Final Report

Caitlyn Yin, Zach Hendon

1 Introduction

In recent years, Retrieval-Augmented Generation (RAG) has emerged as a critical technique for enhancing the capabilities of Large Language Models (LLMs). By providing external context, we can improve accuracy and reduce hallucinations. However, traditional RAG systems typically rely on static similarity metrics (e.g., kNN) to select documents independently, ignoring the utility of the documents for the specific reasoning task.

To address this, recent approaches model retrieval as a sequential decision-making process, employing Reinforcement Learning (RL) to learn a robust retrieval policy. In this work, we implement a comprehensive RL-based retrieval framework for mathematical reasoning (GSM8K). We aim to answer two critical questions that remain underexplored:

1. **Architecture:** How does the choice of policy architecture (Sequential vs. Independent) interact with the underlying generator's size (0.5B vs 1.5B)?
2. **Reward Signals:** Can dense, engineered rewards (e.g., LLM-as-a-judge) outperform simple sparse rewards in guiding the retrieval policy?

We demonstrate that while RL consistently outperforms static baselines by significant margins (+13-15%), the key to performance lies in the *policy architecture* rather than complex reward shaping. Specifically, we find that larger models favor independent retrieval, and that Judge rewards can inadvertently degrade performance compared to sparse signals.

2 Literature Review

Recent work has highlighted a preference gap between retrievers optimized for relevance and generators optimized for final output quality; Jiang et al. (2024) proposed a bridge model to filter and re-rank the retriever's selections to improve alignment with the generator. A paper by Yu et al. (2025) further explores the connection between the retriever and the generator. They observed a gap between a document's relevance for the retriever and its utility for the generator. This analysis provides a strong theoretical motivation for our work.

2.1 RAG

Most document retrieval systems are based on vector indexing. Each document corresponds with a vector and scores are generated by computing similarity scores between vectors (Salton et al., 1975). A more traditional non-vector baseline is sparse retrieval, named BM25, which uses term-frequency statistics for ranking (Robertson and Zaragoza, 2009). While BM25 is a strong baseline, it likes Salton's model, operates independently on documents.

Modern dense retrieval was popularized by Dense Passage Retrieval (DPR), which trains dual-encoders (one for the query, one for the documents) using a contrastive learning objective (Karpukhin et al., 2020). DPR is similar to our project as it provides the foundation for most modern RAG systems, which we aim to improve. However, DPR is a one-shot retriever that fetches the top-k documents based on dot-product similarity, ignoring inter-document relationships. Our sequential RL agent directly addresses this by modeling dependencies. Evaluating these various retrievers is a complex task in itself, often requiring great benchmarks like BEIR to measure performance across diverse tasks (Thakur et al., 2021).

A core weakness of top-k retrieval is content redundancy. Other fields have addressed this, for instance through the use of k-Determinantal Point Processes (k-DPP) to sample a diverse subset of items in recommendation systems (Wang, 2024). This is similar to our goal of improving document set diversity. Our project differs in that k-DPP is a one-shot sampling method based on an explicit diversity metric, whereas our sequential RL agent will learn an implicit policy for diversity by observing how redundant documents fail to improve the step-wise or final reward.

2.2 Basic retrieval methods

Several works have applied RL to selection tasks. Shao et al. (2022) used policy gradient RL to select in-context learning (ICL) examples for math problems. While this demonstrates using RL for retrieval, it treats each example selection as an independent action. Our project, however, selection as a sequential MDP, where the state explicitly includes previously selected documents to capture dependencies.

The field of recommendation systems has also explored RL for list generation, such as generating a list of recommendations by modeling user preferences (Zheng et al., 2018). This is similar to our goal of generating a document list.

A blog post by Samanta (2024) describes an RL-based scoring system for RAG that includes a diversity score in the reward. Gao et al. (2024) introduced UPR, which uses RL to train a retriever for free using only a frozen LLM as a reward signal. This is very similar to our goal of using an LLM's final output as a reward. Our work will be novel by focusing on stepwise rewards for credit assignment, rather than just a final reward, and by exploring a dynamic number of retrieved documents.

Another approach proposed by Chen et al. (2024) is to treat each document as an agent and model the retrieval problem as a multi-agent cooperative task.

The idea of sequential RL for selection was explored by Liu et al. (2022) for ICL. Sun et al. (2023) built directly on this, implementing a sequential MDP for ICL that selects a fixed number (T) of documents. This provides a strong baseline for our MDP design. A key challenge in this sequential model is the expanding state, which includes all previously selected documents. Efficiently encoding this history is crucial. Architectures like the Longformer, which scales transformers to long sequences, offer a viable mechanism for encoding this growing state without quadratic complexity (Beltagy et al., 2020). Our project will use such architectures to handle a large set of retrieved documents. Our novelty will be in applying this to RAG, including dense intermediate rewards, and investigating adaptive stopping criteria.

2.3 RL-based retrieval

Several works have applied RL to selection tasks. Shao et al. (2022) used policy gradient RL to select in-context learning (ICL) examples for math problems. While this demonstrates using RL for retrieval, it treats each example selection as an independent action.

A blog post by Samanta (2024) describes an RL-based scoring system for RAG that includes a diversity score in the reward. Another approach to this problem proposed by Chen et al. (2024) is to treat each document as an agent and model the retrieval problem as a multi-agent cooperative task. Doing so enables efficient optimization for metrics such as diversity. Both of these are similar to our goal of improving RAG with RL, but we will model this problem as a sequential decision-making problem instead.

The idea of sequential RL for selection was explored by Liu et al. (2022) for active example selection in in-context learning (ICL). This is a very closely related concept. Our work adapts this sequential framework from the specific task of ICL example selection to the broader RAG problem, which involves retrieving general informational documents rather than just formatted prompt-completion pairs. Sun et al. (2023) built directly on this, implementing a sequential MDP for ICL that selects a fixed number (T) of documents. This provides a strong baseline for our MDP design. Our project will be novel in multiple ways: first, by applying this to general RAG, second, by including dense intermediate rewards after each selection, and lastly, by investigating dynamic stopping criteria rather than always selecting a fixed T documents.

3 Methodology

3.1 Problem Formulation (MDP)

We formulate RAG as a sequential decision-making problem. Given a query q and a candidate pool $\mathcal{D}$, the agent selects a sequence of documents $S_t = \{q, d_1, \ldots, d_t\}$. The goal is to maximize the probability of the frozen LLM generating the correct answer. The process terminates after k steps.

3.2 Policy Architecture

We investigate two distinct policy architectures:

1. **Sequential (LSTM):** Uses an LSTM to encode the history of selections, capturing dependencies between documents (e.g., "if I selected a geometry example, I should next select an algebra one").

2. **Independent:** An ablation variant where the policy conditions only on the query and the current candidate, treating retrieval steps as independent events.

3.3 RL Algorithms

To optimize the policy, we implement and compare:

- **REINFORCE with Baseline (RWB):** A foundational Monte Carlo policy gradient method.
- **Actor-Critic (AC):** Uses a learned Value Function $V(s)$ to reduce variance.
- **PPO (Proximal Policy Optimization):** Improves stability via clipped surrogate objectives.

3.4 Advanced Reward Shaping

Standard RL relies on **Sparse Rewards** ($R = 1$ if correct at the end, else 0). To potentially improve learning, we designed two dense reward mechanisms:

- **Intermediate Marginal Rewards:** Provides feedback at every step t by calculating the incremental gain: $r_t = R(S_t) - R(S_{t-1})$.
- **LLM-as-a-judge:** Uses a separate LLM to score the generator's reasoning on a scale of [0, 1]. The total reward is a weighted sum: $R_{total} = \alpha R_{judge} + (1 - \alpha) R_{correct}$.

4 Empirical Analysis

4.1 Experimental Setup

We evaluated our methods on GSM8K using the **Qwen-2.5** family (0.5B and 1.5B). We compare against **Random** and **kNN** baselines.

4.2 Main Results: RL vs. Baselines

Our primary finding is that RL-based retrieval provides a decisive advantage over static methods. As shown in Figure 1, the Actor-Critic (LSTM) method achieved 77.5%, representing a +15.5% gain over the random baseline.

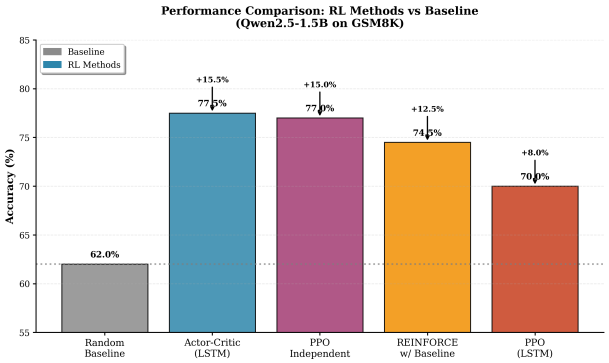

Figure 1: Performance comparison on GSM8K (1.5B Model). RL methods consistently outperform the random baseline, with Actor-Critic achieving the highest accuracy of 77.5%.

4.3 Analysis 1: Architecture & Model Scale

We uncovered a critical interaction between model size and policy architecture. While small models benefited from sequential modeling, larger models favored independent selection.

Figure 2 illustrates this for the 1.5B model, where the Independent architecture (77.0%) outperformed the mean performance of Sequential methods ($\sim$ 74%). Larger models appear robust to context ordering, allowing the Independent policy to benefit from simpler optimization landscapes.

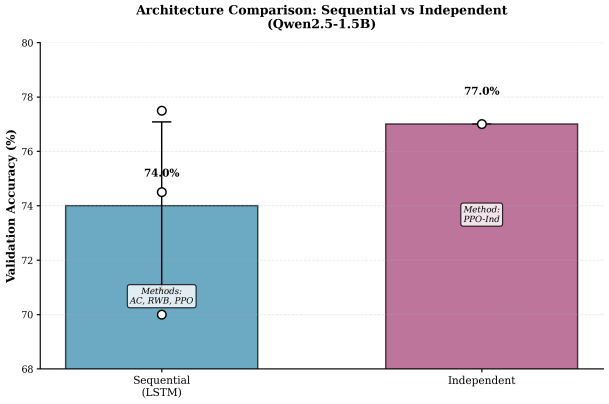

Figure 2: Architecture comparison for the 1.5B model. The Independent architecture (PPO-Ind) outperforms the Sequential (LSTM) variants, suggesting that larger generators do not require strict ordering dependencies in retrieval.

4.4 Analysis 2: Investigation of Reward Shaping

We conducted a focused ablation study to evaluate the efficacy of the "Advanced Reward Shaping" techniques described in Section 3.4. The results are summarized in Figure 3.

Reward Performance Trend:
Sparse Binary > Marginal Correctness > LLM-as-a-Judge

Figure 3: Impact of reward shaping on performance. Contrary to expectations, complex Judge rewards underperformed simple sparse rewards.

Interestingly, our experiments yielded a **negative result** for complex reward engineering. As the mixing ratio α of the Judge reward increased, performance steadily **decreased**. We hypothesize that the "Judge" LLM (Qwen-1.5B) was not robust enough to provide accurate reasoning scores, introducing noise that confused the policy.

Takeaway: For this task, a strong algorithm combined with a simple Sparse Reward is superior to complex reward shaping.

5 Discussion

5.1 Why RL Beats Similarity

Standard similarity (kNN) assumes that problems sharing keywords require similar reasoning. However, our RL agents learned to select examples based on *utility*. For instance, the agent might learn that a specific algebra example is a "universal key" that helps solve various word problems, a relationship kNN would never capture.

5.2 Engineering Challenges

Scaling to 1.5B parameters required significant optimization. We utilized vLLM for inference and implemented a prompt cache, which reduced training time by 90% by avoiding redundant forward passes for identical prompt prefixes.

6 Conclusion

In this work, we framed the RAG retrieval problem as a sequential decision making process and proposed intermediate marginal rewards and using an LLM-based rewards. We showed that having intermediate rewards improves performance, but that the judge-based rewards worsened performance, at least for smaller models.

We demonstrated that RL-based retrieval improves mathematical reasoning by +15% over static baselines. Our architectural analysis provided a key insight: **optimal retrieval strategies are scale-dependent**. Smaller models need sequential guidance (LSTM), while larger models thrive with efficient, independent selection.

References

Beltagy, I., Peters, M. E., and Cohan, A. (2020). Longformer: The long-document transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6728–6741.

Chen, Y., Mao, J., Zhang, Y., Ma, D., Xia, L., Fan, J., Shi, D., Cheng, Z., Gu, S., and Yin, D. (2024). Ma4div: Multi-agent reinforcement learning for search result diversification.

Gao, Y., Li, S., Huang, J., Zhang, J., Zhang, C., Ren, P., Su, Y., and Ren, Z. (2024). Retrieval-augmented generation with unsupervised p-value-based reward. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Jiang, Z., Yu, W., Liu, Z., Liu, S., Wu, Z., and Dou, Z. (2024). Bridging the preference gap between retrievers and llms. *arXiv preprint arXiv:2401.06954*.

Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., and Yih, W.-t. (2020). Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6780.

Liu, Y., Wu, C.-Z., Lin, J., and Yao, Y. (2022). Active example selection for in-context learning. *arXiv preprint arXiv:2211.04486*.

Robertson, S. and Zaragoza, H. (2009). The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Salton, G., Wong, A., and Yang, C. S. (1975). A vector space model for automatic indexing.

Samanta, S. (2024). Building a self-improving rag system with reinforcement learning. Medium.

Shao, Y., Wang, S., Li, Z., Liu, Z., and Liu, Z. (2022). Promptpg: Unsupervised generation of task-specific prompts with policy gradient. *arXiv preprint arXiv:2209.14610*.

Sun, R.-Z., Du, J.-J., Yan, Z.-F., Liu, T.-S., and Wei, Z.-M. (2023). Reticl: A sequential mdp framework for in-context learning. *arXiv preprint arXiv:2305.14502*.

Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., and Gurevych, I. (2021). Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 10023–10046.

Wang, Y., e. a. (2024). k-ddp - uses k-ddp for sampling the top-k items from a subset while accounting for diversity. `https://arxiv.org/pdf/2406.15983v1`. arXiv preprint arXiv:2406.15983.

Yu, W., Jiang, Z., Liu, Z., Wu, Z., Liu, S., and Dou, Z. (2025). Is relevance propagated from retriever to generator in rag? `https://arxiv.org/pdf/2502.15025`. arXiv preprint arXiv:2502.15025.

Zheng, G., Zhang, F., Zheng, Z., Xiang, Y., Yuan, N. J., Xie, X., and Sun, E. (2018). Drn: A deep reinforcement learning framework for news recommendation. In *Proceedings of the 2018 World Wide Web Conference*, pages 167–176.