AUTOREGRESSIVE RETRIEVER: IMPROVING QUERY UNDERSTANDING FROM ITEM FEEDBACK FOR UNIVERSAL MULTIMODAL RETRIEVAL

Anonymous authors
Paper under double-blind review

ABSTRACT

Universal multimodal retrieval typically encodes a query once and ranks independently indexed items by embedding similarity. This design supports efficient search, but leaves the query representation unchanged even when retrieved items could help clarify the information need. We introduce the AutoRegressive Retriever (ARR), a multimodal retrieval model that learns both to select informative items and to use their content to refine subsequent retrieval. ARR alternates between retrieving an item and updating the query embedding, then uses the final embedding to rank the collection. Supervised fine-tuning teaches the encoder to use feedback through stepwise contrastive supervision. Reinforcement learning treats feedback items as actions and optimizes their selection using the final reciprocal rank of a relevant item. A query-side adapter enables this optimization against a fixed item index. ARR demonstrates strong retrieval performance on both in-domain and zero-shot benchmarks, outperforming the compared baselines on average. Further analyses show that feedback improves retrieval at inference time and that training with feedback also improves the initial query embedding, before any item is observed.

1 INTRODUCTION

Retrieval systems increasingly need to search across text, images, and their combinations, supporting a wide range of applications, including retrieval-augmented generation (Sun et al., 2026a; Zhang et al., 2026c), multimedia platforms (Zhao et al., 2026), and e-commerce search (Zhang et al., 2026b; Sun et al., 2026a). Universal multimodal retrieval (UMR) addresses this need with a single model that maps heterogeneous queries and items into a shared embedding space (Wei et al., 2024; Liu et al., 2025; Lin et al., 2025). In the standard embedding-based approach, queries and items are encoded independently and matched by similarity. This separation enables item embeddings to be computed in advance and reused across queries, making large collections searchable through a precomputed index.

However, this separation also limits a query's ability to leverage the collection being searched. A query may omit important details, express its intent only implicitly, or contain visual and textual cues of unequal importance. An independently computed query embedding must resolve such ambiguities before observing any candidate items. Although contrastive training (He et al., 2020; Chen et al., 2020) learns to align queries with relevant items across the training data, at inference time the query representation remains fixed and cannot adapt to the candidates encountered during retrieval. We refer to this lack of candidate-conditioned adaptation as the *query–item gap*.

Recent works improve query interpretation through stronger multimodal large language models (MLLMs) (Jiang et al., 2024; Li et al., 2026a), richer representations (Sun et al., 2026b; Wu et al., 2026), and reasoning before embedding (Cui et al., 2026; Zhang et al., 2026a; Hao et al., 2026). These approaches extract more information from the input, but do not by themselves determine how a retriever should respond to newly encountered items. Retrieval feedback offers a complementary direction. Prior methods either directly refine dense query representations using retrieved documents (Yu et al., 2021), or rewrite queries and re-encode them (Tu et al., 2026; Bigdeli et al., 2026). Yet, the retrieved items are typically processed by an auxiliary refinement mechanism rather than by

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

the retriever itself. We instead ask: **can the retriever learn to use the items it actually encounters to decide how to search better?**

We introduce the **A**uto**R**egressive **R**etriever (ARR), a multimodal retrieval model that alternates between retrieving an item and updating the query embedding (Figure 1). We refer to this iterative process as *autoregressive retrieval*. At each intermediate step, ARR selects an item by embedding similarity, appends its multimodal content to the query history, and encodes the expanded context for the next search. After a fixed number of steps, the final embedding ranks the collection. The process is autoregressive over retrieved items: each selection depends on the original query and earlier feedback. Item embeddings remain independently indexable, preserving the standard embedding-based search interface.

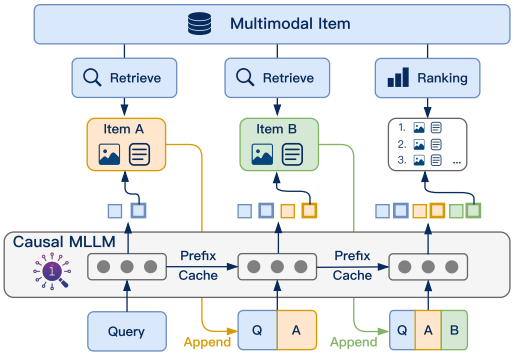

Figure 1: Autoregressive retrieval with ARR. Retrieved items provide context for successive query embeddings; the final embedding ranks the collection.

Learning this process requires distinguishing an item's relevance as an answer from its usefulness as feedback. An item that is not the target may still reveal information that helps retrieve the target later; conversely, a plausible match may distract the encoder. The model receives item content without a relevance label and must learn which evidence to use. We train ARR in two stages to address these challenges. Supervised fine-tuning (SFT) first teaches the encoder to interpret feedback using stepwise contrastive supervision on offline trajectories. Reinforcement learning (RL) then learns feedback selection from trajectories sampled by the current policy, using final retrieval quality as the reward. Its actions are retrieved items rather than generated text tokens. A query-side adapter learns this policy against a fixed item index.

Experiments on M-BEIR and seven zero-shot benchmarks show benefits from both feedback-conditioned inference and feedback-based training. ARR consistently achieves the highest average performance among all compared methods across both benchmarks. Training with feedback also improves first-step retrieval, indicating that its benefits extend to inference without feedback.

Our contributions are threefold:

- We introduce ARR, a multimodal retriever that performs autoregressive retrieval by conditioning query embeddings on a sequence of retrieved items while retaining independently indexed item embeddings.
- We develop a two-stage training procedure that combines supervised feedback interpretation with item-selection policy optimization, linking intermediate retrieval actions to the final ranking outcome.
- We evaluate ARR across in-domain and zero-shot tasks, and use ablation studies to distinguish the benefits of feedback at inference time from improvements learned during training.

2 Related Work

Universal multimodal retrieval. UMR brings diverse query–item modality configurations into a shared retrieval model (Wei et al., 2024). Contrastive models such as CLIP (Radford et al., 2021) provide a foundation for cross-modal alignment, while MLLM-based encoders extend this approach to more flexible input combinations (Jiang et al., 2024; Li et al., 2026a). MM-Embed (Lin et al., 2025) uses modality-aware hard-negative mining to improve multimodal retrieval. LamRA (Liu et al., 2025) combines a retrieval model with a separate reranking component trained with pointwise and listwise objectives. BToks (Sun et al., 2026b) and LaME (Wu et al., 2026) investigate bottleneck representations and latent computation for stronger embeddings. These directions improve representation learning or candidate ranking. ARR instead focuses on how retrieved content can inform subsequent query representations and how feedback items should be selected.

Retrieval feedback. Using retrieved results to refine a query is an established idea in information retrieval. ANCE-PRF (Yu et al., 2021) learns a dense query encoder that consumes the original query and top-ranked documents while keeping the document index fixed. More recent approaches use language models to turn retrieval evidence into query reformulations: GPRF (Tu et al., 2026) learns retrieval-oriented rewrites, and ADORE (Bigdeli et al., 2026) iteratively expands queries using assessments of retrieved content. ARR shares the motivation of adapting a query to the collection. Its focus is on learning a sequence of multimodal feedback actions within the embedding retriever: the encoder both consumes retrieved items and defines the policy that selects the next item, with selection trained for its effect on the final ranking.

Reasoning and reinforcement learning for retrieval. Reasoning-enhanced retrieval enriches inputs before constructing embeddings. TTE (Cui et al., 2026) uses generated context to improve multimodal representations, while TWN (Zhang et al., 2026a) and TRACE (Hao et al., 2026) study when to invoke explicit reasoning. Retrieval-oriented query rewriting provides another way to expose information that is implicit in the input (Zhu et al., 2024). RL has also been used to align generation with retrieval outcomes: GRAPE (Zhang et al., 2025) optimizes query rewriting using ranking rewards, and ELVA (Liu et al., 2026) uses ranking and margin rewards to train representations produced after textual synthesis. ARR differs in its action space and feedback loop. It samples item identities directly from embedding similarities, observes the selected items, and optimizes these selections for downstream retrieval quality. Thus, its autoregressive sequence consists of retrieved items rather than a generated reasoning or rewriting trace.

3 PRELIMINARIES

Task formulation. Let $\mathcal{D}$ denote an item corpus and q a query. Both q and an item $d \in \mathcal{D}$ may contain text, images, or a combination of the two. A query may also specify an instruction describing the desired relevance relation. UMR uses a shared retrieval model across these input configurations to rank items according to their relevance to q. We write a training example as (q, d^+), where d^+ is an annotated relevant item.

MLLM-based encoder. An MLLM-based encoder maps queries and items into a shared embedding space. Text and images are converted into token representations and assembled using an input template. The MLLM processes the resulting sequence, and an embedding readout extracts a fixed-dimensional vector from its hidden states. After ℓ_2 normalization, the embeddings and their similarity are

$$\mathbf{z}_q = f_\theta(q), \qquad \mathbf{v}_d = f_\theta(d), \qquad s_\theta(q, d) = \mathbf{z}_q^\top \mathbf{v}_d, \tag{1}$$

where f_θ denotes the encoder and $\mathbf{z}_q, \mathbf{v}_d \in \mathbb{R}^p$. Their inner product measures cosine similarity in the shared embedding space.

Retrieval objective. The retriever is trained with a query-to-item contrastive objective. For each query q, let $\mathcal{B}_q$ contain its annotated positive d^+ and negative items. We use the InfoNCE loss (He et al., 2020),

$$\ell_{\mathrm{NCE}}\big(\mathbf{z}, d^+, \mathcal{B}_q; \{\mathbf{v}_d\}\big) = -\log \frac{\exp(\mathbf{z}^\top \mathbf{v}_{d^+}/\tau)}{\sum_{d \in \mathcal{B}_q} \exp(\mathbf{z}^\top \mathbf{v}_d/\tau)}, \tag{2}$$

where $\tau > 0$ is the temperature. Minimizing this loss increases the positive item's similarity relative to the negatives in $\mathcal{B}_q$.

4 METHOD

ARR performs autoregressive retrieval by conditioning query representations on retrieved content while preserving independently encoded item embeddings. We first define this retrieval process (Section 4.1), then describe SFT for interpreting feedback (Section 4.2) and RL for selecting feedback that improves the final ranking (Section 4.3).

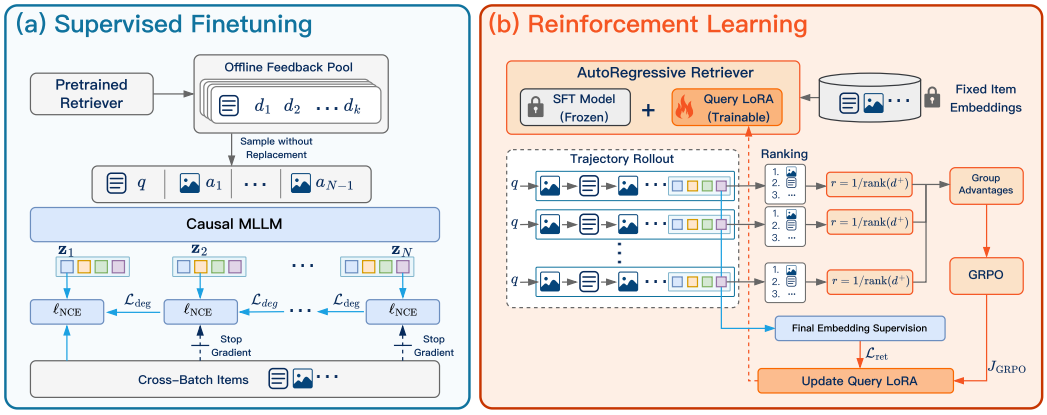

Figure 2: Overview of ARR training. **(a)** SFT uses offline feedback trajectories, stepwise contrastive supervision, and a degradation penalty. Only the initial-step loss backpropagates through standalone item embeddings. **(b)** RL freezes the SFT backbone and item index while training a query-side LoRA adapter. Terminal reciprocal-rank rewards supervise feedback selection through GRPO, complemented by contrastive supervision of the final embedding.

4.1 Autoregressive Retrieval

ARR uses N query states with $N-1$ feedback selections. It first encodes the original query, then repeatedly retrieves an item and appends its content and available metadata to the query context. The encoder processes this context to produce the next query embedding. The Nth embedding determines the final retrieval results. No relevance label is included in the feedback.

At step t, the model observes the original query and all previously selected feedback items. Let $a_t \in \mathcal{D}$ denote the item selected at step t. We define the state and query embedding as

$$h_t = (q, a_1, \ldots, a_{t-1}), \quad \mathbf{z}_t = f_\theta\big(\mathcal{T}(h_t)\big), \quad t = 1, \ldots, N, \tag{3}$$

where $\mathcal{T}$ serializes the history using the MLLM's multi-turn conversation template, illustrated on the right. The <pad> token serves as the representation token.

Query Template

SYSTEM: <instruction>
USER: <query>
ASSISTANT: <pad>
USER: <item>
ASSISTANT: <pad>
...

A history mask prevents repeated selection of the same item. Let $\mathcal{C}(q) \subseteq \mathcal{D}$ denote the feedback candidate pool for query q, with at least $N-1$ items, and let $H_t = \{a_1, \ldots, a_{t-1}\}$ be the set of previously selected items. For $t < N$, embedding similarities define a policy over the remaining candidates:

$$\pi_\theta(d \mid h_t; \mathcal{C}(q)) = \begin{cases} \dfrac{\exp(\mathbf{z}_t^\top \mathbf{v}_d/\tau)}{\sum_{d' \in \mathcal{C}(q) \setminus H_t} \exp(\mathbf{z}_t^\top \mathbf{v}_{d'}/\tau)}, & d \in \mathcal{C}(q) \setminus H_t, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

The probability of a sampled feedback trajectory $\xi = (a_1, \ldots, a_{N-1})$ factorizes as

$$P_\theta(\xi \mid q; \mathcal{C}(q)) = \prod_{t=1}^{N-1} \pi_\theta(a_t \mid h_t; \mathcal{C}(q)). \tag{5}$$

This factorization makes retrieval autoregressive over item identities: earlier selections affect both the next query embedding and subsequent selection probabilities. After incorporating $N-1$ feedback items, ARR uses the final query representation $\mathbf{z}_N$ to retrieve the top-k items:

$$\text{Retrieve}_k(q, \xi) = \text{Top-k}_{d \in \mathcal{D}}\, \mathbf{z}_N^\top \mathbf{v}_d.$$

The history mask is applied only to feedback selection; previously selected items remain eligible for retrieval at each step.

4.2 SUPERVISED FINE-TUNING

SFT teaches the encoder to use retrieved content while preserving the original query intent (Figure 2a). To avoid repeatedly searching the corpus during this stage, we construct an offline feedback pool using the pretrained encoder with parameters θ_0:

$$\mathcal{C}_{\mathrm{SFT}}(q) = \text{Top-}\, K_{d\in\mathcal{D}} f_{\theta_0}(q)^\top f_{\theta_0}(d). \tag{6}$$

Here, $K \geq N-1$. During training, we randomly sample an ordered sequence of $N-1$ items without replacement from this pool and construct the N query states in Eq. 3. We neither inject a missing positive into the feedback pool nor filter feedback by relevance labels. The contrastive set $\mathcal{B}_q$, which contains the annotated positive for supervision, is separate from this feedback pool.

We supervise every query embedding with the relevant item d^+. Let $\mathrm{sg}(\cdot)$ denote stop-gradient. We use the same contrastive set $\mathcal{B}_q$ across the states of a trajectory and define

$$\ell_t = \ell_{\mathrm{NCE}}\big(\mathbf{z}_t, d^+, \mathcal{B}_q; \{\widetilde{\mathbf{v}}_d^{(t)}\}\big), \qquad \widetilde{\mathbf{v}}_d^{(t)} = \begin{cases} \mathbf{v}_d, & t=1, \\ \mathrm{sg}(\mathbf{v}_d), & t>1. \end{cases} \tag{7}$$

The first-step loss backpropagates through both query and item embeddings. Subsequent losses backpropagate through the query branch only, so feedback-conditioned states do not directly pull standalone item embeddings in different directions.

Retrieved content can clarify a query or distract the encoder. We therefore penalize increases in contrastive loss between consecutive states. For $N>1$, the degradation penalty is

$$\mathcal{L}_{\mathrm{deg}} = \frac{1}{N-1}\sum_{t=2}^{N}\big[\log(\ell_t+\delta) - \log\big(\mathrm{sg}(\ell_{t-1})+\delta\big)\big]_+, \tag{8}$$

where $[x]_+ = \max(x,0)$ and $\delta>0$ is a numerical stability constant. The log difference measures a relative increase in loss. Taking the positive part at each step prevents an improvement elsewhere from canceling a harmful update. The stop-gradient prevents the reference loss from being directly increased to reduce the penalty. This encourages, but does not guarantee, improved rankings after feedback.

The overall objective is

$$\mathcal{L}_{\mathrm{SFT}} = \mathbb{E}_{(q,d^+),\,\xi\sim p_{\mathrm{off}}(\cdot|q)}\left[\frac{1}{N}\sum_{t=1}^{N}\ell_t + \lambda_{\mathrm{deg}}\mathcal{L}_{\mathrm{deg}}\right], \tag{9}$$

where p_{off} is the offline trajectory distribution and $\lambda_{\mathrm{deg}}\geq 0$ weights the penalty. For $N=1$, we set $\mathcal{L}_{\mathrm{deg}}=0$. SFT supplies relevance supervision at every state without requiring expert labels for which feedback item to select.

4.3 REINFORCEMENT LEARNING FOR RETRIEVAL

SFT learns to interpret feedback, but does not explicitly optimize which items to select. Its trajectories come from an offline sampling procedure, and its contrastive losses supervise each state rather than assigning credit to actions for their downstream effects. We address these limitations by sampling trajectories from the current retrieval policy and optimizing feedback selection with a terminal ranking reward. Figure 2b illustrates this stage, which adapts group relative policy optimization (GRPO) (Shao et al., 2024) to item-selection actions.

Let θ denote the parameters after SFT. We freeze this model and attach an additional query-side low-rank adaptation (LoRA) (Hu et al., 2022) module with trainable parameters ϕ. The query and item representations are

$$\mathbf{z}_t^{\phi} = f_{\theta,\phi}\big(\mathcal{T}(h_t)\big), \qquad \overline{\mathbf{v}}_d = f_\theta(d). \tag{10}$$

The adapter is active only when encoding query contexts, including their feedback items. Standalone item embeddings are computed without it and remain fixed throughout RL. Consequently, optimization is confined to query interpretation and feedback selection within the SFT embedding space, without requiring the corpus index to be rebuilt after each update.

Terminal retrieval reward. The state is the complete history h_t, the action is a feedback item, and the transition appends that item to the history. This defines a finite-horizon Markov decision process with deterministic transitions. Let $\mathcal{C}_{\mathrm{RL}}(q)$ denote the item pool for query q, and let $\{\overline{\mathbf{v}}_d\}_q$ denote the precomputed static item embeddings. For a trajectory ξ, the terminal reward is the reciprocal rank of d^+ under the final query embedding:

$$r_\phi(q, \xi) = 1/\operatorname{rank}(d^+; \mathbf{z}_N^\phi, \{\overline{\mathbf{v}}_d\}_q). \tag{11}$$

Rank is one-based and determined by descending similarity over the entire training pool, including any previously selected items.

GRPO-style optimization. For each query, we sample $M \geq 2$ trajectories $\{\xi^{(m)}\}_{m=1}^M$ from a rollout policy $\pi_{\phi_{\mathrm{old}}}$, using Eq. 4 over $\mathcal{C}_{\mathrm{RL}}(q)$ with fixed item vectors. We compute rewards $r_m = r_{\phi_{\mathrm{old}}}(q, \xi^{(m)})$ for each trajecory.

For an observed state–action pair, the probability ratio is

$$\rho_{m,t}(\phi) = \frac{\pi_\phi(a_t^{(m)} \mid h_t^{(m)}; \mathcal{C}_{\mathrm{RL}}(q))}{\pi_{\phi_{\mathrm{old}}}(a_t^{(m)} \mid h_t^{(m)}; \mathcal{C}_{\mathrm{RL}}(q))}. \tag{12}$$

The objective for one query is

$$J_{\mathrm{GRPO}}(\phi; q) = \frac{1}{M(N-1)} \sum_{m=1}^{M} \sum_{t=1}^{N-1} \min\Big[\rho_{m,t}(\phi) A_m, \operatorname{clip}\big(\rho_{m,t}(\phi), 1-\epsilon, 1+\epsilon\big) A_m\Big]. \tag{13}$$

Here, $\epsilon > 0$ is the clipping parameter and $A_m = (r_m - \bar{r})/\sigma_r$ is the group-relative advantage, where $\bar{r}$ and σ_r are the mean and standard deviation of the M rewards. Each action receives its trajectory's terminal advantage, encouraging selections associated with better final rankings within the group.

Final embedding supervision. The policy objective acts on the first $N-1$ states. Although the final embedding determines the reward, the detached ranking reward provides no gradient through that embedding. Shared parameters can still change its representation indirectly. We therefore add direct contrastive supervision at the final state, using the fixed item embeddings in $\mathcal{C}_{\mathrm{RL}}(q)$:

$$\mathcal{L}_{\mathrm{ret}}(\phi; q) = \frac{1}{M} \sum_{m=1}^{M} \mathbb{I}[r_m < R]\, \ell_{\mathrm{NCE}}\big(\mathbf{z}_N^{\phi,(m)}, d^+, \mathcal{C}_{\mathrm{RL}}(q); \{\overline{\mathbf{v}}_d\}_q\big). \tag{14}$$

The combined objective is

$$\mathcal{L}_{\mathrm{RL}}(\phi) = \mathbb{E}_{q,\, \{\xi^{(m)}\} \sim \pi_{\phi_{\mathrm{old}}}} \left[-J_{\mathrm{GRPO}}(\phi; q) + \lambda\, \mathcal{L}_{\mathrm{ret}}(\phi; q)\right], \qquad \lambda = \lambda_{\mathrm{ret}}\, \mathbb{I}[r < R], \tag{15}$$

Here, $\mathbb{I}[\cdot]$ is the indicator function, R is a reward threshold, and $\lambda_{\mathrm{ret}} \geq 0$ weights the auxiliary loss. The gate is applied to each trajectory before averaging: only trajectories with rollout reward below R receive the auxiliary loss. The gate remains fixed during the policy update. This concentrates direct supervision on trajectories with poorer final rankings. We do not include a reference-policy KL penalty in the RL objective. Instead of constraining the policy to remain close to a separate reference model, we directly preserve retrieval quality through contrastive supervision on the final representation. This contrastive objective provides task-specific supervision for the final retrieval embedding, while the RL objective optimizes the model's feedback-selection behavior.

5 Experiments

5.1 Experimental Setup

We present only the key experimental settings in this section, while Appendix A provides the full details.

Benchmarks. We train and evaluate ARR on M-BEIR (Wei et al., 2024), which contains 16 dataset–task combinations spanning eight query–item modality configurations.

Table 1: M-BEIR results in the local-pool setting (%). Superscripts t and i denote text and image inputs, respectively. ARR rows report the final state ($N = 4$).

	$q^t \to d^i$			$q^t \to d^t$	$q^t \to d^{i,t}$		$q^i \to d^t$			$q^i \to d^i$	$q^{i,t} \to d^t$		$q^{i,t} \to d^i$		$q^{i,t} \to d^{i,t}$		
Method	**VN** R@5	**CO** R@5	**F200** R@10	**WQ** R@5	**ES** R@5	**WQ** R@5	**VN** R@5	**CO** R@5	**F200** R@10	**NS** R@5	**ON** R@5	**InS** R@5	**FIQ** R@10	**CR** R@5	**ON** R@5	**InS** R@5	**Avg.**
2B models																	
LamRA-Ret (Liu et al., 2025)	30.8	78.8	23.1	82.5	54.3	77.8	31.2	88.5	27.1	28.7	51.1	44.2	28.9	47.7	72.3	60.8	51.6
ELVA (Liu et al., 2026)	**35.6**	80.3	25.0	88.0	56.1	80.5	**33.4**	90.2	25.9	29.3	52.0	**47.4**	30.9	50.0	72.8	61.3	53.8
ARR-SFT	32.6	**84.4**	27.2	93.3	60.4	84.1	32.1	**94.3**	27.6	32.2	55.2	42.9	33.1	**59.7**	74.2	62.8	56.0
ARR-RL	32.0	84.0	**28.2**	**94.0**	**63.9**	**85.1**	31.8	94.2	**28.7**	**34.1**	**57.7**	42.4	**33.3**	59.3	**75.3**	**63.9**	**56.7**
8B/7B models																	
MM-Embed (Lin et al., 2025)	41.0	71.3	17.1	**95.9**	68.8	85.0	41.3	90.1	18.4	32.4	42.1	42.3	25.7	50.0	64.1	57.7	52.7
LamRA-Ret (Liu et al., 2025)	41.6	81.5	28.7	86.0	62.6	81.2	39.6	90.6	30.4	32.1	54.1	52.1	33.2	53.1	76.2	63.3	56.6
TRACE (Hao et al., 2026)	42.1	82.3	**30.5**	87.8	64.1	82.5	41.2	91.3	**33.2**	33.6	57.4	55.8	**36.4**	57.3	**78.5**	67.1	58.8
ELVA (Liu et al., 2026)	**43.5**	83.0	29.2	91.0	63.5	83.1	**41.7**	92.2	32.1	32.8	56.0	55.5	34.6	55.4	77.5	67.1	58.7
ARR-SFT	40.6	**85.8**	27.8	**95.9**	66.8	86.0	40.1	**94.7**	28.1	33.6	58.7	**56.1**	35.0	**64.4**	75.3	67.8	59.8
ARR-RL	41.3	85.6	30.0	95.7	**73.2**	**86.2**	41.0	**94.7**	30.2	**34.0**	**63.5**	55.8	35.1	64.0	78.1	**68.2**	**61.0**

Implementation. We initialize ARR from the 2B and 8B Qwen3-VL-Embedding models (Li et al., 2026a). The visual encoder remains frozen throughout training. SFT fine-tunes the remaining backbone parameters, whereas RL freezes the SFT backbone and trains only a query-side LoRA adapter. SFT uses offline top-10 feedback pools. For RL, we sample 2K queries from each M-BEIR dataset–task combination, giving 32K queries in total. To make training tractable, we use a query-specific pool $\mathcal{C}_{\mathrm{RL}}(q) = \mathcal{C}_{100}(q) \cup \{d^+\}$, where $\mathcal{C}_{100}(q)$ contains the SFT model's top-100 candidates. Feedback sampling, reward computation, and final-step contrastive supervision all use this pool. Unless otherwise stated, we use $N = 4$ for training and inference and select feedback greedily at inference time. ARR-SFT and ARR-RL denote the models after the two training stages.

Baselines. We compare against MM-Embed (Lin et al., 2025) and LamRA-Ret (Liu et al., 2025), as well as TRACE (Hao et al., 2026), which incorporates adaptive reasoning, and ELVA (Liu et al., 2026), which uses ranking-driven RL. These comparisons cover contrastive embedding learning, reasoning-enhanced retrieval, and retrieval-oriented policy optimization.

5.2 In-Domain Evaluation

Table 1 reports results on the M-BEIR test set. ARR-RL achieves the highest average score among the compared methods at both scales: 56.7 for 2B and 61.0 for 8B. RL improves the reported averages over ARR-SFT by 0.7 and 1.2 points, respectively. At 8B, ARR-RL exceeds the strongest baseline average, TRACE's 58.8, by 2.2 points.

The gains vary across tasks. For the 8B model, RL increases EDIS (ES) text-to-multimodal recall from 66.8 to 73.2 and OVEN (ON) multimodal-to-text recall from 58.7 to 63.5. These tasks require connecting the query to an item with a different information structure, making them promising settings for feedback-conditioned adaptation. Overall, RL improves the SFT model on most tasks, demonstrating the effectiveness of policy optimization for ARR. Some task configurations exhibit slight performance degradation, including VisualNews (VN) at 2B and CIRR (CR) at both scales. We hypothesize that these drops may stem from the sensitivity of RL training to the data: since RL is performed intensively on a relatively small subset of training data, variations in data quality and distribution across tasks may introduce task-specific biases. Nevertheless, the gains on most tasks indicate that RL provides an overall improvement over the SFT model.

5.3 Analysis of the Iterative Retrieval Process

Table 2 separates the effects of training with feedback from using feedback at inference time. Subscripts indicate the number of training states; columns report retrieval from each inference state. All RL-stage variants start from ARR-SFT$_{N=4}$. The $N = 1$ RL-stage control receives additional contrastive training on the RL data but uses neither feedback nor GRPO during that training.

Feedback use must be learned. ARR-SFT$_{N=1}$ scores 55.23 at the initial state but falls to 46.90 when extended to four inference states, showing that simply appending retrieved content can harm

Table 3: Zero-shot retrieval results for 8B models (%). Iter-1 uses the initial query embedding without feedback; unqualified ARR rows use the final state ($N = 4$). An asterisk marks benchmarks with images sourced from COCO or FashionIQ.

	$q^{t} \to d^{i}$			$q^{i} \to d^{t}$			$q^{i,t} \to d^{i}$		$q^{\text{dialog}} \to d^{i}$	$q^{i \oplus t} \to d^{i}$	
Method	**Share4V** R@1	**Urban*** R@1	**Flickr** R@1	**Share4V** R@1	**Urban*** R@1	**Flickr** R@1	**CIRCO*** MAP@5	**GeneCIS*** R@1	**VisD*** R@1	**MT-FIQ*** R@5	**Avg.**
LamRA-Ret (Liu et al., 2025)	93.3	95.1	82.8	88.1	94.3	92.7	33.2	18.9	62.8	60.9	72.21
TRACE (Hao et al., 2026)	94.9	94.8	84.5	89.1	94.1	94.5	34.8	20.5	65.4	63.2	73.58
ELVA (Liu et al., 2026)	96.6	96.1	84.4	92.0	95.5	95.2	34.5	20.2	65.3	61.2	74.10
ARR-SFT (Iter-1)	**98.8**	98.9	87.3	**99.0**	**99.0**	97.7	39.3	20.9	75.5	63.1	77.95
ARR-SFT	**98.8**	**99.1**	87.6	**99.0**	**99.0**	97.8	41.6	21.0	76.3	63.8	78.39
ARR-RL (Iter-1)	98.4	99.0	87.5	**99.0**	**99.0**	98.0	39.9	21.0	76.6	65.9	78.42
ARR-RL	98.4	99.0	**88.0**	**99.0**	**99.0**	**98.3**	**43.4**	**21.1**	**77.9**	**66.0**	**79.01**

a model not trained to use it. In contrast, ARR-SFT$_{N=4}$ improves from 55.63 to 56.01. Most of this gain occurs after the first feedback item, at Iter-2 (55.95), with smaller changes thereafter.

Feedback training also improves initial retrieval. Before observing any feedback, ARR-SFT$_{N=4}$ outperforms ARR-SFT$_{N=1}$ by 0.40 points. Thus, training on feedback-conditioned states benefits the shared encoder even when inference stops after the first step. One possible explanation is that supervising several contexts encourages features that also help represent the original query; these results establish the performance effect, rather than a particular feature-learning mechanism.

Table 2: Effect of training and inference steps for ARR-2B on M-BEIR (Avg.%).

Method	**Iter-1**	**Iter-2**	**Iter-3**	**Iter-4**
ARR-SFT$_{N=1}$	55.23	49.85	48.37	46.90
ARR-SFT$_{N=4}$	**55.63**	**55.95**	**56.01**	**56.01**
ARR-RL$_{N=1}$	55.44	55.57	55.54	55.47
ARR-RL$_{N=2}$	**56.40**	56.59	56.58	56.64
ARR-RL$_{N=3}$	**56.40**	56.57	56.62	56.63
ARR-RL$_{N=4}$	56.36	**56.68**	**56.66**	**56.74**

Policy training provides further gains. Policy training consistently improves upon the SFT initialization. ARR-RL$_{N=4}$ achieves scores of 56.36 at Iter-1 and 56.74 at Iter-4, compared with 55.63 and 56.01, respectively, for its SFT initialization. Notably, the improvement at Iter-1 indicates that the benefits of RL are not limited to exploiting feedback from previous retrieval steps. We attribute this gain to the richer supervision provided by the RL objective. While the SFT objective treats each item as either positive or negative, the RL objective assigns credit to a selected item according to its contribution to the eventual retrieval outcome. It therefore provides a softer and more informative learning signal, allowing the model to favor useful intermediate items even when they are not annotated positives and to better distinguish hard negatives based on their utility for subsequent retrieval. The performance gap between ARR-RL$_{N=1}$ and ARR-RL$_{N=2}$ further supports this interpretation. ARR-RL$_{N=1}$, which is optimized only with a contrastive objective, achieves 55.44 at Iter-1, whereas ARR-RL$_{N=2}$, trained with the RL objective, reaches 56.40 at the same step. These results suggest that policy-level credit assignment provides complementary supervision beyond conventional contrastive learning.

Additional steps yield diminishing returns. Training with two, three, or four states gives similar initial scores. Four-state training produces the best final score, but its advantage over two-state training at Iter-4 is only 0.10 points. Together with the marginal improvements beyond Iter-2, these results suggest a favorable trade-off between test-time scaling and retrieval performance in ARR, with Iter-2 offering a good balance between computational cost and performance gains.

5.4 Zero-Shot Evaluation

Table 3 evaluates the 8B models on seven benchmarks without task-specific fine-tuning. ARR-RL obtains the highest reported average, 79.01, compared with 74.10 for the strongest baseline average. Its scores of 77.9 on Visual Dialog and 66.0 on Multi-round FashionIQ also indicate transfer to dialogue and interactive retrieval.

The models from both training stages benefit from feedback at inference time: ARR-SFT improves from 77.95 at Iter-1 to 78.39 at the final step, and ARR-RL improves from 78.42 to 79.01. The gains are most visible on CIRCO and Visual Dialog. RL also raises the average at both the initial and final states relative to SFT, although some individual scores decrease slightly. These results extend the two observed benefits of feedback—better inference with retrieved context and better initial representations after training—beyond the M-BEIR evaluation tasks.

5.5 Ablation Study

Table 4 examines ARR's training objectives using the 2B model.

SFT design choices. Retrieval feedback is intended to help the model better understand the query. Accordingly, contrastive supervision at feedback-conditioned steps should provide a learning signal to the query representation, and we therefore stop gradients through the standalone item embeddings. When item gradients are enabled, the final score drops from 56.01 to 55.71, suggesting that the model may exploit undesirable shortcuts to minimize the training loss rather than learn meaningful item representations. Removing the degradation penalty leaves initial performance nearly unchanged (55.69 vs. 55.63), but eliminates the gain from feedback: the final score is 55.68 instead of 56.01. The penalty therefore improves the usefulness of later states rather than simply increasing first-step accuracy.

Policy and final embedding supervision. Removing the final embedding contrastive loss lowers the final score from 56.74 to 56.43, which remains above the SFT score. Conversely, removing GRPO yields 55.29 at the final step, below the scores of both the full RL model and its SFT initialization. Because the final embedding is conditioned on the retrieval trajectory, supervising only the final embedding can disrupt the model's retrieval policy, which in turn degrades the quality of the final representation. This result further suggests that the gains of ARR-RL stem from learning an effective retrieval policy, rather than from exploiting superficial item-specific patterns. The two objectives are therefore complementary in this setting: policy training improves feedback selection, and direct terminal supervision improves the representation used for ranking.

Table 4: Training ablations for ARR-2B on M-BEIR (Avg.%). "w/o reward gate" applies final-step supervision to all trajectories.

Setting	Iter-1	Iter-2	Iter-3	Iter-4
ARR-SFT	55.63	**55.95**	**56.01**	**56.01**
w/o $\mathrm{sg}(\mathbf{v}_d)$	55.31	55.62	55.72	55.71
w/o $\mathcal{L}_{\mathrm{deg}}$	**55.69**	55.61	55.66	55.68
ARR-RL	56.36	**56.68**	**56.66**	**56.74**
w/o $\mathcal{L}_{\mathrm{ret}}$	56.20	56.40	56.39	56.43
w/o reward gate	**56.39**	56.55	56.52	56.39
w/ KL	56.20	56.46	56.50	56.49
w/o J_{GRPO}	55.74	55.65	55.47	55.29

Reward gating and KL regularization. Applying the auxiliary contrastive loss to every trajectory, instead of gating it by reward, gives a final score of 56.39. This supports concentrating the auxiliary loss on trajectories with lower rewards. The variant with a KL penalty scores 56.49, compared with 56.74 for the default model. Thus, the default objective performs better in this comparison while avoiding a separate reference-model evaluation. Appendix B further examines the weights λ_{deg} and λ_{ret}.

6 Conclusion

We presented ARR, a multimodal retriever that learns to select and use retrieved items as feedback through an autoregressive retrieval process. ARR preserves independently indexed item embeddings while conditioning the query on a sequence of retrieved observations. SFT teaches feedback interpretation, and RL optimizes item selection for the final ranking outcome. Experiments show improvements in average in-domain and zero-shot retrieval performance. Further analyses highlight two complementary benefits of item feedback: using feedback during inference improves subsequent retrieval, while learning from feedback during training improves initial retrieval even before any feedback is observed.

AI USE STATEMENT

AI assistance was used solely for language editing of this manuscript, including correcting typographical and grammatical errors and improving clarity of expression. It was not used to generate or develop any novel scientific content, ideas, or contributions.

ETHICS STATEMENT

This work studies how retrieved items can serve as feedback to refine query representations for universal multimodal retrieval. This research adheres to relevant ethical guidelines, and we identify no specific ethical concerns associated with this work.

REPRODUCIBILITY STATEMENT

All training runs use the default random seed of 42. For a fixed model checkpoint, evaluation is deterministic and involves no random sampling. We provide sufficient details on the method, training settings, and evaluation protocol in Section 4 and Appendix A to enable reproduction of our experiments and results.

REFERENCES

Alberto Baldrati, Lorenzo Agnolucci, Marco Bertini, and Alberto Del Bimbo. Zero-shot composed image retrieval with textual inversion. In *ICCV*, pp. 15338–15347, 2023.

Amin Bigdeli, Negar Arabzadeh, Radin Hamidi Rad, Sajad Ebrahimi, Charles LA Clarke, and Ebrahim Bagheri. Adore: Iterative query expansion with retrieval-grounded relevance feedback. *arXiv preprint arXiv:2606.13905*, 2026.

Yingshan Chang, Mridu Narang, Hisami Suzuki, Guihong Cao, Jianfeng Gao, and Yonatan Bisk. WebQA: Multihop and multimodal QA. In *CVPR*, pp. 16495–16504, 2022.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. ShareGPT4V: Improving large multi-modal models with better captions. In *ECCV*, pp. 370–387. Springer, 2024.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pp. 1597–1607. PmLR, 2020.

Yang Chen, Hexiang Hu, Yi Luan, Haitian Sun, Soravit Changpinyo, Alan Ritter, and Ming-Wei Chang. Can pre-trained vision and language models answer visual information-seeking questions? In *EMNLP*, pp. 14948–14968, 2023.

Xuanming Cui, Jianpeng Cheng, Hong-you Chen, Satya Narayan Shukla, Abhijeet Awasthi, Xichen Pan, Chaitanya Ahuja, Shlok Mishra, Taipeng Tian, Qi Guo, et al. Think then embed: Generative context improves multimodal embedding. In *ICLR*, volume 2026, pp. 2690–2709, 2026.

Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José MF Moura, Devi Parikh, and Dhruv Batra. Visual dialog. In *CVPR*, pp. 326–335, 2017.

Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. DreamSim: Learning new dimensions of human visual similarity using synthetic data. In *NeurIPS*, 2023.

Xintong Han, Zuxuan Wu, Phoenix X Huang, Xiao Zhang, Menglong Zhu, Yuan Li, Yang Zhao, and Larry S Davis. Automatic spatially-aware fashion concept discovery. In *ICCV*, pp. 1463–1471, 2017.

Xiangzhao Hao, Shijie Wang, Tianyu Yang, Tianyue Wang, Haiyun Guo, and Jinqiao Wang. Trace: Task-adaptive reasoning and representation learning for universal multimodal retrieval. *arXiv preprint arXiv:2603.02929*, 2026.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pp. 9726–9735. IEEE, 2020.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Hexiang Hu, Yi Luan, Yang Chen, Urvashi Khandelwal, Mandar Joshi, Kenton Lee, Kristina Toutanova, and Ming-Wei Chang. Open-domain visual entity recognition: Towards recognizing millions of wikipedia entities. In *ICCV*, pp. 12065–12075, 2023.

Ting Jiang, Minghui Song, Zihan Zhang, Haizhen Huang, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, and Fuzhen Zhuang. E5-v: Universal embeddings with multimodal large language models. *arXiv preprint arXiv:2407.12580*, 2024.

Mingxin Li, Yanzhao Zhang, Dingkun Long, Keqin Chen, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, et al. Qwen3-vl-embedding and qwen3-vl-reranker: A unified framework for state-of-the-art multimodal retrieval and ranking. *arXiv preprint arXiv:2601.04720*, 2026a.

Xiaojie Li, Chu Li, Shi-Zhe Chen, and Xi Chen. U-marvel: Unveiling key factors for universal multimodal retrieval via embedding learning with mllms. In *ICLR*, volume 2026, pp. 121298–121319, 2026b.

Sheng-Chieh Lin, Chankyu Lee, Mohammad Shoeybi, Jimmy Lin, Bryan Catanzaro, and Wei Ping. Mm-embed: Universal multimodal retrieval with multimodal llms. In *ICLR*, volume 2025, pp. 44215–44234, 2025.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, pp. 740–755. Springer, 2014.

Fuxiao Liu, Yinghan Wang, Tianlu Wang, and Vicente Ordonez. Visual News: Benchmark and challenges in news image captioning. In *EMNLP*, 2021a.

Siqi Liu, Weixi Feng, Tsu-jui Fu, Wenhu Chen, and William Yang Wang. EDIS: Entity-driven image search over multimodal web content. In *EMNLP*, pp. 4877–4894, 2023.

Yikun Liu, Yajie Zhang, Jiayin Cai, Xiaolong Jiang, Yao Hu, Jiangchao Yao, Yanfeng Wang, and Weidi Xie. Lamra: Large multimodal model as your advanced retrieval assistant. In *CVPR*, pp. 4015–4025. IEEE, 2025.

Yuhan Liu, Pei Fu, Hang Li, Yukun Qi, Chao Jiang, Jingwen Fu, Zhen Liu, Bin Qin, Zhenbo Luo, Jian Luan, et al. Elva: Exploring ranking-driven universal multimodal retrieval. *arXiv preprint arXiv:2606.20280*, 2026.

Zheyuan Liu, Cristian Rodriguez-Opazo, Damien Teney, and Stephen Gould. Image retrieval on real-life images with pre-trained vision-and-language models. In *ICCV*, pp. 2125–2134, 2021b.

Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k Entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *ICCV*, pp. 2641–2649, 2015.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, pp. 8748–8763. PmLR, 2021.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Dike Sun, Zheng Zou, Jingtong Zang, Qi Sun, Huaipeng Zhaoand Tao Luo, and Xiaoyi Zeng. Queryagent-r1: Bridging query generation and product retrieval for e-commerce query recommendation. *arXiv preprint arXiv:2606.05671*, 2026a.

Siyu Sun, Jing Ren, Zhaohe Liao, Dongxiao Mao, Xiangyuan Ren, Yiyi Zhang, Haohua Zhao, Weixiong Lin, Jiang Shaohua, Liqing Zhang, et al. Bottleneck tokens for unified multimodal retrieval. *arXiv preprint arXiv:2604.11095*, 2026b.

Yiteng Tu, Weihang Su, Yujia Zhou, Yiqun Liu, Fen Lin, Qin Liu, and Qingyao Ai. Generalized pseudo-relevance feedback. In *Proceedings of the ACM Web Conference 2026*, pp. 1876–1886, 2026.

Sagar Vaze, Nicolas Carion, and Ishan Misra. GeneCIS: A benchmark for general conditional image similarity. In *CVPR*, pp. 6862–6872, 2023.

Cong Wei, Yang Chen, Haonan Chen, Hexiang Hu, Ge Zhang, Jie Fu, Alan Ritter, and Wenhu Chen. Uniir: Training and benchmarking universal multimodal information retrievers. In *ECCV*, pp. 387–404. Springer, 2024.

Hui Wu, Yupeng Gao, Xiaoxiao Guo, Ziad Al-Halah, Steven Rennie, Kristen Grauman, and Rogerio Feris. Fashion IQ: A new dataset towards retrieving images by natural language feedback. In *CVPR*, pp. 11307–11317, 2021.

Peixi Wu, Biao Yang, Feipeng Ma, Bosong Chai, Bo Lin, Wei Yuan, Fan Yang, Tingting Gao, Hebei Li, and Xiaoyan Sun. Lame: Learning to think in latent space for multimodal embedding via information bottleneck. *arXiv preprint arXiv:2606.13061*, 2026.

HongChien Yu, Chenyan Xiong, and Jamie Callan. Improving query representations for dense retrieval with pseudo relevance feedback. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, 2021. URL `https://arxiv.org/abs/2108.13454`.

Yifei Yuan and Wai Lam. Conversational fashion image retrieval via multiturn natural language feedback. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 839–848, 2021.

Beichen Zhang, Pan Zhang, Xiaoyi Dong, Yuhang Zang, and Jiaqi Wang. Long-CLIP: Unlocking the long-text capability of CLIP. In *ECCV*, pp. 310–325. Springer, 2024.

Longxiang Zhang, Weilong Dai, Guanghao Zhang, Hao Jiang, and Pipei Huang. Think when needed: Adaptive reasoning-driven multimodal embeddings with a dual-lora architecture. *arXiv preprint arXiv:2605.14448*, 2026a.

Xuxin Zhang, Ben Chen, Yue Lv, Siyuan Wang, Yupeng Li, Yufei Ma, Zihan Liang, Tong Zhao, Ying Yang, Huangyu Dai, et al. Oneretrieval: Unifying multi-branch e-commerce retrieval with an editable generative model. *arXiv preprint arXiv:2606.13533*, 2026b.

Yihua Zhang, Mingfu Liang, Jiyan Yang, Rong Jin, Wen-Yen Chen, Yiping Han, Huayu Li, Buyun Zhang, Liang Luo, Luke Simon, et al. Reasonrec: A reasoning-augmented multimodal agent for unified recommendation. In *Findings of ACL*, pp. 7960–7980, 2026c.

Zhaohua Zhang, Jianhuan Zhuo, Muxi Chen, Chenchen Zhao, Wenyu Jiang, Tianwen Jiang, Mingyang Chen, Qiuyong Xiao, Jihong Zhang, Zhixun Su, et al. Grape: Let grpo supervise query rewriting by ranking for retrieval. *arXiv preprint arXiv:2509.23370*, 2025.

Jinghan Zhao, Wenwei Jin, Anqi Li, Jintao Tong, Luya Mo, Jiawei Li, Yao Hu, and Bin Li. Uninote: A unified embedding model for multimodal representation and ranking. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 8555–8564, 2026.

Hongyi Zhu, Jia-Hong Huang, Stevan Rudinac, and Evangelos Kanoulas. Enhancing interactive image retrieval with query rewriting using large language models and vision language models. In *Proceedings of the 2024 international conference on multimedia retrieval*, pp. 978–987, 2024.

A Detailed Experimental Settings

A.1 Benchmarks

In-domain evaluation. We use the M-BEIR local-pool setting (Wei et al., 2024), in which candidates from the task-specific collection are ranked for each query. M-BEIR comprises 16 dataset–task combinations from 10 datasets, covering eight query–item modality configurations and four domains: news, fashion, Wikipedia, and miscellaneous content. Table 5 lists rounded benchmark split sizes and candidate-pool sizes. We report Recall@10 for Fashion200K and FashionIQ and Recall@5 for all other tasks.

Zero-shot evaluation. Table 6 lists query counts, candidate-pool sizes, and metrics for the seven benchmarks in Table 3. ShareGPT4V and Urban-1K evaluate retrieval between images and long captions, and Flickr30K evaluates image–caption retrieval. CIRCO and GeneCIS condition retrieval on an image and text. Visual Dialog evaluates dialogue-to-image retrieval, while Multi-round FashionIQ uses multi-turn image–text interactions. We evaluate these tasks without task-specific fine-tuning. Several benchmarks reuse images from COCO or FashionIQ, so this protocol does not imply that all visual content is unseen.

A.2 Implementation and Training

Model and feedback pools. We use the 2B and 8B Qwen3-VL-Embedding backbones (Li et al., 2026a). The visual encoder remains frozen in both stages. SFT fine-tunes the remaining backbone parameters; RL freezes the SFT backbone and trains an additional query-side LoRA adapter. The adapter is active when encoding the query and its feedback context, but inactive when encoding standalone corpus items.

For SFT, the pretrained encoder retrieves a top-10 feedback pool for each query. We sample three items without replacement to construct four query states. Missing positives are not added to this pool, and relevance labels are not used to filter feedback. The annotated positive is included separately in the contrastive supervision set. We train the SFT model for two epochs. In the first epoch, we disable the degradation penalty by setting $\lambda_{\text{deg}} = 0$ in Eq. 9, allowing the model to first acquire basic multi-round retrieval capability. In the second epoch, we enable the degradation penalty to encourage the model to leverage retrieved items to refine the query representation.

For RL, we sample 2K queries from each of the 16 M-BEIR dataset–task combinations, giving 32K queries. We form $\mathcal{C}_{\text{RL}}(q) = \mathcal{C}_{100}(q) \cup \{d^+\}$ from the SFT model's top-100 results and the annotated positive, removing duplicates. The resulting pool is shared by feedback sampling, reward computation, and final-step contrastive supervision. The history mask applies only to feedback selection. The reward gate in equation 14 is applied to each trajectory before averaging its auxiliary loss. Unless otherwise stated, $N = 4$ at both training and inference, and inference selects the highest-scoring unvisited feedback item at each step.

For queries with multiple positive items, during SFT we randomly select one positive as the ground-truth target and mask the remaining positives. During RL, the final-embedding supervision averages the losses over all positive items, while the reward function applies max pooling over their rewards.

Tables 7 and 8 list settings shared by ARR-2B and ARR-8B. SFT batch sizes count query–positive pairs. RL rollout batch sizes count queries, whereas optimization batch sizes count trajectories.

SFT takes 24 hours on 8 NVIDIA H20 GPUs for the 2B model and 48 hours on 16 NVIDIA H20 GPUs for the 8B model. RL uses 8 NVIDIA H20 GPUs for both models, taking 8 hours for 2B and 24 hours for 8B.

A.3 Baselines

MM-Embed. MM-Embed (Lin et al., 2025) adapts an MLLM for universal multimodal retrieval through contrastive learning and modality-aware hard-negative mining, with text retrieval data providing additional supervision.

Table 5: M-BEIR tasks and rounded benchmark statistics (Wei et al., 2024; Li et al., 2026b). Here, t denotes text and i denotes image; i+t denotes a joint image–text input. Split sizes count queries; "Pool" counts candidates in the local evaluation collection. These are benchmark statistics, not counts after ARR-specific preprocessing.

Dataset	Query → item	Train / dev / test queries	Pool
VisualNews (VN) (Liu et al., 2021a)	t → i	99K / 20K / 20K	542K
MSCOCO (CO) (Lin et al., 2014)	t → i	100K / 24.8K / 24.8K	5K
Fashion200K (F200) (Han et al., 2017)	t → i	15K / 1.7K / 1.7K	201K
WebQA (WQ) (Chang et al., 2022)	t → t	16K / 1.7K / 2.4K	544K
EDIS (ES) (Liu et al., 2023)	t → i+t	26K / 3.2K / 3.2K	1M
WebQA (WQ)	t → i+t	17K / 1.7K / 2.5K	403K
VisualNews (VN)	i → t	100K / 20K / 20K	537K
MSCOCO (CO)	i → t	113K / 5K / 5K	25K
Fashion200K (F200)	i → t	15K / 4.8K / 4.8K	61K
NIGHTS (NS) (Fu et al., 2023)	i → i	16K / 2K / 2K	40K
OVEN (ON) (Hu et al., 2023)	i+t → t	150K / 50K / 50K	676K
InfoSeek (InS) (Chen et al., 2023)	i+t → t	141K / 11K / 11K	611K
FashionIQ (FIQ) (Wu et al., 2021)	i+t → i	16K / 2K / 6K	74K
CIRR (CR) (Liu et al., 2021b)	i+t → i	26K / 2K / 4K	21K
OVEN (ON)	i+t → i+t	157K / 14.7K / 14.7K	335K
InfoSeek (InS)	i+t → i+t	143K / 17.6K / 17.6K	481K

Table 6: Zero-shot evaluation tasks and metrics. Counts are listed separately when retrieval directions differ. Candidate counts refer to the pool searched by each query; GeneCIS uses query-specific candidate sets.

Benchmark	Query → item	Queries / candidates
ShareGPT4V (Share4V) (Chen et al., 2024)	t → i; i → t	1K / 1K (both directions)
Urban-1K (Urban) (Zhang et al., 2024)	t → i; i → t	1K / 1K (both directions)
Flickr30K (Flickr) (Plummer et al., 2015)	t → i; i → t	t → i: 5K / 1K; i → t: 1K / 5K
CIRCO (Baldrati et al., 2023)	i+t → i	800 / 120K
GeneCIS (Vaze et al., 2023)	i+t → i	8K / 10–15
Visual Dialog (VisD) (Das et al., 2017)	dialogue → i	2K / 2K
Multi-round FashionIQ (MT-FIQ) (Yuan & Lam, 2021)	interactive i/t → i	2.4K / 6.2K

LamRA-Ret. LamRA (Liu et al., 2025) provides separate retrieval and reranking components. Its retrieval component, LamRA-Ret, learns multimodal embeddings through language-only pretraining and multimodal instruction tuning. We compare with the retrieval component in our main tables.

TRACE. TRACE (Hao et al., 2026) integrates reasoning and representation learning in an MLLM, adapting its reasoning behavior to the retrieval task. It represents an approach that improves query interpretation before embedding.

ELVA. ELVA (Liu et al., 2026) constructs embeddings after generating a textual synthesis of the input and uses ranking and margin rewards for RL. Its policy training targets retrieval-oriented representations, providing a comparison with ARR's policy over feedback items.

B Hyperparameter Analysis

Table 9 examines the degradation-penalty weight λ_{deg} in equation 9 and the final-step supervision weight λ_{ret} in equation 15, using ARR-2B. We use 0.1 for both weights in the default configuration.

Degradation penalty. Without the penalty ($\lambda_{\text{deg}} = 0$), the score remains nearly constant across inference states: 55.69 at Iter-1 and 55.68 at Iter-4. With $\lambda_{\text{deg}} = 0.1$, it rises from 55.63 to 56.01.

Table 7: SFT training settings shared by ARR-2B and ARR-8B.

Setting	Value
Optimizer	AdamW
Weight decay	0.01
Learning rate	5×10^{-5}
Learning-rate schedule	Cosine
Warmup steps	150
Training epochs	2
Effective global batch size (pairs)	1,024
Gradient clipping (maximum norm)	1.0
Contrastive temperature (τ in Eq. 2)	0.02
Degradation penalty weight λ_{deg}	0.1
Hard negatives	0
Maximum text tokens	1,024
Maximum image pixels	$1{,}280 \times 32 \times 32$
Minimum image pixels	$64 \times 32 \times 32$
Training precision	BF16
Checkpoint selection	Final

Table 8: RL training settings shared by ARR-2B and ARR-8B. All other settings follow those used for SFT.

Setting	Value
Learning rate	5×10^{-6}
Learning-rate schedule	Cosine
Warmup steps	50
Training epochs	2
Effective global batch size (trajectories)	256
LoRA rank	128
LoRA scaling α	256
LoRA dropout	0.0
Global rollout batch size (queries)	1,024
Trajectories per query M	8
Policy sampling temperature (τ in Eq. 4)	0.1
GRPO clipping ϵ	0.2
Optimization passes per rollout batch	1
Final-step contrastive temperature (τ in Eq. 14)	0.05
Final-step contrastive weight λ_{ret}	0.1
Reward threshold R	0.2

Table 9: Sensitivity to auxiliary loss weights for ARR-2B on M-BEIR (average recall, %). Bold marks the best result in each column within each training stage.

Setting	Iter-1	Iter-2	Iter-3	Iter-4
ARR-SFT				
$\lambda_{\text{deg}} = 0$	**55.69**	55.61	55.66	55.68
$\lambda_{\text{deg}} = 0.1$ (default)	55.63	**55.95**	**56.01**	**56.01**
$\lambda_{\text{deg}} = 1.0$	55.60	55.76	55.86	55.89
ARR-RL				
$\lambda_{\text{ret}} = 0$	56.20	56.40	56.39	56.43
$\lambda_{\text{ret}} = 0.01$	56.15	56.45	56.46	56.47
$\lambda_{\text{ret}} = 0.1$ (default)	**56.36**	**56.68**	**56.66**	**56.74**
$\lambda_{\text{ret}} = 1.0$	55.59	55.92	56.01	56.02

Increasing the weight to 1.0 also produces improvements across states, but yields a lower final score of 55.89. Thus, greater emphasis on preventing relative degradation does not necessarily maximize absolute retrieval accuracy. Among the tested values, 0.1 gives the best final score while maintaining comparable initial performance.

Final-step contrastive supervision. Without the auxiliary loss ($\lambda_{\mathrm{ret}} = 0$), policy training reaches 56.43 at Iter-4, above the SFT score of 56.01. Setting $\lambda_{\mathrm{ret}} = 0.1$ further raises this score to 56.74. A weight of 1.0 yields 56.02, suggesting that excessive emphasis on the auxiliary objective can reduce the gains from policy training. The intermediate value 0.1 performs best among the tested settings.