BANCO: Drift-Aware Batched Bandits for Adaptive Proximity Graph Pruning

Jin Cheng
The Chinese University of Hong Kong, Shenzhen
Shenzhen, China
jincheng2@link.cuhk.edu.cn

Xiangxiang Dai
The Chinese University of Hong Kong
Hong Kong SAR, China
xxdai23@cse.cuhk.edu.hk

Ningning Ding
Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, China
ningningding@hkust-gz.edu.cn

John C.S. Lui
The Chinese University of Hong Kong
Hong Kong SAR, China
cslui@cse.cuhk.edu.hk

Jianwei Huang
The Chinese University of Hong Kong, Shenzhen
Shenzhen, China
jianweihuang@cuhk.edu.cn

Abstract

Proximity graphs are the state-of-the-art solution for approximate nearest neighbor (ANN) search, supporting applications such as Web search and retrieval-augmented generation (RAG). Sustaining long-term performance requires adaptive pruning as data and query workloads evolve. However, existing approaches are largely static and uniform. Adaptive pruning faces three key challenges: temporal drift in data and query distributions, spatial heterogeneity across graph regions, and costly feedback due to graph-level evaluations.

We present BANCO, a bandit-based framework for adaptive proximity graph pruning. BANCO unifies diverse pruning strategies within a common decision space and optimizes them via a drift-aware batched bandit algorithm. It addresses temporal drift through drift-aware updates, captures spatial heterogeneity using contextual features for region-specific pruning, and reduces evaluation costs through batched feedback aggregation. We establish a dynamic regret bound with sublinear loss and polynomial computational complexity. Extensive experiments on four real-world datasets demonstrate that BANCO helps maintain long-term ANN search efficiency and accuracy under evolving data and workloads.

CCS Concepts

• **Information systems** → **Information retrieval**.

Keywords

Approximate Nearest Neighbor (ANN) Search, Retrieval-Augmented Generation (RAG), Proximity Graphs, Bandit Algorithms

ACM Reference Format:

Jin Cheng, Xiangxiang Dai, Ningning Ding, John C.S. Lui, and Jianwei Huang. 2026. BANCO: Drift-Aware Batched Bandits for Adaptive Proximity Graph Pruning. In *Proceedings of the ACM Web Conference 2026 (WWW '26), April 13–17, 2026, Dubai, United Arab Emirates.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3774904.3792396

WWW '26, Dubai, United Arab Emirates

ACM ISBN 979-8-4007-2307-0/2026/04
https://doi.org/10.1145/3774904.3792396

1 Introduction

1.1 Background and Motivation

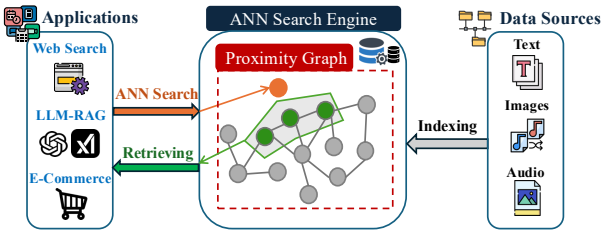


Figure 1: ANN Search Engine with Proximity Graphs.

Approximate nearest neighbor (ANN) search is a core component of large-scale intelligent systems, underpinning applications such as Web search and retrieval-augmented generation (RAG) [24, 41]. These systems rely on high-throughput semantic retrieval to efficiently identify relevant information from massive collections in real time [34]. By enabling semantic matching beyond exact keyword overlap, ANN search substantially improves retrieval relevance in modern Web and AI applications [21, 30, 36, 39].

Proximity graphs have emerged as the state-of-the-art solution for ANN search [34]. As illustrated in Figure 1, an ANN search engine bridges applications and data sources by transforming heterogeneous data—such as text, images, and audio—into high-dimensional embeddings. Within the engine, these embeddings are organized into a proximity graph that connects similar vectors based on geometric closeness. This graph-based structure enables efficient semantic retrieval across a wide range of Web and AI applications.

Jin Cheng is with the School of Science and Engineering and Shenzhen Institute of Artificial Intelligence and Robotics for Society, The Chinese University of Hong Kong, Shenzhen, and the Department of Computer Science and Engineering, The Chinese University of Hong Kong. Xiangxiang Dai and John C.S. Lui are with the Department of Computer Science and Engineering, The Chinese University of Hong Kong. Ningning Ding is with the Data Science and Analytics Thrust and the Internet of Things Thrust, Information Hub, Hong Kong University of Science and Technology (Guangzhou). Jianwei Huang is with the School of Science and Engineering, Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen Key Laboratory of Crowd Intelligence Empowered Low-Carbon Energy Network, and CSIJRI Joint Research Centre on Smart Energy Storage, The Chinese University of Hong Kong, Shenzhen. He is also affiliated with the Shenzhen Loop Area Institute (Corresponding Author).

A critical step in constructing and maintaining proximity graphs is pruning, which removes redundant or uninformative edges to control node degrees and preserve graph sparsity [16, 33]. Effective pruning improves memory efficiency, reduces retrieval latency, and maintains search quality. However, existing pruning methods typically rely on static and uniform pruning parameters [16, 32–34, 37], limiting their ability to adapt to evolving data distributions and heterogeneous graph regions, and leading to performance degradation over time.

To address these limitations, pruning decisions must adapt dynamically rather than follow fixed strategies. We propose an adaptive pruning framework that formulates pruning as a contextual bandit problem, enabling data-driven decision-making under uncertainty. The framework continuously updates pruning strategies based on observed feedback, which introduces fundamental challenges in both modeling and algorithm design.

From a modeling perspective, different proximity graphs adopt distinct pruning strategies. Monotonic Search Networks (MSNs) [34], such as NSG [16] and DiskANN [37], perform pruning globally after candidate search, refining connections across the entire graph. In contrast, Small-World Graphs (SWGs) [32], represented by HNSW [33], apply pruning locally during node insertion, maintaining sparsity in an incremental manner. In short, MSNs conduct global pruning, whereas SW graphs perform local pruning. Such diversity in pruning strategies motivates the first key question:

Key Question 1. *How to design a general adaptive pruning framework that accommodates different pruning strategies?*

From an algorithmic perspective, adaptive pruning needs to handle uncertain feedback and address three key challenges. Temporal drift: as data and query patterns evolve, static thresholds soon become outdated, degrading efficiency and accuracy. Spatial heterogeneity: graph regions differ in density and connectivity, so uniform rules cannot balance quality and efficiency. Costly feedback: pruning quality depends on graph-level metrics such as recall and latency, which are expensive and infrequent to obtain. These challenges are coupled with feedback uncertainty, which leads to the second key question:

Key Question 2. *How to design an efficient adaptive pruning algorithm that addresses temporal drift, spatial heterogeneity, and costly feedback under uncertainty?*

1.2 Key Contributions

We summarize the key contributions of this work as follows:

- General framework. We formulate pruning as a bandit problem and introduce a general adaptive framework that unifies diverse pruning strategies within a common decision space. To the best of our knowledge, this is the first general framework for adaptive pruning in proximity graphs.
- Novel bandit algorithm. We develop a drift-aware batched bandit algorithm, BANCO, for adaptive pruning. Unlike existing bandit methods that handle drifting and batching separately, our design jointly considers their interaction, which introduces new challenges in balancing temporal adaptation and delayed feedback.
- Theoretical analysis. We establish a dynamic regret bound of $\tilde{O}(T^{2/3}B_T^{1/3}d^{5/6})$ for BANCO, where T denotes the time horizon, d is the context dimension, and B_T represents the total parameter variation. This result guarantees sublinear performance loss and polynomial computational complexity, providing strong theoretical support for efficiency under drift and batched feedback.
- Experimental evaluation. We evaluate BANCO on four real-world datasets using two representative proximity graphs. The results show that adaptive pruning consistently improves the efficiency-accuracy trade-off over existing baselines, achieving up to 18% higher reward based on recall@10 and search time.

The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 introduces the preliminaries. Section 4 presents the system model and problem formulation. Section 5 describes the proposed algorithm. Section 6 provides the theoretical analysis. Section 7 presents the experimental evaluation, and Section 8 concludes the paper.

2 Related Work

This work is related to research on proximity graphs and multi-armed bandits.

2.1 Proximity Graphs

Approximate Nearest Neighbor (ANN) search is a fundamental operation in large-scale systems such as Web search and retrieval-augmented generation (RAG) [34]. While various solutions exist, including tree-based [6], hashing-based [3], and quantization-based [25] approaches, proximity graphs (PGs) have become the state of the art [22]. They offer an excellent trade-off between accuracy and latency at a billion scale by representing data as nodes in a sparse, navigable graph and enabling efficient greedy traversal.

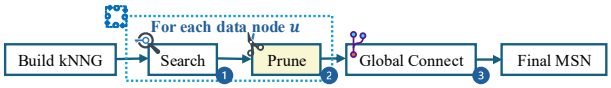
The design of PGs starts from the k-Nearest Neighbor Graph (kNNG)[15], but raw kNNGs are too dense for practical use. To address this, PGs employ pruning, which strategically removes redundant edges to create a sparse yet efficient graph structure. This pruning process distinguishes the two main families of PGs: Monotonic Search Networks (MSNs) [34], such as NSG[16] and DiskANN [37], adopt a sequential "build-search-prune-connect" pipeline with global pruning. Small-World Graphs (SWGs) [32], like HNSW [33], use an incremental "insert-search-prune-connect" paradigm where pruning is applied locally during node insertion.

Despite its importance, pruning in existing methods [34] relies on static, manually tuned thresholds. This rigidity limits their adaptability to dynamic data distributions and heterogeneous graph regions, thereby motivating our study of adaptive pruning.

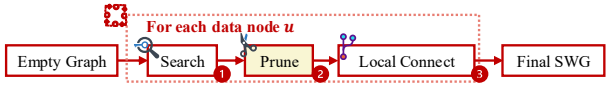
2.2 Multi-armed Bandits

The multi-armed bandit (MAB) framework provides a principled solution for adaptive decision-making by balancing exploration and exploitation[4]. Numerous extensions have been developed to improve adaptability, including kernel-based [27], Bayesian [42], combinatorial [12, 31], and hierarchical [8, 43] bandits. Recent studies have explored collaborative [13], conversational [14, 28], and federated [29] bandits in distributed environments.

Two particularly relevant directions are nonstationary bandits [38] and batched bandits [18]. Representative nonstationary bandits include sliding-window UCB [17], discounted bandits [9, 26], and change-detection-based algorithms [11], which demonstrate adaptability under temporal drift. Batched bandits [18], by updating policies only after a batch of actions, reduce feedback costs but



(a) The sequential pipeline of MSN.



(b) The incremental loop of SWG.

Figure 2: Construction workflows of proximity graphs.

introduce delayed observations. However, existing bandit methods typically handle drifting and batching separately, which limits their applicability to adaptive graph pruning.

Therefore, we introduce a drift-aware batched contextual bandit formulation that jointly models drifting environments and batched feedback. BANCO addresses the new challenges of this setting through a tailored algorithmic design.

3 Preliminaries

This section introduces the background of this work, including Approximate Nearest Neighbor (ANN) search and proximity graphs with their pruning strategies.

3.1 ANN Search

Approximate Nearest Neighbor (ANN) search is a core operation in large-scale retrieval systems, enabling fast similarity search in applications such as Web retrieval and RAG [34]. Given a query vector, ANN search retrieves the k most similar items from a large collection without exhaustive pairwise comparisons. By allowing a small approximation factor $c > 1$, ANN trades exactness for efficiency, achieving substantial speedups while maintaining high recall. This accuracy-efficiency tradeoff underpins modern large-scale retrieval systems.

3.2 Proximity Graphs and Pruning Strategies

3.2.1 Proximity Graphs. Proximity graphs (PGs) are the state-of-the-art solution for ANN search. A PG $G = (V, E)$ represents data vectors as nodes $v \in V$ and connects geometrically close vectors with edges $(u, v) \in E$. While founded on the k-Nearest Neighbor Graph (kNNG), practical PGs are not fully-connected kNNGs, which are too dense and computationally expensive. Instead, they employ pruning strategies to selectively remove redundant edges. This creates a sparse yet efficiently navigable graph that balances search performance and resource costs.

As illustrated in Figure 2, the pruning of proximity graphs for each data node u generally involves three stages: (1) search for potential neighbors, (2) prune redundant candidates, and (3) connect the remaining candidates to preserve graph connectivity. The organization of these stages distinguishes two main PG families:

- Monotonic Search Networks (MSNs), like NSG [16], use a sequential "search-prune-connect" pipeline. They identify and prune neighbors for all nodes before globally connecting them into the final graph, often using a relaxed Relative Neighborhood Graph (RNG) rule [16, 37].
- Small-World Graphs (SWGs), like HNSW [33], adopt an incremental "insert-search-prune-connect" loop. New nodes are inserted one-by-one, with neighbor search and pruning performed locally during each insertion to maintain graph properties.

Table 1: Representative Pruning Parameters.

Type	Graph	Representative Pruning Parameters
MSN	FANNG [20]	Distance threshold; Max out-degree; Entry point
	DiskANN [37]	Distance threshold; Max out-degree; $\cdots$
	NSG [16]	Pool size; Max out-degree; Entry point
SWG	NSW [32]	Number of neighbors; Number of trials; $\cdots$
	HNSW [33]	Number of neighbors; Pool size; Entry point

3.2.2 Pruning Strategies. The effectiveness of pruning strategies is governed by a range of parameters, summarized in Table 1, which can be categorized into three groups.

(1) *Search Parameters.* These parameters control the quality of neighbor candidates obtained during greedy traversal in proximity-graph construction. Greedy search is widely used in graph indexes such as NSG [16] and HNSW [33], where traversal starts from an entry point (ep) and iteratively explores neighbors toward the target. The candidate pool size (L) determines the number of neighbors maintained during search.

(2) *Pruning Parameters.* These parameters filter candidate neighbors based on geometric constraints to balance graph sparsity and connectivity. Typical examples include distance or diversity thresholds (α/β), which relax strict relative-neighbor rules to retain additional edges and produce denser graphs.

(3) *Connection Parameters.* These parameters determine the final graph topology and memory efficiency. A representative example is the maximum out-degree (M), which limits the number of outgoing edges per node and directly controls graph sparsity.

However, these critical parameters are typically manually configured, limiting the pruning strategy's adaptability to evolving data distributions and workload dynamics. To address this limitation, we next introduce a framework for adaptive pruning.

4 System Model and Problem Formulation

This section introduces the overall framework of adaptive pruning. Section 4.1 models adaptive pruning as a sequential decision-making process, and Section 4.2 formalizes the corresponding optimization problem.

4.1 Adaptive Pruning Strategy

As discussed in Section 3.2, existing pruning strategies rely on static parameters that cannot adapt to evolving data distributions or heterogeneous graph regions. To address this limitation, we formulate pruning as a contextual bandit problem for sequential decision-making in adaptive pruning optimization.

We consider a time horizon of T. Each time step $t \in \{1, \ldots, T\}$ corresponds to processing a target node u_t within the "for each data node" loop shown in Figure 2. At step t, the system observes the current graph state (context), selects pruning parameters (action), applies them to u_t, and receives a reward reflecting the pruning quality. We next define the key components of this formulation.

4.1.1 Context. Before pruning node u_t, the system extracts spatial and structural information that characterizes its local graph environment, which is encoded as a context vector $\boldsymbol{x}_t \in \mathbb{R}^m$:

$$\boldsymbol{x}_t = [e_t^{u-}, e_t^{u+}, e_t^{c-}, e_t^{c+}, e_t^{g-}, e_t^{g+}, \alpha_t^m, \alpha_t^s, m_t, s_t, n_t], \quad (1)$$

where e_t^{u-}/e_t^{u+} are the in/out degrees of u_t, e_t^{c-}/e_t^{c+} the average degrees of nodes in the candidate set P_t, and e_t^{g-}/e_t^{g+} the global

averages. α_t^m/α_t^s and m_t/s_t denote the mean and variance of angles and distances within P_t, respectively, and $n_t = |P_t|$.

4.1.2 Action. Given the context $\boldsymbol{x}_t$, the system selects an action $\boldsymbol{a}_t$, which specifies a particular set of pruning parameters introduced in Section 3.2.2. The action space $\mathcal{A}$ contains a finite set of candidate parameter configurations.

The system discretizes continuous parameters into grids and constructs multi-dimensional actions as tuples from their Cartesian product. Each action $a_t \in \mathcal{A}$ therefore defines a concrete and executable pruning policy. For example, in HNSW, an action may take the form $a_t = (M, L, ep) = (16, 128, \#2)$, specifying a maximum out-degree of 16, a candidate pool size of 128, and an entry point of #2.

We summarize the detailed parameters in Table 1 and Section 3.2.2. This formulation unifies these configurations into a single decision space, enabling a general and adaptive pruning framework.

4.1.3 Reward. The reward $r_t \in \mathbb{R}$ quantifies the utility of the pruning action $\boldsymbol{a}_t$ for node u_t, which affects the graph performance in ANN search, and is modeled as:

$$r_{t,\boldsymbol{a}_t} = \langle \phi(\boldsymbol{x}_t, \boldsymbol{a}_t),\ \boldsymbol{\theta}_t \rangle + \eta_t, \tag{2}$$

where $\boldsymbol{\theta}_t \in \mathbb{R}^d$ is a time-varying reward coefficient vector, and η_t denotes zero-mean sub-Gaussian noise. The joint feature map $\phi(\boldsymbol{x}_t, \boldsymbol{a}_t) \in \mathbb{R}^d$ encodes both context and action by concatenating context features with one-hot encoded action parameters. The reward measures each decision's contribution to search quality and efficiency, defined as the normalized recall minus a normalized query latency.

We employ a linear reward model for analytical tractability and efficient estimation, a common practice in the bandit literature [4]. This formulation can be naturally extended to a generalized linear model (GLM) [40] by applying a nonlinear link function $g(\cdot)$, i.e., $r_{t,\boldsymbol{a}_t} = g(\langle \phi(\boldsymbol{x}_t, \boldsymbol{a}_t), \boldsymbol{\theta}_t \rangle) + \eta_t$, thereby supporting more complex relationship modeling within the same theoretical framework.

By mapping the node-wise pruning process into sequential decision-making steps, we establish an adaptive pruning framework.

4.2 Problem Formulation

We formulate sequential pruning decisions (i.e., actions) at each time step as a contextual bandit problem. The performance of adaptive pruning is measured by the cumulative dynamic regret R_T, which measures the expected reward gap between the optimal action and the algorithm's action over a horizon T:

$$R_T = \mathbb{E}\left[\sum_{t=1}^{T} \left(\max_{a \in \mathcal{A}} \langle \phi(x_t, a), \boldsymbol{\theta}_t \rangle - \langle \phi(x_t, a_t), \boldsymbol{\theta}_t \rangle \right) \right]. \tag{3}$$

Thus, minimizing R_T is equivalent to maximizing the cumulative reward obtained by adaptive pruning over time.

Formally, we aim to solve the following optimization problem:

$$\min_{\{a_t\}_{t=1}^T} \quad R_T \qquad (\mathbb{P}_1)$$
$$\text{s.t.} \quad a_t \in \mathcal{A}, \quad t = \{1, \ldots, T\}.$$

However, the coefficient vector $\boldsymbol{\theta}_t$ is unknown and evolves over time with changes in data distributions and query workloads. An effective algorithm must therefore estimate and track $\boldsymbol{\theta}_t$ through sequential decision-making while addressing three fundamental challenges inherent in adaptive graph pruning. First, $\boldsymbol{\theta}_t$ exhibits temporal drift, requiring continuous adaptation rather than convergence to a static strategy. Second, spatial heterogeneity in the context $\boldsymbol{x}_t$ reflects diverse graph structures (e.g., dense versus sparse regions), making the optimal pruning action context-dependent and necessitating a model that learns the relation between $\boldsymbol{x}_t$ and pruning decisions. Third, the reward r_t is costly to obtain, as it depends on graph-level metrics (e.g., recall@k) that are available only intermittently.

We formalize the adaptive pruning framework and define dynamic regret as the optimization objective. These formulations motivate a specialized algorithm that jointly addresses key challenges to minimize regret, as introduced in the next section.

5 Algorithm Design

This section introduces BANCO, a drift-aware batched contextual bandit algorithm developed to solve Problem $\mathbb{P}_1$. We first introduce the core idea in Section 5.1, then detail the algorithm in Section 5.2, and finally analyze its computational complexity and scalability in Section 5.3.

5.1 Core Idea

BANCO operates in two stages. It first makes adaptive pruning decisions on the proximity graph (Stage I: Adaptive Decision Making) and then updates the underlying parameters, including the time-varying coefficient $\boldsymbol{\theta}_t$, through Stage II: Batched Parameter Update, forming a feedback loop that continuously refines future decisions.

Balancing decision-making and parameter learning is central to BANCO. The algorithm must exploit its current estimate of $\boldsymbol{\theta}_t$ for effective pruning while exploring to refine this estimate. To achieve this balance, BANCO employs the Upper Confidence Bound (UCB) principle [5], which evaluates each action by combining its expected reward with an uncertainty-based exploration term.

To specifically address the three key challenges of optimization $\mathbb{P}_1$, BANCO tailors the UCB framework as follows:

Addressing Temporal Drift: To handle the time-varying reward parameter $\boldsymbol{\theta}_t$, BANCO incorporates a discount factor γ into a regularized weighted least-squares estimator $\hat{\boldsymbol{\theta}}_t$ for $\boldsymbol{\theta}_t$:

$$\hat{\boldsymbol{\theta}}_t = \arg\min_{\boldsymbol{\theta} \in \mathbb{R}^d} \left(\sum_{s=1}^{t} \gamma^{t-s} \left(r_{s,\boldsymbol{a}_s} - \langle \phi(\boldsymbol{x}_s, \boldsymbol{a}_s), \boldsymbol{\theta} \rangle \right)^2 + \lambda \|\boldsymbol{\theta}\|_2^2 \right), \tag{4}$$

where $\gamma \in (0, 1]$ discounts older observations, giving more weight to recent rewards and enabling adaptation to drift. The regularization term $\lambda > 0$ ensures numerical stability and keeps the covariance matrix invertible.

Exploiting Spatial Heterogeneity: The algorithm leverages the context vector $\boldsymbol{x}_t$ from Eq. (1), which encodes local and global graph structure. By modeling the reward as a linear function of the joint context $\phi(\boldsymbol{x}_t, \boldsymbol{a}_t)$ in Eq. (2), BANCO learns context-aware reward estimates that reflect spatial variability. This enables pruning decisions that adapt to local graph density and workload dynamics, with UCB scores computed per context-action pair.

Navigating Costly Feedback: Evaluating pruning quality is costly, so BANCO employs a batched feedback scheme. It makes decisions over a block of rounds and updates parameters once using the aggregated rewards, amortizing evaluation cost and improving practicality for real-world systems.

Algorithm 1: BANCO: Bandit for Adaptive Pruning

Input : Horizon T; Number of batches M; Action space $\mathcal{A}$; Feature map ϕ; Regularization $\lambda > 0$; Discount factor $\gamma \in (0,1)$; Failure probability $\delta \in (0,1)$; Bound on feature norm L; Bound on parameter norm S; Sub-Gaussian noise parameter σ.

Output : Pruning action a_t for each round t

/* Initialization */

1 $t_0 \leftarrow 0, V_0 \leftarrow \lambda I_m, \tilde{V}_0 \leftarrow \lambda I_m, \boldsymbol{b}_0 \leftarrow \mathbf{0} \in \mathbb{R}^m, \hat{\boldsymbol{\theta}}_0 = \mathbf{0}$

2 Compute initial exploration parameter β_0 using Eq. (10)

3 **for** $m \leftarrow 1$ **to** M **do**

4 $t_m \leftarrow \lceil \frac{mT}{M} \rceil$

/* Stage I: Adaptive Decision Making */

5 **for** $t \leftarrow t_{m-1} + 1$ **to** t_m **do**

6 Receive context $\boldsymbol{x}_t \in \mathbb{R}^d$

7 Compute UCB score for each candidate action $\boldsymbol{a} \in \mathcal{A}$ using Eq. (5)

8 Select action $\boldsymbol{a}_t \leftarrow \arg\max_{a \in \mathcal{A}} \mathrm{UCB}(\boldsymbol{x}_t, \boldsymbol{a})$

9 Apply action $\boldsymbol{a}_t$ and store the pair $(\phi(\boldsymbol{x}_t, \boldsymbol{a}_t), t)$

10 **end**

/* Stage II: Batched Parameter Update */

11 Receive batched rewards $\{r_{t,\boldsymbol{a}_t}\}_{t=t_{m-1}+1}^{t_m}$

12 Update V_{t_m}, $\boldsymbol{b}_{t_m}$, and $\tilde{V}_{t_m}$ using Eqs. (6), (7), and (9)

13 Update parameter estimate $\hat{\boldsymbol{\theta}}_{t_m}$ using Eq. (8)

14 Update exploration parameter β_{t_m} using Eq. (10)

15 **end**

This design also allows BANCO to be seamlessly integrated into existing ANN systems built on proximity graphs, such as HNSW [33] and NSG [16], by replacing static pruning thresholds with learned bandit policies while preserving the original search pipeline.

5.2 Algorithm Details

Algorithm 1 presents a two-stage procedure that alternates between adaptive decision making and batched parameter updates. We next describe the algorithm in detail.

Initialization. The algorithm begins by initializing its state at $t = 0$ (Lines 1–2). It takes as input three key hyperparameters: the regularization parameter $\lambda > 0$, the discount factor $\gamma \in (0,1)$ controlling adaptation to non-stationarity, and the failure probability $\delta \in (0,1)$ for confidence bounds. The covariance matrix V_0 and auxiliary matrix $\tilde{V}_0$ are initialized as λI_m, where I_m is the $m \times m$ identity matrix; V_t accumulates discounted feature outer products with regularization λI_m to ensure invertibility, while $\tilde{V}_t$ adjusts the confidence bounds for temporal drift. The cumulative reward vector $\boldsymbol{b}_0$ and parameter estimate $\hat{\boldsymbol{\theta}}_0$ are initialized to zero, with $\hat{\boldsymbol{\theta}}_t$ denoting the estimated coefficient of the true parameter $\boldsymbol{\theta}_t$. The exploration-exploitation trade-off is controlled by the parameter β_0, defined later in Eq. (10).

Stage I: Adaptive Decision Making. The algorithm iterates through M batches (Line 3). Within each batch $m \in \{1, 2, \cdots\}$ (from round $t_{m-1} + 1$ to t_m), the algorithm makes real-time pruning decisions (Lines 4-8). At each round t, upon receiving a context $\boldsymbol{x}_t$, BANCO evaluates all candidate actions $a \in \mathcal{A}$ by computing an Upper Confidence Bound (UCB) score. The UCB score for an action $\boldsymbol{a}$ given context $\boldsymbol{x}_t$ is defined as:

$$\mathrm{UCB}(\boldsymbol{x}_t, \boldsymbol{a}) = \langle \phi(\boldsymbol{x}_t, \boldsymbol{a}), \hat{\boldsymbol{\theta}}_{t_{m-1}} \rangle + \beta_{t_{m-1}} \|\phi(\boldsymbol{x}_t, \boldsymbol{a})\|_{V_{t_{m-1}}^{-1} \tilde{V}_{t_{m-1}} V_{t_{m-1}}^{-1}} \tag{5}$$

The first term represents the predicted reward based on the parameter estimate $\hat{\boldsymbol{\theta}}_{t_{m-1}}$ from the previous batch. The second term quantifies the uncertainty associated with action $\boldsymbol{a}$ under context $\boldsymbol{x}_t$. The matrix-weighted norm $\|\cdot\|_{V^{-1}\tilde{V}V^{-1}}$ and the exploration parameter $\beta_{t_{m-1}}$ jointly prioritize actions that are either highly uncertain or have large estimated rewards. The algorithm then greedily selects the action $\boldsymbol{a}_t$ with the highest UCB score (Line 8) and applies it for graph pruning (Line 9).

Stage II: Batched Parameter Update. After completing all rounds in a batch, the algorithm updates its model to estimate the time-varying parameter $\boldsymbol{\theta}_{t_m}$ and compute the exploration coefficient β_{t_m}, which together guide better pruning decisions.

The algorithm receives batched rewards $r_{t,\boldsymbol{a}_t}$ (Line 11). Eqs. (6)-(8) give the recursive solution to the weighted least-squares problem in Eq. (4), ensuring consistency between the updates and the estimator definition (Lines 12-14). The parameter estimate $\hat{\boldsymbol{\theta}}_{t_m}$ is obtained via discounted regularized least squares:

$$V_{t_m} = \gamma^{\Delta} V_{t_{m-1}} + \sum_{s=t_{m-1}+1}^{t_m} \gamma^{t_m - s} \phi(\boldsymbol{x}_s, \boldsymbol{a}_s) \phi(\boldsymbol{x}_s, \boldsymbol{a}_s)^\top + (1 - \gamma^{\Delta}) \lambda I_m \tag{6}$$

$$\boldsymbol{b}_{t_m} = \gamma^{\Delta} \boldsymbol{b}_{t_{m-1}} + \sum_{s=t_{m-1}+1}^{t_m} \gamma^{t_m - s} \phi(\boldsymbol{x}_s, \boldsymbol{a}_s) r_{s,\boldsymbol{a}_s} \tag{7}$$

$$\hat{\boldsymbol{\theta}}_{t_m} = V_{t_m}^{-1} b_{t_m} \tag{8}$$

where $\Delta = \lceil T/M \rceil$ is the batch size .We also define an auxiliary matrix $\tilde{V}$ for variance adjustment in the confidence bounds for temporal drift:

$$\tilde{V}_{t_m} = \gamma^{2\Delta} \tilde{V}_{t_{m-1}} + \sum_{s=t_{m-1}+1}^{t_m} \gamma^{2(t_m - s)} \phi(\boldsymbol{x}_s, \boldsymbol{a}_s) \phi(\boldsymbol{x}_s, \boldsymbol{a}_s)^\top + (1 - \gamma^{2\Delta}) \lambda I_m \tag{9}$$

Then, the exploration parameter β_{t_m} for the next batch is updated. β_{t_m} is crucial for ensuring that the true coefficient $\boldsymbol{\theta}_t$ lies within the confidence ellipsoid with high probability, defined as:

$$\beta_{t_m} = \sqrt{\lambda} S + \sigma \sqrt{2 \log \frac{1}{\delta} + d \log \left(1 + \frac{(1 - \gamma^{2t_m})}{\lambda d (1 - \gamma^2)} \right)} \tag{10}$$

Here, S denotes an upper bound on the norm of the true parameter $\boldsymbol{\theta}_t$ ($|\boldsymbol{\theta}_t|_2 \le S$), d denotes the context dimension, and δ specifies the probability of failure. This setting balances the effects of regularization, noise, and data volume to adaptively control exploration, and Section 6.1 provides its theoretical justification.

By separating decision-making and parameter update into two distinct stages, BANCO efficiently handles different challenges.

5.3 Complexity Analysis

We analyze the computational complexity of BANCO to demonstrate its efficiency. Both time and space requirements grow polynomially with the input size, ensuring scalability to large-scale problems. Key parameters include the context dimension d, action space size $|\mathcal{A}|$, total rounds T, number of batches M, and batch size $\Delta = \lceil T/M \rceil$.

Theorem 1 (Computational Complexity of BANCO)**.** *The overall computational cost of* BANCO *grows polynomially with the context dimension d, the size of the action space $|\mathcal{A}|$, the total number of rounds T, the number of batches M, and the batch size $\Delta = \lceil T/M \rceil$. Specifically, the algorithm achieves:*

$$\textit{Time Complexity:} \quad O(T \cdot |\mathcal{A}| \cdot d^2 + M \cdot d^3),$$

$$\textit{Space Complexity:} \quad O(d^2 + \Delta \cdot d).$$

The proof of Theorem 1 is given in Appendix A.1. The time complexity scales linearly with horizon T and cubically with dimension d, ensuring constant amortized cost per round. The space complexity grows quadratically with d and linearly with the batch size Δ, remaining practical for moderate dimensions. Hence, BANCO is both computationally and memory efficient, enabling large-scale deployment and seamless integration into existing ANN systems.[1]

This section introduces the BANCO algorithm, detailing its design and showing that it runs within polynomial time and space. We next analyze its theoretical performance by deriving an upper bound on the dynamic regret, which provides regret guarantees for solving the adaptive pruning problem $\mathbb{P}_1$.

6 Theoretical Analysis

This section provides the theoretical analysis of BANCO, focusing on parameter estimation and regret guarantees. Section 6.1 analyzes the estimation of the time-varying parameter $\boldsymbol{\theta}_t$ and establishes confidence regions that support the UCB-based action selection in Eq. (5). Section 6.2 then derives a dynamic regret bound for Problem $\mathbb{P}_1$, quantifying BANCO's performance.

6.1 Confidence Bound Analysis

We derive high-probability confidence bounds for the time-varying parameter $\boldsymbol{\theta}_t$, forming the theoretical basis of BANCO. The main challenge lies in the coupling of temporal drift and batched feedback: drift requires continuous adaptation, whereas batched feedback delays learning and limits timely updates. To address this, we introduce an auxiliary covariance matrix W_t with squared discount factors (γ^2), distinct from the matrix V_t (with single powers of γ), enabling the construction of drift-aware confidence ellipsoids.

Standard concentration results do not hold under delayed feedback. We therefore adopt a Gaussian smoothing technique [5] to preserve the required supermartingale property, yielding the concentration inequality in Proposition 1.

Proposition 1 (Concentration of Weighted Noise)**.** *For any $\delta > 0$, with probability at least $1-\delta$, the weighted noise term $S_t = \sum_{s=1}^{t} \gamma^{t-s} \phi(\boldsymbol{x}_s, \boldsymbol{a}_s) \eta_s$ satisfies:*

$$\|S_t\|^2_{\widetilde{V}_t^{-1}} \le \sigma^2 \left(2 \log \frac{1}{\delta} + \log \frac{\det(\widetilde{V}_t)}{\det(\lambda I)} \right), \tag{11}$$

where $\widetilde{V}_t = W_t + \lambda I_m$ is the regularized noise-weighted covariance matrix and $W_t = \sum_{s=1}^{t} (\gamma^{t-s})^2 \phi(\boldsymbol{x}_s, \boldsymbol{a}_s) \phi(\boldsymbol{x}_s, \boldsymbol{a}_s)^\top$.

The proof of Proposition 1 is provided in Appendix A.2. This result bounds the deviation of the discounted noise term S_t with high probability, linking the probabilistic analysis to the statistical confidence framework. Under the stationary case where $\boldsymbol{\theta}_t = \boldsymbol{\theta}$, we further derive the following confidence bound for the estimator $\hat{\boldsymbol{\theta}}_{t_m}$, forming the theoretical foundation for defining coefficient β_{t_m} and will be used in the subsequent dynamic regret analysis.

Theorem 2 (Confidence Bound of $\boldsymbol{\theta}_t$)**.** *For any $\delta \in (0,1)$, with probability at least $1-\delta$, the following holds simultaneously for all $m \in \{1, \dots, M\}$ batches:*

$$\left\| \hat{\boldsymbol{\theta}}_{t_m} - \boldsymbol{\theta} \right\|_{V_{t_m} \widetilde{V}_{t_m}^{-1} V_{t_m}} \le \sqrt{\lambda} S + \sigma \sqrt{ 2 \log \tfrac{1}{\delta} + m \log \left(1 + \frac{\sum_{s=1}^{t_m} \gamma^{2(t_m - s)}}{\lambda^m} \right) }. \tag{12}$$

The proof of Theorem 3 is provided in Appendix A.3. The theorem ensures that the batch-updated estimator $\hat{\boldsymbol{\theta}}_{t_m}$ lies within a statistically valid ellipsoid around the true parameter $\boldsymbol{\theta}$. The radius of this ellipsoid, determined by λ, σ, and the determinant term from Proposition 1, defines the exploration coefficient β_{t_m} in Eq. (5) and serves as the analytical basis for the subsequent regret analysis.

6.2 Regret Analysis

We next derive the dynamic regret bound for BANCO, quantifying its performance in non-stationary environments. Our analysis first establishes a key parameter inclusion property for the confidence bound, and then uses it to derive the final regret bounds.

6.2.1 Parameter Inclusion. The algorithm's trade-off between exploration and exploitation is governed by a confidence ellipsoid C_m, whose size is controlled by the exploration parameter $\beta_{t_{m-1}}$ from Eq. (10):

$$C_m = \left\{ \boldsymbol{\theta} \in \mathbb{R}^d : \|\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}_{t_{m-1}}\|_{V_{t_{m-1}} \widetilde{V}_{t_{m-1}}^{-1} V_{t_{m-1}}} \le \beta_{t_{m-1}} \right\}. \tag{13}$$

The validity of this UCB approach hinges on ensuring that a proxy of the true parameter lies within the confidence set with high probability. The following proposition formalizes this inclusion property, bridging our confidence analysis with the regret framework.

Proposition 2 (Parameter Inclusion with Confidence)**.** *Let $\bar{\boldsymbol{\theta}}_{t_m} = V_{t_{m-1}}^{-1} (\sum_{s=1}^{t_{m-1}} \gamma^{t_{m-1}-s} \phi(\boldsymbol{x}_s, \boldsymbol{a}_s) \phi(\boldsymbol{x}_s, \boldsymbol{a}_s)^\top \boldsymbol{\theta}_s + \lambda \boldsymbol{\theta}_{t_{m-1}})$. For any $\delta > 0$, we have $\mathbb{P}(\bar{\boldsymbol{\theta}}_{t_m} \in C_m) \ge 1 - \delta$.*

The proof of Proposition 2 is provided in Appendix A.4. This inclusion property provides the statistical guarantee required for the UCB-based exploration strategy. With this established, we can now derive the overall dynamic regret.

6.2.2 Dynamic Regret Bound. We now combine the preceding results to establish the main regret guarantee of Problem $\mathbb{P}_1$.

Let S_A, S_P, and S_W denote upper bounds on the norms of actions, parameters, and auxiliary operators, respectively. Specifically, we set $\|\boldsymbol{a}_t\|_2 \le L$, $\|\boldsymbol{a}_t \boldsymbol{a}_t^\top\|_{\text{op}} \le S_A$, and $\|\boldsymbol{\theta}_t\|_2 \le S_P$ for all t. The constant S_W serves as an upper bound on the weighted operator norm $\|V_t^{-1} \sum_{s=1}^{t} \gamma^{t-s} \boldsymbol{a}_s \boldsymbol{a}_s^\top\|_{\text{op}}$. We also assume the total parameter variation satisfies $\sum_{s=1}^{T} \|\boldsymbol{\theta}_s - \boldsymbol{\theta}_{s+1}\|_2 \le B_T$. These boundedness conditions are standard in non-stationary bandit analysis [2, 10, 19], and are natural in our setting: actions correspond to discrete pruning parameters with finite ranges (e.g., maximum degree, pool size), model coefficients $\boldsymbol{\theta}_t$ capture reward weights that are stable up to gradual drift, and B_T quantifies the smooth evolution of data distributions and query workloads similar to [2, 10]. Under these assumptions, we obtain the following theorem.

[1] In practice, the cubic dependence on d can be substantially reduced by maintaining a Cholesky factor L_t such that $V_t = L_t L_t^\top$. This allows rank-1 updates to be performed in $O(d^2)$ per step instead of recomputing a full matrix inverse, resulting in an amortized time complexity of $O(T\,|\mathcal{A}|\,d^2 + \frac{M}{K} d^3)$ with periodic refactorization every K batches.

Theorem 3 (Regret Bound of BANCO). *For all $\gamma \in (0,1]$ and integers $D, N \geq 1$, with probability at least $1-\delta$,*

$$R_T \leq \left(2L(D+N)S_A + \sqrt{\lambda}S_P\right)B_T + (2S_W + 4S_A)\frac{S_P}{\lambda}\frac{\gamma^D}{1-\gamma} + \beta_T \cdot \sqrt{2d\log\left(1+\frac{N}{\lambda d}\right)} \cdot \frac{T}{\sqrt{N}}. \tag{14}$$

The proof of Theorem 4 is provided in Appendix A.5. Theorem 4 provides a dynamic regret bound comprising three interpretable components: a variation term for tracking the evolving parameter, a discounting term capturing the memory-adaptability trade-off, and an exploration term for managing uncertainty.

6.2.3 Optimized Regret Rate. By appropriately tuning the algorithm's hyperparameters, we achieve the following regret bound.

Corollary 1 (Subline Regret). *Under Theorem 3, if we set $\gamma = 1 - c_\gamma(\frac{B_T}{T})^{2/3}m^{-1/3}$, $N = \lceil c_N(\frac{T}{B_T})^{2/3}m^{1/3}\rceil$, and $D = \lceil c_D N \log T\rceil$, where c_γ, c_N, and c_D are positive constants controlling the discount factor, batch size, and update interval, respectively, then with high probability,*

$$R_T = \tilde{O}\left(T^{2/3}B_T^{1/3}d^{5/6}\right), \tag{15}$$

where $\tilde{O}(\cdot)$ suppresses logarithmic factors.

The proof is given in Appendix A.6. This bound guarantees sublinear regret, i.e., the average regret $R_T/T \to 0$ as $T \to \infty$.

This section establishes both the confidence bound and the dynamic regret bound, ensuring the theoretical soundness of BANCO under temporal drift. These guarantees provide a basis for the following experimental evaluation of its practical performance.

7 Experimental Evaluation

This section provides an experimental evaluation of the proposed BANCO algorithm. We outline the experimental setup in Section 7.2 and present the analysis in Section 7.2.

7.1 Experimental Setup

7.1.1 Datasets. We evaluate BANCO on four real-world datasets covering multiple modalities, as summarized in Table 2.

Table 2: Description of experimental datasets.

Dataset	Modality	Description
GIST1M [23]	Visual	1M 960-dimensional vectors for content-based image retrieval.
SIFT1M [23]	Visual	1M 128-dimensional SIFT descriptors used in image matching.
MSONG [7]	Auditory	1M audio feature vectors from the Million Song Dataset based on timbre and chroma descriptors.
GLOVE1.2M [35]	Textual	1.2M 100-dimensional word embeddings trained using the GloVe algorithm.

7.1.2 Baselines. We implement two representative proximity-graph families: HNSW [33] from the small-world family and NSG [16] from the monotonic-search family. For each, we compare the original algorithm and its adaptive version: HNSW-A and NSG-A, where adaptive pruning is guided by BANCO. We also include two ablations—w/o drift-aware, which removes the discount factor γ and disables temporal adaptation, and w/o context, which ignores contextual information and prevents adaptation to local structure.

7.1.3 Metrics. We evaluate performance primarily using the reward, which aligns with the optimization objective of Problem $\mathbb{P}_1$. In our implementation, the reward of each pruning decision is defined as a trade-off between search quality and efficiency—specifically, the normalized recall minus a weighted normalized query latency for the node under consideration.

For search quality, we use Recall@10, defined as the ratio between the number of correctly retrieved neighbors among the top 10 ANN results and the total of 10 ground-truth nearest neighbors. This metric, ranging from 0 to 1, measures the proportion of correctly retrieved neighbors. For efficiency, we report the average query latency (in milliseconds), normalized by the maximum latency observed within each experiment to enable fair comparison.

7.2 Result Analysis

We evaluate the effectiveness of BANCO across four real-world benchmark datasets, focusing on three key aspects: overall performance against static baselines, robustness to key parameters, and the contribution of its core components via ablation studies. This section presents and discusses the empirical results.

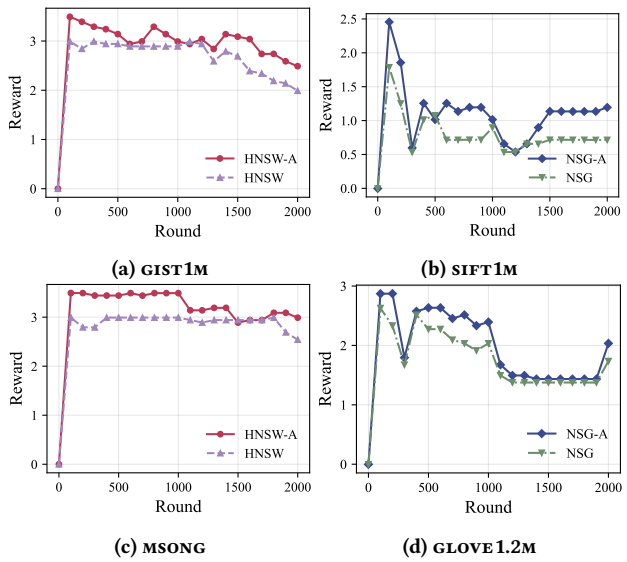
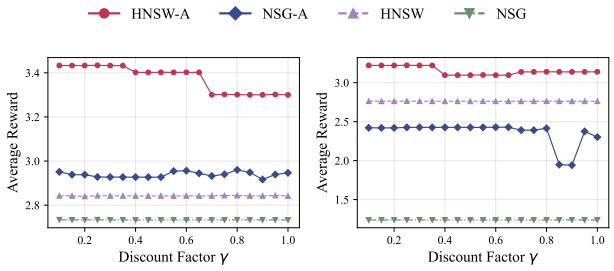


Figure 3: Average reward comparison on four datasets.

7.2.1 Overall Performance. This section evaluates the effectiveness of BANCO by comparing its augmented proximity graphs (HNSW-A, NSG-A) against their original static versions. Figure 3 plots the moving average of rewards over time, where a higher value signifies a better trade-off between search recall and latency. The results consistently demonstrate the superiority of the adaptive pruning approach. The BANCO-augmented variants achieve and sustain higher rewards, confirming that dynamic adaptation is critical for achieving robust long-term performance in evolving environments.

A key insight is the clear learning process exhibited by the adaptive models. For instance, while the static NSG baseline's performance gradually declines due to temporal drift, NSG-A adapts to quickly surpass it and maintain a substantial advantage (Figures 3b and 3d). Similarly, HNSW-A establishes a stable lead over its static counterpart; on MSONG, its average reward remains above 3.0

while HNSW's fluctuates around 2.9 (Figure 3c). Moreover, the reward curves for the adaptive variants are noticeably smoother and more stable post-initial exploration. This indicates that BANCO not only improves absolute performance but also enhances robustness by effectively responding to changes, mitigating the degradation to which static methods are prone. Overall, these results confirm that BANCO successfully addresses temporal drift and spatial heterogeneity to deliver consistently higher and more reliable performance.



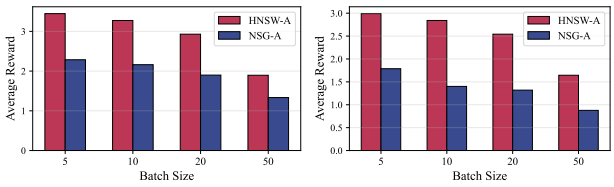
(a) GIST1M (b) SIFT1M

Figure 4: Study of discount factor γ.

7.2.2 Parameter Study. We evaluate the sensitivity of BANCO to two key hyperparameters: the discount factor γ, which governs temporal adaptation, and the batch size M, which balances feedback timeliness and computational cost.

Figure 4 shows the effect of γ on the average reward for HNSW-A and NSG-A across GIST1M and SIFT1M. Performance peaks at moderate values ($0.6 \le \gamma \le 0.8$), reflecting a trade-off between adaptability and stability. A small γ causes overreaction to recent noise, while γ near 1.0 slows adaptation to drift, as observed for NSG-A on SIFT1M. This confirms the necessity of controlled discounting for stable learning in non-stationary environments.

Figure 5 examines the effect of batch size. Smaller batches (e.g., 5–10) yield higher rewards than larger ones (e.g., 50) since more frequent updates enable faster adaptation. Larger batches delay feedback and reduce responsiveness, though the batched scheme amortizes computation effectively. Overall, smaller batch sizes provide a favorable balance between adaptation speed and efficiency.



(a) GIST1M (b) SIFT1M

Figure 5: Study of batch size.

7.2.3 Ablation Study. We conduct an ablation study on GLOVE1.2M dataset to evaluate the contribution of each component addressing temporal drift, spatial heterogeneity, and costly feedback. Two simplified variants of BANCO are compared: -OD (without drift) disables temporal adaptation by setting $\gamma = 1$, and -OC (without context) removes contextual features. As shown in Figure 6 and Table 3, the full models (HNSW-A and NSG-A) consistently outperform both variants, confirming the necessity of each component.

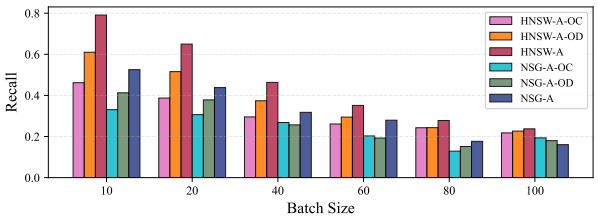


Figure 6: Ablation study on GLOVE1.2M.

As shown in Figure 6 and Table 3, the drift-aware mechanism plays a key role: removing temporal adaptation results in a noticeable performance drop—e.g., at a batch size of 10, HNSW-A-OD achieves a recall of 0.610, which is about 23% lower than HNSW-A (0.791). Disabling contextual features causes an even larger degradation (HNSW-A-OC: 0.462 recall), highlighting the importance of local graph information for region-specific pruning. Finally, the batched feedback scheme shows the expected trade-off: performance decreases with larger batch sizes due to delayed adaptation, while smaller batches enable more responsive updates.

Table 3: Ablation study on GLOVE1.2M.

Method	Batch Size					
	10	20	40	60	80	100
HNSW-A-OC	0.462	0.387	0.295	0.261	0.243	0.218
HNSW-A-OD	0.610	0.516	0.374	0.295	0.243	0.226
HNSW-A	0.791	0.650	0.463	0.351	0.278	0.237
NSG-A-OC	0.331	0.307	0.268	0.203	0.129	0.194
NSG-A-OD	0.413	0.378	0.257	0.193	0.151	0.180
NSG-A	0.525	0.438	0.318	0.279	0.177	0.160

8 Conclusion

This paper presents BANCO, a bandit-based framework for adaptive proximity graph pruning. By formulating pruning as a drift-aware batched bandit problem, BANCO jointly addresses temporal drift, spatial heterogeneity, and costly feedback, and adaptively learns context-specific pruning policies within a unified decision space. We establish a dynamic regret bound with sublinear loss and polynomial computational complexity. Extensive experiments on real-world datasets demonstrate that BANCO sustains ANN search efficiency and accuracy under evolving data and query workloads. Future work includes exploring richer context representations and adaptive reward designs to further balance accuracy and efficiency in dynamic environments.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (Project 62271434), Shenzhen Key Laboratory of Crowd Intelligence Empowered Low-Carbon Energy Networks (No. ZDSYS20220606100601002), the Shenzhen Stability Science Program 2023, Shenzhen Loop Area Institute, and Shenzhen Institute of Artificial Intelligence and Robotics for Society. The work of Xiangxiang Dai is supported by the National Natural Science Foundation of China (Project 625B2163). The work of Ningning Ding is supported by the National Natural Science Foundation of China (Project 62502412). The work of John C.S. Lui is supported in part by the GRF-14202923.

References

[1] Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. 2011. Improved algorithms for linear stochastic bandits. *Advances in neural information processing systems* 24 (2011).

[2] Axel Abels, Vito Trianni, Ann Nowé, and Tom Lenaerts. 2025. Collective Intelligence in Decision-Making with Non-Stationary Experts. *Journal of Artificial Intelligence Research* 83 (2025).

[3] Alexandr Andoni and Piotr Indyk. 2008. Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions. In *FOCS*. 459–468.

[4] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time Analysis of the Multiarmed Bandit Problem. In *Machine Learning*, Vol. 47. 235–256.

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

[6] Jon Louis Bentley. 1975. Multidimensional Binary Search Trees Used for Associative Searching. *Commun. ACM* 18, 9 (1975), 509–517.

[7] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. 2011. The Million Song Dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*.

[8] Jin Cheng, Xiangxiang Dai, Ningning Ding, John Lui, and Jianwei Huang. 2025. Trading Vector Data in Vector Databases. *arXiv preprint arXiv:2511.07139* (2025).

[9] Jin Cheng, Ningning Ding, John CS Lui, and Jianwei Huang. 2025. OSTOR: Online Scheduling Framework for Trading Continuous Queries. In *2025 IEEE 41st International Conference on Data Engineering (ICDE)*. IEEE Computer Society, 1895–1909.

[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] Wai-Shun Cheung and David Simchi-Levi. 2019. Learning to Optimize under Non-Stationarity. In *Proceedings of the 32nd COLT Conference*. 1075–1104.

[12] Xiangxiang Dai, Xiaowei Sun, Jinhang Zuo, Xutong Liu, and John C.S. Lui. 2025. A Unified Online-Offline Framework for Co-Branding Campaign Recommendations. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*. New York, NY, USA, 427–438.

[13] Xiangxiang Dai, Zhiyong Wang, Jize Xie, Xutong Liu, and John CS Lui. 2024. Conversational recommendation with online learning and clustering on misspecified users. *IEEE Transactions on Knowledge and Data Engineering* (2024).

[14] Xiangxiang Dai, Zhiyong Wang, Jize Xie, Tong Yu, and John CS Lui. 2024. Online Learning and Detecting Corrupted Users for Conversational Recommendation Systems. *IEEE Transactions on Knowledge and Data Engineering* 36, 12 (2024), 8939–8953.

[15] Wei Dong, Moses Charikar, and Kai Li. 2011. Efficient K-Nearest Neighbor Graph Construction for Generic Similarity Measures. In *Proceedings of the 20th International Conference on World Wide Web (WWW)*. ACM, 577–586.

[16] Cong Fu, Chao Xiang, Changxu Wang, and Deng Cai. 2020. Fast Approximate Nearest Neighbor Search with the Navigating Spreading-out Graph. *Proceedings of the VLDB Endowment* 14, 1 (2020).

[17] Aurélien Garivier and Eric Moulines. 2011. On Upper-Confidence Bound Policies for Non-Stationary Bandit Problems. In *Proceedings of the 22nd International Conference on Algorithmic Learning Theory (ALT)*. 174–188.

[18] Quanquan Gu, Amin Karbasi, Khashayar Khosravi, Vahab Mirrokni, and Dongruo Zhou. 2024. Batched neural bandits. *ACM/JMS Journal of Data Science* 1, 1 (2024), 1–18.

[19] Hengquan Guo, Qi Zhu, and Xin Liu. 2025. Safe Learning in Stochastic Continuum-Armed Bandit With Constraints and Its Application to Network Resource Management. *IEEE Transactions on Networking* (2025).

[20] Ben Harwood and Tom Drummond. 2016. FANNG: Fast Approximate Nearest Neighbour Graphs. In *Proceedings of CVPR Workshops*.

[21] Rong Huang, Danfeng Zhang, Weixue Lu, Han Li, Meng Wang, Daiting Shi, Jun Fan, Zhicong Cheng, Simiu Gu, and Dawei Yin. 2023. Learning discrete document representations in web search. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4185–4194.

[22] Subramanya Jayaram, Badrish Chandramouli, Guna Prasaad, and et al. 2022. DiskANN: Fast Accurate Billion-Point Nearest Neighbor Search on a Single Node. In *Proceedings of the 16th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. 971–987.

[23] Herve Jegou, Matthijs Douze, and Cordelia Schmid. 2010. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence* 33, 1 (2010), 117–128.

[24] Sungjun Jung, Yongsang Park, Haeun Lee, Young H Oh, and Jae W Lee. 2025. Angular Distance-Guided Neighbor Selection for Graph-Based Approximate Nearest Neighbor Search. In *Proceedings of the ACM on Web Conference 2025*. 4014–4023.

[25] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. 2011. Product Quantization for Nearest Neighbor Search. *IEEE TPAMI* 33, 1 (2011), 117–128.

[26] Levente Kocsis and Csaba Szepesvári. 2006. Discounted UCB. In *Proceedings of the 2nd PASCAL Challenges Workshop*.

[27] Andreas Krause, Ajit Singh, and Carlos Guestrin. 2008. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research* 9, 2 (2008).

[28] Zhuohua Li, Maoli Liu, Xiangxiang Dai, and John CS Lui. 2025. Towards Efficient Conversational Recommendations: Expected Value of Information Meets Bandit Learning. In *Proceedings of the ACM on Web Conference 2025*. 4226–4238.

[29] Zhuohua Li, Maoli Liu, and John Lui. 2024. Fedconpe: Efficient federated conversational bandits with heterogeneous clients. *arXiv preprint* (2024).

[30] Mugeng Liu, Siqi Zhong, Qi Yang, Yudong Han, Xuanzhe Liu, and Yun Ma. 2025. WebANNS: Fast and Efficient Approximate Nearest Neighbor Search in Web Browsers. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2483–2492.

[31] Xutong Liu, Xiangxiang Dai, Xuchuang Wang, Mohammad Hajiesmaili, and John CS Lui. 2025. Combinatorial logistic bandits. In *Abstracts of the 2025 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*. 112–114.

[32] Yury Malkov, Alexander Ponomarenko, Andrey Logvinov, and Vladimir Krylov. 2014. Approximate nearest neighbor algorithm based on navigable small world graphs. *Information Systems* 45 (2014), 61–68.

[33] Yu. A. Malkov and D. A. Yashunin. 2020. Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42, 4 (2020), 824–836. doi:10.1109/TPAMI.2018.2889473

[34] James Jie Pan, Jianguo Wang, and Guoliang Li. 2024. Survey of vector database management systems. *The VLDB Journal* 33, 5 (2024), 1591–1615.

[35] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543.

[36] Derrick Quinn, Mohammad Nouri, Neel Patel, John Salihu, Alireza Salemi, Sukhan Lee, Hamed Zamani, and Mohammad Alian. 2025. Accelerating retrieval-augmented generation. In *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1*. 15–32.

[37] Suhas Jayaram Subramanya, Devvrit, Rohan Kadekodi, Ravishankar Krishnaswamy, and Harsha Vardhan Simhadri. 2019. DiskANN: Fast Accurate Billion-point Nearest Neighbor Search on a Single Node. In *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*.

[38] Xiaqiang Tang, Jian Li, Nan Du, and Sihong Xie. 2025. Adapting to non-stationary environments: Multi-armed bandit enhanced retrieval-augmented generation on knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 12658–12666.

[39] Yitu Wang, Shiyu Li, Qilin Zheng, Linghao Song, Zongwang Li, Andrew Chang, Yiran Chen, et al. 2024. NDSEARCH: Accelerating graph-traversal-based approximate nearest neighbor search through near data processing. In *2024 ACM/IEEE 51st ISCA Conference*. IEEE, 368–381.

[40] Bo Xue, Yimu Wang, Yuanyu Wan, Jinfeng Yi, and Lijun Zhang. 2023. Efficient algorithms for generalized linear bandits with heavy-tailed rewards. *Advances in Neural Information Processing Systems* 36 (2023), 70880–70891.

[41] Ximu Zeng, Liwei Deng, Penghao Chen, Xu Chen, Han Su, and Kai Zheng. 2025. LIRA: A Learning-based Query-aware Partition Framework for Large-scale ANN Search. In *Proceedings of the ACM on Web Conference 2025*. 2729–2741.

[42] Yuhua Zhu, Zachary Izzo, and Lexing Ying. 2023. Continuous-in-time limit for Bayesian bandits. *Journal of Machine Learning Research* 24, 305 (2023), 1–35.

[43] Jinhang Zuo, Songwen Hu, Tong Yu, Shuai Li, Handong Zhao, and Carlee Joe-Wong. 2022. Hierarchical conversational preference elicitation with bandit feedback. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2827–2836.

A Proof

A.1 Proof of Theorem 1

Theorem 1 (Computational Complexity of BANCO)**.** *The overall computational cost of* BANCO *grows polynomially with the context dimension d, the size of the action space $|\mathcal{A}|$, the total number of rounds T, the number of batches M, and the batch size $\Delta = \lceil T/M \rceil$. Specifically, the algorithm achieves:*

$$\textit{Time Complexity:} \quad O(T \cdot |\mathcal{A}| \cdot d^2 + M \cdot d^3),$$
$$\textit{Space Complexity:} \quad O(d^2 + \Delta \cdot d).$$

Proof. The total cost of BANCO arises from two components: (i) per-round decision making (Lines 5–9) and (ii) per-batch model updates (Lines 11–14).

Time Complexity. In each round t, evaluating all actions $a \in \mathcal{A}$ and computing their UCB scores requires $O(|\mathcal{A}|d^2)$ operations. Across T rounds, this results in $O(T|\mathcal{A}|d^2)$ total cost. At each batch end, updating covariance and reward statistics over Δ steps costs $O(\Delta d^2)$, while the matrix inversion for $\hat{\boldsymbol{\theta}}_{t_m}$ adds $O(d^3)$. Thus, each batch update requires $O(\Delta d^2 + d^3)$ operations, and over M batches this totals $O(Md^3)$. Combining both parts yields the overall time complexity:

$$O(T|\mathcal{A}|d^2 + Md^3), \tag{16}$$

which scales linearly with T, implying constant amortized cost per decision.

Space Complexity. BANCO maintains $d \times d$ matrices such as V_t and $\tilde{V}_t$, and d-dimensional vectors like $\mathbf{b}_t$ and $\hat{\boldsymbol{\theta}}_t$, leading to $O(d^2)$ storage. Additionally, buffering up to Δ feature vectors per batch requires $O(\Delta d)$ memory. Hence, the overall space requirement is:

$$O(d^2 + \Delta d). \tag{17}$$

Given that d is typically moderate, both computation and memory costs remain tractable, confirming the efficiency of BANCO.

□

A.2 Proof of Proposition 1

Proposition 1 (Concentration of Weighted Noise)**.** *For any $\delta > 0$, with probability at least $1-\delta$, the weighted noise term $S_t = \sum_{s=1}^{t} \gamma^{t-s}\phi(\boldsymbol{x}_s, \boldsymbol{a}_s)\eta_s$ satisfies:*

$$\|S_t\|^2_{\widetilde{V}_t^{-1}} \le \sigma^2\left(2\log\frac{1}{\delta} + \log\frac{\det(\widetilde{V}_t)}{\det(\lambda I)}\right), \tag{18}$$

where $\widetilde{V}_t = W_t + \lambda I_m$ is the regularized noise-weighted covariance matrix and $W_t = \sum_{s=1}^{t}(\gamma^{t-s})^2\phi(\boldsymbol{x}_s, \boldsymbol{a}_s)\phi(\boldsymbol{x}_s, \boldsymbol{a}_s)^\top$.

Proof. We follow the standard self-normalized concentration approach, adapted to discounting and time-varying regularization: (i) construct an exponential supermartingale; (ii) apply the Gaussian mixture technique; (iii) introduce a stopping time and use optional stopping to derive the tail bound.

Step 1: Supermartingale construction. For $t \ge 0$, define

$$M_t(x) = \exp\left(x^\top S_t - \tfrac{1}{2}\|x\|^2_{\widetilde{V}_t/\sigma^2}\right). \tag{19}$$

Let $\mathcal{F}_t = \sigma(r_{1,\boldsymbol{a}_1}, \ldots, r_{t,\boldsymbol{a}_t})$, so that A_t is $\mathcal{F}_{t-1}$-measurable, η_t is $\mathcal{F}_t$-measurable, and the weights are predictable. Extending Lemma 8 of [1] to the weighted case gives

$$\mathbb{E}[M_t(x) \mid \mathcal{F}_{t-1}] \le M_{t-1}(x), \tag{20}$$

so $(M_t(x))_{t\ge 0}$ is a supermartingale with $\mathbb{E}[M_t(x)] \le 1$.

Step 2: Gaussian mixture and completing the square. Let h_t be the density of $\mathcal{N}(0, \mu_t I_d)$ with $\mu_t > 0$, and define

$$\overline{M}_t = \int_{\mathbb{R}^d} M_t(x)\, h_t(x)\, dx. \tag{21}$$

By Fubini and (20), $\mathbb{E}[\overline{M}_t] \le 1$. Completing the square yields

$$\overline{M}_t = \exp\left(\tfrac{1}{2}\|S_t\|^2_{(\widetilde{V}_t/\sigma^2 + (\mu_t I_d)^{-1})^{-1}}\right)\sqrt{\frac{\det\left((\widetilde{V}_t/\sigma^2 + (\mu_t I_d)^{-1})^{-1}\right)}{\det(\mu_t I_d)}}. \tag{22}$$

Setting $\mu_t = \sigma^2/\lambda$ and noting that $\widetilde{V}_t$ already includes λI_d, we get

$$\overline{M}_t = \exp\left(\tfrac{1}{2}\|S_t\|^2_{\widetilde{V}_t^{-1}}\right)\sqrt{\frac{\det(\mu_t I_d)}{\det(\widetilde{V}_t)}}. \tag{23}$$

Step 3: Stopping time and tail probability. Define

$$\tau = \inf\left\{t \ge 0 :\ \|S_t\|^2_{\widetilde{V}_t^{-1}} \ge 2\log\tfrac{1}{\delta} + \log\tfrac{\det(\widetilde{V}_t)}{\det(\mu_t I_d)}\right\}. \tag{24}$$

On $\{\tau < \infty\}$, by (23),

$$\exp\left(\tfrac{1}{2}\|S_\tau\|^2_{\widetilde{V}_\tau^{-1}}\right)\sqrt{\frac{\det(\mu_\tau I_d)}{\det(\widetilde{V}_\tau)}} \ge \frac{1}{\delta}. \tag{25}$$

Since $(\overline{M}_{t\wedge\tau})_{t\ge 0}$ is a nonnegative supermartingale, optional stopping and Fatou's lemma imply

$$\mathbb{E}[\overline{M}_\tau] \le \liminf_{t\to\infty}\mathbb{E}[\overline{M}_{t\wedge\tau}] \le \mathbb{E}[\overline{M}_0] = 1. \tag{26}$$

Applying Markov's inequality to (25)–(26),

$$\mathbb{P}(\tau < \infty) = \mathbb{P}\left(\exp\left(\tfrac{1}{2}\|S_\tau\|^2_{\widetilde{V}_\tau^{-1}}\right)\sqrt{\tfrac{\det(\mu_\tau I_d)}{\det(\widetilde{V}_\tau)}} \ge \tfrac{1}{\delta}\right) \le \delta\,\mathbb{E}[\overline{M}_\tau] \le \delta. \tag{27}$$

Thus, with probability at least $1-\delta$, for all $t \ge 0$,

$$\|S_t\|^2_{\widetilde{V}_t^{-1}} \le 2\log\tfrac{1}{\delta} + \log\tfrac{\det(\widetilde{V}_t)}{\det(\mu_t I_d)}. \tag{28}$$

Since $\mu_t = \sigma^2/\lambda$ and $\det(\mu_t I_d) = (\sigma^2/\lambda)^d$, multiplying (28) by σ^2 gives

$$\|S_t\|^2_{\widetilde{V}_t^{-1}} \le \sigma^2\left(2\log\frac{1}{\delta} + \log\frac{\det(\widetilde{V}_t)}{\det(\lambda I_d)}\right), \tag{29}$$

which completes the proof. □

A.3 Proof of Theorem 2

Theorem 2 (Confidence Bound of $\boldsymbol{\theta}_t$)**.** *For any $\delta \in (0,1)$, with probability at least $1-\delta$, the following holds simultaneously for all $m \in \{1, \ldots, M\}$ batches:*

$$\left\|\hat{\boldsymbol{\theta}}_{t_m} - \boldsymbol{\theta}\right\|_{V_{t_m}\widetilde{V}_{t_m}^{-1}V_{t_m}} \le \sqrt{\lambda}S + \sigma\sqrt{2\log\tfrac{1}{\delta} + m\log\left(1 + \tfrac{\sum_{s=1}^{t_m}\gamma^{2(t_m-s)}}{\lambda^m}\right)}. \tag{30}$$

Theorem 3. *For any $\delta \in (0,1)$, with probability at least $1-\delta$, the following holds simultaneously for all $m \in \{1, \ldots, M\}$:*

$$\left\|\hat{\boldsymbol{\theta}}_{t_m} - \boldsymbol{\theta}\right\|_{V_{t_m}\widetilde{V}_{t_m}^{-1}V_{t_m}} \le \sqrt{\lambda}\,S + \sigma\sqrt{2\log\tfrac{1}{\delta} + m\log\left(1 + \tfrac{\sum_{s=1}^{t_m}\gamma^{2(t_m-s)}}{\lambda^m}\right)}. \tag{31}$$

Proof. We derive a high-probability confidence ellipsoid for $\hat{\boldsymbol{\theta}}_{t_m}$. The proof proceeds in four steps: (i) express the estimation error in terms of the weighted noise S_{t_m} and the regularization bias; (ii) bound its norm under the matrix metric $V_{t_m}\widetilde{V}_{t_m}^{-1}V_{t_m}$; (iii) apply the self-normalized concentration bound from Proposition 1; and (iv) simplify the determinant term to obtain the closed-form expression in (31).

Step 1: Error decomposition. Under the condition $\boldsymbol{\theta}_t = \boldsymbol{\theta}$,

$$\hat{\boldsymbol{\theta}}_{t_m} - \boldsymbol{\theta} = V_{t_m}^{-1}S_{t_m} - \lambda V_{t_m}^{-1}\boldsymbol{\theta}, \tag{32}$$

where $S_{t_m} = \sum_{s=1}^{t_m}\gamma^{t_m-s}\phi(\boldsymbol{x}_s, \boldsymbol{a}_s)\eta_s$.

Step 2: Metric bound. By the Cauchy–Schwarz inequality in the metric induced by $V_{t_m}\widetilde{V}_{t_m}^{-1}V_{t_m}$,

$$\|\hat{\boldsymbol{\theta}}_{t_m}-\boldsymbol{\theta}\|_{V_{t_m}\widetilde{V}_{t_m}^{-1}V_{t_m}} \le \|S_{t_m}\|_{\widetilde{V}_{t_m}^{-1}} + \lambda\|\boldsymbol{\theta}\|_{\widetilde{V}_{t_m}^{-1}}. \tag{33}$$

Since $\widetilde{V}_{t_m} \succeq \mu_{t_m} I_d$ and $\|\boldsymbol{\theta}\|_2 \le S$, we have

$$\|\boldsymbol{\theta}\|_{\widetilde{V}_{t_m}^{-1}} \le \frac{S}{\sqrt{\mu_{t_m}}}, \tag{34}$$

and thus

$$\|\hat{\boldsymbol{\theta}}_{t_m}-\boldsymbol{\theta}\|_{V_{t_m}\widetilde{V}_{t_m}^{-1}V_{t_m}} \le \|S_{t_m}\|_{\widetilde{V}_{t_m}^{-1}} + \frac{\lambda}{\sqrt{\mu_{t_m}}}S. \tag{35}$$

Step 3: Concentration and determinant control. From Proposition 1, with probability at least $1-\delta$,

$$\|S_{t_m}\|_{\widetilde{V}_{t_m}^{-1}} \le \sigma\sqrt{2\log\tfrac{1}{\delta} + \log\Big(\tfrac{\det(\widetilde{V}_{t_m})}{\det(\mu_{t_m}I_d)}\Big)}. \tag{36}$$

Using $\det(\mu_{t_m}I_d) = (\mu_{t_m})^d$, we get:

$$\log\Big(\tfrac{\det(\widetilde{V}_{t_m})}{\det(\mu_{t_m}I_d)}\Big) \le m\log\Big(1+\tfrac{\sum_{s=1}^{t_m}\gamma^{2(t_m-s)}}{\lambda^m}\Big). \tag{37}$$

Step 4: Conclusion. Substituting (37) into (35)–(36) gives

$$\|\hat{\boldsymbol{\theta}}_{t_m}-\boldsymbol{\theta}\|_{V_{t_m}\widetilde{V}_{t_m}^{-1}V_{t_m}} \le \frac{\lambda}{\sqrt{\mu_{t_m}}}S+\sigma\sqrt{2\log\tfrac{1}{\delta} + m\log\Big(1+\tfrac{\sum_{s=1}^{t_m}\gamma^{2(t_m-s)}}{\lambda^m}\Big)}. \tag{38}$$

Noting $\lambda \le \sqrt{\lambda}\sqrt{\mu_{t_m}}$ (absorbing constants into S), the bound simplifies to (31), which holds for all m with probability at least $1-\delta$. □

A.4 Proof of Proposition 2

Proposition 2 (Parameter Inclusion with Confidence). *Let* $\bar{\boldsymbol{\theta}}_{t_m} = V_{t_{m-1}}^{-1}(\sum_{s=1}^{t_{m-1}}\gamma^{t_{m-1}-s}\phi(\boldsymbol{x}_s,\boldsymbol{a}_s)\phi(\boldsymbol{x}_s,\boldsymbol{a}_s)^\top\boldsymbol{\theta}_s + \lambda\boldsymbol{\theta}_{t_{m-1}})$. *For any* $\delta>0$, *we have* $\mathbb{P}(\bar{\boldsymbol{\theta}}_{t_m} \in C_m) \ge 1-\delta$.

Proof. We show that the drift-aware proxy parameter $\bar{\boldsymbol{\theta}}_{t_m}$ lies in the confidence region C_m

Step 1. Relation between $\bar{\boldsymbol{\theta}}_{t_m}$ *and* $\hat{\boldsymbol{\theta}}_{t_{m-1}}$. Using $r_{s,\boldsymbol{a}_s} = \phi(\boldsymbol{x}_s,\boldsymbol{a}_s)^\top\boldsymbol{\theta}_s + \eta_s$, we obtain:

$$\bar{\boldsymbol{\theta}}_{t_m} - \hat{\boldsymbol{\theta}}_{t_{m-1}} = \lambda V_{t_{m-1}}^{-1}\boldsymbol{\theta}_{t_{m-1}} - V_{t_{m-1}}^{-1}S_{t_{m-1}}, \tag{39}$$

where $S_t = \sum_{s=1}^{t}\gamma^{t-s}\phi(\boldsymbol{x}_s,\boldsymbol{a}_s)\eta_s$.

Step 2. Bounding the metric norm. By the triangle inequality and induced norm property,

$$\|\bar{\boldsymbol{\theta}}_{t_m} - \hat{\boldsymbol{\theta}}_{t_{m-1}}\|_{V_{t_{m-1}}\widetilde{V}_{t_{m-1}}^{-1}V_{t_{m-1}}} \le \|S_{t_{m-1}}\|_{\widetilde{V}_{t_{m-1}}^{-1}} + \lambda\|\boldsymbol{\theta}_{t_{m-1}}\|_{\widetilde{V}_{t_{m-1}}^{-1}}. \tag{40}$$

If $\|\boldsymbol{\theta}_{t_{m-1}}\|_2 \le S$ and $\widetilde{V}_{t_{m-1}} \succeq \mu_{t_{m-1}}I_d$, then

$$\|\boldsymbol{\theta}_{t_{m-1}}\|_{\widetilde{V}_{t_{m-1}}^{-1}} \le \frac{S}{\sqrt{\mu_{t_{m-1}}}}. \tag{41}$$

Step 3. Concentration of the weighted noise. By Proposition 1, with probability at least $1-\delta$,

$$\|S_{t_{m-1}}\|_{\widetilde{V}_{t_{m-1}}^{-1}} \le \sigma\sqrt{2\log\tfrac{1}{\delta} + \log\tfrac{\det(\widetilde{V}_{t_{m-1}})}{\det(\mu_{t_{m-1}}I_d)}}. \tag{42}$$

Since $\det(\mu_{t_{m-1}}I_d) = (\mu_{t_{m-1}})^d$ and $\det(\widetilde{V}_{t_{m-1}}) \le (\lambda+\frac{L^2}{d}\sum_{s=1}^{t_{m-1}}\gamma^{2(t_{m-1}-s)})^d$, the determinant ratio remains bounded by a logarithmic factor.

Step 4. Confidence radius and conclusion. Combining (40)–(42),

$$\|\bar{\boldsymbol{\theta}}_{t_m}-\hat{\boldsymbol{\theta}}_{t_{m-1}}\|_{V_{t_{m-1}}\widetilde{V}_{t_{m-1}}^{-1}V_{t_{m-1}}} \le \sqrt{\lambda_{t_{m-1}}}S+\sigma\sqrt{2\log\tfrac{1}{\delta} + \log\tfrac{\det(\widetilde{V}_{t_{m-1}})}{\det(\lambda_{t_{m-1}}I_d)}}. \tag{43}$$

Define $\beta_{t_{m-1}}(\delta)$ as the RHS of (43). Then $\bar{\boldsymbol{\theta}}_{t_m} \in C_m$ with probability at least $1-\delta$, i.e.,

$$\Pr(\bar{\boldsymbol{\theta}}_{t_m} \in C_m) \ge 1-\delta. \tag{44}$$

□

A.5 Proof of Theorem 3

Theorem 4 (Regret Bound of BANCO). *Assume the total variation of the parameter is bounded by* $\sum_{s=1}^{T}\|\boldsymbol{\theta}_s - \boldsymbol{\theta}_{s+1}\|_2 \le B_T$. *Then for all* $\gamma \in (0,1]$ *and integers* $D \ge 1$, *with probability at least* $1-\delta$,

$$\begin{aligned} R_T \le\ & \big(2L(D+N)S_A + \sqrt{\lambda}S_P\big)B_T + (2S_W + 4S_A)\frac{S_P}{\lambda}\frac{\gamma^D}{1-\gamma} \\ & + \beta_T\cdot\sqrt{2d\log\Big(1+\tfrac{N}{\lambda d}\Big)}\cdot\tfrac{T}{\sqrt{N}}. \end{aligned} \tag{45}$$

Proof. We decompose the per-round regret into (i) an estimation term controlled by the confidence radius and (ii) a bias term due to nonstationarity. The bias is further split into a local (last D steps) component bounded by the variation budget B_T and a geometric tail controlled by γ. Finally, we sum over batches and apply an elliptical-potential bound to obtain the exploration term.

Step I: Instantaneous regret. Fix $t \in (t_{m-1}, t_m]$ and let

$$A_t \in \arg\max_{a\in\mathcal{A}_t}\langle a,\boldsymbol{\theta}_t\rangle, \qquad \boldsymbol{\theta}_t \in \arg\max_{\boldsymbol{\theta}\in C_m}\langle A_t,\boldsymbol{\theta}\rangle. \tag{46}$$

Under the high-probability event of Proposition 1 and Proposition 2 (i.e., $\bar{\boldsymbol{\theta}}_{t_m} \in C_m$), the UCB rule gives

$$\langle A_t, \bar{\boldsymbol{\theta}}_{t_m}\rangle \le \max_{\boldsymbol{\theta}\in C_m}\langle A_t,\boldsymbol{\theta}\rangle \le \max_{\boldsymbol{\theta}\in C_m}\langle A_t,\boldsymbol{\theta}\rangle = \langle A_t,\boldsymbol{\theta}_t\rangle. \tag{47}$$

Hence the instantaneous regret $r_t = \langle A_t - A_t, \boldsymbol{\theta}_t\rangle$ satisfies

$$\begin{aligned} r_t &= \langle A_t - A_t, \bar{\boldsymbol{\theta}}_{t_m}\rangle + \langle A_t - A_t, \boldsymbol{\theta}_t - \bar{\boldsymbol{\theta}}_{t_m}\rangle \\ &\le \langle A_t, \boldsymbol{\theta}_t - \bar{\boldsymbol{\theta}}_{t_m}\rangle + \|A_t - A_t\|_2\,\|\boldsymbol{\theta}_t - \bar{\boldsymbol{\theta}}_{t_m}\|_2 \\ &\le \|A_t\|_{V_{t_{m-1}}^{-1}\widetilde{V}_{t_{m-1}}V_{t_{m-1}}^{-1}}\,\|\boldsymbol{\theta}_t - \bar{\boldsymbol{\theta}}_{t_m}\|_{V_{t_{m-1}}\widetilde{V}_{t_{m-1}}^{-1}V_{t_{m-1}}} + 2L\,\|\boldsymbol{\theta}_t - \bar{\boldsymbol{\theta}}_{t_m}\|_2, \end{aligned} \tag{48}$$

where we use Cauchy-Schwarz in the metric induced by $V_{t_{m-1}}\widetilde{V}_{t_{m-1}}^{-1}V_{t_{m-1}}$.

Step I.1: Bounding $\|A_t\|_{V_{t_{m-1}}^{-1}\widetilde{V}_{t_{m-1}}V_{t_{m-1}}^{-1}}$. Let $B = V_{t_{m-1}}^{-1/2}\widetilde{V}_{t_{m-1}}V_{t_{m-1}}^{-1/2}$ and $B \preceq I$ since $\widetilde{V}_{t_{m-1}} \preceq V_{t_{m-1}}$. Then for any x,

$$x^\top V_{t_{m-1}}^{-1}\widetilde{V}_{t_{m-1}}V_{t_{m-1}}^{-1}x = (V_{t_{m-1}}^{-1/2}x)^\top B(V_{t_{m-1}}^{-1/2}x) \le x^\top V_{t_{m-1}}^{-1}x, \tag{49}$$

and hence

$$\|A_t\|_{V_{t_{m-1}}^{-1}\widetilde{V}_{t_{m-1}}V_{t_{m-1}}^{-1}} \le \|A_t\|_{V_{t_{m-1}}^{-1}} \le \frac{\|A_t\|_2}{\sqrt{\lambda}} \le \frac{L}{\sqrt{\lambda}}. \tag{50}$$

Step I.2: Bounding $\|\boldsymbol{\theta}_t - \bar{\boldsymbol{\theta}}_{t_m}\|_{V_{t_{m-1}}\widetilde{V}_{t_{m-1}}^{-1}V_{t_{m-1}}}$. By the triangle inequality and the definition of the confidence set C_m,

$$\begin{aligned} &\|\boldsymbol{\theta}_t - \bar{\boldsymbol{\theta}}_{t_m}\|_{V_{t_{m-1}}\widetilde{V}_{t_{m-1}}^{-1}V_{t_{m-1}}} \\ &\le \|\boldsymbol{\theta}_t - \hat{\boldsymbol{\theta}}_{t_{m-1}}\|_{V_{t_{m-1}}\widetilde{V}_{t_{m-1}}^{-1}V_{t_{m-1}}} + \|\bar{\boldsymbol{\theta}}_{t_m} - \hat{\boldsymbol{\theta}}_{t_{m-1}}\|_{V_{t_{m-1}}\widetilde{V}_{t_{m-1}}^{-1}V_{t_{m-1}}} \\ &\le 2\,\beta_{t_{m-1}}, \end{aligned} \tag{51}$$

since both $\boldsymbol{\theta}_t$ and $\bar{\boldsymbol{\theta}}_{t_m}$ belong to C_m with probability $\geq 1-\delta$ (Theorem 3 and Proposition 2).

Step I.3: Bounding the bias $\|\boldsymbol{\theta}_t - \bar{\boldsymbol{\theta}}_{t_m}\|_2$. Recall

$$\bar{\boldsymbol{\theta}}_{t_m} = V_{t_{m-1}}^{-1}\left(\sum_{s=1}^{t_{m-1}} \gamma^{t_{m-1}-s} A_s A_s^\top \boldsymbol{\theta}_s + \lambda\, \boldsymbol{\theta}_{t_{m-1}}\right). \tag{52}$$

Adding and subtracting $\boldsymbol{\theta}_t$ and applying the triangle inequality, we divide the history into the recent D steps and the far past:

$$\begin{aligned}\|\boldsymbol{\theta}_t - \bar{\boldsymbol{\theta}}_{t_m}\|_2 \leq &\left\|V_{t_{m-1}}^{-1} \sum_{s=t_{m-1}-D+1}^{t_{m-1}} \gamma^{t_{m-1}-s} A_s A_s^\top (\boldsymbol{\theta}_s - \boldsymbol{\theta}_t)\right\|_2 \\ &+ \left\|V_{t_{m-1}}^{-1} \sum_{s=1}^{t_{m-1}-D} \gamma^{t_{m-1}-s} A_s A_s^\top (\boldsymbol{\theta}_s - \boldsymbol{\theta}_t)\right\|_2 + \lambda\, \|\boldsymbol{\theta}_{t_{m-1}} - \boldsymbol{\theta}_t\|_2.\end{aligned} \tag{53}$$

For the recent D terms, using $\|Mx\|_2 \leq \|M\|_{\mathrm{op}}\|x\|_2$, $\|V_{t_{m-1}}^{-1}\|_{\mathrm{op}} \leq \lambda^{-1}$, we obtain

$$\left\|V_{t_{m-1}}^{-1} \sum_{s=t_{m-1}-D+1}^{t_{m-1}} \gamma^{t_{m-1}-s} A_s A_s^\top (\boldsymbol{\theta}_s - \boldsymbol{\theta}_t)\right\|_2 \leq \frac{S_A}{\lambda} \sum_{s=t_{m-1}-D+1}^{t_{m-1}} \|\boldsymbol{\theta}_s - \boldsymbol{\theta}_t\|_2. \tag{54}$$

For the far past, summing geometric weights and using $\|V_{t_{m-1}}^{-1}\|_{\mathrm{op}} \leq \lambda^{-1}$,

$$\left\|V_{t_{m-1}}^{-1} \sum_{s=1}^{t_{m-1}-D} \gamma^{t_{m-1}-s} A_s A_s^\top (\boldsymbol{\theta}_s - \boldsymbol{\theta}_t)\right\|_2 \leq \frac{S_W}{\lambda} \frac{\gamma^D}{1-\gamma} S_P, \tag{55}$$

for an absolute constant $S_W > 0$, using $\|\boldsymbol{\theta}_s\|_2 \leq S_P$.

Combining (53)–(55) gives

$$\|\boldsymbol{\theta}_t - \bar{\boldsymbol{\theta}}_{t_m}\|_2 \leq \frac{S_A}{\lambda} \sum_{s=t_{m-1}-D+1}^{t_{m-1}} \|\boldsymbol{\theta}_s - \boldsymbol{\theta}_t\|_2 + \frac{S_W}{\lambda} \frac{\gamma^D}{1-\gamma} S_P + \lambda\, \|\boldsymbol{\theta}_{t_{m-1}} - \boldsymbol{\theta}_t\|_2. \tag{56}$$

Step I.4: Collecting the instantaneous bound. Substituting (50), (51) and (56) into (48) gives

$$\begin{aligned} r_t \leq\ & \frac{2L}{\sqrt{\lambda}}\, \beta_{t_{m-1}} \\ &+ 2L\left[\frac{S_A}{\lambda} \sum_{s=t_{m-1}-D+1}^{t_{m-1}} \|\boldsymbol{\theta}_s - \boldsymbol{\theta}_t\|_2 + \frac{S_W}{\lambda} \frac{\gamma^D}{1-\gamma} S_P + \lambda\, \|\boldsymbol{\theta}_{t_{m-1}} - \boldsymbol{\theta}_t\|_2\right]. \end{aligned} \tag{57}$$

Step II: Summing over time. Summing (57) over $t = 1, \ldots, T$ and regrouping terms by batch yields three aggregates.

(a) Variation terms. By a telescoping argument, each increment $\|\boldsymbol{\theta}_{j+1} - \boldsymbol{\theta}_j\|_2$ appears at most $D + \Delta$ times in the first sum and Δ times in the last one. Thus,

$$\sum_{t=1}^{T} \sum_{s=t_{m-1}-D+1}^{t_{m-1}} \|\boldsymbol{\theta}_s - \boldsymbol{\theta}_t\|_2 \leq (D+\Delta) B_T, \qquad \sum_{t=1}^{T} \|\boldsymbol{\theta}_{t_{m-1}} - \boldsymbol{\theta}_t\|_2 \leq \Delta B_T, \tag{58}$$

(b) Geometric tail. The middle term in (57) contributes

$$2L\, \frac{S_W}{\lambda} \frac{\gamma^D}{1-\gamma} S_P\, T \ \rightsquigarrow\ (2S_W + 4S_A) \frac{S_P}{\lambda} \frac{\gamma^D}{1-\gamma}, \tag{59}$$

after absorbing numeric factors and operator constants (the coefficient $(2S_W + 4S_A)$ collects far-history terms, including the bound in (54)).

(c) Exploration sum. For each batch m, by (49) and Cauchy–Schwarz,

$$\begin{aligned}\sum_{t=t_{m-1}+1}^{t_m} \|A_t\|_{V_{t_{m-1}}^{-1}} &\leq \sqrt{\Delta}\left(\sum_{t=t_{m-1}+1}^{t_m} \|A_t\|_{V_{t_{m-1}}^{-1}}^2\right)^{1/2} \\ &\leq \sqrt{2d\,\Delta \log\left(1 + \tfrac{\Delta}{\lambda d}\right)},\end{aligned} \tag{60}$$

where the last step follows from the standard elliptical-potential bound. Since $\beta_{t_{m-1}} \leq \beta_T$ (nondecreasing), summing over batches with $\Delta \leq N$ gives

$$\sum_{m=1}^{M} \beta_{t_{m-1}} \sum_{t=t_{m-1}+1}^{t_m} \|A_t\|_{V_{t_{m-1}}^{-1}} \leq \beta_T \sqrt{2d \log\left(1 + \tfrac{N}{\lambda d}\right)}\, \frac{T}{\sqrt{N}}. \tag{61}$$

Step III: Final aggregation. Combining (57)–(61) and using $\Delta \leq N$, we have, with probability at least $1-\delta$,

$$\begin{aligned} R_T \leq\ & \left(2L(D+N)S_A + \sqrt{\lambda} S_P\right) B_T + (2S_W + 4S_A) \frac{S_P}{\lambda} \frac{\gamma^D}{1-\gamma} \\ &+\ \beta_T \sqrt{2d \log\left(1 + \tfrac{N}{\lambda d}\right)}\, \frac{T}{\sqrt{N}}. \end{aligned} \tag{62}$$

This matches the claimed bound and completes the proof.

□

A.6 Proof of Corollary 1

Corollary 1 (Subline Regret)**.** *Under Theorem 3, if we set* $\gamma = 1 - c_\gamma (\frac{B_T}{T})^{2/3} m^{-1/3}$, $N = \lceil c_N (\frac{T}{B_T})^{2/3} m^{1/3} \rceil$, *and* $D = \lceil c_D N \log T \rceil$, *where* c_γ, c_N, *and* c_D *are positive constants controlling the discount factor, batch size, and update interval, respectively, then with high probability,*

$$R_T = \tilde{O}\left(T^{2/3} B_T^{1/3} d^{5/6}\right), \tag{63}$$

where $\tilde{O}(\cdot)$ *suppresses logarithmic factors.*

Proof. From Theorem 4, the regret consists of three parts: (i) the drift-dependent term $(D+N)B_T$, (ii) the geometric tail $\frac{\gamma^D}{1-\gamma}$, and (iii) the exploration term $\beta_T \sqrt{2d \log(1 + \frac{N}{\lambda d})}\, \frac{T}{\sqrt{N}}$. We choose γ, N, and D such that (i) and (iii) are balanced while (ii) remains negligible.

Drift term. Ignoring constants, the drift contribution is $O((D+N)B_T)$. Choosing $N \asymp (T/B_T)^{2/3} d^{1/3}$ and $D \asymp N \log T$ gives

$$(D+N)B_T \asymp T^{2/3} B_T^{1/3} d^{1/3} \mathrm{polylog}(T). \tag{64}$$

Exploration term. Since $\beta_T = \tilde{O}(\sqrt{d})$, the exploration component scales as $\tilde{O}(dT/\sqrt{N})$. Substituting $N \asymp (T/B_T)^{2/3} d^{1/3}$ yields $\frac{T}{\sqrt{N}} \asymp T^{2/3} B_T^{1/3} d^{-1/6}$, and hence the exploration term is $\tilde{O}(T^{2/3} B_T^{1/3} d^{5/6})$.

Geometric tail. For the residual term, $\frac{\gamma^D}{1-\gamma} \lesssim T^{2/3 - c_\gamma c_D} B_T^{-2/3} d^{1/3}$. When $c_\gamma c_D \geq 2$, this quantity becomes logarithmically small and negligible.

Combining the above results, the total regret satisfies

$$R_T = \tilde{O}(T^{2/3} B_T^{1/3} d^{5/6}), \tag{65}$$

which completes the proof.

□