Dynamically Increasing Agents Set-Size in Bayesian Multi-agent Multi-armed Bandits Framework

Mohammad Essa Alsomali
Lancaster University
Lancaster, United Kingdom
m.alsomali@lancaster.ac.uk

Leandro Soriano Marcolino*
VinUniversity
Hanoi, Vietnam
leandro.sm@vinuni.edu.vn

Barry Porter
Lancaster University
Lancaster, United Kingdom
b.f.porter@lancaster.ac.uk

Roberto Rodrigues-Filho
Federal University of Santa Catarina
Santa Catarina, Brazil
roberto.filho@ufsc.br

ABSTRACT

Modern systems usually face environments that change over time and often unpredictably, making static or fixed agent learning strategies inadequate for sustained performance. The Likelihood-Adaptive Multi-Agent Systems (LA-MAS) introduces a novel framework for lifelong adaptation in Multi-Armed Bandits under non-stationary environments. In contrast to other dynamic environment-based approaches, LA-MAS maintains and incrementally expands a dynamic pool of agent models, each with its own Q-table and learning policy, specialized for distinct environmental characteristics. LA-MAS uses Bayesian inference to compute the likelihood with which each agent explains recently observed rewards, updating a probability distribution over all agents. When all current agents are unable to identify the current environment due to the low likelihoods, the framework detects a changepoint and automatically adds a new agent while retaining memory of previous agents for rapid re-adaptation when the environment recurs. Our empirical results on both synthetic and real-world server environments showed LA-MAS's ability to achieve a lower regret of around 50% and faster environment detection compared to state-of-the-art methods.

KEYWORDS

On-line Learning, Multi-Armed Bandits, Bayesian Inference

ACM Reference Format:
Mohammad Essa Alsomali, Leandro Soriano Marcolino, Barry Porter, and Roberto Rodrigues-Filho. 2026. Dynamically Increasing Agents Set-Size in Bayesian Multi-agent Multi-armed Bandits Framework. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/YQTC6947

1 INTRODUCTION

Multi-armed bandits (MAB) offers a fundamental framework of on-line learning and sequence decision-making where an agent is going to choose one of K actions according to its past observation to get maximum cumulative reward. In the traditional stochastic MAB problems, the rewards distributions of each action are assumed to be stationary [1, 8].

However, the real world uses of this model often incorporate non-stationary settings in which the distribution of rewards varies with time. This non-stationarity poses significant challenges to decision-making algorithms, and has been applied in recommendation systems, adaptive networking and online allocation of resources. Reward distributions in these areas are dynamically changing, so the algorithms must be capable of recognizing and responding to the change in the environment [1, 10, 19, 21].

The essence of the challenge is not just that environment statistics and the identity of optimal arms can vary, but that in many real-world systems, such changes may be unpredictable occurring suddenly or sometimes following a repeating pattern. Traditional bandit algorithms, like UCB and Thompson Sampling, are not effective in such cases because of their stationary assumption. The approach of maintaining distinct models of different contexts and dynamically updating belief weights has been shown to be promising using both changepoint-based and meta-algorithmic composition, although most have an issue of needing prior knowledge of the rate of change [13, 14].

The existing approaches for non-stationary environments face core trade-offs. Passively adaptive algorithms like discounted and sliding-window UCB forget past data [11] but are unable to detect changepoints and their performance is sensitive to hand tuned window sizes and decays in the event of abrupt change. On the other hand, change-point detectors and restarting on detecting changepoints are built into actively adaptive algorithms such as CUSUM-UCB [16], GLR-klUCB [5], and M-UCB [6], which exhibit almost optimal regret, but forget what they have learned based on their restart mechanism. More recent methods have added diminishing exploration [15] or auto-regressive [8] models to deal with certain non-stationarity structures. None of these approaches implements the multi-agent aspect and at the same time enables the agents to retain expertise from previous environments.

Therefore, we present the Likelihood-Adaptive Multi-Agent System (LA-MAS), a new approach to non-stationary Bayesian multi-agent multi-armed bandits. The framework maintains a pool of agents, where each agent corresponds to a UCB1 learner specialized to one observed environment. Rather than resetting a single learner when a change is detected, LA-MAS dynamically spawns a

[*] Corresponding author. Author also affiliated with Lancaster University

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/YQTC6947

new learner and assigns weights to all learners based on their ability to explain the newly observed rewards. The key contributions of this method are as follows: (i) **Dynamic agent specialization**: LA-MAS maintains a dynamic pool of agents, each specific to the maximization in performance in a particular stationary segment determined online by a likelihood-based changepoint detection mechanism. (ii) **Agent pool memory**: Using previously learned agents that are stored and updated enables LA-MAS to recover performance in a cyclic or recurring environment and removes the need to restart the learning process when there is a recurrence.

LA-MAS shows better scalability and coordination, with empirical results on both synthetic and real-world benchmarks that outperform state-of-the-art baselines in adaptation speed, regret reduction, and ability to cope with environment reoccurrence.

2 RELATED WORK

Lifelong Learning and Memory in Multi-Agent Bandits: Traditional multi-armed bandit models predominantly assume stationary environments or apply simple forgetting mechanisms such as sliding windows and discounting to approximate adaptation in non-stationary settings. Recent advances in lifelong learning for bandits and reinforcement learning recognize the need for agents to not only adapt online but also retain memory of past regimes, ensuring rapid re-adaptation when environments recur. Episodic memory, dynamic memory networks, and explicit retention of Q-tables associated with distinct regimes have all been explored as memory mechanisms, but many implementations still rely on manual resets or fail to scale to real-world deployments [5, 9, 12].

The LA-MAS framework distinguishes itself by introducing a principled mechanism to spawn new agents when encountering novel environments, while retaining and continuously updating existing agents' knowledge.

Moreover, recent work has made significant progress in the study of distributed multi-agent bandit algorithms within stationary environments, especially concerning optimal regret bounds via social information sharing. Madhushani and Leonard proposed protocols where agents aggregate observations from their network, either by adaptively selecting whom to observe with explicit cost constraints [18], or via heterogeneous stochastic interaction protocols [17] that reflect agent sociability. These frameworks rigorously prove logarithmic regret bounds and analytically link performance improvements to network structure and observation strategies. However, both models assume static reward distributions and environments, limiting their responsiveness to non-stationary changes.

In contrast, LA-MAS advances the field by integrating multi-agent specialization and Bayesian inference to optimize adaptation in dynamic and evolving environments. Our framework extends beyond static interactions, continually updating beliefs about the underlying state.

Changepoint Detection and Adaptive Control: Changepoint detection is a central challenge in adaptive bandit algorithms as it enables timely resets or model updates in response to abrupt environmental shifts. Existing strategies are categorized as passively adaptive methods (e.g., sliding window, discounting UCB [11], Mean Discounted Sliding-Window [7], Sliding-Window TS [20]) and actively adaptive approaches that explicitly detect changepoints using likelihood ratios, hypothesis testing, or comparison against statistical thresholds. Regarding active change detection approaches, several algorithms embed a statistical changepoint detector and restart the bandit policy when a shift is signalled. Liu et al. [16] proposed the change detection UCB (CD-UCB) framework, where the UCB indices are restarted using tests such as CUSUM or the Page–Hinkley Test; they proved that the resulting CUSUM-UCB achieves the best known regret bound for piecewise-stationary bandits. More recent frameworks like GLR-klUCB [5], and M-UCB [6], combine statistical changepoint detection with popular bandit algorithms. While these methods typically restart or reinitialize a single learner when a change is signaled, our approach differs by maintaining a dynamic pool of specialized agents and continuously updating their belief weights according to observed data. This enables both rapid adaptation and more effective reuse of prior knowledge, particularly when environments may reoccur.

Another related framework is the DAMAS-BO (Dynamic Adaptation Multi-agent Multi-armed Bandit Systems with Bayesian Optimization) [3], which also maintains a pool of agents, but relies on a pre-training phase over a fixed set of environments for initialization. During execution, it can generalize to similar unseen environments by assigning them to the most similar pre-trained agent using a soft-assignment mixture, and it benefits from Bayesian optimization for exploration. However, DAMAS-BO lacks the ability to dynamically spawn agents in response to novel environmental shifts. Our framework, by comparison, requires no pre-training and adapts in real-time through likelihood-driven agent specialization and changepoint-triggered agent creation, enabling more scalable and responsive learning in non-stationary settings.

Recent contributions, such as the Bayesian-CPD-Bandit algorithm [2], combine any multi-armed bandit policy with an online Bayesian changepoint detector. Like LA-MAS, this approach leverages online statistical inference to trigger resets or model updates, advancing adaptive control for non-stationary environments. However, a key distinction is that Bayesian-CPD-Bandit manages a single instance of the base bandit policy and resets it upon detecting changes. In contrast, our proposed approach dynamically maintains a pool of specialized agents.

3 METHODOLOGY

Motivation and Problem Formulation: we consider a *non-stationary multi-armed bandit* with K arms. Time is discrete, indexed by $t = 1, 2, \dots, T$. The environment switches among a set of unknown stationary segments (or "regimes") $\mathcal{E} = \{e_0, e_1, \dots, e_{E-1}\}$. At each segment change, the mean rewards and variance of all arms may change. (i) Let $e_t \in \mathcal{E}$ denote the *current environment* at time t. There exist unknown changepoints $1 = \tau_0 < \tau_1 < \cdots < \tau_E \leq T$ such that $e_t = e_i$ for $\tau_i \leq t \leq \tau_{i+1} - 1$. The set of changepoints is unknown to the learner. (ii) Each environment e is associated with a set of arms $\mathcal{A} = \{a_1, a_2, \dots, a_K\}$. Pulling arm $a \in \mathcal{A}$ at time t yields a reward $r_{t,a}$, drawn from a Gaussian distribution $\mathcal{N}(\mu_{e_t,a}, \sigma^2_{e_t,a})$. (iii) The learner selects an arm $a_t \in \mathcal{A}$, observes reward r_{t,a_t}, and aims to minimise the mean cumulative regret: $\mathrm{Regret}(T) = \frac{1}{T} \sum_{t=1}^{T} \left(\mu_{e_t,a^*(e_t)} - \mu_{e_t,a_t}\right)$, where $a^*(e) = \arg\max_{a \in \mathcal{A}} \mu_{e,a}$ is the optimal arm in environment e.

The challenge arises because the learner does not observe e_t or the changepoints; thus it must both identify the current environment and adapt to changes swiftly.

Many stationary-bandit algorithms use an *upper-confidence bound (UCB)* index to trade off exploration and exploitation [4]. A standard UCB1 agent maintains, for each arm a: (i) the empirical mean reward $\hat{\mu}_{t,a}$; (ii) the number of times arm a has been selected, $n_{t,a}$. At time t, the UCB1 index for arm a is: $\mathrm{UCB}_{t,a} = \hat{\mu}_{t-1,a} + c\sqrt{\frac{2\log t}{n_{t-1,a}}}$, where $c > 0$ controls exploration. The agent selects the arm with the largest $\mathrm{UCB}_{t,a}$, then updates $\hat{\mu}_{t,a}$ and $n_{t,a}$. In non-stationary settings, a single UCB1 agent struggles because past observations from previous environments bias $\hat{\mu}_{t,a}$.

Likelihood-Adaptive Multi-Agent System (LA-MAS): LA-MAS, as shown in Algorithm 1, addresses non-stationarity by maintaining a *pool of agents*, each specialised on a different stationary segment. Instead of resetting a single agent when change is detected, LA-MAS spawns a new agent while retaining past agents. A likelihood-based weighting influences which agent to trust at each time.

(i) Let $\Phi_t = \{\phi_t^{(0)}, \phi_t^{(1)}, \ldots, \phi_t^{(N_t-1)}\}$ denote the set of active agents at time t; N_t is the number of agents (initially 1). (ii) Each agent $\phi^{(i)}$ maintains a UCB1 policy with exploration constant c_i and tracks internal statistics $(\hat{\mu}_{t,a}^{(i)}, n_{t,a}^{(i)})$ for each arm a. (iii) A *belief weight vector* $\mathbf{p}_t = (p_t^{(0)}, \ldots, p_t^{(N_t-1)})$ sums to 1 and indicates how likely each agent is to match the current environment. At $t = 1$, $\mathbf{p}_1 = (1)$. (iv) LA-MAS uses a *local likelihood* function to evaluate how well each agent explains the observed reward. Specifically, once an action is taken and a reward r_{t,a_t} is observed, the system computes the likelihood of that observation under each agent's current belief, i.e., based on its estimated mean $\mu_{e,a}$ and variance $\sigma_{e,a}^2$ for the selected arm. For a Gaussian reward r, agent $\phi^{(i)}$'s local likelihood represents the probability that the observed reward falls within an adaptive tolerance band $[r - \Delta(a), r + \Delta(a)]$ under agent $\phi^{(i)}$'s current belief, where $\Delta(a) = \max\{\delta_{\min}, \kappa \cdot \hat{\sigma}(a)\}$ scales with the estimated standard deviation. Formally: $\ell_i(r;a,t) = F\left(\frac{r+\Delta(a)-\hat{\mu}_{t,a}^{(i)}}{\sigma_{t,a}^{(i)}}\right) - F\left(\frac{r-\Delta(a)-\hat{\mu}_{t,a}^{(i)}}{\sigma_{t,a}^{(i)}}\right)$, where $\sigma_{t,a}^{(i)}$ is the empirical standard deviation of rewards from arm a for agent $\phi^{(i)}$, $F(\cdot;\mu,\sigma))$ is the Gaussian cumulative distribution function, and $\Delta(a)$ is the action-dependent bandwidth that adapts to the uncertainty in each arm.

Action Selection: At each time t, the LA-MAS algorithm uses its current set of N_t internal agents $\Phi_t = \{\phi_t^{(0)}, \phi_t^{(1)}, \ldots, \phi_t^{(N_t-1)}\}$ to decide which arm to pull. Hence, at each time t, we perform the following steps: (i) *Agent-wise UCB indices*: Each agent $\phi^{(i)}$ computes UCB indices $\mathrm{UCB}_{t,a}^{(i)}$ for all arms $a \in \mathcal{A}$ using its own statistics. (ii) *Candidate actions*: Each agent recommends an arm $a_t^{(i)} = \arg\max_{a\in\mathcal{A}} \mathrm{UCB}_{t,a}^{(i)}$. (iii) *Sampling*: LA-MAS samples a single action a_t from the candidate set $\{a_t^{(i)}\}_{i=0}^{N_t-1}$ according to the weights $\mathbf{p}_t$; that is: $\Pr(a_t = a_t^{(i)}) = p_t^{(i)}$.

(iv) *Reward observation*: After an arm a_t is selected and pulled, the algorithm observes a reward r_{t,a_t}. In the subsequent update steps, this reward is used to update the statistics of all agents, weighted by their likelihood. The mechanism for updating all agents and computing these likelihoods will be discussed in detail in the following part.

Agent Update and Likelihood Weighting: After receiving reward r_{t,a_t}, LA-MAS performs: (i) *Per-agent update* (similar to DAMAS-BO [3]): Each agent $\phi^{(i)}$ updates its statistics $(\hat{\mu}_{t,a_t}^{(i)}, n_{t,a_t}^{(i)})$ as if it had observed reward r_{t,a_t} with weight $p_t^{(i)}$ (e.g., $S(\phi^{(i)}, a_t) \leftarrow S(\phi^{(i)}, a_t) + p_t^{(i)} \cdot r_t$, $N(\phi^{(i)}, a_t) \leftarrow N(\phi^{(i)}, a_t) + p_t^{(i)}$, $Q(\phi^{(i)}, a_t) \leftarrow \frac{S(\phi^{(i)}, a_t)}{N(\phi^{(i)}, a_t)}$). (ii) *Compute local likelihoods*: For the observed reward r_{t,a_t}, compute each agent's local likelihood $\ell_i(r_{t,a_t}; a_t, t)$. Enforce a minimum value $\ell_{\min}$ to prevent zero likelihoods. (iii) *Update belief weights*: Perform a Bayesian update of the posterior probabilities over agents: $p_{t+1}^{(i)} = \frac{\ell_i(r_{t,a_t};a_t,t)\, p_t^{(i)}}{\sum_{j=0}^{N_t-1} \ell_j(r_{t,a_t};a_t,t)\, p_t^{(j)}}$. Here, $\ell_i(r_{t,a_t}; a_t, t)$ is the likelihood of the observed reward under agent i's current model, and $p_t^{(i)}$ is the prior belief at time t. The resulting $p_{t+1}^{(i)}$ is the posterior belief that agent i corresponds to the current environment. If all likelihoods are extremely small, set $p_{t+1}^{(i)} = 1/N_t$ to avoid numerical instability.

Agent Spawning: When a changepoint is detected (i.e., all agents' likelihoods fall below threshold η for M consecutive steps), LA-MAS adds a new agent to Φ_t. The new agent is a fresh UCB1 instance with the same exploration constant c as the original agent. It enters a learning phase of τ steps during which its belief weight is temporarily set to 1 while it collects statistics. After the learning phase, it joins the ensemble, and the weight vector $\mathbf{p}_t$ is reset to a uniform distribution over all agents. This additive strategy retains the knowledge from previous segments, enabling quick recovery if the environment reverts.

LA-MAS-BO: Bayesian-Optimised Exploration: We also propose an extension of our approach, called LA-MAS-BO, where we perform an online Bayesian optimisation (BO) over the exploration parameter c used in UCB1. The performance of UCB1 depends on the exploration parameter c. Rather than fixing c or hand-tuning it, LA-MAS-BO performs Bayesian optimisation (BO) over c online. (i) A set of candidate values $c_1, \ldots, c_L$ are maintained. (ii) A Gaussian-process surrogate model is trained on pairs (c_j, score_j), where score_j is a performance metric (such as average response time) observed when using c_j for a time window. (iii) Every B steps, each agent uses its GP surrogate to select a new c^* that minimises a lower-confidence-bound acquisition function, then updates its exploration parameter to c^*. (iv) When LA-MAS detects a changepoint and spawns a new agent, the new agent also uses the same BO mechanism and begins with an initial exploration constant, then adapts it online via BO. (v) The LA-MAS belief weights operate over this *ensemble of agents with different c*. The BO layer thus adapts exploration to dynamics. The pseudocode for LA-MAS-BO is identical to Algorithm 1, except that when a new agent is spawned, its c value is chosen by Bayesian optimisation.

Illustrative Example: Consider $K = 3$ arms and two environments: (i) *Environment 0* ($t = 1, \ldots, 500$): mean rewards $\mu_0 = (0.2, 0.5, 0.4)$; variances $\sigma = (0.05, 0.05, 0.05)$; arm 0 is optimal. (ii) *Environment 1* ($t = 501, \ldots, 1000$): means $\mu_1 = (0.6, 0.3, 0.5)$; arm 1 becomes optimal. Step-by-step operation: (i) *Initial exploration*: At $t = 1$, LA-MAS has one agent $\phi^{(0)}$, using UCB1 with $c = 1$. After a few pulls, it estimates arm 0 as optimal, belief weight $\mathbf{p}_t = (1)$. (ii) *Environment*

```
Algorithm 1: LA-MAS
Input: Horizon $T$, action space $\mathcal{A} = \{a_1, a_2, \ldots, a_K\}$, base agent constructor $\mathcal{B}(\theta)$ (e.g., UCB1 or UCB1+BO), likelihood threshold $\eta$, minimum consecutive count $M$, learning phase length $\tau$, safety margin $\varepsilon$, bandwidth parameters $\delta_{\min}$, $\kappa$
$N \leftarrow 1$; create agent $\phi^{(1)} \leftarrow \mathcal{B}(\theta)$
$\mathbf{p} \leftarrow [1]$ // initial belief
Changepoint counter $m \leftarrow 0$; learning flag learn $\leftarrow$ false; learning timer $q \leftarrow 0$
for $t = 1$ to $T$ do
    Propose: For $i = 1$ to $N$, obtain
        $a_t^{(i)} \leftarrow \arg\max_{a \in \mathcal{A}} \mathrm{UCB}_{\phi^{(i)}}(a)$
    Act: Sample agent index $I \sim \mathrm{Categorical}(\mathbf{p})$, play $a_t \leftarrow a_t^{(I)}$, observe reward $r_t$
    Update agents: For $i = 1$ to $N$, update $\phi^{(i)}$ with $(a_t, r_t)$ weighted by $p_i$    // if available, $\phi^{(i)}$.bo_step()
    Local likelihoods: For $i = 1$ to $N$,
        $(\hat{\mu}_i, \hat{\sigma}_i) \leftarrow$ estimates for $a_t$ in $\phi^{(i)}$
        $\Delta(a_t) \leftarrow \max\{\delta_{\min}, \kappa\,\hat{\sigma}_i\}$    // adaptive bandwidth
        $\ell_i \leftarrow F\left(\frac{r_t + \Delta(a_t) - \hat{\mu}_i}{\hat{\sigma}_i}\right) - F\left(\frac{r_t - \Delta(a_t) - \hat{\mu}_i}{\hat{\sigma}_i}\right)$
    Belief update: $Z \leftarrow \sum_{i=1}^{N} \ell_i p_i$
    if $Z > \varepsilon$ then $p_i \leftarrow \ell_i p_i / Z$ for all $i$ else $p_i \leftarrow 1/N$ for all $i$
    Change test: $m \leftarrow (m + 1)$ if $\max_i \ell_i < \eta$ else $m \leftarrow 0$
    if $m \geq M$ and $\neg$ learn then
        Add new agent $\phi^{(N+1)} \leftarrow \mathcal{B}(\theta)$, $N \leftarrow N + 1$
        $\mathbf{p} \leftarrow \mathbf{e}_N$ (one-hot for new agent)
        learn $\leftarrow$ true, $q \leftarrow 0$, $m \leftarrow 0$
    end if
    if learn then
        $q \leftarrow q + 1$
        if $q \geq \tau$ then
            $p_i \leftarrow 1/N$ for all $i$; learn $\leftarrow$ false
        end if
    end if
end for
```

change at $t = 501$: The sudden shift in reward distribution causes local likelihood ℓ_0 to drop. After M consecutive events, LA-MAS spawns a new agent $\phi^{(1)}$ and assigns all weight to it: $\mathbf{p}_t = (0, 1)$. (iii) *Learning phase for* $\phi^{(1)}$: Over the next τ steps, agent 1 explores arms and finds arm 1 as optimal. After learning, belief reset: $\mathbf{p}_t = (1/2, 1/2)$. (iv) *Belief adaptation*: Subsequent rewards make ℓ_1 large and ℓ_0 small. Weights update towards $\mathbf{p}_t \approx (0, 1)$. Agent 0 is retained. and Q-values for all agents are updated.

This example illustrates LA-MAS, which includes active detection of changes, spawning new agents, and reweighting based on likelihood of the reward signal.

The methodology is further supported by a theoretical analysis with formal proofs, provided in Appendix C in the technical appendix (https://github.com/MedoEA/LA-MAS/blob/master/Technical%20Appendix.pdf).

4 EXPERIMENTAL RESULTS

We evaluated LA-MAS through both synthetic and real-world experiments, comparing it with classical and state-of-the-art approaches including *UCB1* [4], Sliding-Window UCB (*SW-UCB*) [11], and Thompson Sampling variants: Mean Discounted Sliding-Window, Sliding-Window TS (*SW-TS*) for non-stationary bandits, which forms posteriors using only the last W observations, and Discounted Thompson Sampling (*DIS-TS*) with Gaussian priors [7, 20], BO-UCB, and DAMAS-BO. Moreover, we compared our method with actively adaptive algorithms such as GLRklUCB [5] and M-UCB [6] (Our code is publicly available on GitHub at https://github.com/MedoEA/LA-MAS).

We study online selection among server configurations (actions) in non-stationary environments. Our evaluation consists of two complementary experimental tracks:

Real-world scenario: We evaluate performance with $|\mathcal{A}| = 38$ server configurations. At time t, selecting action a in environment e_t yields a delay $X_{t,a} \sim \mathcal{N}(\mu_{e_t,a}, \sigma_{e_t,a})$; we define reward $r_{t,a} = -X_{t,a}$ so higher is better.

Synthetic scenarios: To assess scalability and generalization, we conduct controlled experiments across varying numbers of actions and environments. For each synthetic configuration, we sample means $\mu_{e,a}$ from uniform distributions and variances $\sigma_{e,a}^2$ from appropriate variance distributions, enabling systematic evaluation of algorithmic performance under different problem scales and complexity levels (more details in the appendix).

All methods run for T steps per run, and we report mean ± std over multiple runs across both experimental tracks.

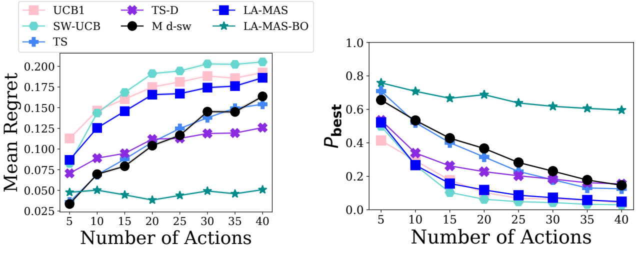
Evaluation Metrics: We report: (i) **Mean Response Time** (lower is better): running average of delays. (ii) **Mean Regret** (iii) $P_{\mathbf{best}}$: frequency of selecting the instantaneous best action $a^*(e_t) = \arg\min_a \mu_{e_t,a}$. (iv) **Environment ID accuracy** (LA-MAS variants): proportion of steps where the top posterior matches the ground-truth environment index. (v) **Changepoint alignment**: scatter of detected changes vs. true schedule changes.

Scalability Analysis: Performance Across Different Action Spaces: The scalability analysis reveals how LA-MAS variants perform as the problem complexity increases. Looking at Figure 1, it is apparent that LA-MAS maintains its performance advantage in different sizes of action spaces, with LA-MAS-BO consistently achieving lower mean regret around 50% improvement over DIS-TS for $|\mathcal{A}| = 40$, and higher probability of selecting the best action with $\approx 60\%$ and $\approx 20\%$ for TS variants.

40-Action Scenario Analysis: We provide a detailed analysis of the 40-action scenario using two different experimental setups to understand the temporal dynamics and comparative performance.

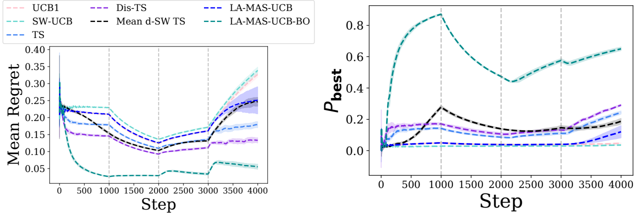
Regarding temporal performance, Figure 2 reveals the temporal evolution of performance with environment changes every 1000 time steps. The results illustrate that LA-MAS-BO maintains stable performance and faster convergence after environment changes, while other methods exhibit slower adaptation.

Active Selection Strategy Comparison: The active selection strategy provides different performance characteristics. As presented in Figure 3a, for $|\mathcal{A}| = 40$, the LA-MAS-BO achieves the lowest Mean Regret, recording a value approximately 45% lower than the second best method (GLR-klUCB). In the same evaluation setting (Figure



(a) Average performance **(b) Probability of selecting best action**

Figure 1: Performance comparison of LA-MAS variants against baseline algorithms across different numbers of actions.



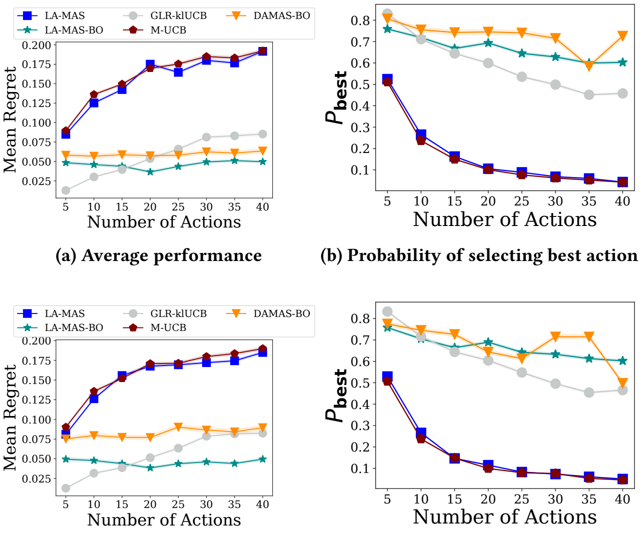
(a) Average performance **(b) Probability of selecting best action**

Figure 2: Performance evaluation with 40 actions with environment change every 1000 time steps. Comparison of mean regret and best action selection probability over time.

3a), LA-MAS achieves around 16% improvement over DAMAS-BO, which was pre-trained on four environments from the same distribution as the test set sharing the same reward ranges, but not the exact realizations used during testing.

As shown in Figure 3c, when DAMAS-BO is pre-trained on only 3 environments and tested on 4, LA-MAS-BO showed better performance and reduced regret by around 50% (for $|\mathcal{A}| = 40$) and maintained a high probability of selecting the optimal action approximately 30% compared with the DAMAS-BO as depicted in Figure 3d. The degradation reflects the fixed, soft-assignment mixture in DAMAS-BO with one unseen environment; its agents cannot specialize to the new mode, and the mixture weights spread, which becomes more costly as $|\mathcal{A}|$ increases.

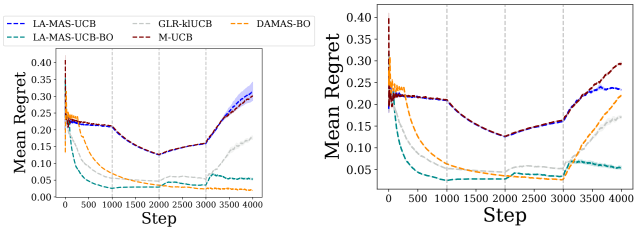
Moreover, Figure 4 shows detailed temporal analysis comparing LA-MAS variants against changepoint detection methods, several interesting aspects emerge. In Figure 4a, a clear trend of decreasing mean regret is observed for the LA-MAS-BO method following the learning phase, particularly after approximately 3000 time steps, as the system adapts to the final environment, while the GLR-klUCB exhibits an increasing trend in mean regret during the same period. Interestingly, LA-MAS-BO maintained lower regret during the DAMAS-BO pre-training phase. However, once DAMAS-BO had access to full environment knowledge, it eventually achieved lower regret than LA-MAS-BO. In Figure 4b DAMAS-BO shows high regret when one or more environments are unseen in the pre-train time.



(a) Average performance **(b) Probability of selecting best action**

(c) Average performance **(d) Probability of selecting best action**

Figure 3: Summary aggregate results showing regret and best action selection across algorithm variants



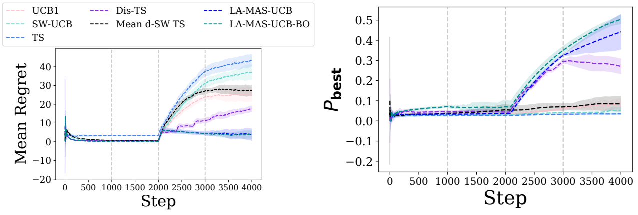
(a) Average performance **(b) Average performance**

Figure 4: Mean regret over time with $|A| = 40$. Vertical dashed lines mark environment switches. (a) DAMAS-BO full-knowledge pre-train. (b) DAMAS-BO pre-trained on 3 environments and evaluated on 4 and LA-MAS variants.

Real-World Web Server Configuration: This study evaluates LA-MAS on realistic web server scenarios, drawing upon measured statistics from four content types requests, namely: `HTML file`, `MP4 video`, `high-entropy MP4`, and `JPG`.

The workload simulates server response behaviors under multiple caching strategies (MRU, LRU, LFU, Round-Robin, file system) and compression mechanisms (GZIP/ZLIB/GZ), applied individually or combined. This creates variability in response times and entropy for adaptive decision-making *(implementation details in Appendix B)*.

Real Web-Server Benchmark: This study uses measured per-action statistics for four content types. For each type, the dataset provides per-action $\mu_{c,a}$ and $\sigma_{c,a}$. At any step, the active environment selects one content type and uses the corresponding row of



(a) Average performance **(b) Probability of selecting best action**

Figure 5: Real-world web server configuration selection using LA-MAS variants against baseline algorithms. Performance on content type distributions.

(μ, σ), thereby reflecting the diversity of server-side request patterns. Hence, in this context, each action corresponds to one of the 38 possible Web server configurations.

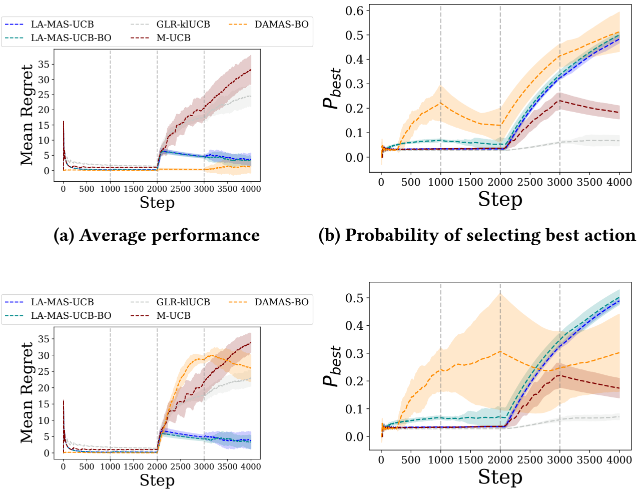
Schedule construction: In order to simulate realistic traffic patterns, we generate a schedule of content types with NASA-like class weights 0.35, 0.35, 0.15, 0.15 for HTML file, MP4 video, high-entropy MP4, and JPG respectively. And the sample *stay durations* was uniformly sampled from the interval [800, 1000] steps before re-sampling a new type by the same weights. The default time horizon for all experiments was set to T=8000 steps.

For realistic traffic patterns, we drew inspiration from the publicly available NASA HTTP Web Server Log (July 1995) dataset, commonly referred to as *NASA-log-jul95*. This dataset captures real-world request behavior.

Other Baselines vs. Active Selection Results: Figure 5 shows the performance using the selection of other baselines or passive adaptive approaches that use a fixed forgetting strategy, such as using a discounting factor or window size [5], while Figure 6 shows the results with the active selection strategy. Both LA-MAS and LA-MAS-BO approaches show a significant improvement over baseline methods in realistic web server scenarios. Interestingly, LA-MAS and LA-MAS-BO showed faster convergence particularly for complex workload type after 2000 time steps, while other methods struggle to identify the best configuration for these patterns.

In Figure 6a, DAMAS-BO was pre-trained on three environments and tested on four. In this setting, the system successfully assigned the unseen environment to the most similar agent, resulting in improved performance. In contrast, Figure 6c shows DAMAS-BO pre-trained on only two environments and tested on four. Here, the system struggled to find suitable agents for the unseen environments, which differed significantly from the ones seen during pre-training. This mismatch led to increased regret, highlighting the cost of relying on a fixed, soft-assignment mixture when generalization is limited. On the other hand, LA-MAS and its variants are able to adapt in real-time, resulting in lower regret.

Extended Traffic Pattern Analysis: In order to investigate the consistency of the results across different traffic pattern realizations, we performed extended evaluations using two distinct random seeds to initialize the request generation process. Each seed produces a different sequence of incoming web requests, simulating variability in user behavior and workload dynamics. This allows us to assess LA-MAS under various traffic conditions.



(a) Average performance **(b) Probability of selecting best action**

(c) Average performance **(d) Probability of selecting best action**

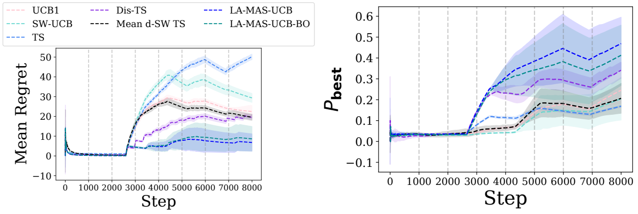
Figure 6: Real-world web server configuration selection using active strategy. Evaluation on different content types: HTML, MP4, high-entropy MP4, and JPG.

Traffic Pattern Analysis: Table 1 presents the specific content transitions for this configuration, including formats such as MP4, HTML file, JPG, and MP4 with high entropy. These transitions reflect realistic web server dynamics, where the user requests shift unpredictably between media types with varying latency, and entropy levels.

Figures 7 and 8 demonstrate how different approaches respond to these dynamics. LA-MAS and its variant consistently outperforms baseline methods in terms of regret reduction for this traffic pattern. This performance is especially evident during periods of recurring content type, where other baselines methods such as GLR-klUCB and M-UCB struggle to adapt quickly due to reset mechanisms and relearn optimal action from scratch. In contrast, LA-MAS and its variants maintain a memory of previously encountered environments through its agent-based model. When a familiar traffic pattern re-emerges, LA-MAS can rapidly reassign belief to the corresponding agent and adapt quickly.

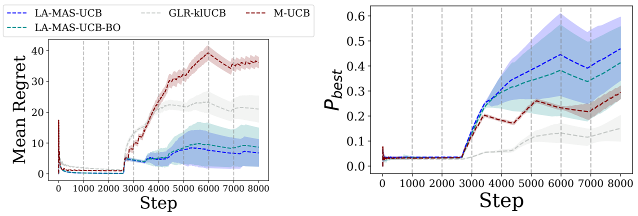
from	to	duration
MP4	HTML_text	810
HTML_text	HTML_text	867
HTML_text	MP4_high_entropy	835
MP4_high_entropy	JPG	890
JPG	MP4_high_entropy	855
MP4_high_entropy	JPG	805
JPG	HTML_text	903
HTML_text	MP4_high_entropy	959
MP4_high_entropy	MP4_high_entropy	842

Table 1: Tested transitions of web content and their durations.



(a) Average performance (b) Probability of selecting best action

Figure 7: Performance on realistic traffic pattern. Performance on NASA-like traffic patterns transitions with realistic content type distributions.



(a) Average performance (b) Probability of selecting best action

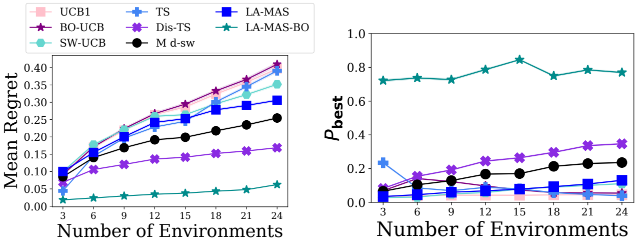
Figure 8: Performance on realistic traffic pattern using active selection. The plot shows adaptation to realistic web server workload changes.

Synthetic Multi-Environment Challenge: The most demanding evaluation uses a synthetic benchmark with 24 distinct environments, here we test two different scenarios. First, we test how the different algorithms behave across different numbers of environments to test scalability (*for setup, see Table 4 in the Appendix*). Second scenario, to test algorithm adaptability across heterogeneous environments to show generality (*for setup, see Table 5 in the Appendix*).

Scalability Testing Benchmark: To study how performance scales with the number of environments, we design a second synthetic suite with different number of environments $N \in \{3, 6, \ldots, 24\}$. Each environment e specifies per-action mean reward $\mu_{e,a}$ and noise $\sigma_{e,a}$ using uniform ranges; one action per environment is made faster and slightly noisier, to test the convergence of algorithms to optimal action. For a given N, we evaluate all N environments in order to change every 1000 step each (horizon $T = N \times 1000$), with A actions ($|\mathcal{A}| = 40$). For each N we run R independent trials ($R = 5$) and compute the average across number of trials.

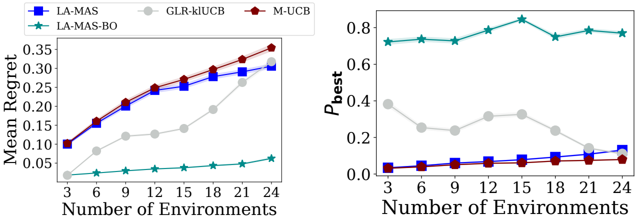
Synthetic 24-Environment Benchmark: To test adaptability across heterogeneous regimes, we synthesize $|\mathcal{E}| = 24$ environments with distinct latency patterns. Patterns include micro-latency, high-latency, log-normal/heavy-tail, beta/gamma/exponential, and high-variance environments. In each environment one randomly chosen action is made ~20% faster and slightly noisier. We evaluated the 24 environments in order for 1000 steps each (T=24000).

Performance Results: Figure 9 demonstrates that LA-MAS-BO consistently outperforms TS variants in all number of environments.



(a) Average performance (b) Probability of selecting best action

Figure 9: Summary aggregate results showing regret and best action selection across algorithm variants for different number of environments for scalability testing.



(a) Average performance (b) Probability of selecting best action

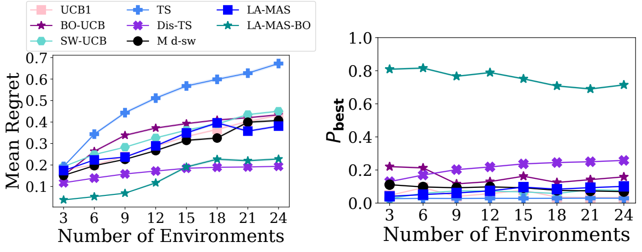
Figure 10: Summary of aggregate results showing regret and best-action selection for active-selection algorithms and LA-MAS variants across different environment counts used for scalability testing.

This finding was further supported by Figure 10, which shows that LA-MAS-BO provides the most consistent performance compared to GLR-klUCB and M-UCB. Notably, LA-MAS also showed robust performance as the number of environments increased.

The generalization analysis reveals how LA-MAS variants perform as the complexity of the problem increases. Figure 11 shows that LA-MAS-BO maintains a performance advantage across varying environment counts, achieving ≈ 50% lower mean regret than Dis-TS at $|\mathcal{E}| = 12$ and higher best-action selection probability (≈ 80% vs. ≈ 25%). However, Dis-TS outperforms LA-MAS-BO as $|\mathcal{E}|$ increases. As shown in Figure 12, LA-MAS-BO also achieves lower mean regret and higher best-action selection probability compared to GLR-klUCB and M-UCB.

Changepoint Detections Performance: To evaluate the effectiveness of different algorithms in non-stationary environments, we assess their ability to detect changepoints accurately and promptly. We measure two key metrics: detection accuracy (the proportion of true changepoints successfully identified) and detection delay (the average number of steps between a true changepoint occurrence and its detection).

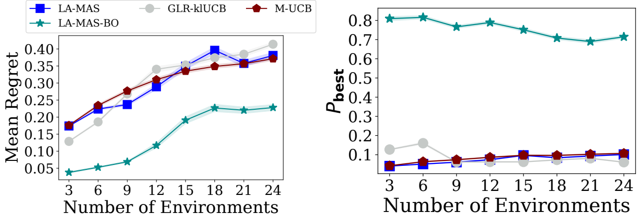
In Figure 13, each subfigure corresponds to a different number of environments specifically 3, 12, 18, and 24, to evaluate each method's ability to detect changepoints under varying complexity. The horizontal axis ("Step") represents discrete time from 0 to the end of the run, with ground-truth changepoints occurring every



(a) Average performance

(b) Probability of selecting best action

Figure 11: Performance on synthetic 24-environment suite. Evaluation across diverse latency patterns.



(a) Average performance

(b) Probability of selecting best action

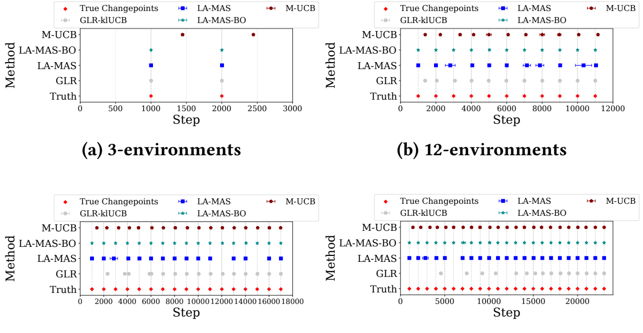
Figure 12: Performance on synthetic 24-environment suite using active selection, testing adaptability across diverse distributions.

1,000 steps. The vertical axis lists the methods as categorical rows: GLR-klUCB, LA-MAS, LA-MAS-BO, M-UCB, and the ground-truth reference ("Truth") at the bottom. Each marker's vertical position indicates the method, and its horizontal position shows the mean detected step for a given changepoint, aggregated across multiple runs (For a full performance comparison across all environment counts, please refer to Table 3 in the Appendix).

The scaling behavior across different environment counts provides insights into each algorithm's robustness in increasingly complex scenarios. Figure 13 shows that both M-UCB and LA-MAS-BO detect most changepoints, with LA-MAS-BO demonstrating rapid detection and lower delay than M-UCB.

Computational Efficiency: A 5-run benchmark over 4,000-step simulations shows LA-MAS averages 2.9–3.4s per run (1.45–3.36× slower than GLR-klUCB), while LA-MAS-BO incurs additional GP/BO overhead at 4.09–5.97s (1.88–5.80× slower). M-UCB is the most lightweight at 0.06–0.08s (16–25× faster than GLR-klUCB). As we can see in Table 2, our raw time remains low, showing that our algorithm is still suitable for real time performance. Overhead remains stable as actions scale to 160, with full timings in Table 2.

Notably, while LA-MAS-BO showed better performance in terms of regret and faster convergence through its Bayesian optimisation component, LA-MAS achieves comparable performance on real-world scenarios (e.g., the web server optimisation) at lower computational cost. This suggests a future direction, a performance efficiency trade-off that could be exploited through adaptive selection as a meta-learning level.



(a) 3-environments

(b) 12-environments

(c) 18-environments

(d) 24-environments

Figure 13: Aggregated changepoint detections across varying environment counts, with true changepoints every 1000 steps; points show the mean detected step across runs.

Actions	GLR-kLUCB	LA-MAS	LA-MAS-BO	M-UCB
20	2.080	3.369	4.670	0.082
40	2.022	2.923	5.689	0.074
60	1.014	3.408	5.576	0.063
80	1.030	3.250	5.974	0.065
100	1.006	3.186	5.505	0.066
120	1.285	3.101	5.578	0.066
140	1.197	3.431	4.095	0.077
160	2.169	3.208	4.086	0.077

Table 2: Raw wall-clock timings (seconds per *one* 4000-step)

5 CONCLUSION

We present the Likelihood-Adaptive Multi-Agent System (LA-MAS), which addresses learning in non-stationary environments by maintaining and weighting a pool of agents, each specializing in stationary segments. Through local likelihood assessment, dynamic agent spawning, and memory-based reuse, LA-MAS enables fast changepoint detection and robust re-adaptation. Its Bayesian-optimized exploration extends adaptability, achieving lower dynamic regret and environment identification errors compared to Thompson Sampling variants, Sliding-Window UCB, and active changepoint models. Empirical results show that LA-MAS consistently outperforms baselines in mean regret and optimal action selection across varying action space sizes and environment counts. LA-MAS scales effectively with more environments, maintaining high identification accuracy and rapid changepoint detection. Its practical strengths are further demonstrated in realistic web-server configuration tasks under diverse content and traffic patterns. Future work will explore bounded agent pools to address scalability in indefinitely changing environments.

ACKNOWLEDGMENTS

We wish to thank the Saudi Arabian Ministry of Education, the Saudi Arabian Cultural Bureau in London, and Qassim University for their support, including the studentship funding that enabled the completion of this research.

REFERENCES

[1] Yasin Abbasi-Yadkori, András György, and Nevena Lazić. 2023. A new look at dynamic regret for non-stationary stochastic bandits. *Journal of Machine Learning Research* 24, 288 (2023), 1–37.

[2] Reda Alami. 2023. Bayesian change-point detection for bandit feedback in non-stationary environments. In *Asian Conference on Machine Learning*. PMLR, 17–31.

[3] Mohammad Essa Alsomali, Leandro Soriano Marcolino, Barry Porter, and Roberto Rodrigues-Filho. 2025. Decision-Making in Evolving Environments: A Bayesian Multi-Agent Bandit Framework. In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*. 2402–2404.

[4] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning* 47, 2 (2002), 235–256.

[5] Lilian Besson, Emilie Kaufmann, Odalric-Ambrym Maillard, and Julien Seznec. 2022. Efficient change-point detection for tackling piecewise-stationary bandits. *Journal of Machine Learning Research* 23, 77 (2022), 1–40.

[6] Yang Cao, Zheng Wen, Branislav Kveton, and Yao Xie. 2019. Nearly optimal adaptive procedure with change detection for piecewise-stationary bandit. In *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 418–427.

[7] Emanuele Cavenaghi, Gabriele Sottocornola, Fabio Stella, and Markus Zanker. 2021. Non stationary multi-armed bandit: Empirical evaluation of a new concept drift-aware algorithm. *Entropy* 23, 3 (2021), 380.

[8] Qinyi Chen, Negin Golrezaei, and Djallel Bouneffouf. 2023. Non-stationary bandits with auto-regressive temporal dependency. *Advances in Neural Information Processing Systems* 36 (2023), 7895–7929.

[9] Srivas Chennu, Andrew Maher, Jamie Martin, and Subash Prabanantham. 2022. Dynamic Memory for Interpretable Sequential Optimisation. *arXiv preprint arXiv:2206.13960* (2022).

[10] Wang Chi Cheung, David Simchi-Levi, and Ruihao Zhu. 2019. Learning to optimize under non-stationarity. In *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 1079–1087.

[11] Aurélien Garivier and Eric Moulines. 2008. On upper-confidence bound policies for non-stationary bandit problems. *arXiv preprint arXiv:0805.3415* (2008).

[12] Matthieu Jedor, Jonathan Louëdec, and Vianney Perchet. 2020. Lifelong Learning in Multi-Armed Bandits. *arXiv preprint arXiv:2012.14264* (2020).

[13] Haozhe Jiang, Qiwen Cui, Zhihan Xiong, Maryam Fazel, and Simon S Du. 2023. A black-box approach for non-stationary multi-agent reinforcement learning. *arXiv preprint arXiv:2306.07465* (2023).

[14] Junpei Komiyama, Edouard Fouché, and Junya Honda. 2024. Finite-time analysis of globally nonstationary multi-armed bandits. *Journal of Machine Learning Research* 25, 112 (2024), 1–56.

[15] Kuan-Ta Li, Ping-Chun Hsieh, and Yu-Chih Huang. 2024. Diminishing Exploration: A Minimalist Approach to Piecewise Stationary Multi-Armed Bandits. *arXiv preprint arXiv:2410.05734* (2024).

[16] Fang Liu, Joohyun Lee, and Ness Shroff. 2018. A change-detection based framework for piecewise-stationary multi-armed bandit problem. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[17] Udari Madhushani and Naomi Ehrich Leonard. 2019. Heterogeneous stochastic interactions for multiple agents in a multi-armed bandit problem. In *2019 18th European Control Conference (ECC)*. IEEE, 3502–3507.

[18] Udari Madhushani and Naomi Ehrich Leonard. 2020. A dynamic observation strategy for multi-agent multi-armed bandit problem. In *2020 European control conference (ECC)*. IEEE, 1677–1682.

[19] Zepeng Ning and Lihua Xie. 2024. A survey on multi-agent reinforcement learning and its application. *Journal of Automation and Intelligence* 3, 2 (2024), 73–91.

[20] Han Qi, Fei Guo, and Li Zhu. 2025. Thompson Sampling for Non-Stationary Bandit Problems. *Entropy* 27, 1 (2025), 51.

[21] Anna L Trella, Walter Dempsey, Asim H Gazi, Ziping Xu, Finale Doshi-Velez, and Susan A Murphy. 2024. Non-Stationary Latent Auto-Regressive Bandits. *arXiv preprint arXiv:2402.03110* (2024).