

Diagnosing and Repairing Factual Errors in RAG under Budget Constraints

Soroush Hashemifar[†], Havva Alizadeh Noughabi[‡], Fattane Zarrinkalam^{†,‡,*}, Ali Dehghantanha[‡]

[†] College of Engineering, University of Guelph, Guelph, ON, Canada

[‡] Cyber Science Lab, School of Computer Science, University of Guelph, Guelph, ON, Canada

Abstract

Retrieval-Augmented Generation (RAG) improves the factuality of large language models by grounding responses in external evidence, yet real-world deployments remain fragile. Failures often stem from missing or weakly relevant evidence, as well as from generation that does not faithfully reflect the retrieved context. Many existing approaches rely on fine-tuning, privileged access to internal model signals, or resource-insensitive escalation strategies, which limits their practicality in black-box and budget-constrained settings. We propose D2R-RAG (Diagnose-to-Repair RAG), a model-agnostic and resource-aware framework that combines lightweight failure diagnosis with adaptive repair. D2R-RAG derives interpretable failure signatures from observable signals in the query, retrieved evidence, and generated response, and then selects from a small set of corrective actions under explicit latency and VRAM constraints. Experiments on FEVER and HotpotQA show that D2R-RAG improves reliability over recent baselines and achieves better accuracy–efficiency trade-offs across multiple compute budgets. The code is available at <https://github.com/CyberScienceLab/D2R-RAG/>.

Keywords: RAG, Factuality Verification, Contextual Bandits, Adaptive Repair.

1. Introduction

Retrieval-Augmented Generation (RAG) [1] has become a standard approach for improving the factuality of large language model (LLM) outputs by grounding generation in external evidence rather than relying solely on parametric memory. In many real deployments, however, RAG remains unreliable: correct answers may fail when the retriever misses key evidence, when retrieved passages are relevant but incomplete, or when the generator produces content that is not supported by the provided context [2]. These failure modes are stochastic and non-uniform, and they are most problematic in settings with strict latency and hardware budgets (e.g., limited GPU memory, rate-limited APIs, or cost-constrained edge/cloud deployments), where iterative retrieval or resource-intensive reranking are computationally prohibitive [3].

Recent work has proposed dynamic and self-correcting RAG variants from three directions: (1) training-time modifications, (2) inference-time control, and (3) learning-based decision policies. Self-RAG [4] trains generators to emit reflection control tokens that decide when to retrieve and how to critique intermediate outputs, improving factuality through self-assessment, but requires fine-tuning and architectural coupling that limits use in black-box or API-based settings. Moreover, DRAGIN [5] uses token-level uncertainty and other internal dynamics to trigger retrieval during inference, offering lightweight, training-free control, yet relies on logit access (unavailable in closed-source LLM APIs). Finally, learning-based policies like MBA-RAG [6] (multi-armed bandit balancing quality and efficiency, dependent on an external query-complexity predictor) and QueryBandits [7] (bandit-based query rewriting using semantic and lexical signals) demonstrate that online decision-making can improve robustness, but typically target specific pipeline stages or require additional supervision or modeling components.

* fzarrink@uoguelph.ca

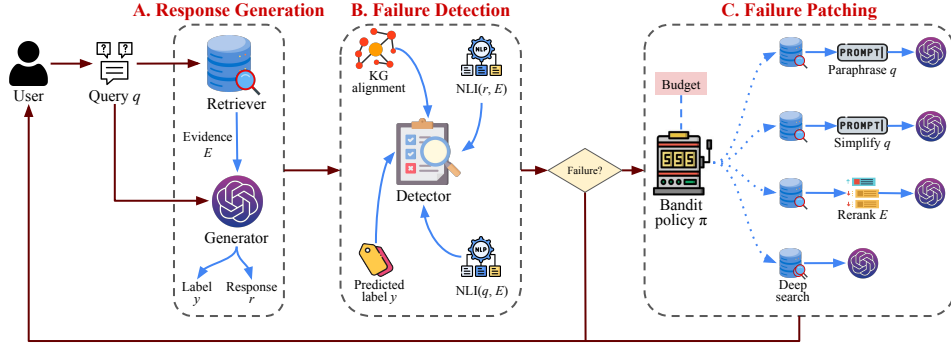


Figure 1. Overview of D2R-RAG.

This paper addresses *model-agnostic, resource-aware RAG recovery in black-box settings*, arguing that reliable recovery requires two capabilities: a *lightweight diagnostic* distinguishing retrieval-side evidence insufficiency from generation-side unfaithfulness using only observable artifacts, and the *least-cost corrective action* under explicit latency and VRAM budgets. We operationalize these principles in D2R-RAG (Diagnose-to-Repair RAG), which performs triangulated verification by combining (1) textual entailment checks between retrieved evidence and both the query and response, with (2) structured consistency checks that align relational triples extracted from the response against a knowledge graph to identify entity/predicate mismatches. These signals yield an interpretable failure signature separating missing evidence from unsupported generation. Repair selection is then formulated as a contextual multi-armed bandit [8] where each arm corresponds to a specific intervention (query rewriting, retrieval depth or mode adjustment, cross-encoder reranking, or index refresh), and a LinUCB policy [9] learns to balance expected factual recovery against computational cost. Notably, the framework requires neither generator fine-tuning nor access to internal logits, enabling deployment with closed-source LLM APIs.

We evaluate D2R-RAG on fact verification and multi-hop question answering using FEVER [10] and HotpotQA [11]. Our evaluation emphasizes both factual performance and operational efficiency under multiple latency/VRAM budget regimes. Beyond aggregate results, we analyze how the learned policy allocates high-cost interventions across different failure signatures and how performance changes as resource budgets tighten.

2. Methodology

Figure 1 overviews the D2R-RAG, which augments a RAG pipeline with two modules.

(1) Failure Diagnosis. This stage determines whether the initial RAG output is trustworthy and identifies the likely failure type, using query q , evidence E , response r , and predicted label y , which is suitable for black-box deployment. The goal is to produce an interpretable failure signature that separates retrieval-side insufficiency from generation-side unfaithfulness, rather than perfectly explaining every error. We derive the signature using three complementary signals: *query entailment* e^q , *response entailment* e^r , and *KG alignment status* κ [12]. See Appendix A for additional details.

Given (κ, e^q, e^r, y) , our failure types are: wrong-predicate (WP) for KG conflict, insufficient-evidence (IE) when retrieved evidence does not entail the query ($e^q = \text{neutral}$), wrong-response (WR) when the response is not entailed by retrieved evidence ($e^r \in \{\text{neutral}, \text{contradict}\}$), and label-evidence mismatch (LEM) for FEVER when the predicted y conflicts the query-context entailment. For FEVER, IE and LEM cases emit the abstention

label *Unverified* rather than a potentially correct label for the wrong reason, improving trustworthiness and prioritizing retrieval-oriented repairs (evidence-gated label prediction).

(2) Adaptive Repair via Contextual Bandits. D2R-RAG performs single-shot repair by selecting a patch and re-running the RAG pipeline, as a contextual multi-armed bandit (CMAB) [8]: the learner observes context vector $\mathbf{x} \in \mathbb{R}^d$, chooses action $a \in \mathcal{A}$, and receives a reward reflecting output quality and resource cost. The action space \mathcal{A} is intentionally small and black-box deployable: (1) prompt-level rewrites (paraphrasing and simplification) to reduce query ambiguity [7]; (2) reranker activation using a cross-encoder to improve evidence precision; and (3) retrieval-level interventions adjusting depth k , switching between BM25 and dense retrieval, or refreshing the index. We learn the repair policy with LinUCB [9], which assumes a linear reward model $\hat{r}(a | \mathbf{x}) = \mathbf{x}^\top \boldsymbol{\theta}_a$ and selects actions by maximizing an upper-confidence bound (more details in Appendix B):

$$a^* = \arg \max_{a \in \mathcal{A}} \left(\mathbf{x}^\top \boldsymbol{\theta}_a + \alpha \sqrt{\mathbf{x}^\top \mathbf{A}_a^{-1} \mathbf{x}} \right). \quad (2.1)$$

where α controls exploration and \mathbf{A}_a is the action-specific covariance matrix. After applying a^* , the revised output and measured action cost define the observed reward. We use a resource-aware reward scoring repairs by output reliability and budget compliance. Let L_a and V_a denote the latency and additional VRAM usage incurred by action a , with per-action budgets B_L and B_V . The reward is:

$$r(a) = \frac{1}{4} (\mathbb{1}_{NF} + \mathbb{1}_{KG} + 2 \cdot \mathbb{1}_{NLI}) \prod_{x \in \{L, V\}} \mathbb{1}[x_a \leq B_x] \left(1 - \frac{x_a}{B_x} \right). \quad (2.2)$$

where $\mathbb{1}_{NF}$ indicates a **NoFailure** diagnosis after repair, $\mathbb{1}_{KG}$ indicates triple-level consistency, and $\mathbb{1}_{NLI}$ indicates entailment support. Hard gates $\mathbb{1}[x_a \leq B_x]$ zero out rewards for budget-violating actions, while multiplicative terms encourage cheaper actions among feasible ones. Following each interaction, LinUCB updates only the selected action:

$$\mathbf{A}_a \leftarrow \mathbf{A}_a + \mathbf{x}\mathbf{x}^\top, \quad \mathbf{b}_a \leftarrow \mathbf{b}_a + r(a)\mathbf{x}, \quad \boldsymbol{\theta}_a \leftarrow \mathbf{A}_a^{-1} \mathbf{b}_a. \quad (2.3)$$

3. Experiments

We investigate three research questions: **RQ1:** Does D2R-RAG improve factual quality while respecting deployment costs compared to baselines? **RQ2:** Does the policy adapt interventions to specific failure modes rather than using a uniform strategy? **RQ3:** How does the reward formulation in D2R-RAG determine the learned repair strategy, and how stable is this under varying latency/VRAM budgets?

3.1. Experimental Setup

Datasets. We evaluate D2R-RAG on FEVER [10] (2,000 claim verification instances) and HotpotQA [11] (1,000 multi-hop question answering instances).

Baselines. We evaluate D2R-RAG against *Naive-RAG* (a single-pass RAG), *Query Paraphrase* [13] (rewriting the query conditioned on the retrieved context), *Context Expansion* (increasing dense retrieval depth from k to $k' = 20$), and *Thompson Sampling* [14] (replacing LinUCB with a context-free policy). All methods utilize a *GPT-4o-mini* generator and 128-token chunks with 16-token overlap, within LlamaIndex.

Evaluation Metrics. We report *Exact Match (EM)* for HotpotQA, *Accuracy (ACC)* for FEVER, and response *Relevance* and *Faithfulness*. Efficiency is measured by Latency (retrieval to final output) and peak VRAM (maximum GPU memory during execution).

Implementation Settings. We use *DeBERTa-v3-large-nli* for entailment check and *Babelscape/rebel-large* for triple extraction. We impose 3s latency and 6 MB VRAM per query. The repair policy is trained online for two epochs, with exploration parameter $\alpha=2$.

3.2. Results

RQ1. Table 1 shows that D2R-RAG improves factual quality while maintaining efficiency constraints: bold and underlined values represent best and second-best results per dataset/metric, respectively. On FEVER, it improves ACC from 56.3% (*Naive-RAG*) to 61.5% (*Thompson Sampling*) and 60.8% (*LinUCB*), with *Thompson Sampling* achieving the best cost profile (1.14s, 0.17 MB) compared to *LinUCB* (1.47s, 0.36 MB). On HotpotQA, *Thompson Sampling* reaches 40.4% EM with 71.52% faithfulness, while *LinUCB* matches 39.8% EM but at higher latency (3.41s vs 2.37s); *LinUCB* occasionally exceeds the VRAM budget (3.41 MB), since budget constraints are enforced per individual repair action rather than end-to-end. Overall, *LinUCB* prioritizes quality at higher resource cost, while *Thompson Sampling* offers a stronger quality–efficiency trade-off (see Appendix C).

Dataset	Method	Latency	VRAM	Relevance	Faithfulness	ACC	EM
FEVER	Naive-RAG	<u>1.16</u>	<u>0.20</u>	50.16	91.67	56.3	-
	Query Paraphrase	1.32	<u>0.20</u>	50.09	<u>92.69</u>	56.7	-
	Context Expansion	1.35	<u>0.20</u>	<u>50.24</u>	92.82	61.8	-
	D2R-RAG (LinUCB)	1.47	0.36	50.80	92.34	60.8	-
	D2R-RAG (T.Sampling)	1.14	0.17	50.00	92.37	<u>61.5</u>	-
HotpotQA	Naive-RAG	1.52	0.36	28.39	68.58	-	36.2
	Query Paraphrase	<u>1.60</u>	0.36	27.90	74.65	-	39.8
	Context Expansion	1.52	0.36	32.14	70.70	-	40.6
	D2R-RAG (LinUCB)	3.41	<u>1.04</u>	<u>31.66</u>	69.89	-	39.8
	D2R-RAG (T.Sampling)	2.37	1.23	30.69	<u>71.52</u>	-	<u>40.4</u>

Table 1. Comparison of D2R-RAG and baselines on FEVER and HotpotQA.

RQ2. Figure 2 shows that D2R-RAG conditions actions on failure type. On FEVER, IE cases favor retrieval escalation ($\sim 60\text{--}80\%$ on deeper dense/BM25), while on HotpotQA, WR and WP failures route to reranker activation, indicating that selecting the right passages (not merely retrieving more) is critical for multi-hop questions. *LinUCB* explores broader action distribution, while *Thompson Sampling* exploits more (concentrates on high-utility repairs), yet both preserve the same failure-to-action mapping.

RQ3. Table 2 ablates the two mechanisms that encode deployment constraints: *Unweighted*, which removes the soft resource-weighting terms, and *Unconstrained*, which removes the hard budget gates. On HotpotQA, *Unconstrained* reduces latency (3.41 \rightarrow 2.03s) and VRAM (1.04 \rightarrow 0.84 MB) with minimal EM drop, while *Unweighted* improves quality metrics (Relevance 31.66 \rightarrow 32.20%; Faithfulness 69.89 \rightarrow 70.59%) with near-identical EM. On FEVER, both ablations improve ACC (60.8 \rightarrow 61.3-61.4%) and reduce memory. Budget sensitivity tests (Table 3) show *Stringent* (0.7 \times) cuts VRAM on both datasets while preserving performance, and *Relaxed* (1.5 \times) lowers latency and boosts faithfulness. Notably, D2R-RAG achieves the highest HotpotQA EM at highest cost, demonstrating that budget shaping controls quality–cost trade-offs. More results are provided in Appendix C.

4. Conclusion

We introduced D2R-RAG, a repair framework for black-box RAG systems with lightweight diagnostics and adaptive patch decisions. Experiments suggest that reliability improves most when repair is targeted to specific failure signatures. Limitations include: (1) diagnostic signal imperfections, since KG alignment and NLI models can struggle with noisy evidence and multi-hop reasoning, cascading errors into suboptimal repairs; and (2) the policy is constrained to exposed repair actions, limiting its capability or overusing costly patches when cheaper alternatives are unavailable.

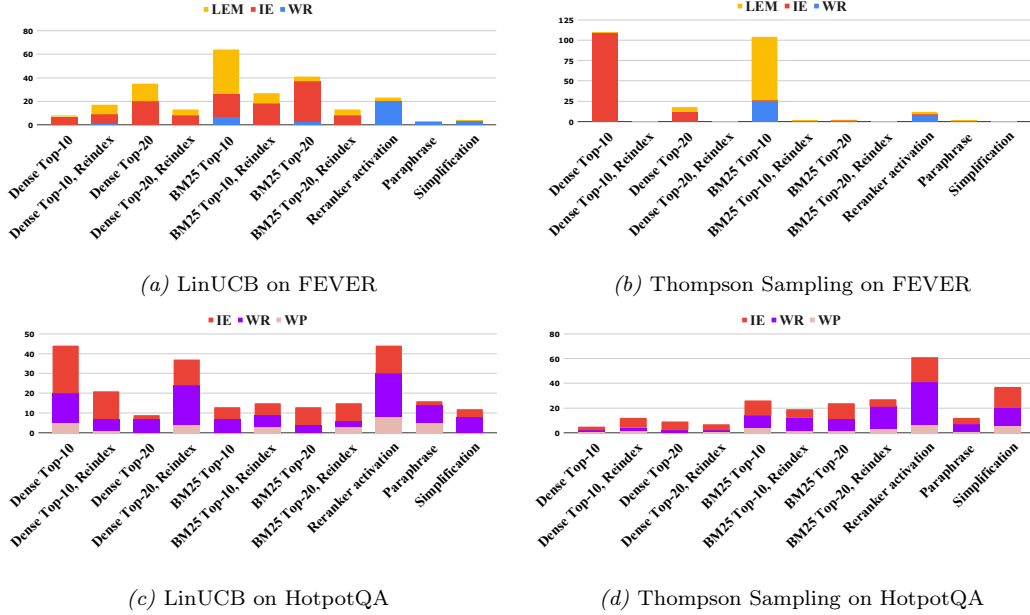


Figure 2. Analysis of failure type frequency exposed by D2R-RAG.

Dataset	Variant	Latency	VRAM	Relevance	Faithfulness	ACC	EM
FEVER	Unconstrained	1.40	<u>0.27</u>	50.91	<u>92.61</u>	<u>61.3</u>	-
	Unweighted	<u>1.46</u>	0.19	50.64	92.62	61.4	-
	D2R-RAG	1.47	0.36	<u>50.80</u>	92.34	60.8	-
HotpotQA	Unconstrained	2.03	<u>0.84</u>	30.36	69.46	-	39.0
	Unweighted	<u>2.79</u>	0.68	32.20	70.59	-	<u>39.4</u>
	D2R-RAG	3.41	1.04	<u>31.66</u>	<u>69.89</u>	-	39.8

Table 2. Overall performance comparison of reward variants.

Dataset	Variant	Latency	VRAM	Relevance	Faithfulness	ACC	EM
FEVER	Stringent	2.22	0.24	51.01	92.80	<u>60.6</u>	-
	Relaxed	1.42	<u>0.32</u>	50.63	<u>92.76</u>	<u>60.6</u>	-
	D2R-RAG	<u>1.47</u>	0.36	<u>50.80</u>	92.34	60.8	-
HotpotQA	Stringent	2.23	0.66	29.99	<u>71.46</u>	-	39.2
	Relaxed	<u>2.31</u>	<u>0.72</u>	<u>30.81</u>	71.57	-	<u>39.4</u>
	D2R-RAG	3.41	1.04	31.66	69.89	-	39.8

Table 3. Overall performance comparison across budget variants.

GenAI Usage Disclosure

OpenAI’s ChatGPT was used to refine sentence clarity and grammatical correctness during manuscript preparation.

Acknowledgments

This work was supported in part by the NSERC-CSE Research Community Grants (ALLRP 598786-24), NSERC Canada Research Chair Grant (CRC-2024-00017), and the

National Cybersecurity Consortium (2025-1601) projects. Researchers funded through the NSERC-CSE Research Communities Grants do not represent the Communications Security Establishment Canada or the Government of Canada. Any research, opinions or positions they produce as part of this initiative do not represent the official views of the Government of Canada.

References

- [1] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al. “Retrieval-augmented generation for knowledge-intensive nlp tasks”. In: *Advances in neural information processing systems* 33 (2020), pp. 9459–9474.
- [2] J. Xie, K. Zhang, J. Chen, R. Lou, and Y. Su. “Adaptive Chameleon or Stubborn Sloth: Revealing the Behavior of Large Language Models in Knowledge Conflicts”. In: *The Twelfth International Conference on Learning Representations*. 2024.
- [3] S. Ray, R. Pan, Z. Gu, K. Du, S. Feng, G. Ananthanarayanan, R. Netravali, and J. Jiang. “Metis: fast quality-aware rag systems with configuration adaptation”. In: *Proceedings of the ACM SIGOPS 31st symposium on operating systems principles*. 2025, pp. 606–622.
- [4] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi. “Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection”. In: *The Twelfth International Conference on Learning Representations*. 2024.
- [5] W. Su, Y. Tang, Q. Ai, Z. Wu, and Y. Liu. “DRAGIN: Dynamic Retrieval Augmented Generation based on the Real-time Information Needs of Large Language Models”. In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2024, pp. 12991–13013.
- [6] X. Tang, Q. Gao, J. Li, N. Du, Q. Li, and S. Xie. “Mba-rag: a bandit approach for adaptive retrieval-augmented generation through question complexity”. In: *Proceedings of the 31st International Conference on Computational Linguistics*. 2025, pp. 3248–3254.
- [7] N. Cho, W. Watson, A. Koppel, S. Ganesh, and M. Veloso. “QueryBandits for Hallucination Mitigation: Exploiting Semantic Features for No-Regret Rewriting”. In: *arXiv preprint arXiv:2508.16697* (2025).
- [8] J. Langford and T. Zhang. “The epoch-greedy algorithm for multi-armed bandits with side information”. In: *Advances in neural information processing systems* 20 (2007).
- [9] L. Li, W. Chu, J. Langford, and R. E. Schapire. “A contextual-bandit approach to personalized news article recommendation”. In: *Proceedings of the 19th international conference on World wide web*. 2010, pp. 661–670.
- [10] J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal. “FEVER: a Large-scale Dataset for Fact Extraction and VERification”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*. Association for Computational Linguistics, 2018, pp. 809–819.
- [11] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. Cohen, R. Salakhutdinov, and C. D. Manning. “HotpotQA: A dataset for diverse, explainable multi-hop question answering”. In: *Proceedings of the 2018 conference on empirical methods in NLP*. 2018, pp. 2369–2380.
- [12] N. Badathala, A. Saxena, and P. Bhattacharyya. “NLI to the Rescue: Mapping Entailment Classes to Hallucination Categories in Abstractive Summarization”. In: *Proceedings of the 20th International Conference on Natural Language Processing (ICON)*. NLP Association of India (NLP AI), 2023, pp. 120–132.
- [13] Y. Deng, W. Zhang, Z. Chen, and Q. Gu. “Rephrase and respond: Let large language models ask better questions for themselves”. In: *arXiv preprint arXiv:2311.04205* (2023).
- [14] W. R. Thompson. “On the Likelihood that One Unknown Probability Exceeds Another in View of the Evidence of Two Samples”. In: *Biometrika* 25.3/4 (1933), pp. 285–294. ISSN: 00063444. (Visited on 02/17/2026).

Appendix A. Diagnostic Signals

We derive failure signatures from three complementary diagnostic signals. The first and second are semantic support via textual entailment: an NLI model evaluates whether retrieved passages address the query and whether the response is supported by evidence. Concretely, each passage in E is treated as a premise and the query (or response) as a hypothesis; passage-level predictions are aggregated into two coarse entailment states e^q and e^r . Intuitively, e^q captures whether retrieved evidence is relevant and sufficient for the information need, while e^r captures whether the generated output remains faithful to the evidence. The third source is structured consistency derived from relational triples. Entailment signals can miss fine-grained relational errors, e.g., response mentions the correct entities but asserts an incorrect predicate or swaps roles. To capture such errors, we extract relations from the response and check whether the stated relations align with the underlying knowledge source. In practice, this signal is most useful for distinguishing “topically plausible but relationally wrong” answers from answers that are merely missing evidence. Algorithm 1 summarizes the rule-based diagnosis.

Appendix B. Context Representation and Training Process

To form the context vector \mathbf{x} used by the repair policy, in our implementation, we concatenate: (1) a semantic representation of the query, (2) the diagnosed failure type, (3) the entailment and triple-alignment statuses, and (4) normalized remaining latency and VRAM budgets. This representation supports cost-sensitive choices: evidence-insufficiency signatures favor retrieval-side escalation, while evidence-present but unsupported-response signatures prioritize higher-precision evidence selection (e.g., reranking) or query rewrites. Algorithm 2 summarizes training procedure. The process is one-shot: for each diagnosed failure, the policy selects a single action and performs exactly one additional RAG pass.

Algorithm 1 FAILURE DIAGNOSIS

Require: query q , evidence set E , response r , predicted task label y
Ensure: failure type $f \in \{\text{NoFail}, \text{WP}, \text{IE}, \text{WR}, \text{LEM}\}$

- 1: $\kappa \leftarrow \text{TRIPLEALIGNSTATUS}(r)$
- 2: $e^q \leftarrow \text{AGGREGATENLI}(E, q)$
- 3: $e^r \leftarrow \text{AGGREGATENLI}(E, r)$
- 4: **if** $\kappa = \text{conflict}$ **then**
- 5: **return** WP
- 6: **end if**
- 7: **if** $e^q = \text{neutral}$ **then**
- 8: **return** IE
- 9: **end if**
- 10: **if** $e^r \neq \text{entail}$ **then**
- 11: **return** WR
- 12: **end if**
- 13: **if** y is provided **and** $\text{LABELCOMPATIBLE}(y, e^q) = \text{false}$ **then**
- 14: **return** LEM
- 15: **end if**
- 16: **return** NoFailure

Algorithm 2 Bandit Policy Learning

Require: dataset \mathcal{D} , default RAG configuration P
Ensure: learned bandit parameters $\{\mathbf{A}_a, \mathbf{b}_a\}_{a \in \mathcal{A}}$ (equivalently $\{\theta_a\}_{a \in \mathcal{A}}$)

- 1: Initialize $\mathbf{A}_a \leftarrow \mathbf{I}$, $\mathbf{b}_a \leftarrow \mathbf{0}$ for all $a \in \mathcal{A}$
- 2: **for** each epoch **do**
- 3: **for** each query $q \in \mathcal{D}$ **do**
- 4: $(r, y, E) \leftarrow \text{RAG}(q, P)$
- 5: $f \leftarrow \text{DETECTFAILURE}(q, E, r, y)$
- 6: **if** $f \neq \text{NoFailure}$ **then**
- 7: $\mathbf{x} \leftarrow \text{BUILDCONTEXT}(q, f, E, r)$
- 8: $a \leftarrow \text{SELECTACTION}(\mathbf{x}, \{\mathbf{A}_a, \mathbf{b}_a\})$
- 9: $P' \leftarrow \text{APPLYPATCH}(P, a)$
- 10: $(r', y', E') \leftarrow \text{RAG}(q, P')$
- 11: $f' \leftarrow \text{DETECTFAILURE}(q, E', r', y')$
- 12: $r(a) \leftarrow \text{COMPUTEREWARD}(f', L_a, V_a)$
- 13: $\text{UPDATELINUCB}(a, \mathbf{x}, r(a))$
- 14: **end if**
- 15: **end for**
- 16: **end for**
- 17: **return** $\{\mathbf{A}_a, \mathbf{b}_a\}_{a \in \mathcal{A}}$

Appendix C. Diagnostic Analysis and Budget Sensitivity

Regarding factual quality, Table 4 shows that correct predictions concentrate in entailment-supportive regimes: on HotpotQA, both policies yield 170+ correct answers under response-entailment versus far fewer under contradiction, while on FEVER, most correct predictions fall under entailed response with contradiction states remaining rare and mostly incorrect. Frequent missing KG alignment indicates triple extraction falls short for many queries, yet policies still succeed when entailment signals provide adequate support.

Diagnostic	Status	HotpotQA		FEVER	
		LinUCB	T.Sampling	LinUCB	T.Sampling
KG Alignment	Consistent	80 (65)	73 (60)	112 (81)	111 (75)
	Conflict	9 (16)	8 (19)	0 (0)	0 (0)
	Missing	110 (220)	121 (219)	494 (309)	502 (308)
	No Triplet	0 (0)	0 (0)	2 (2)	2 (2)
Query Entailment	Entail	115 (148)	112 (155)	350 (203)	351 (195)
	Contradict	59 (107)	57 (95)	258 (139)	264 (140)
	Neutral	25 (46)	33 (48)	0 (50)	0 (50)
Resp. Entailment	Entail	173 (229)	174 (230)	600 (377)	611 (375)
	Contradict	7 (30)	6 (22)	8 (10)	4 (9)
	Neutral	19 (42)	22 (46)	0 (5)	0 (1)

Table 4. Diagnostic signal counts. Each cell reports the # correct (incorrect) instances, where correctness means EM=1 on HotpotQA and label accuracy (ACC)=1 on FEVER.

Further budget sensitivity analysis (Figure 3) shows that under tight budgets, the policy relies more heavily on low-cost retrieval adjustments, especially on HotpotQA.

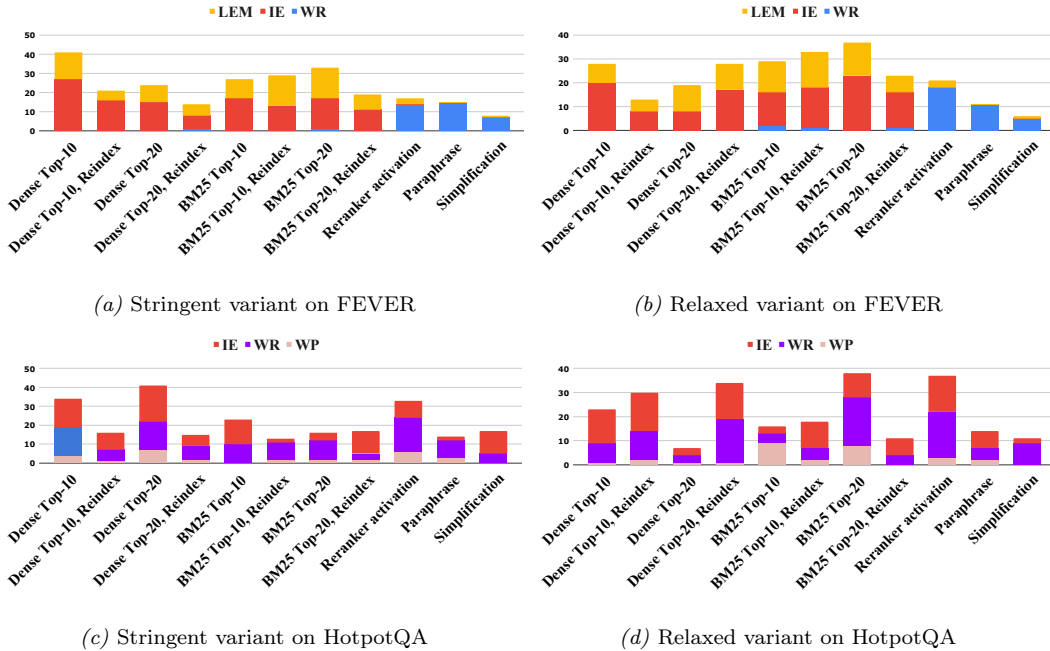


Figure 3. Analysis of failure type frequency under stringent and relaxed budgets.