

Bridging the Semantic-Utility Gap in Multimodal RAG via Generator-in-the-Loop Alignment

Zhan-Lun Chang, *Student Member, IEEE*, Dong-Jun Han, *Member, IEEE*, Seyyedali Hosseinalipour, *Senior Member, IEEE*, Mung Chiang, *Fellow, IEEE*, and Christopher G. Brinton, *Senior Member, IEEE*

Abstract—Vision-language models (VLMs) augmented with retrieval-augmented generation (RAG) benefit from access to external evidence. However, standard retrievers and rerankers optimize for semantic similarity rather than answer utility, creating a preference gap: documents that appear relevant may not help the generator produce a correct answer. Motivated by this, we propose a two-stage *generator-in-the-loop alignment framework* that closes this gap without human document-level relevance annotations. Our framework consists of two stages: in Stage 1, a VLM generates a hypothetical text passage from the image-query pair, which is used as the retrieval query for dense text search, bridging the image-to-text modality gap. In Stage 2, a cross-encoder reranker adapted with low-rank adaptation (LoRA) is finetuned using answer-supervised preference pairs mined from the frozen VLM: given the dataset answer label, a candidate document is labeled positive if the VLM produces the correct answer when given that document as context, and negative otherwise. This generator-guided signal is compatible with multiple alignment loss functions, including contrastive (triplet) loss, pairwise direct preference optimization (DPO), and supervised fine-tuning (SFT), and supports periodic re-mining to refresh preference pairs as the reranker improves. Experiments on VQA-X and A-OKVQA with Qwen3.5-2B and Qwen3-VL-4B-Instruct show that our proposed framework consistently outperforms rank-order, random, and REPLUG-style likelihood baselines under various alignment losses and pool size settings, suggesting that answer-level generator feedback is an effective supervision signal for preference alignment.

Impact Statement—RAG-enabled multimodal systems are becoming a key component of knowledge-intensive visual reasoning applications, where retrieving misleading evidence can cause models to produce confident yet incorrect answers. This work addresses a fundamental limitation of existing multimodal RAG pipelines by shifting retrieval optimization from semantic similarity toward downstream answer utility. Rather than relying on costly human document-level relevance annotations, the proposed *generator-in-the-loop alignment framework* leverages a frozen VLM to automatically identify evidence that improves answer correctness and uses this signal to train a lightweight LoRA-adapted reranker. The framework is compatible with multiple alignment objectives and consistently improves performance across different datasets and model architectures, demonstrating that answer-level supervision is a more effective retrieval objective

than semantic similarity alone. By improving the reliability and faithfulness of multimodal RAG systems, this approach has the potential to benefit high-stakes applications such as medical decision support, scientific knowledge retrieval, education, document understanding, and other VLM-empowered systems that require trustworthy multimodal reasoning while reducing the dependence on expensive manual supervision.

Index Terms—Cross-modal retrieval, Information retrieval, Retrieval-augmented generation, Visual question answering.

I. INTRODUCTION

LARGE language models (LLMs) and multimodal vision-language models (VLMs) have fundamentally transformed artificial intelligence, demonstrating unprecedented capabilities in natural language understanding, visual reasoning, and content generation [1], [2]. However, despite their scale, these models remain plagued by intrinsic limitations: they are prone to hallucinations, restricted by static parametric knowledge cutoffs, and lack transparency in their reasoning processes [3].

To mitigate these issues, retrieval-augmented generation (RAG) has emerged as the de facto standard by replacing reliance on static parametric knowledge alone with the retrieval of external evidence at inference time [4], [5]. By grounding generation in retrieved evidence, RAG systems significantly enhance factual accuracy and interpretability, particularly in knowledge-intensive domains where hallucination remains a persistent reliability concern [6].

The design of RAG systems has evolved from simple retrieve-then-generate pipelines to architectures that incorporate query rewriting, iterative retrieval, retrieval verification, and multi-step reasoning [7], [8], [9]. While these techniques improve candidate retrieval and evidence refinement, retrievers and rerankers continue to optimize semantic similarity rather than the generator’s downstream objective. Consequently, a document that appears highly relevant to the query may still fail to provide the evidence required for the generator to produce the correct answer.

More critically, semantically relevant documents can contain incorrect or misleading information that steers the generator toward an erroneous answer. Such documents may therefore increase, rather than reduce, the likelihood of hallucination [10]. A reranker must thus distinguish documents that help the generator produce the correct answer from those that are

Z.-L. Chang, M. Chiang, and C. G. Brinton are with the Elmore Family School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907, USA.

D.-J. Han is with the Department of Computer Science and Engineering, Yonsei University, Seoul 03722, Republic of Korea.

S. Hosseinalipour is with the Department of Electrical Engineering, University at Buffalo–SUNY, Buffalo, NY 14260, USA.

semantically relevant but unhelpful or misleading. Training a reranker to make this distinction typically requires document-level relevance annotations; however, manually determining whether each candidate document improves the generator’s answer correctness requires costly, task-specific annotation at scale. This motivates the central question of this work:

Q: How can we train a reranker to identify evidence that improves the generator’s answer correctness using only ground-truth answer labels with no human document-level relevance annotations?

Answering this question becomes more challenging in *multimodal RAG* (MM-RAG), where retrieval must account for information contained in both visual and textual inputs. Specifically, visual embeddings optimized for general image-text alignment may fail to preserve fine-grained visual information, such as object relationships or structural details in scientific diagrams, that is necessary to retrieve evidence for downstream reasoning. This mismatch between the multimodal query and the retrieved textual evidence compounds the misalignment between retrieval relevance and answer correctness. Prior approaches, such as ATLAS [11] and REPLUG [12], attempt to align retrieval relevance with the generator’s downstream objective through likelihood- or perplexity-based training signals. However, these signals measure how a retrieved document changes the probability of a target output rather than whether the generator actually produces the correct answer. Therefore, MM-RAG requires addressing two coupled problems: (i) translating multimodal inputs into effective retrieval queries and (ii) selecting retrieved evidence based on its contribution to downstream answer correctness.

To jointly address these problems, we propose a two-stage *generator-in-the-loop alignment framework*. In Stage 1, a frozen VLM converts the image-question pair into a hypothetical textual rationale using hypothetical document embeddings (HyDE). This rationale is concatenated with the original question and encoded by a *Sentence Transformer* [13] to retrieve a broad pool of candidate documents using Facebook AI similarity search (FAISS). Stage 2 addresses what we term the *Preference Gap*: conventional rerankers prioritize documents based on semantic relevance, whereas the generator requires documents that help it produce the correct answer. A semantically relevant document may therefore lack the evidence needed to answer the question [14] or may even act as a distractor that degrades performance below the closed-book baseline [15], [16]. To directly align reranking with answer correctness, we provide each candidate document to the frozen VLM as context and compare the generated answer with the ground-truth answer. A document is labeled *positive* if the model produces the correct answer and *negative* otherwise, yielding preference pairs without human document-level relevance annotations. We use these pairs to fine-tune a cross-encoder reranker through low-rank adaptation (LoRA) and evaluate the same generator-derived supervision under contrastive learning, direct preference optimization (DPO), and supervised fine-tuning (SFT). By comparing the same supervision signal across three training objectives, we evaluate whether reranking based on the generator’s answer correctness

consistently improves evidence selection independent of the specific training objective.

The main contributions of this paper can be summarized as follows:

- 1) We propose a two-stage multimodal RAG framework that separates multimodal query translation from generator-aligned reranking. In Stage 1, a frozen VLM generates a hypothetical textual rationale from the image-question pair using HyDE. The rationale is concatenated with the question and encoded by a Sentence Transformer to retrieve a broad pool of candidate documents using FAISS.
- 2) We introduce an *answer-supervised preference mining* procedure that trains the Stage 2 reranker without human document-level relevance annotations. Specifically, for each candidate document, the frozen VLM generates an answer using the document as context; documents that lead to correct and incorrect answers are used to construct positive-negative preference pairs for reranker training.
- 3) We use the generator-derived preference pairs to fine-tune a LoRA-adapted cross-encoder reranker under three training objectives: contrastive triplet loss, DPO, and SFT. Experiments on VQA-X and A-OKVQA with Qwen3.5-2B and Qwen3-VL-4B-Instruct show that generator-guided preference mining consistently outperforms rank-order, random, and REPLUG-style likelihood preference mining across the considered training objectives and various candidate pool sizes.
- 4) We develop an iterative preference-alignment mechanism with *periodic preference re-mining* that dynamically reconstructs generator-derived preference pairs as the reranker evolves, thereby keeping the training signal aligned with the reranker’s current ranking behavior rather than relying on a fixed set of initially mined pairs. Experiments across various generators and datasets demonstrate consistent improvements over fixed-pair training, with gains of up to 2.58 percentage points.

II. RELATED WORK

We organize the related literature into the following categories: (i) evolution of RAG; (ii) multimodal retrieval and the semantic gap; (iii) retriever-generator alignment.

A. Evolution of RAG

RAG has emerged as a standard method for mitigating the limitations of LLMs/VLMs, specifically their susceptibility to hallucinations and outdated parametric knowledge [4], [17]. Following recent surveys [7], [18], we summarize the progression of RAG through three phases: *Naive RAG methods*, *Advanced RAG methods*, and *Modular RAG methods*.

The initial phase, *Naive RAG methods*, followed a linear “retrieve-then-generate” protocol. These systems utilized sparse (BM25) or dense passage retrieval (DPR) retrievers to fetch context, which was then directly concatenated with the user query [19]. While effective for simple factual retrieval, Naive RAG methods struggle with low precision and noise induction, where irrelevant retrieved context degrades the generator’s

performance [20]. To mitigate these shortcomings, approaches such as *Chain-of-Note* [21] were introduced, teaching models to explicitly generate *notes* assessing the relevance of retrieved documents before answering, thereby improving robustness against noisy inputs.

To further improve performance, *Advanced RAG* methods introduced sophisticated pre-retrieval and post-retrieval optimizations. Specifically, pre-retrieval techniques, such as query rewriting [9], multi-aspect query retrieval [22], and HyDE [23], aim to align the user’s intent with the document space. HyDE in particular prompts a language model to generate a hypothetical answer-oriented document from the query and uses that generated text, rather than the raw query, as the retrieval input; because this text resembles the corpus documents more closely, its embedding can have higher similarity to answer-bearing passages and improve recall without retrieval-specific training. Also, post-retrieval strategies focused on refining the context window. Notably, reranking models [24] and cross-encoder filtering [25] were developed to identify the most relevant retrieved passages, thereby reducing irrelevant context before generation. Despite these improvements, recent studies on the “lost in the middle” phenomenon [15] suggest that simply stuffing context windows is insufficient, as models struggle to access information buried in the middle of long retrieval sequences.

Most recently, the field has shifted toward *modular and active RAG* [26]. In this paradigm, the pipeline is decomposed into independent, swappable modules orchestrated by programming frameworks such as DSPy [8], which optimizes the flow of information between modules. Beyond static retrieval, recent frameworks further employ *active* strategies. For instance, FLARE [27] drafts upcoming generated text and triggers retrieval when low-confidence tokens suggest that additional evidence is needed, while CRAG (corrective RAG) [28] introduces a lightweight retrieval evaluator that triggers a web search fallback when the retrieved documents are deemed incorrect. These systems improve *when* and *how* retrieval is invoked, but evidence selection often remains based on *semantic similarity*, meaning how closely a retrieved document matches the query in topic, wording, or embedding space. However, a semantically similar document may still be unhelpful or even misleading for answering the question. Therefore, evidence should instead be selected according to its *downstream answer utility*, namely whether providing that document as context actually helps the generator produce the correct answer.

B. Multimodal Retrieval and the Semantic Gap

The success of RAG in the textual domain has extended to large multimodal models, giving rise to MM-RAG [29], [30], systems that retrieve external textual or visual evidence to support reasoning over image–text queries. Unlike text-only RAG, MM-RAG must align heterogeneous query inputs, such as images and natural-language questions, with evidence stored in an external knowledge base. Standard retrieval architectures commonly rely on dual-encoder backbones such as CLIP [31] to map visual and textual content into a shared semantic space. Meanwhile, foundational models such as

Flamingo [32] demonstrated the effectiveness of conditioning generation on interleaved image–text inputs. However, neither multimodal conditioning nor coarse image–text alignment alone guarantees that the retrieved evidence contains the fine-grained information needed for downstream reasoning. This leads to a critical *semantic gap*: visual embeddings optimized for general image–caption matching may overlook subtle objects, relationships, and structural details that are essential for retrieving answer-supporting evidence [26]. For example, Qwen-VL [33] attempts to preserve finer visual information through resolution-aware visual encoders, but still relies on conventional retrieval mechanisms. Recent efforts have therefore sought to narrow this gap by integrating stronger generative or fine-grained visual representations into retrieval. RA-CM3 [34] was among the first to jointly retrieve and generate mixed-modal content within a unified architecture. More recently, ColPali [35] avoids the lossy conversion of document pages into text by using a VLM to produce multi-vector embeddings for page patches, enabling late-interaction retrieval that better preserves visual layout and textual semantics.

Our work takes a complementary approach to these methods by decoupling multimodal query alignment from evidence-utility estimation. Rather than requiring the retriever itself to learn a shared representation that preserves all task-relevant visual and textual information, we use a frozen VLM to translate the image–question pair into a hypothetical textual rationale. This converts the multimodal query into the same modality as the document corpus, enabling effective text-based dense retrieval without training a dedicated multimodal retriever. However, resolving the modality gap alone does not ensure that the retrieved documents will help the generator answer correctly. We therefore introduce a second stage in which a separately trained cross-encoder reranker is aligned with the frozen generator’s downstream behavior, prioritizing documents according to their answer utility rather than semantic similarity.

C. Retriever-Generator Alignment

The most significant challenge in RAG is the *alignment gap* between retrieval or reranking objectives and the generator’s reasoning objective [7]. In particular, retrievers are typically optimized for query–document semantic similarity, an objective decoupled from whether the retrieved context actually helps the generator produce the correct answer. This mismatch frequently leads to failure in complex reasoning tasks [36], despite evidence that LLMs can effectively utilize in-context knowledge when the right documents are retrieved [37].

Several methods have sought to reduce the alignment gap by incorporating generator feedback into retrieval training. For instance, Atlas [11] jointly trains the retriever and generator using generation perplexity, while REPLUG [12] treats the language model as a black box and trains the retriever using each document’s effect on the probability of the ground-truth continuation. These methods make retrieval more generator-aware, but their likelihood-based signals remain indirect proxies for downstream correctness and were developed primarily for text-only generation settings.

BGM [38] addresses a related *preference gap*, in which the retriever’s highest-ranked documents do not necessarily

produce the best generation outcomes. Rather than updating the frozen retriever and generator, BGM inserts a trainable sequence-to-sequence bridge between them. The bridge is trained using supervised learning on synthesized silver passages and reinforcement learning to rerank and reorganize the retrieved context. Although this approach moves closer to task-level optimization, it introduces an additional large trainable component into the RAG pipeline.

Our framework instead aligns reranking directly with the deployed generator’s answer behavior: for each candidate document, the frozen VLM is prompted to produce the constrained dataset answer output, either yes/no or one of the provided options, and the document is labeled useful when that output matches the dataset-provided ground-truth. The resulting signal directly measures whether a document helps the generator answer correctly, rather than whether it increases the likelihood of the ground-truth answer when preceding answer tokens are supplied. These generator-specific document-level preferences are then distilled into a lightweight cross-encoder reranker. Our method therefore complements conventional semantic rerankers [24], [25], likelihood- and reward-based retrieval alignment methods [11], [12], [38], and multimodal retrieval approaches that primarily address cross-modal representation alignment [34], [?].

Our proposed supervision mechanism is related to *LLM-as-judge* and synthetic-data methods. LLM-as-judge approaches use language models to assign general quality scores or preferences to open-ended outputs [39], while self-generated instruction methods synthesize training examples to reduce dependence on manual annotation [40]. In our setting, however, the VLM neither produces a general evaluation nor generates new instructions. Instead, it reveals whether a particular retrieved document causes the same frozen generator used at deployment to produce the correct answer. The resulting labels differ from general-purpose LLM judgments or synthetic training data: they provide generator-specific, document-level utility signals tailored explicitly to reranker alignment.

III. PROPOSED ARCHITECTURE

In this section, we present the architecture of our two-stage MM-RAG system and the alignment framework used to close the gap between candidate ranking and answer utility. The key idea is to separate broad multimodal recall from generator-specific utility alignment: Stage 1 uses the frozen deployment generator as an image-to-text bridge for dense retrieval, while Stage 2 uses the same frozen generator to mine answer-level preferences for reranker training. Table I summarizes the mathematical notations used throughout.

A. Problem Formulation

We formulate reranking as an answer-utility alignment problem rather than a standard relevance-ranking problem. Formally, we presume that the corpus $\mathcal{D}$ consists of text documents $d \in \mathcal{D}$ (e.g., textual justifications for VQA-X or rationales for A-OKVQA) that serve as retrieval targets: each document provides external textual evidence that the generator $\mathcal{G}$ can use as context to help answer a question. Each query consists of an image v and a natural-language question q ;

TABLE I
MATHEMATICAL NOTATIONS

Symbol	Meaning
v	Input image
q	Natural-language question string
a_{gold}	Ground-truth answer
$\mathcal{G}$	Frozen VLM generator
$h = \mathcal{G}(v, q)$	HyDE hypothetical rationale text
$q_{\text{text}} = q \oplus h$	Concatenated query string
$\phi(\cdot)$	Qwen3-Embedding-8B text encoder
$\mathbf{e}_q = \phi(q_{\text{text}})$	ℓ_2 -normalized query embedding
$\mathcal{I}$	FAISS IndexFlatIP index
$\mathcal{C}_q$	Stage 1 top- K candidate pool
K	Stage 1 retrieval depth (default 100)
K'	Stage 2 preference-mining pool cap, with $K' \leq K$
E_θ	Cross-encoder ($\theta = \theta_{\text{base}} \cup \theta_{\text{LoRA}}$)
E_{ref}	Frozen reference reranker (no LoRA)
$s_\theta(q_{\text{text}}, d)$	Scalar relevance logit for pair (q_{text}, d)
$\mathcal{D}$	Mined preference dataset
d	A text document (e.g., textual justification or rationale)
d^+	Positive (preferred) document
d^-	Negative (rejected) document
$\mathcal{S}_+, \mathcal{S}_-$	Correct / incorrect candidate sets
B_m	VLM mini-batch size during mining
N	Re-mining frequency (epochs between re-mines)
η	AdamW learning rate
β	DPO inverse-temperature
m	Triplet margin
Ψ	Persistent VLM prediction cache

the dataset also provides a ground-truth answer a_{gold} that is available only during preference mining. The textual query representation used by the retriever and reranker is denoted q_{text} and is formed by concatenating the question q with the hypothetical rationale h generated by $\mathcal{G}$. Retrieving with q_{text} yields the candidate pool $\mathcal{C}_q$ consisting of K documents. Given a frozen generator $\mathcal{G}$, a query (v, q) , and a candidate document $d \in \mathcal{C}_q$, we define the *answer utility* of d as the binary indicator

$$u(d) = \mathbf{1}[\mathcal{G}(v, q, d) = a_{\text{gold}}], \quad (1)$$

which equals 1 if and only if $\mathcal{G}$ produces the correct answer when given d as context. We further partition the candidate pool $\mathcal{C}_q$ accordingly into a set of *useful* documents $\mathcal{S}_+ = \{d \in \mathcal{C}_q : u(d) = 1\}$ and a set of *harmful* documents $\mathcal{S}_- = \{d \in \mathcal{C}_q : u(d) = 0\}$.

Over this pool, a pre-trained cross-encoder E_θ scores each pair (q_{text}, d) with a relevance logit $s_\theta(q_{\text{text}}, d)$ based on *semantic similarity*, selecting $d^* = \arg \max_{d \in \mathcal{C}_q} s_\theta(q_{\text{text}}, d)$. However, semantic similarity does not imply answer utility: a document semantically close to the query may fail to support correct reasoning, while a document that appears less similar may provide exactly the missing evidence. This gives rise to the **preference gap**, which can be interpreted as follows:

$$E_\theta \text{ is misaligned when } d^* \notin \mathcal{S}_+. \quad (2)$$

Research Problem. Given a candidate pool $\mathcal{C}_q$ and a frozen generator $\mathcal{G}$, we aim to learn reranker parameters θ such that

$$s_\theta(q_{\text{text}}, d^+) > s_\theta(q_{\text{text}}, d^-) \quad \forall d^+ \in \mathcal{S}_+, d^- \in \mathcal{S}_-, \quad (3)$$

without any human-annotated document relevance labels. In this paradigm, the supervision signal is the answer utility $u(d)$, which is computed by querying the frozen $\mathcal{G}$ and comparing its output to the dataset-provided answer label.

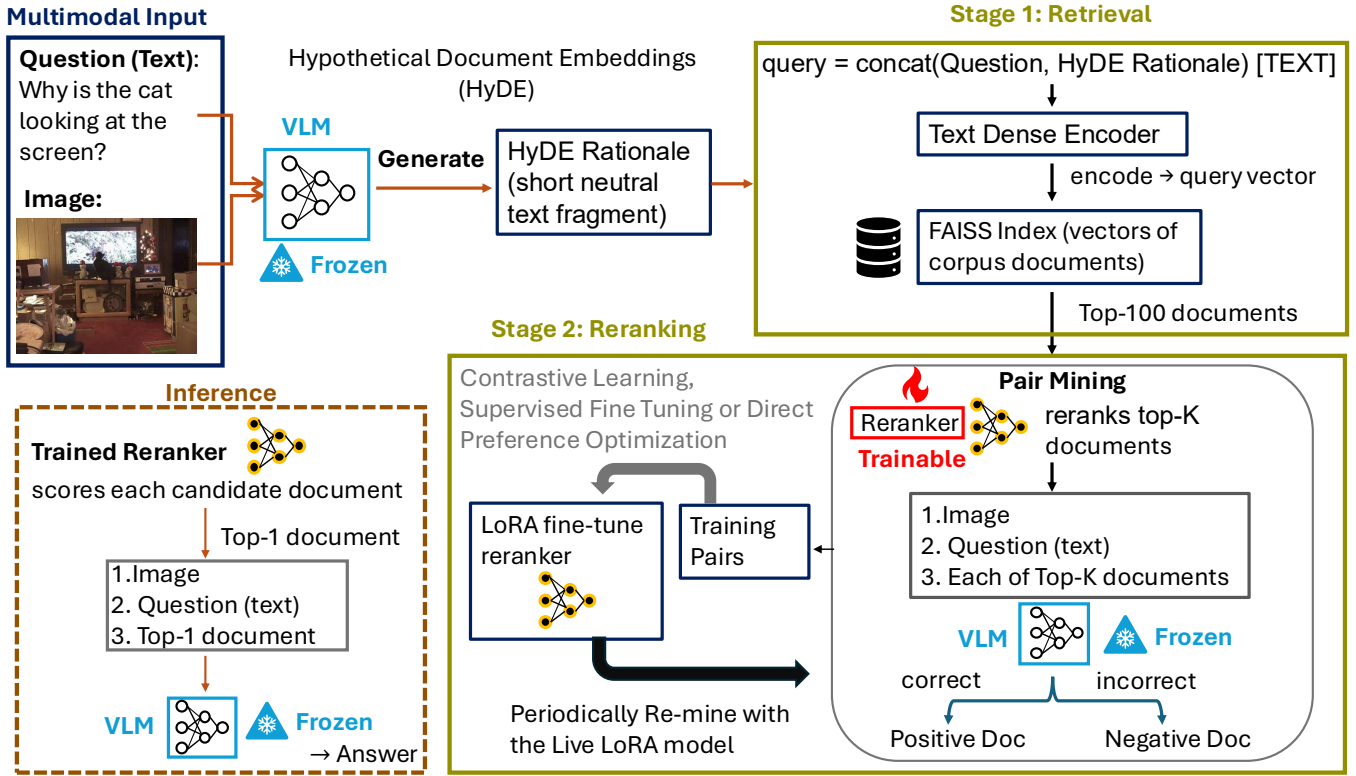


Fig. 1. A schematic of our proposed two-stage multimodal RAG system. **Stage 1 (Retrieval)**: given a multimodal input (v, q) , the frozen VLM $\mathcal{G}$ generates a short HyDE rationale h ; the composite query $q_{\text{text}} = q \oplus h$ is encoded by a text dense encoder and used to retrieve the top- k corpus documents from a FAISS index. **Stage 2 (Reranking)**: a LoRA-adapted cross-encoder scores each of the k candidate documents and is fine-tuned on preference pairs (d^+, d^-) mined by querying the frozen VLM for answer correctness. Preference pairs are periodically re-mined using the live LoRA model, and the reranker can be trained under contrastive (triplet), SFT, or DPO losses. **Inference**: the trained reranker selects the top-1 document, which is passed together with the image and question to the frozen VLM to produce the final answer.

B. Architecture Overview

Our proposed framework is built on a two-stage pipeline, illustrated in Fig. 1. This stage-wise decomposition is intentional: Stage 1 is optimized for recall across the text corpus, while Stage 2 learns which recalled evidence actually improves the frozen generator’s answer behavior. More specifically, Stage 1 converts the multimodal input (v, q) into a short hypothetical text rationale h via HyDE, which is concatenated with the question to form the composite text query $q_{\text{text}} = q \oplus h$. This query is encoded by a text encoder (e.g., Qwen3-Embedding-8B) ϕ and used to search a FAISS index $\mathcal{I}$, producing a candidate pool $\mathcal{C}_q$ of up to K documents. Subsequently, Stage 2 takes $\mathcal{C}_q$ and applies a LoRA-adapted cross-encoder E_θ to rerank the candidates by assigning a scalar relevance logit $s_\theta(q_{\text{text}}, d)$ to each text pair (q_{text}, d) . The reranker is fine-tuned on preference pairs mined by querying the frozen $\mathcal{G}$ for answer correctness; pairs are periodically re-mined using the live model to keep the training signal fresh. At inference, the top-scored document $d^* = \arg \max_{d \in \mathcal{C}_q} s_\theta(q_{\text{text}}, d)$ is passed together with (v, q) to the frozen $\mathcal{G}$ for final answer generation.

One of our key design choices is to separate multimodal understanding from reranker training. Specifically, the frozen generator $\mathcal{G}$ is the only component that processes the image v : in Stage 1, it translates the image–question pair into a textual rationale that enables text-based retrieval, while in Stage 2, it evaluates whether each retrieved document helps produce the correct answer. In contrast, the cross-encoder E_θ

operates exclusively on text, allowing it to leverage strong pre-trained language representations without requiring multimodal adaptation. This separation keeps the architecture lightweight: $\mathcal{G}$ remains frozen and provides both modality conversion and supervision, while only the text-based reranker is trained.

C. Datasets and Document Corpora

Fig. 2 illustrates what a real-world multimodal query looks like in the two datasets considered in this work: each query consists of an image v and a natural-language question q , while the knowledge base is populated with dataset-specific textual documents (justifications for VQA-X, rationales for A-OKVQA) that serve as the retrieval targets for Stage 1. The following paragraphs briefly describe how each dataset defines the multimodal query, constructs the external knowledge base, and uses the retrieved evidence for answer generation.

- **A-OKVQA (Augmented Outside Knowledge VQA):**

This dataset focuses on visual question answering tasks that require commonsense or world knowledge beyond what is directly visible in the image. As shown in Fig. 2, the pipeline accepts the query image (e.g., a cat watching TV) and the associated question text (e.g., Why is the cat looking at the screen?). The external database is populated with the dataset’s *Rationales*, which provide the necessary background information (e.g., The cat is intrigued by the image on the screen). The generator uses these retrieved rationales to select the correct option from multiple-choice answers or provide a direct response.

VQA-X	A-OKVQA	
Question: Does the giraffe have a goofy look on his face?	Question: Why is the cat looking at the screen?	Query Text
		Query Image
Textual Justification: "his jaw is all out of whack he is chomping with his mouth its bottom lip is rolled up."	Rationale(s): 1. The cat is curious about the tv. 2. The cat is intrigued by the image on the screen. 3. The cat is wondering about what it sees on the screen.	Documents
Answer: yes	Multiple-Choices Answers: (a) warmth (b) no reason (c) light (d) curiosity [CORRECT]	

Fig. 2. Dataset Integration into the MM-RAG Pipeline. For each dataset (A-OKVQA and VQA-X), we map the original inputs to a unified query format comprising the image and question text. The external knowledge base is populated with the dataset-specific ground-truth explanations (rationales and justifications). The retrieval target is defined as the textual evidence required to bridge the reasoning gap between the visual query and the final answer.

- **VQA-X (Visual Question Answering with eXplanations):** This dataset extends standard VQA by providing textual justifications for answers. As shown in Fig. 2, the query is formed by the image (e.g., a giraffe) and the question (e.g., Does the giraffe have a goofy look on his face?). The *Textual Justification* serves as the retrievable document. By retrieving these ground-truth reasons (e.g., ...his jaw is all out of whack...), the pipeline aligns the generator’s output with the expected reasoning process.

To prevent test-set evidence leakage, the retrieval corpus is constructed exclusively from the training split and fixed before Stage 2 training. For A-OKVQA, the FAISS index contains the unique, non-empty rationales from the training set, whereas for VQA-X, it contains the unique, non-empty explanations associated with yes/no training examples. Stage 1 retrieves candidates from this fixed corpus and stores them in split-specific candidate files. Stage 2 then combines these files with the corresponding dataset split to recover the images, answer choices, and ground-truth labels required for preference mining and evaluation. In this way, the reranker is trained on candidates drawn only from the training-derived corpus, while its supervision is obtained from answer correctness rather than human document-level relevance annotations.

D. (Stage 1) HyDE-Based Modality Bridge and Dense Recall

A fundamental challenge in multimodal RAG is the *modality gap* between the multimodal query and the text-only retrieval space: the text encoder ϕ accepts only text, while the query is inherently multimodal, comprising an image v and a natural-language question q . Using q alone therefore discards potentially essential visual information from v , while training a dedicated cross-modal retriever would require substantial amounts of aligned image–text data and additional model adaptation.

To bridge this gap, we use HyDE [23] as a multimodal-to-text conversion mechanism rather than merely as a generic

query-expansion technique. Specifically, rather than searching with the raw question q or training a separate multimodal retriever, the frozen generator $\mathcal{G}$ is first prompted with the full multimodal input (v, q) to produce a *hypothetical rationale*

$$h = \mathcal{G}(v, q), \quad (4)$$

where h is a short, image-grounded passage that captures the visual information relevant to the question and resembles the type of evidence contained in the retrieval corpus. The rationale is then concatenated with the original question to form the composite textual query $q_{\text{text}} = q \oplus h$. Since q_{text} is entirely textual, it can be processed directly by ϕ without any multimodal adaptation. HyDE thus serves as an image-to-text modality bridge, projecting the task-relevant information in v into the textual domain and enabling text-only dense retrieval for an inherently multimodal query.

The resulting query embedding is computed as $\mathbf{e}_q = \phi(q_{\text{text}}) \in \mathbb{R}^n$, where ϕ prepends an instruction prefix and applies ℓ_2 normalization. The corpus documents are encoded offline using the same encoder, and their embeddings are stored in a FAISS IndexFlatIP index $\mathcal{I}$ that supports exact inner-product search. Because each rationale or justification is already a short, self-contained unit, no document chunking is applied; instead, each document is indexed in its entirety. At query time, the top- K candidates are retrieved as

$$\mathcal{C}_q = \text{top-}K(\mathbf{e}_q, \mathcal{I}). \quad (5)$$

To avoid repeated VLM inference, the hypothetical rationale h is cached per question identifier with atomic disk writes. The resulting candidate set is represented as $\mathcal{C}_q = \{(d_1, r_1), \dots, (d_K, r_K)\}$, where d_i denotes the i -th retrieved document and r_i its retrieval rank. These candidate sets are then stored and passed to Stage 2 for generator-aligned reranking.

E. (Stage 2) Generator-Guided Cross-Encoder Alignment

Stage 2 is responsible for reranking the candidate pool $\mathcal{C}_q$ retrieved by Stage 1. Its key objective is to replace indirect supervision signals, such as semantic relevance or ground-truth-token likelihood, with direct feedback on whether the generator’s constrained answer output matches the ground-truth. Specifically, the reranker is trained to prefer documents that enable the frozen generator $\mathcal{G}$ to produce the dataset-provided ground-truth answer a_{gold} . To learn this preference while keeping training lightweight, we adapt a GTE-ModernBERT-base cross-encoder [41] with LoRA ($r = 16$, $\alpha = 32$, dropout = 0.1), where only the low-rank adapter weights θ_{LoRA} are updated during training while the pre-trained encoder θ_{base} is kept frozen. For each query–document pair, the cross-encoder jointly processes the composite textual query q_{text} and the candidate document d , truncated to a maximum of 512 tokens, and produces a scalar score

$$s_{\theta}(q_{\text{text}}, d) = E_{\theta}([q_{\text{text}}; d]) \in \mathbb{R}, \quad (6)$$

This score represents the reranker’s learned estimate of how useful the document is for supporting the generator’s answer. To control VLM inference cost during preference mining, Stage 2 evaluates only the top- K' prefix of the Stage 1 pool, where $K' \leq K$.

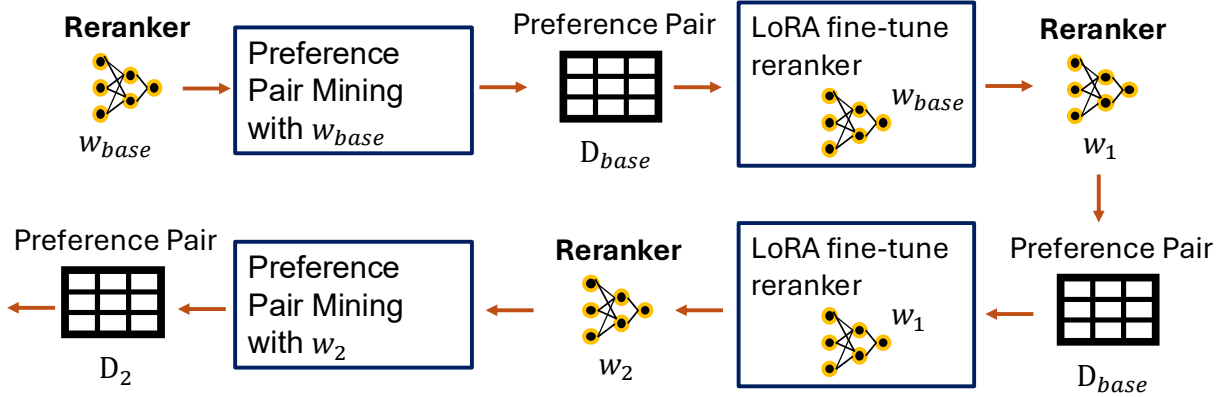


Fig. 3. Iterative preference alignment with periodic re-mining. Training begins with the base reranker weights w_{base} . Preference pairs are first mined under w_{base} , producing dataset D_{base} . The reranker is then LoRA fine-tuned on D_{base} , advancing the weights from w_{base} to w_1 and then to w_2 . After a fixed number of epochs, the preference pairs are re-mined under the current model w_2 , yielding a refreshed dataset D_2 that reflects the model’s updated ranking. This cycle can keep the training signal aligned with the reranker’s current behavior, although the benefit of additional refreshes may saturate once the model has stabilized.

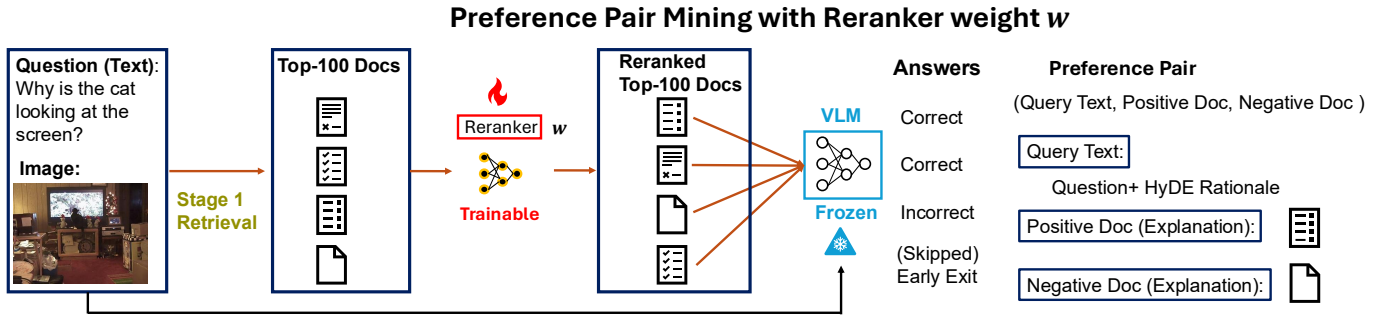


Fig. 4. Generator-guided preference pair mining (proposed). Given a multimodal query (v, q) , Stage 1 retrieves 100 candidate documents. The trainable reranker with current weights w re-scores those candidates, and Stage 2 retains the top- $K' = 10$ subset for mining. The frozen VLM then evaluates each retained document in descending score order: if the generated answer is correct, the document is taken as d^+ and subsequent correct answers are skipped; if the answer is incorrect, the document is taken as d^- . Evaluation exits immediately once both d^+ and d^- are found, avoiding unnecessary VLM calls. The resulting training triple is (query text = question $\oplus$ HyDE rationale, d^+ , d^-).

Training supervision is derived automatically through *preference mining*: for each training query, the current reranker E_θ first scores the Stage 1 candidates and retains only the top- K' prefix for mining. A preference pair (d^+, d^-) is then constructed, where d^+ denotes a preferred document and d^- denotes a rejected one. All four mining strategies follow this same pairwise formulation but differ in how they assign the positive and negative labels. The proposed **generator-guided** strategy feeds each retained candidate document to the frozen VLM $\mathcal{G}$ together with the image v and question q , and checks whether $\mathcal{G}$ produces the correct answer under the dataset answer label. The first retained candidate (in reranker-score order) that leads to a correct answer becomes d^+ , and the first retained candidate that leads to an incorrect answer becomes d^- . If the retained pool does not contain both outcomes, the query is skipped and contributes no training pair for that epoch. The controlled baselines construct the same pairwise labels from different signals. The **REPLUG-style likelihood** baseline [12] does not rely on constrained answer generation; instead, it scores each retained candidate according to the likelihood of a_{gold} when preceding ground-truth answer tokens are supplied and selects the highest- and lowest-scoring documents as d^+ and d^- . The **first** baseline requires no VLM: it uses only the current reranker E_θ , assigning its highest-ranked candidate to d^+ and its lowest-ranked candidate to d^- . The **random**

baseline ignores both reranker scores and VLM feedback: d^+ and d^- are drawn uniformly at random from the retained pool. The mined pairs are used to fine-tune E_θ under one of three alignment loss functions: triplet margin loss, SFT, or pairwise DPO [42]. To keep preference data fresh as the model improves, pairs are periodically re-mined against the current E_θ checkpoint rather than a fixed initial ranking. Section IV provides pseudocode and walk-throughs for the training loop and mining strategies.

IV. RERANKER ALIGNMENT AND PREFERENCE MINING

Section III introduced the two-stage MM-RAG architecture, in which Stage 1 translates the multimodal query into text and retrieves a candidate document pool, while Stage 2 reranks these candidates according to their utility to the frozen generator. Building on this architecture, this section describes how the reranker is trained and evaluated. Specifically, we present the iterative LoRA-based alignment procedure with periodic re-mining, define the three alignment loss functions, and describe both the proposed generator-guided preference-mining strategy and the First, Random, and REPLUG-style likelihood baselines.

Figs. 3 and 4 illustrate the two central components of our proposed alignment procedure. Specifically, Fig. 3 depicts the iterative mine–train–re-mine loop, whereas Fig. 4 shows how generator feedback is used to construct a positive–negative preference pair for an individual query. In essence, our proposed

procedure is enabled by three key mechanisms: generator-guided preference mining uses downstream answer correctness as the supervision signal; batching, early exit, and persistent caching reduce the cost of repeated VLM evaluation; and periodic re-mining refreshes the preference pairs as the reranker evolves.

Fig. 3 shows the iterative alignment loop. At the start, the reranker E_θ is used to score the Stage 1 candidates, and a preference dataset $\mathcal{D}_1$ is mined from those scores. The reranker then undergoes LoRA fine-tuning using $\mathcal{D}_1$ for N consecutive epochs. After N epochs, the now-improved reranker is used to re-score the same candidates and produce a refreshed dataset $\mathcal{D}_2$. Fine-tuning continues from $\mathcal{D}_2$ for another N epochs, and this mine-then-train cycle repeats until the total number of training epochs E is reached. The key intuition is that as the reranker changes, re-mining can refresh the preference pairs so that training continues to reflect the model’s current ranking behavior.

Further, Fig. 4 shows how a single preference pair (d^+, d^-) is constructed for one query. The reranker scores all retrieved candidate documents and evaluates them in descending score order using the frozen VLM $\mathcal{G}$. For each candidate, $\mathcal{G}$ generates an answer conditioned on the image, the question, and that candidate as context. The first candidate for which $\mathcal{G}$ produces the correct answer is taken as d^+ , and the first candidate for which $\mathcal{G}$ produces a wrong answer is taken as d^- . Once both documents have been identified, the evaluation terminates through an early-exit mechanism, thereby avoiding unnecessary VLM calls for the remaining candidates. If the entire candidate pool is examined without finding both a positive and a negative document (e.g., when all candidates produce the same correctness outcome), the query is skipped and contributes no preference pair for that mining round.

Subsection IV-A presents the iterative loop. Section IV-B defines the three alignment loss functions. Subsection IV-C details the preference mining subroutine for all four strategies.

A. Iterative Reranker Alignment

Algorithm 1 describes the reranker-alignment procedure in two phases: initialization and iterative training. In the *initialization phase* (Lines 1–3), the LoRA parameters θ_{LoRA} are randomly initialized and the preference dataset $\mathcal{D}$ is mined using the unmodified base cross-encoder $E_{\theta_{\text{base}}}$. In other words, the randomly initialized adapters are not used to rank candidates during the initial mining step, preventing them from introducing arbitrary ordering effects into the first set of preference pairs. The LoRA adapters are then incorporated into the reranker, and the base parameters θ_{base} are frozen. Only θ_{LoRA} is updated during training.

In the *iterative training phase* (Lines 4–11), the outer loop over epochs e triggers a re-mining step at the start of epoch e whenever $(e - 1) \bmod N = 0$ and $e > 1$. Re-mining replaces stale preference pairs (scored under an older model snapshot) with fresh pairs scored by the current E_θ , so that d^+ and d^- reflect the model’s present ranking. This can provide a useful refresh when the reranker has changed substantially, but additional refreshes may yield diminishing returns after the model saturates. Within each epoch, the

Algorithm 1 Iterative LoRA Fine-Tuning with Periodic Re-mining

Require: Queries $\mathcal{Q}$, base cross-encoder $E_{\theta_{\text{base}}}$, frozen VLM $\mathcal{G}$
Require: Pool cap K' , re-mine frequency N , epochs E , learning rate η , method

- 1: randomly initialize θ_{LoRA} ; freeze θ_{base}
- 2: $\mathcal{D} \leftarrow \text{PreferenceMine}(\mathcal{Q}, E_{\theta_{\text{base}}}, \mathcal{G}, K')$
- 3: $E_\theta \leftarrow E_{\theta_{\text{base}}} + \theta_{\text{LoRA}}$
- 4: **for** $e = 1$ **to** E **do**
- 5: **if** $N > 0$ **and** $(e - 1) \bmod N = 0$ **and** $e > 1$ **then**
- 6: $\mathcal{D} \leftarrow \text{PreferenceMine}(\mathcal{Q}, E_\theta, \mathcal{G}, K')$
- 7: **for each** $(q_{\text{text}}, d^+, d^-) \in \mathcal{D}$ **do**
- 8: $s^+ \leftarrow s_\theta(q_{\text{text}}, d^+)$
- 9: $s^- \leftarrow s_\theta(q_{\text{text}}, d^-)$
- 10: $\mathcal{L} \leftarrow \text{ComputeLoss}(s^+, s^-, \text{method})$
- 11: update θ_{LoRA} via AdamW on $\nabla_{\theta_{\text{LoRA}}} \mathcal{L}$ with learning rate η
- 12: **return** final model E_θ

training set $\mathcal{D}$ is iterated over sample by sample (in simulations, in mini-batches of 16). Each training sample is a triple $(q_{\text{text}}, d^+, d^-)$ for which the cross-encoder produces two scalar logits $s^+ = s_\theta(q_{\text{text}}, d^+)$ and $s^- = s_\theta(q_{\text{text}}, d^-)$, which are passed to `ComputeLoss` to compute the alignment loss (the specific loss depends on the chosen method: Contrastive Triplet, SFT, or Pairwise DPO, as defined in Section IV-B). The resulting gradient is then back-propagated through θ_{LoRA} , and the parameters are updated using AdamW. After the final epoch, the aligned reranker E_θ is saved.

B. Alignment Loss Functions

Once a preference pair (d^+, d^-) has been mined, the reranker is trained to assign a higher score to the document that helps the frozen generator answer correctly than to the document that does not. For each pair, the cross-encoder produces two scalar logits, $s^+ = s_\theta(q_{\text{text}}, d^+)$ and $s^- = s_\theta(q_{\text{text}}, d^-)$.

- **SFT:** The first objective treats answer utility as a supervised regression target. The positive document is assigned the target score 1, while the negative document is assigned the target score 0:

$$\mathcal{L}_{\text{SFT}} = (s_\theta(q_{\text{text}}, d^+) - 1)^2 + (s_\theta(q_{\text{text}}, d^-) - 0)^2. \quad (7)$$

This objective directly pushes useful documents toward a high absolute score and unhelpful documents toward a low absolute score. However, because reranking ultimately depends on relative ordering rather than calibrated score values, these fixed targets impose a stronger constraint than is strictly necessary.

- **Contrastive Triplet Loss:** The second objective focuses directly on relative preference. Rather than assigning absolute score targets, the triplet loss trains the reranker by enforcing a relative ordering constraint: the relevance score of the positive document s^+ must exceed that of the negative document s^- by at least a margin m :

$$\mathcal{L}_{\text{triplet}} = \max(0, m - (s^+ - s^-)), \quad m = 0.2. \quad (8)$$

The $\max(0, \cdot)$ hinge structure implies that the loss is zero whenever $s^+ - s^- \geq m$, i.e., the reranker already ranks d^+ sufficiently above d^- . A gradient is produced only when this margin is violated, pushing s^+ up and s^- down until the gap is restored. This makes triplet loss a natural objective for reranking because it optimizes the positive-over-negative ordering directly without requiring the logits to take any particular absolute values. Further, the margin $m = 0.2$ discourages near-ties by encouraging a small score buffer between preferred and rejected documents.

- **Pairwise DPO:** The third objective adapts DPO [42] to reranking. Following the score definition in (6), the trainable logit is mapped to a bounded DPO-style log-score, $\ell_\theta(q_{\text{text}}, d) = \log \sigma(s_\theta(q_{\text{text}}, d)) = \log \sigma(E_\theta([q_{\text{text}}; d]))$, where $\sigma(\cdot)$ denotes the logistic sigmoid. For the frozen reference reranker E_{ref} , the analogous logit is $s_{\text{ref}}(q_{\text{text}}, d) = E_{\text{ref}}([q_{\text{text}}; d])$, yielding $\ell_{\text{ref}}(q_{\text{text}}, d) = \log \sigma(s_{\text{ref}}(q_{\text{text}}, d))$. The preference margins are then defined as

$$\begin{aligned} \Delta_\theta &= \ell_\theta(q_{\text{text}}, d^+) - \ell_\theta(q_{\text{text}}, d^-), \\ \Delta_{\text{ref}} &= \ell_{\text{ref}}(q_{\text{text}}, d^+) - \ell_{\text{ref}}(q_{\text{text}}, d^-), \end{aligned} \quad (9)$$

The DPO loss is then given by

$$\mathcal{L}_{\text{DPO}} = -\log \sigma(\beta[(\Delta_\theta) - (\Delta_{\text{ref}})]). \quad (10)$$

This objective encourages the LoRA-adapted reranker to prefer d^+ over d^- more strongly than the original base reranker does. At the same time, the reference margin anchors the update to the pre-trained model, reducing unnecessary deviation from its initial ranking behavior. The parameter β controls the strength of this reference-relative preference update.

Together, the above objectives provide three complementary views of alignment: SFT learns absolute utility targets, triplet loss enforces relative ordering, and DPO improves the preferred-over-rejected margin relative to the frozen base reranker. Using the same mined preference pairs across all three objectives allows the experiments to isolate the effect of the supervision signal from the effect of the loss function.

C. Preference-Mining Strategies

The previous subsection defined how a mined pair (d^+, d^-) is used to optimize the reranker under SFT, triplet, or DPO losses. We now describe how these positive-negative pairs are constructed. This distinction is important because all training objectives operate on the same pairwise format, while the supervision strategy determines what qualifies a document as preferred or rejected. For mining, $\mathcal{C}_q$ is the Stage 1 top- K pool retrieved in (5), and $\mathcal{C}_q^{K'}$ denotes its top- K' subset after sorting by the current reranker score, with $K' \leq K$. Algorithm 2 presents the proposed generator-guided strategy. Algorithm 3 covers the three baselines: **first**, **random**, and **REPLUG-style likelihood (REPLUG-L)**.

1) *Generator-Guided Preference Mining:* Algorithm 2 specifies the proposed generator-guided mining strategy. Its purpose is not to estimate generic document relevance, but to ask whether each candidate changes the deployed generator’s answer behavior. For each query triple (q, v, a_{gold}) , the retained

Algorithm 2 Generator-Guided Preference Mining (Proposed)

Require: Queries $\mathcal{Q}$, reranker E_θ , frozen VLM $\mathcal{G}$, cap K' , batch size B_m , cache Ψ

```

1:  $\mathcal{D} \leftarrow \emptyset$ 
2: for each  $(q, v, a_{\text{gold}}) \in \mathcal{Q}$  do
3:    $\mathcal{C}_q^{K'} \leftarrow \text{top-}K'(\mathcal{C}_q; s_\theta(q_{\text{text}}, \cdot)) \triangleright \text{descending score}$ 
4:    $d^+ \leftarrow \emptyset; d^- \leftarrow \emptyset$ 
5:   for  $i = 0, B_m, 2B_m, \dots$  do
6:     batch  $\leftarrow \mathcal{C}_q^{K'}[i : i + B_m]$ 
7:     for each  $d \in \text{batch}$  do
8:       key( $d$ )  $\leftarrow (\text{doc-id}(d), \text{pos}(d))$ 
9:       if key( $d$ )  $\notin \Psi$  then
10:         $\Psi[\text{key}(d)] \leftarrow \mathcal{G}(v, q, d)$ 
11:        if  $\Psi[\text{key}(d)] = a_{\text{gold}}$  and  $d^+ = \emptyset$  then
12:           $d^+ \leftarrow d$ 
13:        if  $\Psi[\text{key}(d)] \neq a_{\text{gold}}$  and  $d^- = \emptyset$  then
14:           $d^- \leftarrow d$ 
15:        if  $d^+ \neq \emptyset$  and  $d^- \neq \emptyset$  then
16:          break  $\triangleright$  Early exit: both examples identified
17:        if  $d^+ \neq \emptyset$  and  $d^- \neq \emptyset$  then
18:           $\mathcal{D} \leftarrow \mathcal{D} \cup \{(q_{\text{text}}, d^+, d^-)\}$ 
19: return  $\mathcal{D}$ 

```

candidate set $\mathcal{C}_q^{K'}$ is first sorted by $s_\theta(q_{\text{text}}, \cdot)$ in descending order (Line 3), so that the most confidently ranked documents are evaluated first. The algorithm then iterates over mini-batches of B_m candidates in that order (Lines 5–16). For each candidate d , a cache key is formed from its document identifier and position in the retrieved pool (Line 8); if the key is absent from the persistent cache Ψ , the frozen VLM $\mathcal{G}$ is queried with (v, q, d) and the predicted answer is stored (Lines 9–10). If the key is already cached, the stored prediction is reused instead of prompting the VLM again. If the prediction matches a_{gold} and no positive has been assigned yet, d is taken as d^+ (Lines 11–12). If it does not match and no negative has been assigned yet, d is taken as d^- (Lines 13–14). Once both d^+ and d^- are identified, an *early exit* terminates the remaining mini-batches immediately (Lines 15–16), avoiding superfluous VLM calls. A pair is added to $\mathcal{D}$ only when both d^+ and d^- are found (Lines 17–18); queries for which the pool contains no correct or no incorrect prediction are silently skipped.

2) *Controlled Baselines:* Algorithm 3 defines the three controlled baselines. All baselines use the same Stage 1 candidate pools, reranker architecture, LoRA training procedure, and alignment losses as the proposed method. The only difference is how d^+ and d^- are defined. By comparing the proposed method with these baselines, we can assess whether answer-utility supervision is the key source of the gains.

The **random baseline** (Lines 4–5) draws d^+ and d^- uniformly at random from $\mathcal{C}_q^{K'}$ without replacement. It uses neither reranker scores nor VLM feedback. This baseline therefore measures how the alignment losses behave when the preference labels contain no meaningful supervision and provides a noise-sensitivity reference. The **first baseline** (Lines 6–8) relies only on the current reranker ordering. It selects the highest-scored candidate as d^+ and the bottom-ranked candidate as

d^- , without consulting the VLM. This strategy tests whether simply reinforcing the reranker’s existing semantic ranking is sufficient to improve performance. Because it never prompts the VLM, the selected positive may be semantically relevant without helping the generator answer correctly.

The **REPLUG-style likelihood baseline (REPLUG-L)** (Lines 9–15) uses the VLM differently from our generator strategy. Rather than asking the VLM to produce the constrained answer output, it scores every candidate with a REPLUG-style LM-likelihood signal: the log-likelihood of a_{gold} conditioned on (v, q, d) , where a_{gold} is provided token-by-token as input rather than being generated:

$$\ell(d) = \frac{1}{|a_{\text{gold}}|} \sum_{t=1}^{|a_{\text{gold}}|} \log p_{\mathcal{G}}(a_{\text{gold},t} \mid v, q, d, a_{\text{gold},<t}). \quad (11)$$

Here, $p_{\mathcal{G}}$ denotes the frozen VLM’s next-token probability, $|a_{\text{gold}}|$ is the number of target answer tokens, $a_{\text{gold},t}$ is the t -th token, and $a_{\text{gold},<t}$ denotes preceding target tokens. At each step t , the VLM $\mathcal{G}$ is conditioned on $(v, q, d, a_{\text{gold},<t})$, and the log probability assigned to $a_{\text{gold},t}$ is averaged across the target answer tokens to produce $\ell(d)$. This is a REPLUG-style ranking score rather than the full REPLUG LSR objective, because we use it only to choose d^+ and d^- instead of training a retriever with a document-level KL loss. A higher $\ell(d)$ means that $\mathcal{G}$ assigns greater probability to the target answer when conditioned on document d and the preceding ground-truth answer tokens. The highest-scoring document is selected as d^+ and the lowest-scoring document as d^- (Lines 14–15), so all K' candidates must be evaluated before selection and early exit is not possible. In contrast, our generator-guided strategy asks the VLM to produce the constrained answer output and checks whether it matches the ground-truth. Thus, a document may have a high $\ell(d)$ even if the VLM would fail to produce the correct constrained answer without those ground-truth answer tokens.

V. EXPERIMENTS

We first describe the implementation details, followed by the results and analysis.

A. Implementation Details

We apply one default hyperparameter configuration across both datasets, summarized in Table II; experiments that deviate from these defaults state the changed value explicitly. Stage 1 retrieves 100 candidates per query using the Qwen3-Embedding-8B dense encoder with a FAISS index. For Stage 2, the cross-encoder backbone is Alibaba-NLP/gte-reranker-modernbert-base. Preference mining uses the top 10 retrieved candidates by default ($K' = 10$), with re-mining disabled unless otherwise stated. We evaluate two frozen VLM generators, Qwen3-VL-4B-Instruct and Qwen3.5-2B, to assess whether the proposed training-signal strategy generalizes across generators with different capacities.

The reranker is fine-tuned with LoRA using rank $r = 16$, scaling factor $\alpha = 32$, and dropout 0.1. Optimization uses AdamW with a learning rate of 5×10^{-5} , batch size of 16,

Algorithm 3 Baseline Preference Mining

Require: Queries $\mathcal{Q}$, reranker E_{θ} , cap K' , method m

Require: If $m = \text{replug-l}$: frozen VLM $\mathcal{G}$, batch size B_m , cache Ψ

```

1:  $\mathcal{D} \leftarrow \emptyset$ 
2: for each  $(q, v, a_{\text{gold}}) \in \mathcal{Q}$  do
3:    $\mathcal{C}_q^{K'} \leftarrow \text{top-}K'(\mathcal{C}_q; s_{\theta}(q_{\text{text}}, \cdot))$   $\triangleright$  descending score
4:   if  $m = \text{random}$  then
5:      $d^+, d^- \sim \text{Uniform}(\mathcal{C}_q^{K'})$  without replacement
6:   if  $m = \text{first}$  then
7:      $d^+ \leftarrow \arg \max_{d \in \mathcal{C}_q^{K'}} s_{\theta}(q_{\text{text}}, d)$ 
8:      $d^- \leftarrow \arg \min_{d \in \mathcal{C}_q^{K'}} s_{\theta}(q_{\text{text}}, d)$ 
9:   if  $m = \text{replug-l}$  then
10:    for each  $d \in \mathcal{C}_q^{K'}$  in mini-batches of  $B_m$  do
11:       $\text{key}(d) \leftarrow (\text{doc-id}(d), \text{pos}(d))$ 
12:      if  $\text{key}(d) \notin \Psi$  then
13:         $\Psi[\text{key}(d)] \leftarrow \ell(d)$ 
14:       $d^+ \leftarrow \arg \max_{d \in \mathcal{C}_q^{K'}} \Psi[\text{key}(d)]$ 
15:       $d^- \leftarrow \arg \min_{d \in \mathcal{C}_q^{K'}} \Psi[\text{key}(d)]$ 
16:    if  $d^+ \neq \emptyset$  and  $d^- \neq \emptyset$  then
17:       $\mathcal{D} \leftarrow \mathcal{D} \cup \{(q_{\text{text}}, d^+, d^-)\}$ 
18: return  $\mathcal{D}$ 

```

TABLE II
DEFAULT HYPERPARAMETER CONFIGURATION FOR VQA-X AND A-OKVQA

Category	Hyperparameter	Value
Architecture	VLM Generator	Qwen3-VL-4B-Instruct, Qwen3.5-2B
	Sentence Transformer	Qwen3-Embedding-8B
	Reranker Backbone	GTE-ModernBERT reranker
	Trainable Module	LoRA-adapted reranker
LoRA Adapter	Rank (r)	16
	Alpha (α)	32
	Dropout	0.1
Optimization	Optimizer	AdamW
	Learning Rate	5×10^{-5}
	Training Epochs	2
	Batch Size	16
Objective	DPO Beta (β)	0.1
Mining	Stage 2 Pool Size	$K' = 10$ of 100 candidates
	Mining Batch Size	16
	Re-mining Frequency	0 (disabled)
	REPLUG-L Pool Size	$K' = 10$ of 100 candidates

and 2 epochs unless otherwise stated. For Pairwise DPO, the inverse-temperature parameter is set to $\beta = 0.1$.

We do not apply periodic re-mining in the main comparison (`remining_every=0`); re-mining frequency is studied separately in Table VII. The REPLUG-style likelihood baseline scores all 10 candidates by ground-truth-token log-likelihood.

Stage 1 and Stage 2 are split-aware: the retrieval index is built from training-split documents only, while Stage 2 loads the selected dataset split to recover images and labels for the candidate question identifiers. A-OKVQA is evaluated as a multiple-choice task: the prompt lists choices as A),

TABLE III

IMPACT OF TRAINING-SIGNAL CONSTRUCTION ACROSS ALIGNMENT LOSS FUNCTIONS (2 EPOCHS, GENERATOR: QWEN3-VL-4B-INSTRUCT, CANDIDATE POOL: TOP 10). RESULTS REPORTED AS MEAN $\pm$ STD ACROSS 3 RANDOM SEEDS.

Alignment Loss	Training Signal	VQA-X (%)	A-OKVQA (%)
Base	No training	94.75	85.04
Contrastive (Triplet)	First	94.72 $\pm$ 0.22	85.07 $\pm$ 0.11
	Random	92.96 $\pm$ 0.14	82.77 $\pm$ 0.72
	REPLUG-L	95.85 $\pm$ 0.10	82.03 $\pm$ 0.28
	Generator	96.27$\pm$0.03	87.00$\pm$0.13
Pairwise DPO	First	94.82 $\pm$ 0.04	85.15 $\pm$ 0.05
	Random	93.84 $\pm$ 0.82	83.63 $\pm$ 1.16
	REPLUG-L	95.79 $\pm$ 0.09	82.40 $\pm$ 0.27
	Generator	96.11$\pm$0.04	87.16$\pm$0.03
SFT	First	94.75 $\pm$ 0.05	85.16 $\pm$ 0.02
	Random	93.35 $\pm$ 0.37	80.95 $\pm$ 0.72
	REPLUG-L	95.69 $\pm$ 0.04	82.65 $\pm$ 0.14
	Generator	95.73$\pm$0.01	85.75$\pm$0.08

TABLE IV

IMPACT OF TRAINING-SIGNAL CONSTRUCTION ACROSS ALIGNMENT LOSS FUNCTIONS (2 EPOCHS, GENERATOR: QWEN3.5-2B, CANDIDATE POOL: TOP 10). RESULTS REPORTED AS MEAN $\pm$ STD ACROSS 3 RANDOM SEEDS.

Alignment Loss	Training Signal	VQA-X (%)	A-OKVQA (%)
Base	No training	90.28	79.83
Contrastive (Triplet)	First	90.24 $\pm$ 0.13	79.76 $\pm$ 0.11
	Random	89.13 $\pm$ 2.01	77.02 $\pm$ 1.37
	REPLUG-L	92.44 $\pm$ 0.30	76.98 $\pm$ 0.07
	Generator	94.82$\pm$0.11	84.67$\pm$0.21
Pairwise DPO	First	90.20 $\pm$ 0.18	79.99 $\pm$ 0.08
	Random	89.49 $\pm$ 0.78	79.50 $\pm$ 0.61
	REPLUG-L	92.76 $\pm$ 0.18	76.97 $\pm$ 0.11
	Generator	93.32$\pm$0.04	82.50$\pm$0.06
SFT	First	90.12 $\pm$ 0.16	79.94 $\pm$ 0.08
	Random	89.27 $\pm$ 0.71	76.72 $\pm$ 0.63
	REPLUG-L	92.13 $\pm$ 0.14	76.95 $\pm$ 0.12
	Generator	92.54$\pm$0.10	80.75$\pm$0.16

B), etc. Although the Qwen model cards and generation configs provide sampling-oriented default or recommended values, our evaluation uses deterministic greedy decoding by setting `do_sample=False` with `max_new_tokens=10`; temperature and top-*p* sampling are therefore not used. The output is parsed first as a leading valid answer letter, with a fallback regex over standalone letters. Accuracy is exact match against `correct_choice_idx`. VQA-X is evaluated on yes/no questions: the prompt instructs the VLM to answer strictly with “yes” or “no”, generation uses the same deterministic decoding settings, outputs are lowercased, simple markers such as *, _, and periods are removed before yes/no matching, and accuracy is the standard VQA soft score $\min(\# \text{matching annotators}/3, 1)$. During binary preference mining for VQA-X, a prediction is treated as correct when this VQA score reaches 1.

B. Results and Analysis

We evaluate our proposed framework across three dimensions: (i) the superiority of the generator-guided training signal over alternative mining strategies under all three alignment losses (Tables III and IV), (ii) the robustness of the generator strategy to the size of the Stage 1 candidate pool (Tables V and VI), and (iii) the impact of periodic preference re-mining on accuracy (Table VII).

We note that direct comparison with existing retriever-generator alignment systems is difficult because prior methods often target text-only tasks, use different retrieval corpora and generators, or require joint retriever-generator training or an additional trainable bridge module. We therefore adopt a controlled comparison in which the Stage 1 candidate pool, reranker architecture, frozen generator, training configuration, and alignment loss are held fixed, while only the strategy used to construct (d^+, d^-) is varied. Under this design, the First baseline tests whether reinforcing the reranker’s existing score order is sufficient; the Random baseline measures the behavior of pairwise training under uninformative preference labels; and the REPLUG-style likelihood baseline represents likelihood-based generator-aware supervision adapted from prior work [12]. Our proposed method differs by assigning preferences according to whether each candidate enables the frozen VLM to produce the correct constrained answer output.

All trained results are reported as mean $\pm$ standard deviation over three random seeds. The seeds vary the LoRA initialization, training-data order, and random pair sampling where applicable, while the frozen generators and deterministic decoding procedure remain unchanged. We therefore assess a strategy based not only on its mean accuracy, but also on whether its gains are consistent across datasets, generators, alignment losses, and random seeds. The following subsections analyze the three dimensions in sequence.

1) *Generator-Guided Mining Outperforms All Baselines:* Tables III and IV compare all training-signal strategies under three alignment losses for Qwen3-VL-4B-Instruct and Qwen3.5-2B, respectively, on VQA-X and A-OKVQA. The base cross-encoder selects the first document as the context and achieves 94.75% / 85.04% (Qwen3-VL-4B-Instruct) and 90.28% / 79.83% (Qwen3.5-2B), establishing the no-training reference that an effective strategy must surpass.

Our proposed generator strategy achieves the highest accuracy in the evaluated cells of both tables, regardless of alignment loss, generator, or dataset. With Qwen3-VL-4B-Instruct, it reaches 96.27% / 87.00% under Contrastive (Triplet) and 96.11% / 87.16% under Pairwise DPO, with SFT trailing at 95.73% / 85.75%. With the smaller Qwen3.5-2B, Contrastive training achieves the best result (94.82% / 84.67%), followed by Pairwise DPO (93.32% / 82.50%) and SFT (92.54% / 80.75%). The consistent advantage across both generators suggests that the improvement stems from the quality of the training signal rather than any capacity-specific effect.

The controlled baselines further clarify which supervision signal matters. The First strategy represents rank-order self-training: it reinforces the reranker’s own semantic ordering without answer-level feedback, and it often fails to improve over the untrained base (e.g., 94.72% / 85.07% vs. the base 94.75%

/ 85.04% for Qwen3-VL-4B-Instruct under Contrastive). The Random strategy tests whether gains arise merely from LoRA fine-tuning with pairwise objectives; its frequent degradation below the base (e.g., 92.96% on VQA-X for Qwen3-VL-4B-Instruct under Contrastive, and 89.13% for Qwen3.5-2B) shows that arbitrary pair construction can harm the reranker. The REPLUG-style likelihood baseline (REPLUG-L) is the closest prior-work-connected comparison, adapting likelihood-based generator-aware retrieval supervision. It improves VQA-X over the base across all configurations (e.g., 95.85%, 95.79%, and 95.69% for Qwen3-VL-4B-Instruct under the three losses), but it hurts A-OKVQA accuracy relative to the base in these experiments (e.g., 82.03%–82.65% vs. the base 85.04% for Qwen3-VL-4B-Instruct). This asymmetry suggests that ground-truth-token likelihood is an imperfect proxy for constrained-answer correctness: a high likelihood of a_{gold} when preceding answer tokens are supplied does not reliably indicate that $\mathcal{G}$ would produce the correct constrained answer.

Across the three alignment losses, Contrastive (Triplet) emerges as a strong default: it is consistently competitive and becomes the clear best-performing objective with Qwen3.5-2B, which motivates using it in the pool-sensitivity and remaining studies below. At the same time, Pairwise DPO reaches comparable accuracy with Qwen3-VL-4B-Instruct under the same generator-guided mining signal. Thus, the main finding is not that triplet loss universally dominates, but that answer-utility preference mining is the decisive factor, with triplet loss serving as a reliable and simple instantiation.

2) *Robustness to Stage 1 Candidate Pool Size*: Tables V and VI investigate how sensitive each training signal is to K' , the number of Stage 1 candidates used during preference mining, under Contrastive (Triplet) training at 2 epochs for Qwen3-VL-4B-Instruct and Qwen3.5-2B, respectively.

The results show that the generator strategy is stable across pool sizes, though the degree of stability differs by generator. For Qwen3-VL-4B-Instruct, VQA-X accuracy increases modestly from 95.88% at Top 2 to 96.27% at Top 10 (a range of 0.39 pp), and A-OKVQA grows from 86.22% to 87.00% (0.78 pp). For Qwen3.5-2B, the variation is more pronounced: VQA-X rises from 92.76% at Top 2 to 94.82% at Top 10 (2.06 pp), and A-OKVQA from 81.45% to 84.67% (3.22 pp), suggesting that Qwen3.5-2B, despite being smaller and from a different model generation, benefits more from a richer candidate pool when constructing preference pairs. Notably, our proposed generator strategy with the smallest pool of Top 2 already outperforms the evaluated baselines at their largest pool of Top 10 across both generators and datasets. This aligns with recent budget-aware RAG findings that the utility of retrieved chunks, not merely the number of chunks used, is central to effective retrieval augmentation [43]; in our setting, the quality of the training signal matters more than the quantity of candidates evaluated.

Further inspection shows that the Random strategy has the highest variance across pool sizes (e.g., 87.73%–89.13% on VQA-X for Qwen3.5-2B), because the quality of the sampled pair depends heavily on which documents happen to fall inside the pool. The First strategy is essentially insensitive to pool size, remaining near base-level performance regardless of K' , because it always selects the top-1 and bottom-1 of the ranked

TABLE V
SENSITIVITY TO STAGE 1 CANDIDATE POOL IN CONTRASTIVE (TRIPLET) TRAINING (2 EPOCHS, GENERATOR: QWEN3-VL-4B-INSTRUCT). RESULTS REPORTED AS MEAN $\pm$ STD ACROSS 3 RANDOM SEEDS.

Training Signal	Candidate Pool	VQA-X (%)	A-OKVQA (%)
Generator (Ours)	Top 2	95.88 $\pm$ 0.03	86.22 $\pm$ 0.09
	Top 4	96.06 $\pm$ 0.03	86.68 $\pm$ 0.06
	Top 6	96.16 $\pm$ 0.03	86.88 $\pm$ 0.05
	Top 8	96.18 $\pm$ 0.08	86.93 $\pm$ 0.06
	Top 10	96.27$\pm$0.03	87.00$\pm$0.13
First	Top 2	94.73 $\pm$ 0.04	85.05 $\pm$ 0.01
	Top 4	94.76 $\pm$ 0.09	85.05 $\pm$ 0.03
	Top 6	94.68 $\pm$ 0.07	85.03 $\pm$ 0.02
	Top 8	94.61 $\pm$ 0.18	85.06 $\pm$ 0.05
	Top 10	94.72 $\pm$ 0.22	85.07 $\pm$ 0.11
Random	Top 2	92.62 $\pm$ 1.05	82.24 $\pm$ 0.60
	Top 4	93.50 $\pm$ 0.85	82.50 $\pm$ 0.48
	Top 6	93.07 $\pm$ 1.31	84.15 $\pm$ 0.25
	Top 8	93.18 $\pm$ 0.42	82.61 $\pm$ 0.92
	Top 10	92.96 $\pm$ 0.14	82.77 $\pm$ 0.72
REPLUG-L	Top 2	95.52 $\pm$ 0.03	81.45 $\pm$ 0.20
	Top 4	95.75 $\pm$ 0.10	81.58 $\pm$ 0.38
	Top 6	95.86 $\pm$ 0.10	81.89 $\pm$ 0.11
	Top 8	95.83 $\pm$ 0.10	82.04 $\pm$ 0.09
	Top 10	95.85 $\pm$ 0.10	82.03 $\pm$ 0.28

list and the pool boundary only marginally affects the bottom candidate. REPLUG-L scores all K' candidates in the pool, so it naturally uses more information as the pool grows; even so, at Top 10 it remains below our proposed generator strategy by 0.42 pp on VQA-X and 4.97 pp on A-OKVQA for Qwen3-VL-4B-Instruct (95.85% vs. 96.27% and 82.03% vs. 87.00%). For Qwen3.5-2B the gap is even larger: 2.38 pp on VQA-X (92.44% vs. 94.82%) and 7.69 pp on A-OKVQA (76.98% vs. 84.67%). This suggests that REPLUG-L’s shortfall is not only a matter of how many candidates it evaluates, but also that its ground-truth-token likelihood signal can be a weaker proxy for downstream answer correctness than the direct binary judgment used by the generator strategy.

3) *Impact of Periodic Re-mining*: Table VII examines the effect of re-mining frequency on downstream accuracy over 4 training epochs using the generator strategy and Contrastive (Triplet) loss. Three configurations are compared: no re-mining (the preference dataset is fixed throughout training), re-mining twice (once at the start before training and once at mid-training), and re-mining four times (once per epoch).

Periodic re-mining generally enhances the performance compared with using a fixed preference dataset, but the benefit of increasing the re-mining frequency is model- and dataset-dependent. For Qwen3-VL-4B-Instruct on VQA-X, re-mining twice and four times are effectively tied within the reported seed variation (97.45 $\pm$ 0.09% vs. 97.44 $\pm$ 0.09%), indicating saturation rather than a meaningful gain from additional refreshes. In contrast, re-mining four times yields the best A-OKVQA accuracy for Qwen3-VL-4B-Instruct (89.58% vs. 87.00% without re-mining, +2.58 pp). For Qwen3.5-2B, re-mining four times produces the best results on both datasets (95.16% on VQA-X, +0.73 pp; 84.67% on A-OKVQA, +2.43

TABLE VI

SENSITIVITY TO STAGE 1 CANDIDATE POOL IN CONTRASTIVE (TRIPLET) TRAINING (2 EPOCHS, GENERATOR: QWEN3.5-2B). RESULTS REPORTED AS MEAN $\pm$ STD ACROSS 3 RANDOM SEEDS.

Training Signal	Candidate Pool	VQA-X (%)	A-OKVQA (%)
Generator (Ours)	Top 2	92.76 $\pm$ 0.02	81.45 $\pm$ 0.02
	Top 4	93.01 $\pm$ 0.05	81.87 $\pm$ 0.16
	Top 6	93.11 $\pm$ 0.06	82.02 $\pm$ 0.01
	Top 8	93.29 $\pm$ 0.04	81.97 $\pm$ 0.02
	Top 10	94.82$\pm$0.11	84.67$\pm$0.21
First	Top 2	90.28 $\pm$ 0.05	79.85 $\pm$ 0.03
	Top 4	90.25 $\pm$ 0.12	79.76 $\pm$ 0.06
	Top 6	90.24 $\pm$ 0.05	79.85 $\pm$ 0.01
	Top 8	90.33 $\pm$ 0.24	79.74 $\pm$ 0.11
	Top 10	90.24 $\pm$ 0.13	79.76 $\pm$ 0.11
Random	Top 2	87.73 $\pm$ 0.85	78.18 $\pm$ 0.69
	Top 4	88.94 $\pm$ 1.64	76.64 $\pm$ 1.12
	Top 6	88.34 $\pm$ 1.00	77.76 $\pm$ 0.90
	Top 8	88.80 $\pm$ 1.77	78.30 $\pm$ 1.12
	Top 10	89.13 $\pm$ 2.01	77.02 $\pm$ 1.37
REPLUG-L	Top 2	91.58 $\pm$ 0.27	76.89 $\pm$ 0.22
	Top 4	91.92 $\pm$ 0.23	76.91 $\pm$ 0.25
	Top 6	92.17 $\pm$ 0.12	77.13 $\pm$ 0.06
	Top 8	92.33 $\pm$ 0.20	77.10 $\pm$ 0.07
	Top 10	92.44 $\pm$ 0.30	76.98 $\pm$ 0.07

TABLE VII

IMPACT OF RE-MINING FREQUENCY ON DOWNSTREAM ACCURACY (CANDIDATE POOL: TOP 10). RESULTS REPORTED AS MEAN $\pm$ STD ACROSS 3 RANDOM SEEDS.

Generator	# of Re-mines	VQA-X (%)	A-OKVQA (%)
Qwen3-VL-4B-Instruct	0	96.27 $\pm$ 0.03	87.00 $\pm$ 0.13
	2	97.45$\pm$0.09	89.01 $\pm$ 0.23
	4	97.44$\pm$0.09	89.58$\pm$0.04
Qwen3.5-2B	0	94.43 $\pm$ 0.09	82.24 $\pm$ 0.14
	2	94.82 $\pm$ 0.11	84.47 $\pm$ 0.30
	4	95.16$\pm$0.13	84.67$\pm$0.21

pp over no re-mining). These results suggest that refreshing stale preference pairs can improve training, while the marginal gains from more frequent refreshes can saturate once the reranker stabilizes. This is also evident from the fact that the incremental gain from four rather than two re-mines is small and dataset-dependent. For Qwen3-VL-4B-Instruct, VQA-X is effectively tied (97.44% vs. 97.45%), while four re-mines add 0.57 pp on A-OKVQA (89.58% vs. 89.01%). For Qwen3.5-2B, four re-mines give modest gains over two on both datasets (+0.34 pp VQA-X, +0.20 pp A-OKVQA). Re-mining also adds cost because each refresh must re-rank the mining queries, rebuild pairs from the current top- K' subset, and evaluate any newly reached uncached query-document pairs. Thus, re-mining twice offers a favorable accuracy-cost balance, while additional refreshes should be treated as optional.

VI. LIMITATIONS

This work presents several opportunities for further development. First, although the framework eliminates the need for human document-level relevance annotations, preference

mining still relies on answer-labeled training examples to determine whether a candidate document helps the frozen generator produce the correct answer. Future work could reduce this dependence by using self-supervised consistency signals, confidence-based pseudo-labels, or other forms of weak supervision.

Second, preference mining introduces additional computational cost because the VLM must evaluate multiple candidate documents for each query, and periodic re-mining repeats this process as the reranker evolves. This overhead can be further reduced through more selective candidate evaluation, adaptive re-mining schedules, and lightweight surrogate models that approximate the generator’s utility judgments.

Third, the learned reranker is intentionally generator-specific, as it is optimized to select evidence that improves the frozen VLM’s behavior during training rather than to learn a universal notion of document relevance. Future work could improve transferability by incorporating preferences from multiple generators or by developing generator-agnostic utility representations that require only limited adaptation to a new VLM.

Finally, the current pairwise training procedure uses only queries whose candidate pools contain at least one helpful and one unhelpful document. While this requirement produces clean positive–negative pairs, it excludes queries for which all candidates yield the same outcome and uses only a limited portion of the available utility information. This limitation could be addressed through graded, listwise, or multi-document utility supervision that uses more of the information available in the candidate pool.

VII. CONCLUSION

This paper introduced a *generator-in-the-loop alignment framework* for multimodal RAG. Our proposed two-stage framework separates modality bridging from answer-utility alignment: Stage 1 uses a frozen VLM to generate a HyDE-style text query for dense recall, while Stage 2 trains a LoRA-adapted cross-encoder reranker from answer-supervised preference pairs mined with the same frozen generator. We demonstrated that across VQA-X and A-OKVQA datasets, generator-guided preferences improve over rank-order, random, and REPLUG-style likelihood baselines under multiple alignment losses, suggesting that downstream answer behavior is a useful signal for reranker alignment. The results also showed that periodic re-mining can improve accuracy by refreshing preference pairs as the reranker changes, although the marginal value of more frequent refreshes depends on the model and dataset. Overall, the findings support a generator-centered view of RAG reranking: the most useful evidence is not the passage semantically closest to the query, but the passage that helps the deployed generator answer correctly.

VIII. ACKNOWLEDGMENTS

Generative AI tools, including ChatGPT (OpenAI) [44] and Claude (Anthropic) [45], were used to assist with the preparation and refinement of manuscript text, including sentence-level language improvement. All technical content was developed by the authors, and the full manuscript was carefully

reviewed and verified by the authors to ensure accuracy and correctness.

REFERENCES

- [1] L. Yuan, D.-J. Han, S. Wang, and C. Brinton, "Local-cloud inference offloading for llms in multi-modal, multi-task, multi-dialogue settings," in *Proceedings of the Twenty-sixth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, 2025, pp. 201–210.
- [2] W. Fang, D.-J. Han, L. Yuan, E. Chen, and C. Brinton, "Bridging on-device and cloud llms for collaborative reasoning: A unified methodology for local routing and post-training," in *Forty-third International Conference on Machine Learning*, 2026.
- [3] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin *et al.*, "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions," *ACM transactions on information systems*, vol. 43, no. 2, pp. 1–55, 2025.
- [4] 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," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.
- [5] S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. B. Van Den Driessche, J.-B. Lespiau, B. Damoc, A. Clark *et al.*, "Improving language models by retrieving from trillions of tokens," in *International conference on machine learning*. PMLR, 2022, pp. 2206–2240.
- [6] T. R. McIntosh, T. Liu, T. Susnjak, P. Watters, A. Ng, and M. N. Halgamuge, "A culturally sensitive test to evaluate nuanced GPT hallucination," *IEEE Transactions on Artificial Intelligence*, vol. 5, no. 6, pp. 2739–2751, 2023.
- [7] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, H. Wang, and H. Wang, "Retrieval-augmented generation for large language models: A survey," *arXiv preprint arXiv:2312.10997*, vol. 2, no. 1, 2023.
- [8] O. Khattab, A. Singhvi, P. Maheshwari, Z. Zhang, K. Santhanam, S. Vardhamanan, S. Haq, A. Sharma, T. T. Joshi, H. Moazam *et al.*, "DSPy: Compiling declarative language model calls into self-improving pipelines," *arXiv preprint arXiv:2310.03714*, 2023.
- [9] X. Ma, Y. Gong, P. He, H. Zhao, and N. Duan, "Query rewriting in retrieval-augmented large language models," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023, pp. 5303–5315.
- [10] F. Cuconasu, G. Trappolini, F. Siciliano, S. Filice, C. Campagnano, Y. Maarek, N. Tonello, and F. Silvestri, "The power of noise: Redefining retrieval for RAG systems," in *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2024, pp. 719–729.
- [11] G. Izacard, P. Lewis, M. Lomeli, L. Hosseini, F. Petroni, T. Schick, J. Dwivedi-Yu, A. Joulin, S. Riedel, and E. Grave, "Atlas: Few-shot learning with retrieval augmented language models," *Journal of Machine Learning Research*, vol. 24, no. 251, pp. 1–43, 2023.
- [12] W. Shi, S. Min, M. Yasunaga, M. Seo, R. James, M. Lewis, L. Zettlemoyer, and W.-t. Yih, "REPLUG: Retrieval-augmented black-box language models," in *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 2024, pp. 8371–8384.
- [13] Y. Zhang, M. Li, D. Long, X. Zhang, H. Lin, B. Yang, P. Xie, A. Yang, D. Liu, J. Lin *et al.*, "Qwen3 embedding: Advancing text embedding and reranking through foundation models," *arXiv preprint arXiv:2506.05176*, 2025.
- [14] L. Wang, N. Yang, and F. Wei, "Learning to retrieve in-context examples for large language models," in *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024, pp. 1752–1767.
- [15] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang, "Lost in the middle: How language models use long contexts," *Transactions of the association for computational linguistics*, vol. 12, pp. 157–173, 2024.
- [16] O. Yoran, T. Wolfson, O. Ram, and J. Berant, "Making retrieval-augmented language models robust to irrelevant context," in *International Conference on Learning Representations*, vol. 2024, 2024, pp. 29 862–29 883.
- [17] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang, "Retrieval augmented language model pre-training," in *International conference on machine learning*. PMLR, 2020, pp. 3929–3938.
- [18] S. Gupta, R. Ranjan, and S. N. Singh, "A comprehensive survey of retrieval-augmented generation (RAG): Evolution, current landscape and future directions," *arXiv preprint arXiv:2410.12837*, 2024.
- [19] V. Karpukhin, B. Oguz, S. Min, P. S. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih, "Dense passage retrieval for open-domain question answering," in *EMNLP (1)*, 2020, pp. 6769–6781.
- [20] K. Shuster, S. Poff, M. Chen, D. Kiela, and J. Weston, "Retrieval augmentation reduces hallucination in conversation," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, 2021, pp. 3784–3803.
- [21] W. Yu, H. Zhang, X. Pan, P. Cao, K. Ma, J. Li, H. Wang, and D. Yu, "Chain-of-note: Enhancing robustness in retrieval-augmented language models," in *Proceedings of the 2024 conference on empirical methods in natural language processing*, 2024, pp. 14 672–14 685.
- [22] Z. Xu, G. Pang, X. Chen, M. Ding, B. Vucetic, Y. Li, and Z. Li, "Maq-retrieval: Multi-aspect queries retrieval for large language models," *IEEE Transactions on Artificial Intelligence*, 2026.
- [23] L. Gao, X. Ma, J. Lin, and J. Callan, "Precise zero-shot dense retrieval without relevance labels," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2023, pp. 1762–1777.
- [24] R. Nogueira and K. Cho, "Passage re-ranking with bert," *arXiv preprint arXiv:1901.04085*, 2019.
- [25] W. Sun, L. Yan, X. Ma, S. Wang, P. Ren, Z. Chen, D. Yin, and Z. Ren, "Is ChatGPT good at search? investigating large language models as re-ranking agents," in *Proceedings of the 2023 conference on empirical methods in natural language processing*, 2023, pp. 14 918–14 937.
- [26] A. J. Oche, A. G. Folashade, T. Ghosal, and A. Biswas, "A systematic review of key retrieval-augmented generation (RAG) systems: Progress, gaps, and future directions," *arXiv preprint arXiv:2507.18910*, 2025.
- [27] Z. Jiang, F. F. Xu, L. Gao, Z. Sun, Q. Liu, J. Dwivedi-Yu, Y. Yang, J. Callan, and G. Neubig, "Active retrieval augmented generation," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023, pp. 7969–7992.
- [28] S.-Q. Yan, J.-C. Gu, Y. Zhu, and Z.-H. Ling, "Corrective retrieval augmented generation," *arXiv preprint arXiv:2401.15884*, 2024.
- [29] D. Zhu, X. Shen, X. Li, M. Elhoseiny *et al.*, "Minigt-4: Enhancing vision-language understanding with advanced large language models," in *International Conference on Learning Representations*, vol. 2024, 2024, pp. 18 378–18 394.
- [30] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," *Advances in Neural Information Processing Systems*, vol. 36, pp. 34 892–34 916, 2023.
- [31] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.
- [32] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds *et al.*, "Flamingo: a visual language model for few-shot learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 23 716–23 736, 2022.
- [33] J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang *et al.*, "Qwen technical report," *arXiv preprint arXiv:2309.16609*, 2023.
- [34] M. Yasunaga, A. Aghajanyan, W. Shi, R. James, J. Leskovec, P. Liang, M. Lewis, L. Zettlemoyer, and W.-t. Yih, "Retrieval-augmented multi-modal language modeling," *arXiv preprint arXiv:2211.12561*, 2022.
- [35] M. Faysse, H. Sibille, T. Wu, B. Omrani, G. Viaud, C. Hudelot, and P. Colombo, "Colpali: Efficient document retrieval with vision language models," in *International Conference on Learning Representations*, vol. 2025, 2025, pp. 61 424–61 449.
- [36] B. He, N. Chen, X. He, L. Yan, Z. Wei, J. Luo, and Z.-H. Ling, "Retrieving, rethinking and revising: The chain-of-verification can improve retrieval augmented generation," in *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024, pp. 10 371–10 393.
- [37] O. Ram, Y. Levine, I. Dalmedigos, D. Muhlgeay, A. Shashua, K. Leyton-Brown, and Y. Shoham, "In-context retrieval-augmented language models," *Transactions of the Association for Computational Linguistics*, vol. 11, pp. 1316–1331, 2023.
- [38] Z. Ke, W. Kong, C. Li, M. Zhang, Q. Mei, and M. Bendersky, "Bridging the preference gap between retrievers and LLMs," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024, pp. 10 438–10 451.
- [39] Y. Liu, D. Iyer, Y. Xu, S. Wang, R. Xu, and C. Zhu, "G-Eval: NLG evaluation using GPT-4 with better human alignment," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023, pp. 2511–2522.

- [40] Y. Wang, Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi, and H. Hajishirzi, "Self-Instruct: Aligning language models with self-generated instructions," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2023, pp. 13 484–13 508.
- [41] B. Warner, A. Chaffin, B. Clavié, O. Weller, O. Hallström, S. Taghadouini, A. Gallagher, R. Biswas, F. Ladhak, T. Aarsen *et al.*, "Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference," in *Proceedings of the 63rd annual meeting of the association for computational linguistics (volume 1: Long papers)*, 2025, pp. 2526–2547.
- [42] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," *Advances in Neural Information Processing Systems*, vol. 36, pp. 53 728–53 741, 2023.
- [43] S. Al-Maliki, A. Gharaibeh, M. Rahouti, M. R. Amin, M. Abdallah, J. Qadir, and A. Al-Fuqaha, "Budget-constrained online retrieval-augmented generation: The chunk-as-a-service model," *IEEE Transactions on Artificial Intelligence*, 2026.
- [44] OpenAI, "ChatGPT," [Online]. Available: <https://chatgpt.com/>.
- [45] Anthropic, "Claude," [Online]. Available: <https://claude.ai/>.