RSCB: REGIONAL SEMANTIC CONTEXTUAL BANDIT FOR OPTIMAL SELECTION OVER LLM-GENERATED DYNAMIC CANDIDATES

Anonymous authors
Paper under double-blind review

ABSTRACT

Large Language Models (LLMs) increasingly generate multiple candidate solutions for complex tasks such as bug localization, code repair, and tool selection. Selecting among these candidates remains challenging because the available arm set changes across rounds and newly generated arms may have no interaction history. Identity-based or disjoint contextual bandits cannot transfer feedback to such arms, while globally shared semantic linear bandits may suffer negative transfer when heterogeneous context-arm relationships cannot be captured by a single reward parameter. We propose Regional Semantic Contextual Bandits (RSCB), which maintains independent linear reward models over semantic interaction regions and routes each context-arm pair to its corresponding regional model. RSCB selects arms using region-specific upper-confidence scores, transferring feedback among semantically related pairs within each region while restricting transfer across heterogeneous regions. For H semantic regions, we prove a regret bound of $O(m\sqrt{HT}\log T)$ under regional linear realizability and correct fixed routing. Consequently, for fixed H, RSCB retains the $O(m\sqrt{T}\log T)$ regret rate independent of the number of distinct arms. We further establish an $\Omega(\sqrt{K_{\text{total}}T})$ lower bound for arm-independent learning. Finally, under a semantic-continuity assumption, we bound the prediction uncertainty of unseen arms within a correctly routed region. We evaluate RSCB on SWE-bench and an Android/Java security bug dataset, using LLM-generated root-cause hypotheses as arms and objective rewards derived from ground-truth patches. On held-out bugs, RSCB reduces cumulative regret compared to SC-LinUCB and kernelized baselines, with statistically significant improvements on the security dataset. Additionally, RSCB also learns a multi-region structure (H=3) on the public BugsInPy benchmark, achieving the lowest regret among all methods.

1 INTRODUCTION

The widespread adoption of Large Language Models in software engineering has given rise to a paradigm wherein multiple candidate solutions are routinely proposed for a single problem instance. For example, when localizing a fault, an LLM may generate several plausible root causes; when selecting an API tool, it may suggest multiple candidate invocations; and when recovering from a test failure, it may propose alternative remediation strategies. This raises a central question: how should an autonomous agent efficiently select among such dynamically generated candidates? Conventional heuristics fall short: LLM self-ranking is inconsistent (identical reports yield divergent rankings), exhaustive validation is prohibitively expensive, random selection discards accrued knowledge, and static rules cannot adapt to novel failure modes.

Disjoint contextual bandits, such as Disjoint LinUCB (Li et al., 2010), maintain separate parameters for individual arms and therefore cannot transfer arm-specific feedback to newly generated candidates. When LLM-generated arms rarely recur, a new arm has no interaction history, and its estimate is driven primarily by initialization and uncertainty rather than evidence transferred from semantically related arms. SC-LinUCB (Müller, 2025) addresses this limitation by representing arms with semantic features and learning a globally shared reward model, enabling generalization to previ-

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

ously unseen arms. However, a single global parameter can be misspecified when heterogeneous context–arm interactions follow different reward mappings, potentially causing negative transfer.

We introduce Regional Semantic Contextual Bandits (RSCB), a pair-conditioned semantic bandit framework that partitions the context–arm interaction space into multiple regions. For a fixed number of regions, a reward-free gate that routes based solely on semantic interactions assigns each context–arm pair to a region. Each region maintains an independent linear reward estimator and confidence ellipsoid over a common semantic feature space. At each round, RSCB evaluates every candidate using the upper-confidence score of its assigned regional model and updates only the model associated with the selected pair. It therefore shares feedback among context–arm pairs assigned to the same region while restricting direct parameter transfer across different regions. Because both routing and reward prediction operate on semantic context–arm features rather than persistent arm identities, RSCB can immediately score dynamically generated and previously unseen candidates.

Our contributions are as following:

- **Region-based semantic modeling.** We propose RSCB, which routes context–arm pairs to regional linear models via a reward-free semantic gate, transferring feedback within regions while avoiding cross-region negative transfer. RSCB adapts the region count from data, providing a middle ground between independent per-arm estimation and unrestricted global semantic sharing.
- **Regret and generalization analysis.** Under correct fixed routing and regional linear realizability, we establish a pseudo-regret bound of $O\Big(m\sum_{h=1}^{H}\sqrt{T_h}\log T\Big) \leq O(m\sqrt{HT}\log T)$, independent of the number of distinct arms encountered. For fixed H, RSCB therefore retains the $O(m\sqrt{T}\log T)$ dependence of shared semantic linear bandits. We further characterize the contrast with arm-independent learning through an $\Omega(\sqrt{K_{\text{total}}T})$ lower-bound construction and bound prediction uncertainty for unseen arms under semantic-continuity assumptions.
- **Objective cross-domain evaluation.** We evaluate RSCB on three bug-localization benchmarks spanning two software domains. RSCB automatically selects different sharing granularities: it collapses to a single global model on SWE-bench (matching the globally shared baseline) while selecting multiple regions on an industrial security dataset (where it significantly reduces regret) and on the public BugsInPy benchmark (where it attains the lowest regret among all methods). This provides empirical evidence for a domain-dependent boundary between global and regional semantic sharing rather than universal superiority.

2 Problem Formulation

This section formalizes the setting and states our assumptions at the level of a *single* semantic expert (SCB), which is the building block of RSCB. The full model, RSCB (Section 3), is an ensemble of H such experts coordinated by a reward-free router; we analyze one expert here and lift the result to fixed-routing RSCB in Section 4.2 by summing the per-region regrets. Accordingly, the regret analysis assumes a fixed number of regions H and a fixed (oracle) routing; the data-driven selection of H on validation (Section 3.1) is a practical model-selection step and is not covered by this analysis.

2.1 Setting: Contextual Bandits with Dynamic Semantic Arms

At round t, the environment reveals a context encoded by a sentence encoder into $c_t \in \mathbb{R}^p$, and an LLM generates a variable-sized candidate set $\mathcal{A}_t = \{a_t^1, \ldots, a_t^{K_t}\}$. Each arm a_t^i is embedded by the same encoder into a d-dimensional semantic space, giving $\phi'(a_t^i) \in \mathbb{R}^d$, and then projected onto an r-dimensional subspace via PCA, giving $\phi(a_t^i) \in \mathbb{R}^r$. All embeddings are unit-normalized, and the same encoder is used for contexts and arms, so $p = d$. The agent selects $i_t \in \{1, \ldots, K_t\}$ and observes reward $r_t \in [0, 1]$. Let $\mu_t(a) = \mathbb{E}[r_t \mid c_t, a]$ denote the expected reward and $a_t^* \in \arg\max_i \mu_t(a_t^i)$ an optimal arm.

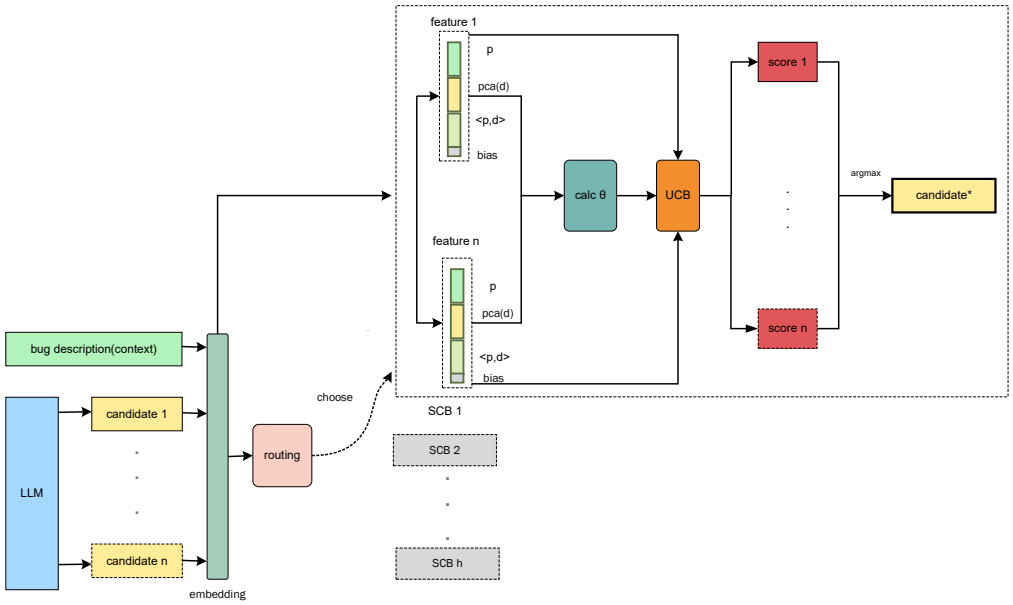

Figure 1: Overview of RSCB. At each round, the LLM produces a variable-sized set of candidate solutions (dynamic arms). For each context–candidate pair, a routing gate selects a region from the embeddings of the context and the candidate. The selected regional SCB estimator encodes the pair into a shared joint-feature space, maintains parameters θ, and forms an upper confidence bound (UCB) for the candidate. The policy plays the candidate with the highest UCB and updates the corresponding region's parameters θ with the observed reward. Because candidates are represented by their semantic embeddings, the learned estimators generalize to never-before-seen candidates.

Definition 1 (Pseudo-Regret)**.** *The pseudo-regret over T rounds is*

$$\bar{R}(T) = \sum_{t=1}^{T} \left[\mu_t(a_t^*) - \mu_t(a_t^{i_t})\right]. \tag{1}$$

2.2 Assumptions

Assumption 2.1 (Linear Realizability)**.** *The expected reward is linear in a joint feature mapping:*

$$\mathbb{E}[r_t \mid c_t, \phi(a_t^i)] = {\theta^*}^\top \psi(c_t, \phi(a_t^i)) \tag{2}$$

where $\psi : \mathbb{R}^p \times \mathbb{R}^d \to \mathbb{R}^m$ is a known feature map.

Remark 2.1 (Approximate realizability). Assumption 2.1 is the standard premise underlying all linear contextual bandits (e.g., LinUCB Li et al. (2010)); like there, we do not claim it holds exactly on real data. When the assumption holds only approximately, standard misspecification analysis (Lattimore et al., 2020) shows that regret degrades gracefully with the approximation error rather than collapsing. Our empirical results (Section 5) are direct measurements on real data and do not depend on Assumption 2.1 holding exactly.

Assumption 2.2 (Boundedness)**.** *There exist constants $B, S > 0$ such that $\|\psi(c, \phi(a))\| \le B$ and $\|\theta^*\| \le S$, and $r_t \in [0, 1]$.*

3 The RSCB Algorithm

RSCB is an ensemble of experts, where each expert is a semantic contextual bandit (SCB) responsible for estimating the optimal arm within a distinct region of the interaction space. A reward-free router governs the ensemble: it assigns each incoming context–arm pair to a region and delegates the corresponding candidate to the associated expert, as shown in Figure 1

3.1 RSCB ROUTINE

The router g controls how far feedback is shared across candidates. It is fitted on the training pool only, uses no reward, and is frozen before evaluation; its assignment unit is the transient context-arm pair. For a context c and the original arm embedding $\phi' \in \mathbb{R}^d$, we form the element-wise interaction $u(c, \phi') = c \odot \phi'$, reduce it with a training-fitted PCA projection P_q (q=8), and ℓ_2-normalize: $z(c, \phi') = P_q^\top (u - \bar{u}) / \|P_q^\top (u - \bar{u})\|$. Running cosine k-means on z gives H region centroids μ_h. At evaluation, each candidate is routed independently by nearest centroid, $g(c, \phi') = \arg\min_h \|z(c, \phi') - \mu_h\|_2^2$, a hard assignment to a single region.

The region count $H \in \{1, 2, 3\}$ is chosen on training-to-validation trajectories by a one-standard-error rule (discarding configurations with any expert below 10% occupancy), which biases toward fewer regions and recovers a single global model (H=1) when extra regions give no validation gain. H is frozen before the test split. The router reuses standard PCA and k-means; unlike clustered bandits over persistent user or arm identities, it clusters transient context-arm interactions as a reward-free controller of feedback-sharing granularity.

3.2 SCB

SCB is a semantic contextual bandit, which uses a shared parameter to estimate the UCB's score to select the optimal arm.

3.2.1 JOINT FEATURE MAPPING

We apply PCA to reduce the dimensionality of the arm features, aiming to reduce computational cost. We define the joint feature $\psi(c, \phi)$ to capture context-arm interactions:

$$\psi(c, \phi) = [\, c;\ \phi;\ c^\top \phi';\ 1 \,] \tag{3}$$

where $\phi = \phi(a) \in \mathbb{R}^r$ is the PCA-projected arm feature defined above, $\phi' = \phi'(a) \in \mathbb{R}^d$ is its original semantic embedding before projection, and $c \in \mathbb{R}^p$ is the context embedding. The similarity term $c^\top \phi'$ is computed in the original embedding space, and the trailing 1 is a bias term. Here $\psi(c, \phi(a))$ is shorthand for $[\, c; \phi(a); c^\top \phi'(a); 1 \,]$, where $\phi'(a)$ is retained for the similarity term.

3.2.2 ALGORITHM

The full per-round selection and update procedure for a single SCB expert is given in Algorithm 1 (Appendix A).

4 THEORETICAL ANALYSIS

4.1 SCB THEORETICAL ANALYSIS

Theorem 4.1 (SCB Regret Bound). Under Assumptions 2.1–2.2, set the exploration radius to

$$\alpha := \max\{\beta_T, 1/2\}, \qquad \beta_T := \sigma_{\text{noise}} \sqrt{m \log\Big(\tfrac{1 + TB^2/\lambda}{\delta}\Big)} + \sqrt{\lambda}\, S, \tag{4}$$

where B is the feature-norm bound. Then with probability at least $1 - \delta$,

$$\bar{R}(T) \;\le\; 2\alpha \sqrt{2mT \log\big(1 + TB^2/(m\lambda)\big)} \;=\; O\Big(m\sqrt{T}\, \log T\Big). \tag{5}$$

Proof in Appendix D.

Theorem 4.2 (Advantage over Independent Learning). Consider any algorithm that treats each distinct arm as a separate action, sharing no information across arms through the embedding. For $K_{\text{total}} \le T$, where K_{total} is the total number of distinct arms across T rounds, there exists a problem instance on which the expected regret of any such algorithm satisfies

$$\bar{R}_{\text{indep}}(T) = \Omega\left(\sqrt{K_{\text{total}} \cdot T}\right). \tag{6}$$

Proof in Appendix D.

Theorem 4.3 (Generalization to Unseen Arms). Suppose the feature map is L_ψ-Lipschitz in the embedding, i.e. $\|\psi(c,\phi_1) - \psi(c,\phi_2)\| \le L_\psi \|\phi_1 - \phi_2\|$. On the event of Theorem 4.1, for any new arm a_{new} never previously selected, with $\psi_{\text{new}} = \psi(c, \phi(a_{\text{new}}))$ and statistics $V_t, \hat{\theta}_t, \beta_t$ at round t,

$$\left|\theta^{*\top}\psi_{\text{new}} - \hat{\theta}_t^\top \psi_{\text{new}}\right| \;\le\; \beta_t\Big(\|\psi_j\|_{V_t^{-1}} + \tfrac{1}{\sqrt{\lambda}}\, L_\psi\, \|\phi(a_{\text{new}}) - \phi(a_j)\|\Big) \tag{7}$$

for every previously observed arm j; in particular for the nearest such j.

Proof in Appendix D. This result formalizes the intuition that SCB predicts accurately for novel arms that are semantically close to previously observed ones.

4.2 RSCB Regret under Fixed Routing

Consider H experts under correct (oracle) routing: each context–arm pair is assigned to its region, and expert h receives T_h samples with $\sum_{h=1}^{H} T_h = T$. Since the router is reward-free and frozen, each routed subsequence has conditional mean $\psi^\top \theta_h^*$ and R-sub-Gaussian noise, so Theorem 4.1 applies per region, and under correct routing the total regret decomposes additively:

$$\bar{R}(T) = \sum_{h=1}^{H} \bar{R}_h(T_h) \le \sum_{h=1}^{H} O\big(m\sqrt{T_h}\log T_h\big) \le O\big(m\sqrt{HT}\log T\big), \tag{8}$$

using $\log T_h \le \log T$ and $\sum_{h=1}^{H}\sqrt{T_h} \le \sqrt{HT}$ (Cauchy–Schwarz). This bound assumes correct routing; it does not account for routing mistakes of the learned gate.

5 Experiments

We evaluate RSCB on two domains: (1) Python bug localization on the full SWE-bench test split (Jimenez et al., 2024), and (2) cross-domain validation on an industrial Android/Java security bug dataset. We further validate on the public BugsInPy benchmark (Widyasari et al., 2020), where RSCB again learns a multi-region structure (Appendix C).

5.1 Baselines

- **Random**: Uniform random selection from candidates.
- **LLM-Top1**: Always selects the LLM's top-ranked candidate.
- **LLM-Confidence**: Selects based on LLM self-reported confidence scores.
- **UCB1** (Auer et al., 2002): Classical UCB with arm-level statistics (arms hashed to buckets).
- **LinUCB** (Li et al., 2010): Standard contextual bandit with per-arm parameters.
- **NeuralUCB** (Zhou et al., 2020): Neural network reward model with UCB exploration.
- **KernelUCB** (Valko et al., 2013): Kernel ridge regression with UCB; a dynamic-arm baseline that generalizes via kernel similarity.
- **SC-LinUCB** (Müller, 2025): a LinUCB variant with semantic features and a single shared parameter vector across arms.
- **RSCB (Ours)**: Regional Semantic Contextual Bandit; routes each context-arm pair to one of H regional SCB experts via a reward-free gate, with H selected on validation.

5.2 Experiment 1: SWE-bench Full Test Split

Setup. We construct a fully real evaluation pipeline on SWE-bench (Jimenez et al., 2024), using the *entire* test split: all 2,294 real Python bug instances across 12 open-source repositories such as `django`, `sympy`, and `scikit-learn`. Of these, 1,723 yield candidate sets with a non-degenerate reward signal; the remaining 571 are excluded because all generated candidates receive identical reward scores under our patch-grounding metric (typically zero), making bandit learning

uninformative—this occurs when none of the LLM-generated hypotheses mention the specific functions, classes, or files modified by the ground-truth patch. This filtering applies uniformly to all methods and does not favor RSCB. For each bug, we query a real LLM (a 9B Qwen-family model) once to generate $K \approx 5$ distinct root-cause hypotheses; these hypotheses are the dynamic arms. We encode both the problem statement and each hypothesis with a pretrained sentence encoder (`all-MiniLM-L6-v2`, 384-dim), so the arm embeddings carry genuine semantics; arm embeddings are PCA-projected to $r = 256$ dimensions (Section 2). Crucially, the reward is *grounded*: we parse the ground-truth patch to extract the files, functions, and classes it actually modifies, and score each hypothesis objectively by whether it identifies those (function/class match $= 1.0$, file/module match $= 0.5$, otherwise 0). This reward is independent of any LLM judgment, eliminating self-preference bias (see Appendix B for empirical validation).

Protocol. To measure generalization to *unseen* arms—the central claim of RSCB—we split the 1,723 usable bugs into train (965), validation (241), and test (517) pools. Only RSCB's region count H is selected on validation, and the test split is used solely for final evaluation. The agent trains online for 1,000 rounds (sampling from the train pool) and is then evaluated on 500 rounds drawn from the held-out test pool (517 bugs whose hypotheses are never seen during training). We report cumulative regret and average reward on the test pool, averaged over 10 random seeds. We additionally report Hit@1, defined as the fraction of test rounds in which the selected candidate attains the maximum reward in its candidate set, i.e. $\frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} 1[\, r_t(i_t) = \max_i r_t(i)\,]$.

Table 1: Performance on the SWE-bench held-out test set (N=517) with patch-grounded rewards. Results averaged over 10 seeds.

Method	**Avg Reward ↑**	**Cum. Regret ↓**	**Hit@1 ↑**
Random	0.384	185.4	0.450
LLM-Top1	0.476	139.2	0.582
LLM-Confidence	0.468	142.9	0.571
UCB1	0.404	175.2	0.475
LinUCB	0.442	156.0	0.525
NeuralUCB	0.391	181.75	0.458
KernelUCB	0.468	142.95	0.565
SC-LINUCB	0.560	97.20	0.696
RSCB H=1(Ours)	**0.562**	**95.95**	**0.697**

On held-out bugs with a patch-grounded, LLM-independent reward, RSCB attains the highest mean reward and lowest mean regret (Table 1), though its margin over the globally shared SC-LinUCB is small and not significant on this domain ($p = 0.57$). The non-semantic baselines (UCB1, NeuralUCB, and LinUCB) trail the semantic bandits by a large margin, confirming that sharing parameters across arms through semantic embeddings is essential when the arm set is dynamic. Two further observations emerge. First, KernelUCB matches the LLM heuristics in reward (0.468 vs. 0.476 for LLM-Top1) yet remains far behind the semantic bandits, and its kernel evaluations scale with the accumulated data, making it the most expensive method per round. Second, LLM-Top1 is a strong heuristic (Hit@1 $= 0.582$), reflecting that modern LLMs produce well-ranked candidates; nonetheless, RSCB significantly surpasses it ($+18.2\%$ reward, $p = 0.002$) by learning online which candidates are truly grounded. On SWE-bench the one-standard-error rule selects $H = 1$ on validation, so RSCB reduces to a single global model; its near-parity with SC-LinUCB therefore reflects the absence of exploitable regional structure in this domain rather than a failure of the method. In contrast, on the security domain the same procedure selects $H > 1$ and yields a significant gain, illustrating that RSCB adapts its sharing granularity to the data.

5.3 Experiment 2: Cross-Domain Validation (Industrial Security Dataset)

A key concern is whether RSCB's advantage is specific to the SWE-bench/Python setting or generalizes across domains. We validate on an industrial dataset of 1,642 Android/Java security bug-fix commits from a proprietary mobile security application. This domain differs from SWE-bench in

language (Java vs. Python), project structure (single monorepo vs. diverse open-source repos), and bug characteristics (security-critical fixes vs. general issues).

Setup. We apply the identical pipeline: for each bug, an LLM generates $K \approx 5$ root-cause hypotheses (dynamic arms), encoded with the same sentence encoder (384-dim). The reward is objectively grounded via gold-patch parsing—no LLM judge. Of 1,642 bugfix instances, 725 yield candidate sets with a non-degenerate reward signal. We split into train (406), validation (101), and test (218) pools with the same protocol as SWE-bench.

Table 2: Cross-domain validation on an industrial Android/Java security dataset (725 usable instances; 218 held-out test bugs; 800 train + 400 test rounds; patch-grounded reward; 10 seeds). RSCB significantly outperforms all baselines (Wilcoxon signed-rank; $p = 0.002$ vs. non-semantic baselines, $p = 0.006$ vs. SC-LinUCB) in a completely different software domain.

Method	**Avg Reward** ↑	**Cum. Regret** ↓	**Hit@1** ↑
Random	0.227	135.05	0.385
LLM-Top1	0.285	111.7	0.490
LLM-Confidence	0.276	115.25	0.477
UCB1	0.256	123.25	0.439
LinUCB	0.325	95.80	0.551
NeuralUCB	0.238	130.55	0.401
KernelUCB	0.294	108.10	0.490
SC-LINUCB	0.375	75.75	0.646
RSCB H=2(Ours)	**0.395**	**67.60**	**0.679**

Results. The results (Table 2, Figure 2, Figure 3) show a consistent trend: RSCB significantly outperforms *every* baseline in reward, regret and Hit@1 (Wilcoxon signed-rank), including KernelUCB (+34.4% reward) and LLM-Top1 (+38.7% reward), with $p = 0.002$ against all non-semantic baselines and $p = 0.006$ against the strongest baseline, SC-LinUCB. The advantage over LLM-Top1 is far larger here than on SWE-bench, because the LLM's self-ranking is weaker on this domain (Hit@1 $= 0.490$ vs. 0.582 on SWE-bench)—precisely the scenario where online learning provides the most value. This shows that RSCB's regional sharing yields significant gains where regional structure exists, and that RSCB remains the best-performing method across both domains, generalizing over programming languages, project structures, and bug types without any domain adaptation.

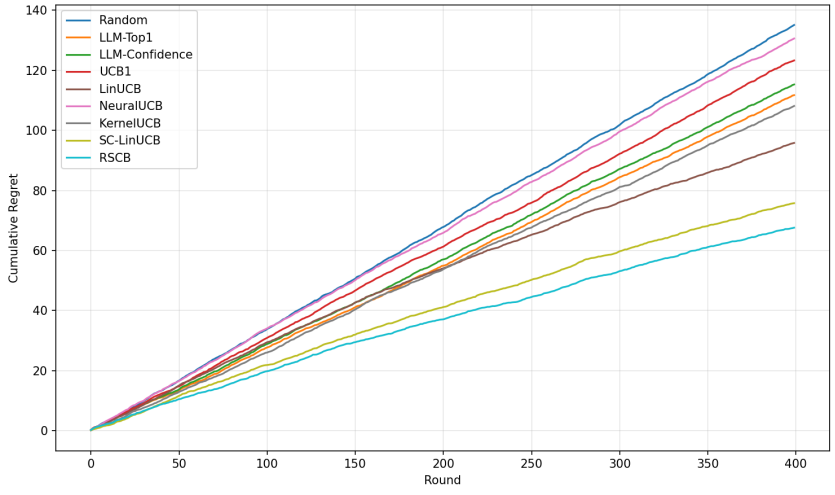

Figure 2: Cumulative regret on held-out industrial security bugs. RSCB's advantage is even more pronounced than on SWE-bench, with all baselines accumulating markedly higher cumulative regret.

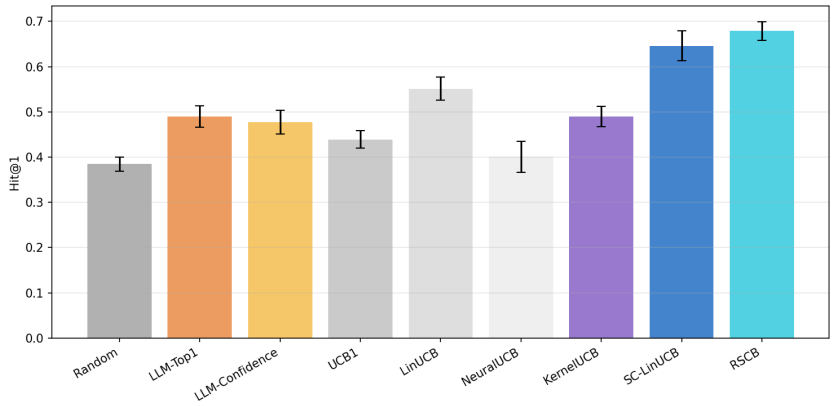

Figure 3: Hit@1 on industrial security bugs. RSCB achieves highest Hit@1 across all baseline methods.

6 Related Work

Linear and Nonlinear Contextual Bandits. The problem of contextual bandits has been studied by several scholars (Pandey et al., 2007; Wang et al., 2005); these studies usually impose assumptions based on prior knowledge of the arms. Langford & Zhang (2007) proposed the Epoch-Greedy algorithm and established the modern framework of contextual bandits. LinUCB (Li et al., 2010) models expected rewards as linear functions of context features. While the original paper focused on practical disjoint or hybrid models, subsequent OFUL analysis by Abbasi-Yadkori et al. (2011) established a regret bound of $\tilde{O}(d\sqrt{T})$ for linear contextual bandits. Subsequent work relaxed the linearity assumption: NeuralUCB (Zhou et al., 2020) replaces the linear predictor with a neural network while retaining UCB-style exploration via a neural tangent kernel approximation, and KernelUCB (Valko et al., 2013) models rewards in a reproducing kernel Hilbert space, achieving flexible nonparametric estimation at $O(T^2)$–$O(T^3)$ computational cost. Thompson Sampling variants (Agrawal & Goyal, 2012; 2013) offer a Bayesian alternative to UCB-style exploration, with provably comparable regret guarantees. Closest to our setting, SC-LinUCB (Müller, 2025) scores dynamically generated candidates through their semantic embeddings under a single globally shared parameter vector, enabling generalization to unseen arms; RSCB adopts this semantic-feature view but replaces the single global model with region-specific estimators to avoid negative transfer, and recovers a single globally shared model when one region is used. A common limitation of the earlier methods is the *fixed arm set* assumption: each arm is either assigned its own parameter vector (LinUCB) or indexed in a fixed kernel matrix (KernelUCB). When arms are dynamically generated and previously unseen, these methods cannot generalize without retraining or re-indexing—precisely the gap RSCB addresses.

Bandits with Large or Dynamic Action Spaces. Lipschitz bandits (Kleinberg et al., 2019) pioneer the study of continuous arm spaces under metric smoothness assumptions, achieving regret that depends on the *zooming dimension* rather than the ambient dimension. Krishnamurthy et al. (2020) address contextual bandits with continuous actions by smoothing the actions and adapting to unknown smoothness parameters, enabling efficient learning in large action spaces without requiring strong Lipschitz assumptions on the loss function. More recently, Lattimore et al. (2020) study bandits with good feature representations, showing that linear structure in the action embedding enables efficient learning. However, these approaches either lack contextual information (Lipschitz bandits), assume a fixed action set that is known in advance (Krishnamurthy et al., 2020), or focus on representation learning rather than dynamic arm generalization. RSCB differs by jointly embedding *both* contexts and arms in a shared space with explicit interaction features, enabling zero-shot generalization to arms generated on-the-fly.

RL-Based Fault Localization. Prior work has explored using reinforcement learning and LLM agents for debugging. RLocator (Chakraborty et al., 2024) formulated bug localization as a Markov

decision process, training a policy to rank suspicious source code files. Ma et al. (2025) trained LLM agents for repository-level tool use and issue localization via a two-stage pipeline of rejection-sampled SFT and tool-integrated RL. While effective, such RL-based approaches require expensive offline training and produce task-specific policies that do not transfer. RSCB is complementary: it operates as a lightweight, online selection layer atop *any* candidate generator, requiring no model fine-tuning and adapting in real time through bandit feedback.

LLM-Based Fault Localization. Recent advances in this domain have explored diverse paradigms. Some methods train lightweight adapters to probe LLM representations for test-free line-level localization (Yang et al., 2024). Others adopt iterative, agentic workflows where models navigate repositories via function calls to reason about and explain suspicious locations (Kang et al., 2024). Hossain et al. (2024) propose Toggle, a framework that performs token-level localization to guide subsequent repair. More recent studies scale these agents to large, real-world systems, such as diagnosing faults in the Linux kernel (Zhou et al., 2026), and critically examine their robustness—showing that LLM localizability often relies on syntactic rather than semantic cues, and degrades sharply under semantic-preserving code changes (Haroon et al., 2026). While these techniques improve candidate generation, they operate within a single-pass inference paradigm. Our approach is complementary: rather than improving the LLM's generation, RSCB learns to select from whatever candidates are produced, exploiting cross-candidate and feedback signals via a contextual bandit. Because generation inevitably leaves residual uncertainty over the top candidate, adaptive selection provides value that persists as base models strengthen; empirically, while RSCB's relative improvement diminishes as baseline Hit@1 increases, the absolute gains remain statistically significant (Section 5), confirming that generation and selection provide independent sources of value.

7 Conclusion and Limitations

We introduced RSCB, a regional semantic contextual bandit that routes each context–arm pair to one of H region-specific estimators through a reward-free gate, providing a middle ground between independent per-arm estimation and unrestricted global semantic sharing. Under oracle routing and regional realizability we prove an $O(m\sqrt{HT}\log T)$ pseudo-regret bound independent of the number of distinct arms, in contrast to an $\Omega(\sqrt{K_{\text{total}}T})$ lower bound for arm-independent learning. Across SWE-bench, an Android/Java security dataset, and the public BugsInPy benchmark, all with objective patch-grounded rewards, RSCB adapts its sharing granularity per domain: it collapses to a single global model on SWE-bench (a small, non-significant improvement over the strongest baseline) while learning multiple regions on the security dataset ($H{=}2$, a significant gain) and on BugsInPy ($H{=}3$, the lowest regret of all methods). This is evidence for a domain-dependent boundary between global and regional semantic sharing rather than universal superiority.

Limitations.

- **Routing guarantee**: our regret analysis assumes correct/oracle routing. A guarantee for the learned, frozen gate—whose routing mistakes may contaminate regional estimators—remains open.
- **Linear realizability**: RSCB assumes rewards are linear in the joint feature map (Assumption 2.1), the standard premise of linear contextual bandits, which holds only approximately on real data. As formalized in Remark 2.1, under misspecification error ε regret degrades gracefully by an additive $O(\varepsilon\sqrt{m}\,T)$ term rather than collapsing.
- **Reward grounding**: patch-grounded rewards are LLM-independent (their key virtue) but have limited recall—a hypothesis phrased differently from the patch may be under-credited; this noise is neutral across methods.

Extending RSCB to other LLM selection tasks (e.g., tool selection, code repair strategy) and to cost-aware selection are natural next steps.

References

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in neural information processing systems*, 2011.

Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Proceedings of the 25th Annual Conference on Learning Theory*. PMLR, 2012.

Shipra Agrawal and Navin Goyal. Thompson sampling for contextual bandits with linear payoffs. In *Proceedings of the 30th International Conference on Machine Learning*. PMLR, 2013.

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

Partha Chakraborty, Mahmoud Alfadel, and Meiyappan Nagappan. Rlocator: Reinforcement learning for bug localization. *IEEE Transactions on Software Engineering*, 50, 2024.

Sabaat Haroon, Ahmad Faraz Khan, Ahmad Humayun, Waris Gill, Abdul Haddi Amjad, Ali R. Butt, Mohammad Taha Khan, and Muhammad Ali Gulzar. Assessing the impact of code changes on the fault localizability of large language models. In *Proceedings of the 2026 IEEE International Conference on Software Testing, Verification and Validation (ICST)*, 2026.

Soneya Binta Hossain, Nan Jiang, Qiang Zhou, Xiaopeng Li, Wen-Hao Chiang, Yingjun Lyu, Hoan Nguyen, and Omer Tripp. A deep dive into large language models for automated bug localization and repair. *Proceedings of the ACM on Software Engineering*, 1, 2024.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? In *International Conference on Learning Representations*, 2024.

Sungmin Kang, Gabin An, and Shin Yoo. A quantitative and qualitative evaluation of llm-based explainable fault localization. In *Proceedings of the ACM on Software Engineering (FSE)*, volume 1. Association for Computing Machinery, 2024.

Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. Bandits and experts in metric spaces. *Journal of the ACM (JACM)*, 66, 2019.

Akshay Krishnamurthy, John Langford, Aleksandrs Slivkins, and Chicheng Zhang. Contextual bandits with continuous actions: Smoothing, zooming, and adapting. *Journal of Machine Learning Research*, 21, 2020.

John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in neural information processing systems*, 2007.

Tor Lattimore, Csaba Szepesvari, and Gellert Weisz. Learning with good feature representations in bandits and in rl with a generative model. In *International conference on machine learning*. PMLR, 2020.

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, 2010.

Zexiong Ma, Chao Peng, Qunhong Zeng, Pengfei Gao, Yanzhen Zou, and Bing Xie. Tool-integrated reinforcement learning for repo deep search. *arXiv preprint arXiv:2508.03012*, 2025.

Robert Müller. Semantic context for tool orchestration. *arXiv preprint arXiv:2507.10820*, 2025.

Sandeep Pandey, Deepayan Chakrabarti, and Deepak Agarwal. Multi-armed bandit problems with dependent arms. In *Proceedings of the 24th international conference on Machine learning*, 2007.

Michal Valko, Nathaniel Korda, Rémi Munos, Ilias Flaounas, and Nelo Cristianini. Finite-time analysis of kernelised contextual bandits. *arXiv preprint arXiv:1309.6869*, 2013.

Chih-Chun Wang, Sanjeev R Kulkarni, and H Vincent Poor. Bandit problems with side observations. *IEEE Transactions on Automatic Control*, 50, 2005.

Ratnadira Widyasari, Sheng Qin Sim, Camellia Lok, Haodi Qi, and Eng Lieh Ouh. Bugsinpy: a database of existing bugs in python programs to enable controlled testing and debugging studies. *ACM*, 2020.

Aidan Z. H. Yang, Claire Le Goues, Ruben Martins, and Vincent Hellendoorn. Large language models for test-free fault localization. In *Proceedings of the 46th IEEE/ACM international conference on software engineering*, 2024.

Dongruo Zhou, Lihong Li, and Quanquan Gu. Neural contextual bandits with ucb-based exploration. In *Proceedings of the 37th International Conference on Machine Learning*. PMLR, 2020.

Zhenhao Zhou, Zhuochen Huang, Yike He, Chong Wang, Jiajun Wang, Yijian Wu, Xin Peng, and Yiling Lou. Taming system complexity: Demystifying software engineering agents in diagnosing linux kernel faults. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2026.

THE USE OF LARGE LANGUAGE MODELS

We used a large language model as a general-purpose writing and coding assistant: to polish prose, check grammar, and help debug and refactor experiment scripts. All research ideas, the method design, the theoretical analysis, and the experimental protocol were conceived and carried out by the authors, who verified every result and take full responsibility for the content of this paper.

A SCB EXPERT ALGORITHM

Algorithm 1 details the per-round selection and update procedure for a single SCB expert. Within RSCB, each region maintains its own instance of these statistics $(V, b, \hat{\theta})$, and the router directs each context–arm pair to the corresponding expert.

Algorithm 1 SCB: Semantic Contextual Bandit

Require: Exploration parameter $\alpha > 0$, regularization $\lambda > 0$

Initialize: $V \leftarrow \lambda I_m$, $b \leftarrow \mathbf{0}_m$, $\hat{\theta} \leftarrow \mathbf{0}_m$, $V^{-1} \leftarrow \frac{1}{\lambda} I_m$
for $t = 1, 2, \ldots, T$ **do**
  Observe context c_t; receive candidates $\mathcal{A}_t = \{a_1, \ldots, a_{K_t}\}$
  for $i = 1, \ldots, K_t$ **do**
    Compute $\psi_i \leftarrow \psi(c_t, \phi(a_i))$
    $\mu_i \leftarrow \hat{\theta}^\top \psi_i$
    $\sigma_i \leftarrow \sqrt{\psi_i^\top V^{-1} \psi_i}$
    $\text{score}_i \leftarrow \mu_i + \alpha \cdot \sigma_i$
  end for
  Select $i_t \leftarrow \arg\max_i \ \text{score}_i$
  Execute arm a_{i_t}, observe reward r_t
  $V^{-1} \leftarrow V^{-1} - \frac{(V^{-1}\psi_{i_t})(\psi_{i_t}^\top V^{-1})}{1+\psi_{i_t}^\top V^{-1}\psi_{i_t}}$
  $V \leftarrow V + \psi_{i_t}\psi_{i_t}^\top; \quad b \leftarrow b + r_t \cdot \psi_{i_t}$
  $\hat{\theta} \leftarrow V^{-1} b$
end for

B WHY NOT AN LLM JUDGE?

A natural alternative to our patch-grounded reward is to use an LLM as judge. We evaluate this by comparing judge scores against objective patch-grounded labels over 4,961 (hypothesis, patch) pairs from 1,000 SWE-bench instances (using the same 9B Qwen-family model as both candidate generator and judge). The two scoring systems are essentially uncorrelated: Pearson $r = 0.016$, Cohen's $\kappa = -0.012$, and binary agreement 47% (near chance). An LLM judge would thus contaminate the reward signal with noise uncorrelated to actual hypothesis quality. This motivates our LLM-independent, patch-grounded reward throughout all experiments.

C PUBLIC BENCHMARK VALIDATION: BUGSINPY

To assess reproducibility on a public benchmark and to test whether RSCB's multi-region behavior is specific to the industrial dataset, we additionally evaluate on BugsInPy (Widyasari et al., 2020), a public database of real Python bugs from open-source projects. We apply the identical pipeline used for our other benchmarks: for each bug, a 9B Qwen-family model generates $K \approx 5$ root-cause hypotheses (dynamic arms), which are encoded with the same sentence encoder (`all-MiniLM-L6-v2`, 384-dim) and PCA-projected to $r = 256$; rewards are objectively patch-grounded with no LLM judge. The context for each bug is the title line of its fix commit message, with issue numbers and file names removed to avoid leaking the ground-truth location. Of 501 collected bugs, 275 yield candidate sets with a non-degenerate reward signal, which we split into train (154), validation (39), and test (82) pools. As before, we run 1,000 training rounds and 500 test rounds, averaged over 10 seeds, with H selected on validation.

On BugsInPy the one-standard-error rule selects $H = 3$ (validation regret 34.3, 39.3, 31.1 for $H = 1, 2, 3$), so RSCB learns a multi-region structure on a public benchmark—echoing its behavior on the industrial security dataset rather than the single-region collapse seen on SWE-bench. RSCB attains the best reward, regret, and Hit@1 of all methods (Table 3). Its advantage over the non-semantic baselines and heuristics is statistically significant (Wilcoxon signed-rank, $p = 0.002$), while its improvement over the strongest baseline SC-LinUCB is not significant on this smaller test pool ($p = 0.084$). This provides public, reproducible evidence that RSCB's regional structure emerges beyond the proprietary setting, while remaining honest that its edge over a well-tuned global semantic model is domain-dependent.

Table 3: Public BugsInPy benchmark (275 usable instances; 82 held-out test bugs; 1,000 train + 500 test rounds; patch-grounded reward; 10 seeds). RSCB selects $H = 3$ and attains the best score on every metric. p-values are Wilcoxon signed-rank on cumulative regret versus RSCB.

Method	**Avg Reward ↑**	**Cum. Regret ↓**	**Hit@1 ↑**	p **vs. RSCB**
Random	0.349	220.55	0.398	0.002
LLM-Top1	0.477	156.70	0.571	0.002
LLM-Confidence	0.477	156.70	0.571	0.002
UCB1	0.446	172.25	0.516	0.002
LinUCB	0.599	95.45	0.715	0.002
NeuralUCB	0.356	216.90	0.412	0.002
KernelUCB	0.483	153.85	0.560	0.002
SC-LinUCB	0.648	71.20	0.790	0.084
RSCB H=3 (Ours)	**0.664**	**62.95**	**0.818**	—

D FULL PROOFS

Throughout, $V_t = \lambda I_m + \sum_{s=1}^{t-1} \psi_s \psi_s^\top$, $\hat{\theta}_t = V_t^{-1} \sum_{s=1}^{t-1} r_s \psi_s$, and $\sigma_{t,i} = \sqrt{\psi_{t,i}^\top V_t^{-1} \psi_{t,i}}$. Recall B bounds $\|\psi\|$ and L_ψ is the Lipschitz constant of the feature map ψ in the embedding (Theorem 4.3); these are distinct.

Lemma D.1 (Self-normalized confidence ellipsoid). Under Assumptions 2.1, 2.2, for any $\delta \in (0, 1)$, with probability at least $1 - \delta$, simultaneously for all $t \geq 1$,

$$\|\hat{\theta}_t - \theta^*\|_{V_t} \;\leq\; \beta_t := \sigma_{\text{noise}} \sqrt{m \log\Big(\tfrac{1+(t-1)B^2/\lambda}{\delta}\Big)} + \sqrt{\lambda}\, S. \tag{9}$$

Proof. Follows directly from Abbasi-Yadkori et al. (2011, Thm. 1) applied to our setting. □

Lemma D.2 (Elliptical potential). If $\|\psi_t\| \leq B$ for all t, then $\sum_{t=1}^{T} \min(1, \sigma_t^2) \leq 2m \log\big(1 + TB^2/(m\lambda)\big)$, where $\sigma_t^2 = \psi_t^\top V_t^{-1} \psi_t$.

Proof. Standard determinant argument; see Abbasi-Yadkori et al. (2011, Lemma 11). □

Proof of Theorem 4.1. Work on the simultaneous $1-\delta$ confidence event of Lemma D.1. Thus, for every round t and every candidate i,

$$\left|(\hat{\theta}_t - \theta^*)^\top \psi_{t,i}\right| \le \beta_t \sigma_{t,i}.$$

Since $\alpha = \max\{\beta_T, 1/2\} \ge \beta_t$, the UCB index

$$U_{t,i} = \hat{\theta}_t^\top \psi_{t,i} + \alpha \sigma_{t,i}$$

is optimistic:

$$U_{t,i} \ge {\theta^*}^\top \psi_{t,i}.$$

Instantaneous regret. Let

$$i_t^* \in \arg\max_i {\theta^*}^\top \psi_{t,i}, \qquad i_t \in \arg\max_i U_{t,i},$$

and define the instantaneous pseudo-regret

$$\Delta_t := \mu_t(a_t^{i_t^*}) - \mu_t(a_t^{i_t}).$$

By linear realizability,

$$\Delta_t = {\theta^*}^\top \psi_{t,i_t^*} - {\theta^*}^\top \psi_{t,i_t}.$$

Therefore,

$$\begin{aligned}
\Delta_t &\le U_{t,i_t^*} - {\theta^*}^\top \psi_{t,i_t} \\
&\le U_{t,i_t} - {\theta^*}^\top \psi_{t,i_t} \\
&= (\hat{\theta}_t - \theta^*)^\top \psi_{t,i_t} + \alpha \sigma_{t,i_t} \\
&\le 2\alpha \sigma_{t,i_t}.
\end{aligned}$$

Because $\mu_t(a) \in [0,1]$, we also have $\Delta_t \le 1$. Hence,

$$\Delta_t \le \min\{1, 2\alpha\sigma_{t,i_t}\} \le 2\alpha \min\{1, \sigma_{t,i_t}\},$$

where the last inequality uses $\alpha \ge 1/2$.

Summation. On the same confidence event,

$$\begin{aligned}
\bar{R}(T) &= \sum_{t=1}^{T} \Delta_t \\
&\le 2\alpha \sum_{t=1}^{T} \min\{1, \sigma_{t,i_t}\} \\
&\le 2\alpha \sqrt{T \sum_{t=1}^{T} \min\{1, \sigma_{t,i_t}^2\}} \\
&\le 2\alpha \sqrt{2mT \log\left(1 + \frac{TB^2}{m\lambda}\right)},
\end{aligned}$$

where the second inequality follows from Cauchy–Schwarz and the last inequality follows from Lemma D.2. Since

$$\alpha = \max\{\beta_T, 1/2\} = O\left(\sqrt{m \log(T/\delta)}\right),$$

we obtain

$$\bar{R}(T) = O\left(m\sqrt{T}\log T\right)$$

with probability at least $1-\delta$. □

Proof of Theorem 4.2. Let $K = K_{\text{total}}$ and assume $2 \le K \le T$. Consider the following subclass of the dynamic-arm setting: the context is fixed, the same set of K arms is available at every round, and the arm representations provide no information that distinguishes their reward distributions. Hence, any arm-independent algorithm reduces to a policy for a standard stochastic K-armed bandit.

Choose an index J uniformly at random from $\{1, \dots, K\}$. Conditional on $J = j$, let arm j have Bernoulli rewards with mean

$$\frac{1}{2} + \Delta, \qquad \Delta = \frac{1}{4}\sqrt{\frac{K}{T}},$$

and let every other arm have Bernoulli rewards with mean $1/2$. Since $K \le T$, we have $\Delta \le 1/4$, so all means lie in $[1/2, 3/4] \subset [0, 1]$.

This is the standard hard family for stochastic K-armed bandits. The standard minimax lower-bound argument implies that, for some universal constant $c > 0$,

$$\frac{1}{K}\sum_{j=1}^{K} \mathbb{E}_j\big[\bar{R}_{\text{indep}}(T)\big] \ge c\sqrt{KT},$$

where $\mathbb{E}_j$ denotes expectation when arm j is the unique optimal arm. Consequently, there exists at least one index j such that

$$\mathbb{E}_j\big[\bar{R}_{\text{indep}}(T)\big] \ge c\sqrt{KT}.$$

Therefore, there exists a problem instance for which

$$\mathbb{E}\big[\bar{R}_{\text{indep}}(T)\big] = \Omega\Big(\sqrt{K_{\text{total}}T}\Big).$$

□

Proof of Theorem 4.3. Work on the simultaneous confidence event of Lemma D.1. For a new arm a_{new} under context c, let

$$\psi_{\text{new}} := \psi(c, \phi(a_{\text{new}})).$$

Then

$$\begin{aligned}\left|\theta^{*\top}\psi_{\text{new}} - \hat{\theta}_t^{\top}\psi_{\text{new}}\right| &= \left|(\theta^* - \hat{\theta}_t)^{\top}\psi_{\text{new}}\right| \\ &\le \|\theta^* - \hat{\theta}_t\|_{V_t}\|\psi_{\text{new}}\|_{V_t^{-1}} \\ &\le \beta_t\|\psi_{\text{new}}\|_{V_t^{-1}}.\end{aligned}$$

For any previously selected arm a_j, define its feature under the same context c by

$$\psi_{t,j} := \psi(c, \phi(a_j)).$$

By the triangle inequality,

$$\|\psi_{\text{new}}\|_{V_t^{-1}} \le \|\psi_{t,j}\|_{V_t^{-1}} + \|\psi_{\text{new}} - \psi_{t,j}\|_{V_t^{-1}}.$$

Since $V_t \succeq \lambda I$,

$$\|x\|_{V_t^{-1}} \le \frac{1}{\sqrt{\lambda}}\|x\|_2.$$

By the Lipschitz assumption on the feature map,

$$\|\psi_{\text{new}} - \psi_{t,j}\|_{V_t^{-1}} \le \frac{L_\psi}{\sqrt{\lambda}}\|\phi(a_{\text{new}}) - \phi(a_j)\|_2.$$

Combining the above inequalities gives

$$\left|\theta^{*\top}\psi_{\text{new}} - \hat{\theta}_t^{\top}\psi_{\text{new}}\right| \le \beta_t\left(\|\psi_{t,j}\|_{V_t^{-1}} + \frac{L_\psi}{\sqrt{\lambda}}\|\phi(a_{\text{new}}) - \phi(a_j)\|_2\right).$$

Since this holds for every previously selected arm a_j, it also holds for the previously selected arm closest to a_{new} in the embedding space. For a fixed feature vector, the first term decreases as the design matrix grows, while the second term is controlled by the embedding distance. □