# Extended Abstract

**Motivation** Retrieval-Augmented Generation (RAG) has emerged as a powerful strategy for grounding large language models (LLMs) in factual documents by fetching relevant passages as context, improving accuracy in information-intensive tasks. However, its reliance on a static retriever trained only for generic relevance creates a critical mismatch: the retriever is never optimized for the downstream goal of producing accurate answers. As a result, even semantically similar passages may fail to support correct responses, and there is no mechanism for the retriever to learn from feedback on answer quality. To address this bottleneck, we model retrieval as a decision problem and optimize it using reinforcement learning (RL). Using a two-phase pipeline, our goal is to use downstream LLM answer quality to train the retriever, maximizing end-to-end question answering (QA) performance.

**Method** We first mitigate cold-start by supervised fine-tuning a dense retriever (BGE) on gold-label passages. For each minibatch, the retriever encodes queries and candidate passages, forms a similarity matrix, and applies a per-query softmax. We train with negative log-likelihood of the one-hot gold labels, using only the batch's positive passages as the search corpus, so the corpus size remains tractable. We then transition to on-policy reinforcement learning by casting passage retrieval as a sequential decision-making problem. Using Proximal Policy Optimization (PPO), we optimize the retriever to select passages that maximize semantic coherence between LLM-generated and reference answers, using SBERT-based cosine similarity as the reward. A multi-head attention critic and GAE-based advantage estimator provide structured value feedback for stable end-to-end fine-tuning.

**Implementation** We evaluate our approach both retrieval and end-to-end QA tasks. For retrieval warm-up and initial comparison, we use the MSMARCO passage ranking dataset, where each query is paired with one or more ground truth gold passages, to train both versions of the behavior cloning. We then fine-tune via PPO on the QA set of MSMARCO, using the MSMARCO corpus. The MSMARCO QA is in the free response style, so we used the SBERT Cosine Similarity to "grade" the quality of the response from our LLM and generate rewards. We compare against static retrieval baselines including the base fine-tuned dense retriever BGE and previous paper's implementations of BGE using the normalized Discounted Cumulative Gain (nDCG) metric, measuring how well all relevant items are at the top of the retrieval list.

**Results** For behavior cloning, SFT collapsed: MSE on cosine scores drove all similarities toward zero, and even with soft-max + regularization, retrieval lagged baseline. The nDCG@3 score doesn't improve from the original model. PPO training likewise failed to yield substantive policy improvement. The mean SBERT reward remained flat throughout training, the actor loss hovered near zero, indicating minimal policy updates, and the critic loss fluctuated without converging, reflecting unstable value estimation.

**Discussion** For behavior cloning, SFT stalled because our simple pipeline still drove cosine-similarities toward zero despite the softmax fix, while PPO faltered because the small in-batch corpus produced near-saturated rewards and MS MARCO's label noise corrupted the learning signal, together yielding gradients too weak to update the retriever. For RL updates, PPO stagnated: rewards plateaued, actor loss stayed flat, and the critic under-performed. Likely reasons are (i) disrupted gradients through the SentenceTransformer, (ii) a non-convex landscape plus a weak SBERT reward that traps the policy, and (iii) a low-capacity, high-variance critic that yields unreliable value estimates.

**Conclusion** We have presented a unified two-phase training paradigm that first employs behavior cloning to warm up a dense retriever on MSMARCO gold passages, and then performs on-policy PPO fine-tuning against actual QA rewards. This approach combines the stability of supervised learning with the adaptability of reinforcement learning to give feedback on downstream answer quality, demonstrating a framework to directly improve downstream question-answering accuracy of RAG systems.

# Reinforcement Learning for Retrieval Optimization in RAG Systems

**Ryan Tan**
Stanford University
tanryan@stanford.edu

**Jeffrey Xue**
Stanford University
jjxue1@stanford.edu

**Richard Gu**
Stanford University
yrichard@stanford.edu

## Abstract

Retrieval-Augmented Generation (RAG) enhances the factuality of large language models (LLMs) by supplying them with relevant contextual passages from a corpus. However, static retrievers used in RAG systems are typically trained on generic relevance signals rather than optimized for downstream answer quality, introducing a training–evaluation mismatch. We address this limitation by formulating dense retrieval as a sequential decision-making task and propose a two-phase training framework that combines behavior cloning and reinforcement learning. In Phase 1, we employ behavior cloning (BC) to warm up the retriever policy, using datasets with passages labeled as relevant (gold passages) demonstrating expert decisions to train the retriever to imitate these choices. In Phase 2, we fine-tune the retriever with Proximal Policy Optimization (PPO), using a Sentence-BERT-based semantic similarity between generated and reference answers as the reward. Despite this principled design, both training phases failed to yield meaningful improvements. Supervised training suffered from vanishing similarity scores, while PPO optimization stagnated due to saturated in-batch rewards, unstable gradient flow through transformer encoders, and unreliable value estimation. Our results underscore the complexity of applying reinforcement learning in retrieval-augmented QA and highlight key challenges in optimizing retrievers for end-to-end task performance.

## 1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in generating natural text across a wide range of domains. However, when applied to specific and knowledge-intensive tasks, these models often fail to recall precise information or sometimes even hallucinate facts. Some strategies have been known to mitigate these effects, specifically Retrieval-Augmented Generation (RAG), where an external retriever fetches relevant passages from a corpus used as context for the LLM which is used as a generator. This aims to ground the generation process in retrieved truths, as RAG systems improve factual accuracy, especially for knowledge-intensive tasks, as the model can reason based on explicit documents. In a typical RAG workflow, shown in Figure 1, a user's query and a large corpus are first embedded and fed into a retriever. This results in retrieved passages which are combined with the original query and fed into the LLM, with the retrieved passages acting as context. This then generates a response.

The static retriever in RAG systems creates a critical weakness: it is not optimized for downstream tasks. Therefore, the retriever might not be able to effectively retrieve passages aligned to the need of downstream LLM to produce the most accurate, factual, or coherent answer to a query. In other words, the retriever is trained based on generic relevance metrics, but the goal is evaluated on end-to-end

Stanford CS224R 2025 Final Report

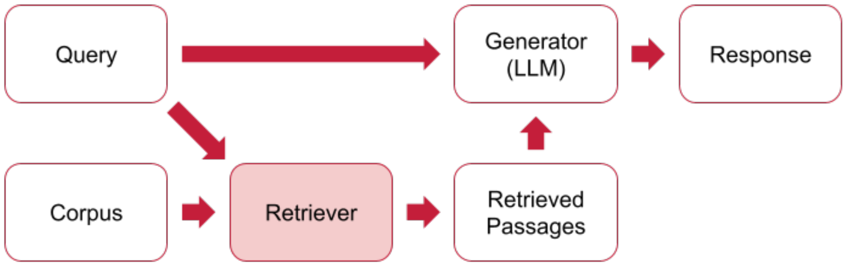


Figure 1: The typical RAG pipeline is shown above: a query and corpus are encoded and fed into a retriever. The relevant passages are retrieved and fed into into a LLM generator as context with the query. The final outputted response is then collected. Our project focuses on just training retriever part of the pipeline, highlighted in red.

answer quality. There is no mechanism in RAG for the retriever to learn directly from feedback on answer correctness.

Our work addresses these limitations by treating retrieval as a decision-making problem in a reinforcement learning (RL) framework. We improve the quality of open-domain question answering (QA) by optimizing the retrieval component of RAG systems using RL to maximize end-to-end QA performance rather than isolated retrieval metrics. To do so, we propose a two-phase training pipeline. In Phase 1, we employ behavior cloning (BC) to warm up the retriever policy, using datasets with passages labeled as relevant (gold passages) demonstrating expert decisions to train the retriever to imitate these choices. By framing imitation learning in this way, we aim to achieve a strong initialization that avoids the cold-start issues typical of pure RL. In Phase 2, we switch evaluating the end-to-end performance of the entire RAG pipeline, training using Proximal Policy Optimization (PPO), freezing the LLM and retriever encoders and treating the retriever policy as an agent whose reward is derived from the factual correctness of answers generated downstream. This on-policy fine-tuning enables the retriever to adapt dynamically based on end-to-end performance, closing the feedback loop between retrieval and generation.

## 2 Related Work

By introducing an external retriever that selects relevant passages from a corpus, RAG allows a frozen generator to produce outputs grounded in retrieved evidence. However, a key limitation remains: the retriever is typically trained independently from the generator, optimizing proxy objectives such as semantic similarity or binary relevance, rather than directly improving downstream answer quality. This disconnect between training supervision and evaluation metrics, commonly referred to as the training–evaluation gap, motivates a growing body of work on more tightly aligned retriever optimization.

One of the earliest and most influential dense retrieval methods is Dense Passage Retrieval (DPR) Karpukhin et al. (2020), which uses a dual-encoder architecture trained via contrastive loss to distinguish relevant from non-relevant passages. While DPR provides strong improvements in retrieval recall compared to sparse methods such as BM25, it is trained solely on semantic similarity and lacks exposure to downstream task performance. As a result, retrieved passages may be topically relevant but suboptimal for grounding accurate or coherent answers.

REALM Guu et al. (2020) introduced an end-to-end framework that jointly optimizes retrieval and generation. It formulates retrieval as a latent variable and backpropagates through retrieval by maximizing the likelihood of producing correct answers. Although this approach significantly improves alignment between retriever training and downstream QA performance, it imposes heavy computational demands, requiring repeated corpus re-encoding and joint gradient updates across both modules. These computational costs make end-to-end models like REALM difficult to scale to corpora with millions of documents or to deploy in production environments.

In contrast, multi-stage architectures decouple retrieval into modular components. PG-Rank Gao et al. (2024) exemplifies this paradigm by appending a neural reranker to a fast first-stage retriever and optimizing it using policy gradient methods. PG-Rank directly optimizes ranking metrics such as nDCG and demonstrates strong performance across QA benchmarks. However, it still inherits a fundamental bottleneck: if relevant passages are missed by the first-stage retriever, the reranker cannot recover them. Moreover, while the reranker is trained with reward signals, the retrieval objective remains detached from end-to-end answer accuracy.

Recent work has also sought to improve the generalization of retrievers across tasks and models. The Augmentation-Adapted Retriever (AAR) Yu et al. (2023) is designed as a plug-in module for a wide range of generation tasks, training on diverse augmented datasets to promote robustness and adaptability. Similarly, LLM-Embedder Zhang et al. (2023) uses frozen large language models to generate retrieval embeddings that are more aligned with generator behavior. While both AAR and LLM-Embedder expand the scope of retriever capabilities, they are still trained on task-agnostic or proxy objectives, and neither optimizes directly for downstream factual correctness.

Finally, reinforcement learning (RL) has been proposed as a means to align retriever training with downstream generation outcomes. Nogueira and Cho Nogueira and Cho (2017) introduced an early formulation of this idea, using REINFORCE to adjust retrieval based on downstream QA success. However, such approaches suffer from severe cold-start issues: during initial training, untrained retrievers fail to retrieve answer-bearing passages, resulting in sparse or uninformative reward signals that hinder learning.

Our work addresses these limitations by proposing a two-phase RL-based retriever training framework. In Phase 1, we use behavior cloning to warm-start the retriever with expert demonstrations, mitigating the cold-start problem. In Phase 2, we freeze both the LLM and the retriever encoder and apply Proximal Policy Optimization (PPO) to fine-tune the retriever using answer-level correctness as the reward signal. This approach enables direct optimization for downstream QA performance while preserving the modularity and scalability of the standard two-stage RAG architecture.

## 3 Method

To enhance our retriever's capability of correct and more LLM-aligned retrieval, we employ two sequential phases to train just the retriever, each leveraging the same data-collection and batch-processing pipeline but differing in their optimization objectives. Both phases use the same framework for handling candidate pools derived from Proximal Policy Optimization (PPO) as follows. The state $s$ is defined as the combination of a query and a batch corpus, which was usually of size 128 given our GPU memory constraints. The action $a$ is defined as a set of $k$ passages and their corresponding probabilities which were to be retrieved by the retriever. These are the passages fed as context to the LLM for generation. The subsequent policy $\pi_\theta(a|s)$ is defined as the retriever that yields retrieval probabilities over the passages as actions. In the training loop, we rollout one batch under the old policy, use multiple mini-batch updates to adjust the current policy, and at the end of all mini-batches set current policy to old policy. This setup accurately models the retriever as a decision maker, which allows us to apply RL to train the behavior of the retriever.

### 3.1 Behavior Cloning Warm-Up with SFT & PPO

In the first phase, we first trained the retriever to pick passages based on ground truth gold passage labels. The objective is to avoid cold start problems by fine-tuning a dense retriever (BGE) to identify gold passages.

To do so, we started with a supervised fine-tuning (SFT) approach. We use our retriever model to produces embeddings for each query and each candidate passage, and then compute the similarity matrix between the queries and the passages. Then we normalize this similarity matrix with softmax on each query and compare the resulting scores with the ground-truth one-hot correlation label from the original dataset. We treat the related passages for all the in-batch queries as the in-batch corpus and therefore control the corpus size to an practical size. Finally, the loss function for each query is the negative log-likelihood of the gold passages, averaged over the batch based on the MSE loss of the comparison.

This behavior cloning was also performed using a more RL style formulation. We use a similar definition of in-batch corpus for PPO. We use the normalized Discounted Cumulative Gain (nDCG) score as the reward, which compares retrieved passages and their respective ranking with gold passages for each query, a metric used by Gao et al. (2024).

### 3.2 Reinforcement Learning with PPO

After the behavior cloning warm-up, we transition to on-policy fine-tuning based on the LLM outputs. Our method formulates dense passage retrieval as a decision-making problem within the Proximal Policy Optimization (PPO) paradigm, enabling end-to-end training that maximizes semantic coherence between generated answers and ground truth responses. This approach fundamentally reconceptualizes retrieval training by optimizing directly for answer quality rather than relying on potentially noisy relevance judgments or click-through data.

#### 3.2.1 Problem Formulation and PPO Framework

We formalize neural information retrieval as a Markov Decision Process where states $s = \{q, \mathcal{P}\}$ encode queries $q$ and candidate passage sets $\mathcal{P} = \{p_1, \ldots, p_k\}$ through dense neural representations. Actions $a \in \{0,1\}^k$ represent binary passage selection decisions, with $a_i = 1$ indicating inclusion of passage $p_i$ in the context provided to downstream language models. The policy $\pi_\theta(a|s)$ is parameterized by a neural retriever with learnable parameters $\theta$.

For computational tractability while maintaining expressiveness, we employ a factorized policy representation:

$$\pi_\theta(a|s) = \prod_{i=1}^{k} \pi_\theta(a_i|s) \tag{1}$$

This factorization assumes conditional independence of passage selections given query and passage representations, a necessary computational approximation that enables scalable optimization while potentially limiting modeling of complex passage interactions.

Our PPO implementation addresses the fundamental challenge of stable policy updates through a clipped surrogate objective that constrains probability ratios between consecutive policy iterations. The complete optimization objective integrates three essential components:

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t\left[\min\left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right] \tag{2}$$

$$L^{\text{VF}}(\theta) = \mathbb{E}_t\left[(V_\theta(s_t) - V_t^{\text{targ}})^2\right] \tag{3}$$

$$L^{\text{ENT}}(\theta) = \mathbb{E}_t\left[H(\pi_\theta(\cdot|s_t))\right] \tag{4}$$

where the importance sampling ratio $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{\text{old}}}(a_t|s_t)$ quantifies policy deviation, $\hat{A}_t$ denotes advantage estimates, and $\epsilon = 0.15$ controls update conservativeness. The combined objective function is:

$$L^{\text{TOTAL}}(\theta) = L^{\text{CLIP}}(\theta) - c_1 L^{\text{VF}}(\theta) + c_2 L^{\text{ENT}}(\theta) \tag{5}$$

with coefficients $c_1 = 0.5$ and $c_2 = 0.01$ balancing actor-critic learning and exploration incentives. The clipping mechanism prevents destructively large policy updates by constraining importance ratios within $[1-\epsilon, 1+\epsilon]$, ensuring training stability while providing theoretical guarantees for monotonic policy improvement.

#### 3.2.2 Multi-Head Attention Critic with Structured Value Estimation

Traditional value function approximators treat state representations as monolithic vectors, failing to capture the inherent structural properties of retrieval states where queries and passages possess distinct semantic roles and complex relational dependencies. We introduce a sophisticated multi-head attention critic that explicitly models query-passage interactions through transformer-inspired attention mechanisms.

Given query embedding $\mathbf{q} \in \mathbb{R}^d$ and passage embeddings $\mathbf{P} \in \mathbb{R}^{k \times d}$, our critic employs multi-head cross-attention where the query serves as the attention query while passages function as both keys and values:

$$\mathbf{Q} = \mathbf{q}W_q \in \mathbb{R}^{1\times d} \tag{6}$$

$$\mathbf{K}, \mathbf{V} = \mathbf{P}W_k, \mathbf{P}W_v \in \mathbb{R}^{k\times d} \tag{7}$$

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V} \tag{8}$$

$$V(s) = \text{MLP}(\text{concat}(\mathbf{q}, \text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}))) \tag{9}$$

where MultiHead employs $h = 8$ attention heads with learned projection matrices $W_q$, $W_k$, and $W_v$. This architecture enables dynamic weighting of passage importance based on query-specific relevance, providing the critic with structured understanding of retrieval states rather than treating them as unstructured feature vectors.

We compute advantage estimates using Generalized Advantage Estimation (GAE) to optimally balance bias and variance in temporal credit assignment:

$$\delta_t^V = r_t + \gamma V(s_{t+1}) - V(s_t) \tag{10}$$

$$\hat{A}_t^{\text{GAE}}(\gamma, \lambda) = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}^V \tag{11}$$

computed recursively as $\hat{A}_t = \delta_t + \gamma\lambda\hat{A}_{t+1}$ with $\lambda = 0.95$ providing empirically effective bias-variance balance. Advantages are normalized to unit variance for training stability.

### 3.2.3 SBERT-Based Semantic Reward Mechanism

Our reward function establishes direct connection between retrieval decisions and ultimate task performance by evaluating passage selection quality through downstream answer generation. Rather than relying on traditional relevance assessments, we measure semantic coherence between generated and ground truth answers using pre-trained Sentence-BERT embeddings:

$$\mathcal{R}(s, a) = \psi(\text{sim}_{\text{SBERT}}(\text{LLM}(q, \mathcal{P}_{\text{selected}}), y^*)) \tag{12}$$

where $\text{LLM}(q, \mathcal{P}_{\text{selected}})$ generates answers from selected passages $\mathcal{P}_{\text{selected}} = \{p_i : a_i = 1\}$, and semantic similarity employs the all-mpnet-base-v2 SBERT model:

$$\text{sim}_{\text{SBERT}}(\hat{y}, y^*) = \frac{\mathbf{E}(\hat{y}) \cdot \mathbf{E}(y^*)}{|\mathbf{E}(\hat{y})||\mathbf{E}(y^*)||} \tag{13}$$

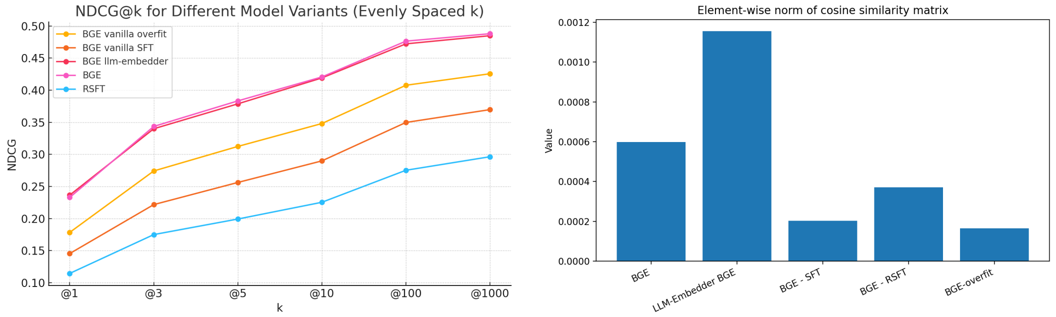
where $\mathbf{E}(\cdot)$ denotes L2-normalized SBERT encoding ensuring similarity scores within the interpretable range $[-1, 1]$.

To prevent exploitation of trivial patterns and encourage meaningful answer generation, we implement sophisticated quality-based penalty mechanisms:

$$\psi(s) = s \times p_{\text{quality}} \times p_{\text{length}} \times p_{\text{content}} \tag{14}$$

where $p_{\text{quality}} = 0.3$ for non-answer responses (e.g., beginning with "question" or "no answer"), $p_{\text{length}} = 0.5$ for responses under two words, and $p_{\text{content}} = 0.7$ for responses primarily repeating the query. These penalties maintain differentiability while discouraging degenerate solutions.

This reward design enables direct optimization for answer quality while circumventing traditional relevance labeling requirements. SBERT provides robust semantic similarity measurement that correlates strongly with human quality judgments, enabling the system to learn sophisticated passage selection strategies that genuinely improve downstream task performance rather than optimizing for potentially misaligned proxy metrics.



(a) nDCG over training epochs for SFT.

(b) Element-wise norm of cosine similarity matrix.

Figure 2: Behavior cloning SFT results.

## 4 Experimental Setup

We evaluate our approach on both retrieval and end-to-end QA tasks. For retrieval warm-up and initial comparison, we use the MSMARCO passage ranking dataset, where each query is paired with one or more ground truth gold passages drawn from a 8.8 M-passage corpus. We then fine-tune via PPO on the QA set of MSMARCO, using the MSMARCO corpus. The MSMARCO QA is in the free response style, but questions tend to be rather open ended, creating long answers which can often be reworded or described differently. Given the MSMARCO QA ground truths only included one answer, we used the SBERT Cosine Similarity to "grade" the quality of the response from our LLM. SBERT provides an in between solution, more complex and understanding of schemantics when compared to linguistic metrics like BLEU and ROUGE, but less compute intensive than LLM judges. We compare against static retrieval baselines including the base fine-tuned dense retriever BGE and previous paper's implementations of BGE (Zhang et al. (2023)). The nDCG metric was used as it compares rankings to an ideal order given by ground truths from the MSMARCO dataset, measuring how well all relevant items are at the top of the retrieval list.

## 5 Results

### 5.1 Behavior Cloning: Supervised Fine-tuning of the retriever

Our SFT training generally failed despite our multiple attempts. The primary obstacle is a pronounced **vanishing-similarity effect**. Because the training objective is the mean-squared error (MSE) between the model's cosine-similarity scores and a one-hot query-passage relevance label, the network can minimize the loss by uniformly suppressing similarity magnitudes, an outcome that is exacerbated by large batch sizes and extremely sparse positive labels.

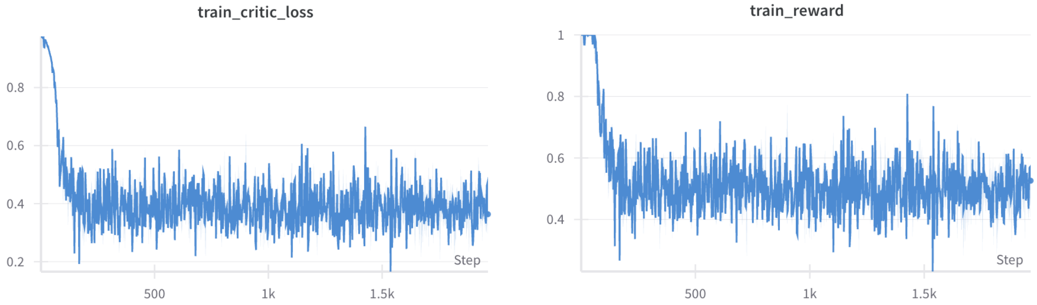
To quantitatively analyze the vanishing similarity effect, we measure the per-element norm of the similarity matrix and found it declining sharply after SFT. Even a deliberately over-fitted model trained on the test set exhibits this collapse, indicating that the vanishing-similarity phenomenon severely degrades the model's capacity to discriminate relevant passages.

To mitigate the vanishing-similarity issue, we introduced a **softmax normalization** over the cosine-similarity scores prior to computing the retrieval probabilities. Because the softmax is scale-invariant, the model can no longer minimize the loss merely by uniformly shrinking the raw similarity values. We further augmented the objective with an additional **regularization** term.

Although these modifications partially alleviate the collapse, as evidenced by an approximate two-fold increase in the average similarity norm, the overall retrieval performance remains sub-optimal.

### 5.2 Behavior Cloning: PPO

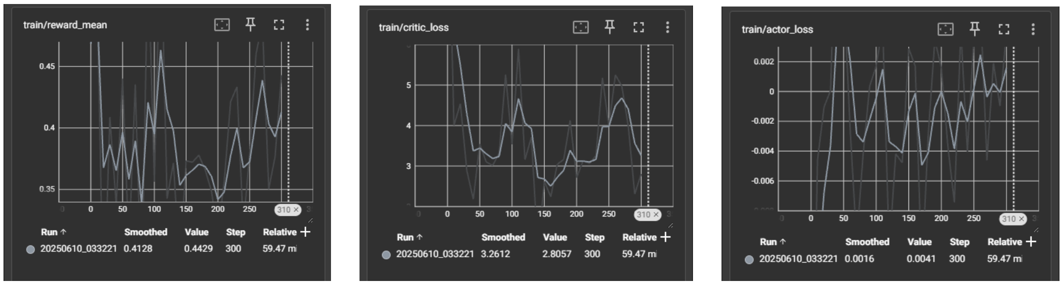
During training, the average reward on the training set steadily declined and eventually plateaued at a sub-optimal level. Although the critic converged successfully, the actor (i.e., the retriever) displayed



(a) Train loss for critic model.

(b) Train reward for actor model. We expect similar trend for evaluation reward.

Figure 3: Behavior cloning RL results.



| Run ↑ | Smoothed | Value | Step | Relative + |
|---|---|---|---|---|
| ● 20250610_033221 | 0.4128 | 0.4429 | 300 | 59.47 mi |

| Run ↑ | Smoothed | Value | Step | Relative + |
|---|---|---|---|---|
| ● 20250610_033221 | 3.2612 | 2.8057 | 300 | 59.47 mi |

| Run ↑ | Smoothed | Value | Step | Relative + |
|---|---|---|---|---|
| ● 20250610_033221 | 0.0016 | 0.0041 | 300 | 59.47 mi |

(a) Mean training reward.

(b) Critic loss across training steps.

(c) Actor loss across training steps.

Figure 4: Reinforcement Learning PPO results.

an increasingly large training loss that paralleled the drop in reward, indicating that the policy updates were degrading retrieval quality rather than improving it.

We suspect that the underlying issue is an excessively dense reward signal. Because the reward is defined as nDCG@3 over an in-batch corpus of 128 passages, its reward is **almost always close to** 1: the retriever nearly always ranks the correct passage among the top three. Such a saturated reward provides little learning signal. Hardware limitations prevent us from using larger batch sizes, and thus from creating a more challenging retrieval task, so the reward distribution remains overly concentrated near its maximum.

### 5.3 Reinforcement Learning: PPO

To evaluate the performance of our reinforcement learning framework for dense passage retrieval, we tracked several key training metrics over 300 optimization steps using the MS MARCO dataset. The principal objective was to maximize semantic similarity between generated answers and reference answers, as measured by SBERT cosine similarity. The results, however, indicate that the model failed to achieve consistent learning or substantial policy improvement during training.

The mean training reward, shown in Figure 4a, oscillates within a relatively narrow band throughout the entire training trajectory. The smoothed average reward plateaus around 0.41, with the final observed value at step 300 being 0.4429. Notably, there is no sustained upward trend, suggesting that the retriever's policy did not improve its ability to select passages that lead to higher-quality answer generation. The observed fluctuations likely reflect high variance in stochastic gradient updates or instability in the value function approximation, but critically, the absence of clear reward escalation implies that the learning signal may have been insufficiently propagated or entirely suppressed.

The critic loss, visualized in Figure 4b, initially decreases from values above 5.0 to approximately 2.8, but exhibits recurrent rebounds and a general lack of monotonic descent. Although some reduction in value estimation error is evident, the overall trend suggests that the critic struggled to stably learn the value of states under the policy. These dynamics are indicative of either poor convergence or noisy target signals, which may stem from high variance in Generalized Advantage Estimation (GAE), unreliable gradient flow from the embedding model, or inadequate representational capacity in the critic.

The actor loss (Figure 4c), reflecting the clipped surrogate PPO objective, hovers around zero for the duration of training. The minimal magnitude of this loss (mean smoothed value $\approx 0.0016$) is consistent with negligible policy shifts between iterations. In PPO, small actor loss values are generally expected when updates are conservative; however, in this context, they more likely reflect the failure of the policy network (the retriever) to respond meaningfully to the advantage signal, again pointing to potential disruptions in gradient flow or ineffective learning dynamics.

Together, these trends reveal that despite architecturally correct implementation of PPO components and reward calculation, the training process did not yield significant policy improvement, nor did it demonstrate robust optimization of the value function.

### 5.3.1 Further Evaluation

The absence of consistent reward increase, the weak optimization of the value function, and the negligible magnitude of actor loss collectively indicate that the system failed to leverage the PPO training loop to optimize retrieval for downstream answer quality.

Several factors may account for this outcome. Most critically, implementation challenges related to gradient flow through the dense embedding model likely compromised the ability of the retriever to learn. While the SentenceTransformer-based retriever architecture was modified to support gradient propagation, ensuring correct and stable flow of gradients through pre-trained transformers remains non-trivial. If gradient paths were inadvertently broken or masked during forward or backward passes, the PPO objective would not be able to update the retriever parameters meaningfully.

Furthermore, the optimization landscape in retrieval-augmented generation is highly non-convex and prone to local minima. Without a strong initialization or an informative reward signal, policy updates may oscillate or stagnate. While SBERT cosine similarity provides a smooth and differentiable reward function, it is still sensitive to superficial lexical overlap and may fail to distinguish between semantically weak but syntactically similar outputs. This may have further contributed to unstable or noisy reward gradients.

Finally, the implementation of the critic via multi-head attention should, in principle, enable effective value estimation by modeling query-passage dependencies. However, the observed critic loss pattern suggests that the critic network may have lacked sufficient capacity or training signal to generalize well, or that high-variance GAE estimates degraded its stability.

## 6 Conclusion & Discussion

We have presented a unified two-phase training paradigm that first employs behavior cloning to warm up a dense retriever on MSMARCO gold passages, and then performs on-policy PPO fine-tuning against actual QA rewards. This approach combines the stability of supervised learning with the adaptability of reinforcement learning to give feedback on downstream answer quality, demonstrating a framework to directly improve downstream question-answering accuracy of RAG systems.

Nevertheless, our empirical evidence demonstrates that these strategies are insufficient for reliable training. As detailed in the Results section, the SFT section is blocked by our overly-simple training pipeline, which caused the unsolved problem of vanishing similarity scores even after we applied softmax. For the failure of PPO, the main reason might be that the restricted in-batch corpus produces an excessively dense reward distribution, depriving the retriever of informative gradients and preventing meaningful parameter updates. In addition, label noise in the MS MARCO dataset, arising from incomplete or erroneous passage-to-query alignments, subjects the PPO optimiser to unreliable supervisory signals, further impeding convergence. Furthermore, PPO failed to improve as rewards plateaued, actor-loss stayed near zero, and the critic optimized poorly. Likely causes include (i) broken or unstable gradient flow through the SentenceTransformer encoder, preventing parameter updates; (ii) a non-convex search space and weak SBERT-based reward signal that left the policy trapped in local minima; and (iii) an under-powered or high-variance critic whose multi-head attention failed to supply reliable value estimates.

## 7 Team Contributions

- **Ryan Tan:** evaluation pipeline and PPO training
- **Jeffrey Xue:** SFT trials and evaluation pipeline
- **Richard Gu:** RAG pipeline, SFT trials, and BC training

**Changes from Proposal** We all took on tasks as they appeared, with each of us completing what was necessary at each stage of the project. We all collaborated to write the proposal, milestone, poster, and final paper.

## References

Ge Gao, Jonathan D. Chang, Claire Cardie, Kianté Brantley, and Thorsten Joachims. 2024. Policy-Gradient Training of Language Models for Ranking. arXiv:2310.04407v2 [cs.CL]

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-Augmented Language Model Pre-Training. arXiv:2002.08909 [cs.CL]

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. arXiv:2004.04906 [cs.CL]

Rodrigo Nogueira and Kyunghyun Cho. 2017. Task-Oriented Query Reformulation with Reinforcement Learning. arXiv:1704.04572 [cs.IR]

Zichun Yu, Chenyan Xiong, Shi Yu, and Zhiyuan Liu. 2023. Augmentation-Adapted Retriever Improves Generalization of Language Models as Generic Plug-In. arXiv:2305.17331v1 [cs.CL]

Peitian Zhang, Shitao Xiao, Zheng Liu, Zhicheng Dou, and Jian-Yun Nie. 2023. Retrieve Anything To Augment Large Language Models. arXiv:2310.07554v2 [cs.IR]