863 \pm 0.013}$	$\mathbf{0.833 \pm 0.015}$
ESCP	$\mathbf{0.879 \pm 0.015}$	$\mathbf{0.847 \pm 0.017}$	0.852 ± 0.019	0.820 ± 0.021	0.845 ± 0.018	0.814 ± 0.020	0.859 ± 0.015	0.827 ± 0.017
PEARL	0.864 ± 0.019	0.833 ± 0.021	$\mathbf{0.866 \pm 0.020}$	$\mathbf{0.835 \pm 0.022}$	0.838 ± 0.022	0.806 ± 0.023	0.856 ± 0.018	0.825 ± 0.020
PPO-GRU	0.786 ± 0.023	0.748 ± 0.024	0.758 ± 0.025	0.721 ± 0.026	0.772 ± 0.024	0.736 ± 0.025	0.772 ± 0.021	0.735 ± 0.022
Prior-only	0.612 ± 0.018	0.584 ± 0.019	0.634 ± 0.018	0.602 ± 0.020	0.626 ± 0.017	0.596 ± 0.019	0.624 ± 0.015	0.594 ± 0.017
Random	0.196 ± 0.030	0.263 ± 0.027	0.178 ± 0.032	0.247 ± 0.029	0.207 ± 0.031	0.272 ± 0.028	0.194 ± 0.027	0.261 ± 0.025

Table 1: Performance in in-distribution (ID) stationary environments (mean $\pm$ standard deviation over 10 seeds; the same 20 pre-generated episodes per mode for all methods). Macro Average weights the three modes equally; the best result in each column is bold.

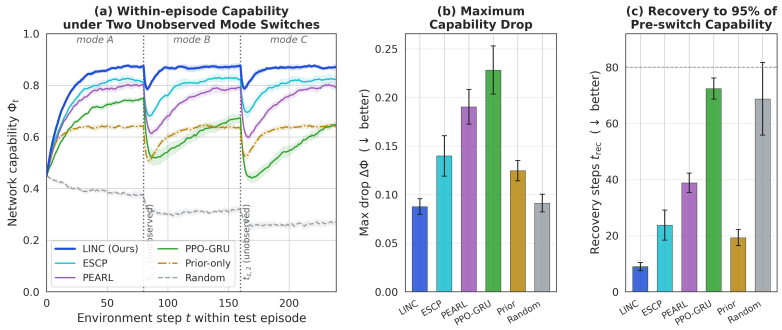

Figure 3: Adaptation under two unobserved switches, from hub attack (mode A) to regional failure (mode B) to random failure (mode C), with policy parameters frozen. (a) Per-step capability Φ_t; dotted lines mark switch times. (b) Maximum capability drop $\Delta\Phi$ (lower is better). (c) Restricted mean steps to recover 95% of pre-switch capability (lower is better); the dashed line marks the right-censoring limit of 80 steps, at which unrecovered samples are counted.

mode differences are within one standard deviation. The adaptation machinery of LINC therefore costs nothing on in-distribution stationary tasks; the differences reported below appear under switches and distribution shift.

Figure 3(a) shows online adaptation of the mean per-step capability Φ_t on a test-episode timeline with two unobserved switches (hub $\rightarrow$ regional $\rightarrow$ random at $t = 80, 160$); this is the single switching schedule evaluated in this paper. Before the first switch, the learned methods settle on a hub-attack plateau (LINC at about 0.85 on these episodes); Prior-only stays flat at about 0.62, and Random decays below 0.35. At each switch, the capability of every learned method drops abruptly. LINC has the shallowest drop and returns to its pre-switch level within roughly ten steps. ESCP recovers within a few tens of steps, whereas PEARL recovers more slowly and PPO-GRU does not return to its pre-switch level before the next switch arrives.

Figures 3(b) and 3(c) quantify this behavior, averaging each metric over the two switches. Among the learned methods, LINC attains the smallest maximum capability drop ($\Delta\Phi \approx 0.09$, versus 0.14 for ESCP, 0.19 for PEARL, and 0.23 for PPO-GRU) and recovers 95% of its pre-switch capability within about nine steps, more than twice as fast as ESCP and about four times as fast as PEARL; PPO-GRU and Random approach the right-censoring limit of 80 steps. Random and Prior-only show comparatively small drops, but only because both policies are insensitive to the failure mode and have little capability to lose.

PPO-GRU has no explicit signal for inferring the latent mode, so a switch corrupts its hidden state for a long horizon; PEARL assumes a fixed task per episode, which delays its response even with the sliding window. LINC re-identifies the mode from the most recent H transitions, so a switch becomes a brief transient.

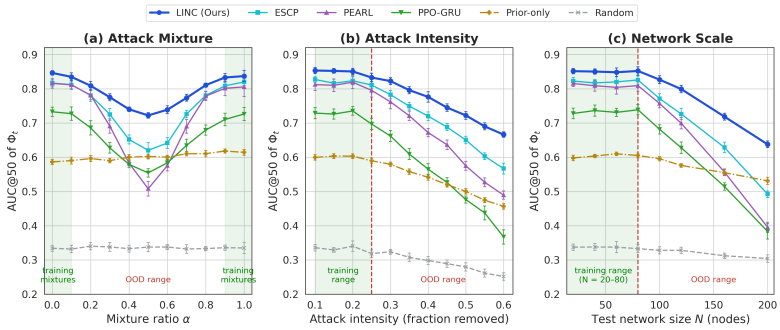

Figure 4: Generalization under three types of test-time distribution shift, measured by AUC@50. Green shading marks the training regime (pure failure modes, removal fractions up to the cap of 0.25, $N \leq 80$) and red dashed lines mark its boundaries. (a) Attack mixture. (b) Attack intensity. (c) Network scale.

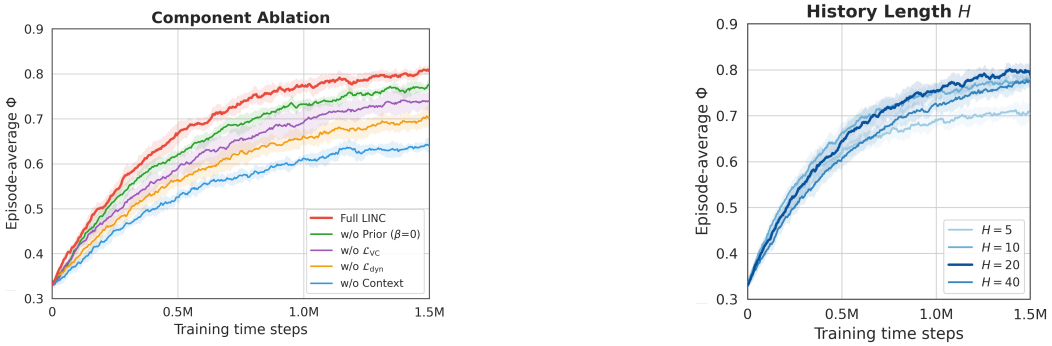

Figure 5: Training curves on stationary-mode training episodes (mean $\pm$ standard deviation over 8 seeds, plotted against environment steps). Left: component ablations of LINC. Right: history lengths $H \in \{5, 10, 20, 40\}$; $H = 20$ is the default.

5.3 Out-of-Distribution Generalization

Figure 4 reports AUC@50 under three types of test-time distribution shift, evaluated on stationary test episodes. Under attack-mixture shift (Figure 4a), where hub attacks and regional failures act simultaneously with intensities weighted by α and $1 - \alpha$ and intermediate mixtures are unseen during training, every learned method degrades toward the middle of the mixture range, but LINC remains the best throughout, declining from about 0.85 at the training mixtures to 0.73 at $\alpha = 0.5$, whereas PEARL drops most sharply to about 0.51. Under attack-intensity shift (Figure 4b), all methods degrade as the removal fraction grows beyond the training bound of 0.25; LINC degrades most gently and keeps the highest AUC@50 at the largest intensity, while PPO-GRU falls below Prior-only. Under network-scale shift (Figure 4c), LINC stays nearly flat within the training range ($N \leq 80$) and remains the best up to $N = 200$; at the largest scale the training-free Prior-only baseline even surpasses all learned baselines except LINC.

The advantage of LINC thus holds at unseen mixtures, at over twice the training intensity, and on networks $2.5\times$ larger than any seen in training. The scale result also suggests why: the bridging prior rests on connected-component statistics that are insensitive to scale, so Prior-only degrades slowly and LINC inherits this robustness, whereas purely learned baselines degrade sharply.

5.4 Ablation Studies

Figure 5 (left) shows training curves of LINC and four ablated variants. Training episodes keep the failure mode fixed (switches appear only at test time), so the curves measure learning efficiency and asymptotic return, not switch recovery. Removing any single component slows learning and lowers final performance, with gaps exceeding the shaded regions for most of training. Removing

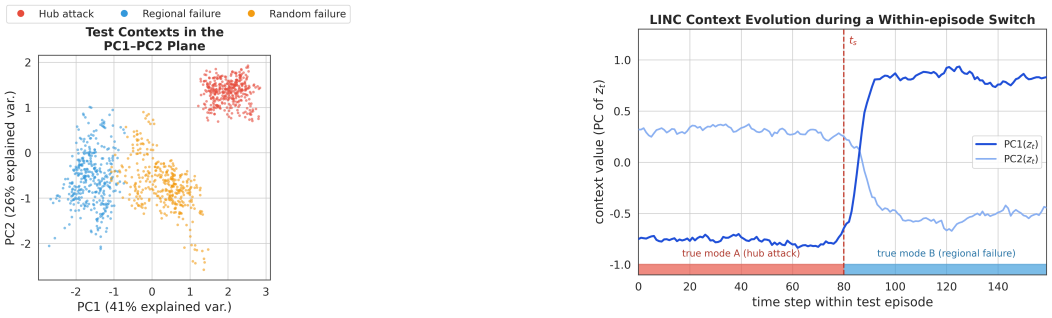

Figure 6: Label-free contexts of LINC in the PC1–PC2 plane of a single PCA fitted on training contexts and then frozen. Left: test-time contexts colored by ground-truth failure mode, for post-hoc visualization only. Right: the two components across an unobserved switch from hub attack to regional failure at t_s.

the context input hurts most, reducing the final episode-average Φ from about 0.81 to 0.64, followed by removing the predictive loss $\mathcal{L}_{\text{dyn}}$ (0.70), the variance–covariance regularizer $\mathcal{L}_{\text{VC}}$ (0.73), and the structural prior (0.77).

Without the context input the policy cannot distinguish failure modes; without $\mathcal{L}_{\text{dyn}}$ the encoder receives no signal tied to the dynamics, and without $\mathcal{L}_{\text{VC}}$ the contexts become partially redundant. The prior gives the smallest yet consistent gain here, while Figure 4(c) shows its contribution grows under scale shift.

Figure 5 (right) compares history lengths under the same protocol. $H = 5$ undershoots (final Φ about 0.71); $H = 10$ and $H = 40$ are similar; $H = 20$ is best (about 0.80) and is the default. A short window gives too little evidence to identify m; a long window dilutes post-switch transitions with stale ones and delays turnover after a test-time switch.

5.5 Context Representation Analysis

To check that the label-free context actually encodes the failure mode (Q4), we project test-time contexts z_t onto the first two principal components of a PCA fitted on training contexts and then frozen (67% of the variance explained), in the spirit of the belief plots of Zintgraf et al. (2020). In Figure 6 (left), hub-attack contexts form a compact region, while regional and random modes overlap partially; the colors are post-hoc ground-truth labels never available to the encoder. Figure 6 (right) tracks the same two components through an unobserved hub→regional switch: both stay in a band before t_s, jump within roughly ten steps after it, and settle again, matching the recovery speed in Figure 3. This is qualitative evidence only; we do not claim that the context separates all three modes.

6 Conclusion

We studied in-context meta-RL when a hidden dynamics parameter can switch inside an episode without a label, instantiated on adaptive network reconfiguration. LINC infers a label-free context by self-supervised next-step prediction, conditions a PPO policy on the detached context and a bridging prior, and adapts at test time entirely in-context. On our benchmarks it matches contextual baselines on stationary tasks, recovers faster and with a shallower drop after unobserved switches in the tested schedule, and degrades most gracefully under shifts in mixture, intensity, and scale.

Limitations. All results use synthetic topologies, stylized failures, and a single-edge action; no standard meta-RL control benchmark is included. Switching is evaluated on one mode order with fixed switch times; other orders, randomized or faster switches are untested. Ablations cover the context pathway and the prior but not the node features or the stop-gradient; no true-mode specialist bounds the optimality gap, and the prior's share of the largest-scale result (Figure 4c) is not isolated. Future work: real infrastructure graphs, flow-based capability, change-point encoders, and multi-edge repair.

AI USE STATEMENT

Generative AI coding assistants were used to write and debug parts of the experimental code under the authors' direction; the method design, experiments, and analysis are the authors' own. Generative AI tools were also used for language editing, LaTeX formatting, and literature lookup, not for generating results, figures, or numerical claims. The authors reviewed all code and text and take responsibility for them.

ETHICS STATEMENT

Experiments use synthetic graphs and simulated failures; no human subjects or personal data are involved. Policies trained only on synthetic topologies should not be deployed on real infrastructure unvalidated.

REPRODUCIBILITY STATEMENT

The training algorithm is given in Appendix A, all hyperparameters in Appendix B (Table 2), and the environment generators, test protocols, and baseline configurations in Appendix C; metrics and budgets are specified in Section 5. Anonymous source code that regenerates every figure and table is supplied as supplementary material (10 seeds for main results, 8 for ablations).

REFERENCES

Réka Albert, Hawoong Jeong, and Albert-László Barabási. Error and attack tolerance of complex networks. *Nature*, 406(6794):378–382, 2000. doi: 10.1038/35019019.

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286 (5439):509–512, 1999. doi: 10.1126/science.286.5439.509.

Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=xm6YD62D1Ub.

Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=F72ximsx7C1.

Xiaoyu Chen, Xiangming Zhu, Yufeng Zheng, Pushi Zhang, Li Zhao, Wenxue Cheng, Peng Cheng, Yongqiang Xiong, Tao Qin, Jianyu Chen, and Tie-Yan Liu. An adaptive deep RL method for non-stationary environments with piecewise stable context. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/e67a667df97d44fa43ffe8d179b44dd8-Abstract-Conference.html.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans (eds.), *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 1724–1734. ACL, 2014. doi: 10.3115/V1/D14-1179. URL https://doi.org/10.3115/v1/d14-1179.

Victor-Alexandru Darvariu, Stephen Hailes, and Mirco Musolesi. Goal-directed graph construction using reinforcement learning. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 477(2254):20210168, 2021. doi: 10.1098/rspa.2021.0168.

Finale Doshi-Velez and George Dimitri Konidaris. Hidden parameter Markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. In Subbarao Kambhampati (ed.), *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pp. 1432–1440. IJCAI/AAAI Press, 2016. URL http://www.ijcai.org/Abstract/16/206.

Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. RL^2: Fast reinforcement learning via slow reinforcement learning. *CoRR*, abs/1611.02779, 2016. URL http://arxiv.org/abs/1611.02779.

Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, 5:17–61, 1960.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135. PMLR, 2017. URL http://proceedings.mlr.press/v70/finn17a.html.

Haotian Fu, Jiayu Yao, Omer Gottesman, Finale Doshi-Velez, and George Konidaris. Performance bounds for model and policy transfer in hidden-parameter MDPs. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=20gBzEzgtiI.

Jake Grigsby, Linxi Fan, and Yuke Zhu. AMAGO: Scalable in-context reinforcement learning for adaptive agents. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=M6XWoEdmwf.

David Ha and Jürgen Schmidhuber. World models. *CoRR*, abs/1803.10122, 2018. URL http://arxiv.org/abs/1803.10122.

Assaf Hallak, Dotan Di Castro, and Shie Mannor. Contextual Markov decision processes. *CoRR*, abs/1502.02259, 2015. URL http://arxiv.org/abs/1502.02259.

Pouya Hamadanian, Arash Nasr-Esfahany, Malte Schwarzkopf, Siddhartha Sen, and Mohammad Alizadeh. Online reinforcement learning in non-stationary context-driven environments. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=l6QnSQizmN.

Michael Laskin, Luyu Wang, Junhyuk Oh, Emilio Parisotto, Stephen Spencer, Richie Steigerwald, DJ Strouse, Steven Stenberg Hansen, Angelos Filos, Ethan Brooks, Maxime Gazeau, Himanshu Sahni, Satinder Singh, and Volodymyr Mnih. In-context reinforcement learning with algorithm distillation. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=hy0a5MMPUv.

Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin. Context-aware dynamics model for generalization in model-based reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5757–5766. PMLR, 2020. URL http://proceedings.mlr.press/v119/lee20g.html.

Lanqing Li, Hai Zhang, Xinyu Zhang, Shatong Zhu, Yang Yu, Junqiao Zhao, and Pheng-Ann Heng. Towards an information theoretic framework of context-based offline meta-reinforcement learning. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/8a30aba6514b56d02976f49797f6338a-Abstract-Conference.html.

Fan-Ming Luo, Shengyi Jiang, Yang Yu, Zongzhang Zhang, and Yi-Feng Zhang. Adapt to environment sudden changes by learning a context sensitive policy. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pp. 7637–7646. AAAI Press, 2022. doi: 10.1609/AAAI.V36I7.20730. URL https://doi.org/10.1609/aaai.v36i7.20730.

Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=B1DmUzWAW.

Yao Mu, Yuzheng Zhuang, Fei Ni, Bin Wang, Jianyu Chen, Jianye Hao, and Ping Luo. DOMINO: Decomposed mutual information optimization for generalized context in meta-reinforcement learning. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/b0b1cfc8ede53f452cabf8b9cf4eef76-Abstract-Conference.html.

Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=HyztsoC5Y7.

Seok Hwa Oh, Yong Tae Yoon, and Seung Wan Kim. Online reconfiguration scheme of self-sufficient distribution network based on a reinforcement learning approach. *Applied Energy*, 280: 115900, 2020. doi: 10.1016/j.apenergy.2020.115900.

Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5331–5340. PMLR, 2019. URL http://proceedings.mlr.press/v97/rakelly19a.html.

Hang Ren, Aivar Sootla, Taher Jafferjee, Junxiao Shen, Jun Wang, and Haitham Bou-Ammar. Reinforcement learning in presence of discrete Markovian context evolution. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=CmsfC7u054S.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In Yoshua Bengio and Yann LeCun (eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1506.02438.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

Chengya Shang, Lijun Fu, Xianqiang Bao, Haipeng Xiao, Xinghua Xu, and Lufeng Liu. Dynamic fault reconfiguration of distribution networks in ship power systems based on deep reinforcement learning approach. *IEEE Transactions on Transportation Electrification*, 10(3):7076–7089, 2024. doi: 10.1109/TTE.2023.3333340.

Ruiqi Si, Siyuan Chen, Jun Zhang, Jian Xu, and Luxi Zhang. A multi-agent reinforcement learning method for distribution system restoration considering dynamic network reconfiguration. *Applied Energy*, 372:123625, 2024. doi: 10.1016/j.apenergy.2024.123625.

Zitao Song, Wendi Ren, and Shuang Li. Amortized network intervention to steer the excitatory point processes. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=8g26Yv1EOu.

Adam Stooke, Kimin Lee, Pieter Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9870–9879. PMLR, 2021. URL http://proceedings.mlr.press/v139/stooke21a.html.

Min Wang, Xin Li, Leiji Zhang, and Mingzhong Wang. MetaCARD: Meta-reinforcement learning with task uncertainty feedback via decoupled context-aware reward and dynamics components. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada*, pp. 15555–15562. AAAI Press, 2024. doi: 10.1609/AAAI.V38I14.29482. URL https://doi.org/10.1609/aaai.v38i14.29482.

Mingyang Wang, Zhenshan Bing, Xiangtong Yao, Shuai Wang, Kai Huang, Hang Su, Chenguang Yang, and Alois Knoll. Meta-reinforcement learning based on self-supervised task representation learning. In Brian Williams, Yiling Chen, and Jennifer Neville (eds.), *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pp. 10157–10165. AAAI Press, 2023. doi: 10.1609/AAAI.V37I8.26210. URL https://doi.org/10.1609/aaai.v37i8.26210.

Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998. doi: 10.1038/30918.

Deunsol Yoon, Sunghoon Hong, Byung-Jun Lee, and Kee-Eung Kim. Winning the L2RPN challenge: Power grid management via semi-Markov afterstate actor-critic. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=LmUJqB1Cz8.

Amy Zhang, Shagun Sodhani, Khimya Khetarpal, and Joelle Pineau. Learning robust state abstractions for hidden-parameter block MDPs. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=fmOOI2a3tQP.

Hongtu Zhou, Ruiling Yang, Yakun Zhu, Haoqi Zhao, Hai Zhang, Di Zhang, Junqiao Zhao, Chen Ye, and Changjun Jiang. CERTAIN: Context uncertainty-aware one-shot adaptation for context-based offline meta reinforcement learning. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*, pp. 79529–79547. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/zhou25ai.html.

Bingyu Zhu, Tianchen Zhu, Jianxi Gao, Shlomo Havlin, and Daqing Li. Design of robust networks via reinforcement learning prompts the emergence of multi-backbones. *Nature Communications*, 17(1):4278, 2026. doi: 10.1038/s41467-026-70745-0.

Luisa M. 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 *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=Hkl9JlBYvr.

A TRAINING PROCEDURE

Algorithm 1 matches the protocol in the main text: the failure mode is sampled once per training episode and kept fixed; switches occur only at evaluation.

Algorithm 1 Training of LINC

Require: task distribution $p(\mathcal{M})$; graph encoder f_θ; context encoder q_ϕ; readout head h_ψ; policy π_ω; value V; window length H; PPO hyperparameters; λ_{VC}; prior strength β

for iteration $= 1, 2, \ldots$ **do**
  Sample $m \sim p(\mathcal{M})$; reset G_0; clear the history and the buffer B
  for $t = 0, \ldots, T-1$ **do**
    $(\{h_v\}, g_t, \{e_{uv}\}, \mathrm{mask}) \leftarrow f_\theta(G_t)$; $\bar{g}_t \leftarrow \mathrm{sg}(g_t)$
    $z_t \leftarrow q_\phi(\tau_{t-H:t})$
    Compute the bridging prior $\tilde{p}_t$; sample $a_t \sim \pi_\omega(\cdot \mid \{e_{uv}\}, \mathrm{sg}(z_t))$ with the mask and prior
    Execute a_t; observe G_{t+1} and r_t; re-encode G_{t+1} to obtain $\bar{g}_{t+1}$; append x_t to the history; store the transition in B
  end for
  Update q_ϕ and h_ψ with $\mathcal{L}_{\mathrm{CE}}$ on B
  Recompute $\{z_t\}$ with the updated q_ϕ; estimate advantages with GAE
  Update π_ω and V (jointly with f_θ) with the clipped PPO objective; clear B
end for

B DEFAULT HYPERPARAMETERS

Unless noted, all learned methods share the defaults in Table 2. Training uses on-policy PPO with 8 episodes of 120 steps per iteration (960 environment steps per iteration); main comparisons run 500 iterations and the ablation/history sweeps run 1,563 iterations ($\approx 1.5 \times 10^6$ steps). Both panels of Figure 5 are plotted against environment steps. Seeds are set from the training `--seed` flag for NumPy and PyTorch.

Quantity	Default
N (train) / H / β / d_z	20–80 / 20 / 0.2 / 16
GNN width / layers / heads	64 / 3 / 8
GRU hidden / action-emb. dim.	64 / 32
PPO γ / GAE λ / clip / entropy	0.99 / 0.95 / 0.2 / 0.01
Learning rates (policy / context)	3×10^{-4} / 10^{-3}
λ_{VC} / variance-term target std.	1.0 / 1.0
Damage cap / action representation	0.25 / edge projection

Table 2: Shared hyperparameters used unless otherwise stated.

C IMPLEMENTATION DETAILS

This appendix records the settings that the main text states only in words. All of them are the defaults in the released code.

Topologies and failure processes. Each training episode samples N uniformly from $\{20, \ldots, 80\}$ and one of three generators with equal probability: Barabási–Albert with $m = 2$ attachment edges, Erdős–Rényi with edge probability $p = 4/(N-1)$ (mean degree 4), and Watts–Strogatz with $k = 4$ ring neighbours and rewiring probability 0.3. The failure intensity is the mean of a Poisson draw per step and is sampled uniformly from $[1.0, 2.5]$ during training. Hub attack removes edges incident to the single node of currently highest degree; regional failure removes edges touching a radius-1 ball around a center node sampled once per episode (the center is re-drawn if its neighborhood runs out of edges, so the mode stays active); random failure removes edges uniformly. The damage cap is

enforced by the environment: the attacker may never push $|E_t|$ below $0.75\,|E_{\text{healthy}}|$. Within a step the order is fixed: apply any scheduled (hidden) switch, add the agent's edge, let the failure process strike, compute Φ and the reward.

Test protocols. Switching episodes last $T = 240$ steps and follow the schedule hub→regional→random with switches at $t = 80$ and $t = 160$; the switch is applied by the environment at the start of the step and is never exposed to the agent. The three OOD sweeps evaluate AUC@50 on stationary episodes (Figure 4): attack mixture, where hub attack and regional failure act in the same step with intensities α and $1 - \alpha$ times the base intensity, swept over $\alpha \in [0, 1]$ in steps of 0.1; attack intensity, where the damage cap is moved from 0.10 to 0.60 and the Poisson intensity is scaled in proportion (the training point is cap 0.25); and network scale, where N is varied up to 200 at training intensity. Intensity and scale results are averaged over the three failure modes. The mixture process is used only at test time.

Graph observation and encoder. Each node carries four features, max-normalized within the graph: degree, betweenness centrality, local clustering coefficient, and the size fraction of its connected component. A virtual node receives an incoming edge from every real node and its embedding after the last GATv2 layer serves as the graph summary g_t. The candidate action set is all node pairs $u < v$ that are currently not edges; pairs that are already edges are masked with a large negative logit. Candidate edges are embedded from the two endpoint embeddings, and the action token α_k is a linear projection of the stop-gradient edge embedding to 32 dimensions, which is what makes the transition token independent of N.

Context encoder and optimization. The GRU has one layer with 64 hidden units followed by a linear map to $d_z = 16$; the readout head is a two-layer MLP with 64 hidden units. Both are trained with Adam (learning rate 10^{-3}) for 4 epochs on each rollout buffer. The actor, critic, and graph encoder are then trained with Adam (learning rate 3×10^{-4}) for 4 PPO epochs with minibatches of 128 transitions, value-loss coefficient 0.5, and gradient-norm clipping at 0.5. When fewer than H transitions are available at the start of an episode, the window is left-padded with zero tokens.

Baseline configurations. All learned baselines reuse the graph encoder, edge-scoring head, and transition tokens of LINC; only the context pathway differs. *PEARL* maps each token in the H-window to a Gaussian factor and forms the product-of-Gaussians posterior with an $\mathcal{N}(0, I)$ prior, re-estimated at every step; z is sampled during training and set to the posterior mean at test time; the encoder is trained through the PPO value loss plus a KL term with weight 0.1, and the policy sees the detached z. *ESCP* uses the same GRU as LINC but trains it with an intra-episode consistency loss (weight 1.0) that pulls contexts toward their episode mean and an RMDM-style log-determinant diversity term over episode means (weight 0.1; RMDM is the relational matrix determinant maximization of Luo et al. (2022)), again with the policy on the detached z. *PPO-GRU* runs the same GRU but trains it by backpropagation through the PPO loss alone, with no auxiliary objective and no stop-gradient. *Prior-only* takes the arg-max of the bridging prior with random tie-breaking, and *Random* samples uniformly from the unmasked candidates.