AMBER: Multi-View Adaptive Budget Allocation for Listwise Vision-Language Reranking

Anonymous ACL submission

Abstract

Vision-language models (VLMs) are powerful listwise rerankers for multimodal retrieval, but high inference costs restrict them to evaluating small local candidate views. Existing multi-call strategies rely on fixed schedules, wasting expensive VLM calls on uninformative candidate pairs and easy queries. To address this, we propose **A**daptive **M**ulti-view **B**udgeted **E**lo **R**eranking (**AMBER**), an online, budgeted multi-view reranking framework that dynamically optimizes global resource allocation. AMBER treats fragmented listwise VLM outputs as local tournaments, using continuous Elo updates to maintain a lightweight global ranking state. Building on this, it allocates computation at two levels: dynamically constructing candidate views with high score ambiguity, and scheduling queries to maximize expected information gain. We theoretically prove AMBER's equivalence to online MLE under the Bradley–Terry model and its submodular approximation guarantees. Experiments on CIRR, CIRCO, and PhotoBench demonstrate that AMBER achieves state-of-the-art head-ranking precision while consuming significantly fewer VLM calls than existing baselines. Our code is publicly available in [1].

1 Introduction

Multimodal retrieval typically follows a two-stage paradigm, where a strong reranker refines candidates recalled by an embedding model. Vision-language models (VLMs) excel at this second stage via listwise reranking, which compares multiple images in a shared context to capture fine-grained visual-textual alignment (Cai, 2026; Qiao et al., 2026). However, this approach introduces a severe resource bottleneck. Due to token costs, context length limits, and inference latency, a single VLM call can only observe a small *local view* of candidates. Reliable reranking thus requires aggregating evidence across multiple VLM calls, which creates a critical **budget allocation problem**: how should a system optimally distribute a limited computation budget across thousands of queries and candidates?

Existing multi-call strategies fail to utilize this budget effectively. Approaches like sliding-window or tournament-style traversals rely on fixed schedules (Chen et al., 2025). They apply the same reranking effort uniformly, failing to distinguish which candidate comparisons are actually informative or which queries are fundamentally more difficult. While recent adaptive methods (Yoon et al., 2025) can stop early for confident queries, they operate in isolation and cannot dynamically redistribute the remaining budget across a global pool of competing queries. Consequently, existing systems waste expensive VLM calls on easy queries or uninformative candidate pairs (Rashid et al., 2024; Liu and Li, 2026).

To address this, we propose **A**daptive **M**ulti-view **B**udgeted **E**lo **R**eranking (**AMBER**), an online, budgeted multi-view reranking framework that jointly optimizes evidence aggregation and dynamic resource allocation. To aggregate fragmented local views, AMBER treats each listwise VLM output as a local tournament, converting the outcomes into online Elo score updates. This yields a lightweight global ranking state that seamlessly integrates first-stage embedding priors with VLM evidence, without requiring an explicit pairwise preference matrix.

Building on this dynamic state, AMBER strategically allocates computation at two levels. At the *candidate level*, it constructs the next local view by prioritizing candidates that are near the top of the ranking, infrequently observed, and have ambiguous scores compared to their neighbors. At the *query level*, AMBER evaluates the expected information gain of these proposed views, schedul-

[1] https://anonymous.4open.science/status/bear-budget-aware-listwise-ranking-5E2A

ing VLM calls exclusively for queries where the comparison is highly uncertain and consequential for head-ranking precision.

Crucially, AMBER's design is heavily grounded in theory. We demonstrate that its Elo updates equate to online Maximum Likelihood Estimation under the Bradley–Terry model (Robbins and Monro, 1951). Furthermore, the query scheduling objective is a monotone submodular function, guaranteeing a tight $(1 - 1/e)$ approximation ratio for our greedy allocation policy (Nemhauser et al., 1978). Extensive experiments on CIRR (Liu et al., 2021), CIRCO (Baldrati et al., 2023), and PhotoBench (Xu et al., 2026) demonstrate that AMBER achieves state-of-the-art head-ranking precision (e.g., R@1 and NDCG) while consuming significantly fewer VLM calls than existing rigid baselines.

Our main contributions are as follows:

- We are the first to formulate VLM-based listwise reranking as a **budgeted multi-view evidence aggregation problem**, moving beyond fixed schedules to dynamic global resource allocation.
- We propose **AMBER**, an online framework that maintains a lightweight global ranking state by converting local listwise VLM outputs into continuous Elo-style updates.
- We design a **two-level allocation strategy** that jointly selects high-uncertainty candidate views and schedules queries based on maximum expected information gain.
- We provide a rigorous **theoretical foundation**, proving AMBER's equivalence to online MLE and its submodular approximation guarantees (Nemhauser et al., 1978), complemented by SOTA results on three composed image retrieval benchmarks under constrained budgets.

2 Preliminary

We study second-stage vision-language model (VLM) reranking for multimodal retrieval. Let $\mathcal{Q} = \{q_1, q_2, \ldots, q_m\}$ denote the full query set of size m. For each query $q_j \in \mathcal{Q}$, a first-stage embedding retriever returns a candidate pool ordered by embedding similarity:

$$\mathcal{C}_{q_j} = \{c_1, c_2, \ldots, c_N\}. \tag{1}$$

The ultimate goal is to produce a refined ranking $\hat{\sigma}_{q_j}$ that places the most relevant candidates at the top.

A listwise VLM reranker takes q_j and a small candidate subset as input to output a local ranking. At any reranking round t, the VLM can only observe a limited *local view*

$$V_{q_j}^{(t)} \subset \mathcal{C}_{q_j}, \qquad \left|V_{q_j}^{(t)}\right| = M, \quad M \ll N. \tag{2}$$

Given the substantial computational cost per VLM call, we assume a fixed global call budget

$$B = b \cdot m, \tag{3}$$

where b is the average number of VLM calls permitted per query.

AMBER maintains a query-specific state for each q_j:

$$\mathcal{S}_{q_j} = \Big(\{s_{ij}\}_{i=1}^{N},\ \{m_{ij}\}_{i=1}^{N},\ \mathcal{O}_{q_j},\ \{r_{ij}\}_{i=1}^{N},\ U_{q_j}\Big), \tag{4}$$

where s_{ij} is the current Elo score of candidate c_i for query q_j, m_{ij} is its VLM observation count, $\mathcal{O}_{q_j}$ is the set of observed candidates, r_{ij} is the global rank of c_i induced by the current Elo state for q_j, and U_{q_j} is the query-level scheduling utility score.

3 Methodology

To address the aforementioned challenges under a strict budget constraint, we propose **AMBER**, a dynamic framework that jointly optimises query allocation, candidate selection, and local-to-global ranking aggregation. We elaborate on the three core mechanisms below.

3.1 Online Cross-View Elo Aggregation

Since individual VLM calls observe different candidate subsets, the aggregation of local rankings must be incremental and cross-view compatible. AMBER conceptualises this process as an online rating mechanism. Given a local VLM ranking $\pi_{q_j}^{(t)}$ over view $V_{q_j}^{(t)}$, we decompose the local ranking into independent pairwise outcomes. If candidate c_i is ranked above c_k in $\pi_{q_j}^{(t)}$, we record a "win":

$$c_i \succ c_k. \tag{5}$$

For each outcome $c_i \succ c_k$, the expected winning probability based on the current state $\mathcal{S}_{q_j}$ is computed as:

$$P(c_i \succ c_k \mid q_j) = \frac{1}{1 + 10^{(s_{kj} - s_{ij})/\tau}}, \tag{6}$$

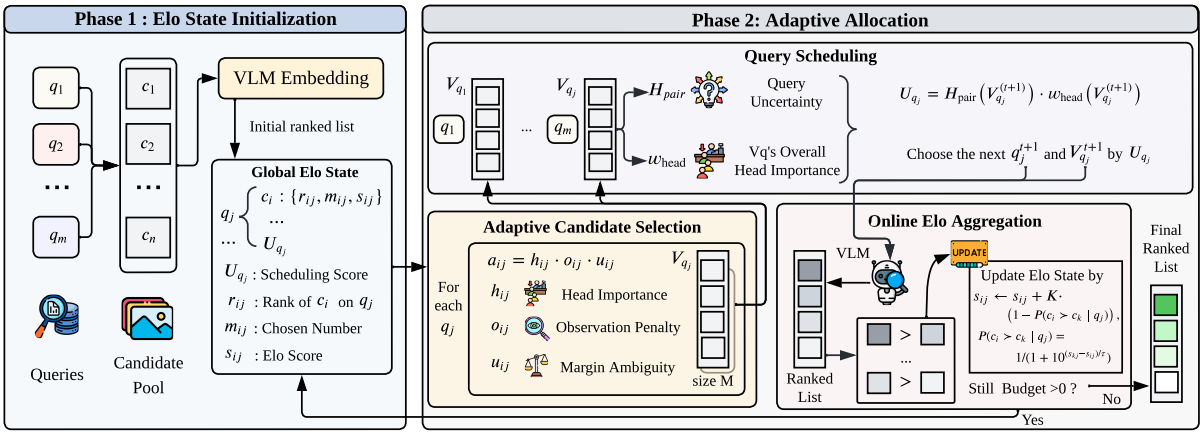

Figure 1: The overall architecture of AMBER.

where τ is the Elo scaling factor. The candidates' scores are then updated simultaneously:

$$s_{ij} \leftarrow s_{ij} + K(1 - P(c_i \succ c_k \mid q_j)), \quad (7)$$

$$s_{kj} \leftarrow s_{kj} - K(1 - P(c_i \succ c_k \mid q_j)), \quad (8)$$

where K dictates the update step size.

Initial Elo scores $\{s_{ij}\}$ are assigned using the first-stage embedding ranking, acting as a prior. Specifically, for the candidate ranked i-th in the embedding-based candidate pool $\mathcal{C}_{q_j}$, we initialise

$$s_{ij}^{(0)} = s_0 + R \cdot \frac{N - i}{N - 1}, \quad (9)$$

where $N = |\mathcal{C}_{q_j}|$, s_0 is the base rating, and R controls the strength of the embedding prior. This linear initialisation gives higher-ranked candidates from the first-stage retriever larger initial Elo scores, while allowing subsequent VLM observations to revise the ranking through online updates.

Following each update, observed candidates are re-sorted by their Elo scores, while unobserved candidates are appended based on their original embedding order.

3.2 Adaptive Candidate Selection

When query q_j is selected for a VLM call, AMBER must construct the most informative local view $V_{q_j}^{(t+1)}$. We assign each candidate c_i a sampling value a_{ij} that integrates three strategic factors.

1. Head Importance. We prioritise candidates near the top of the current ranking, aligning with metrics such as Recall@K and mAP@K:

$$h_{ij} = \frac{1}{\log_2(r_{ij} + 1)}. \quad (10)$$

2. Observation Penalty. To prevent redundant evaluations, we introduce diminishing returns for frequently observed candidates:

$$o_{ij} = \frac{1}{\sqrt{m_{ij} + 1}}. \quad (11)$$

3. Local Margin Ambiguity. We favour candidates whose relative ordering is highly uncertain. Let $\mathcal{N}_\kappa^{(j)}(c_i)$ denote the set of candidates within κ rank positions of c_i under query q_j. The minimal local Elo gap of c_i is

$$g_{ij} = \min_{c_l \in \mathcal{N}_\kappa^{(j)}(c_i)} |s_{ij} - s_{lj}|, \quad (12)$$

yielding the margin ambiguity factor:

$$u_{ij} = \frac{1}{1 + g_{ij}/\tau_c}, \quad (13)$$

where τ_c controls gap sensitivity. Smaller margins denote ambiguous ordering, increasing the sampling value.

The final candidate sampling value is the product of these three factors:

$$\begin{aligned} a_{ij} &= h_{ij} \cdot o_{ij} \cdot u_{ij} \\ &= \frac{1}{\log_2(r_{ij} + 1)} \cdot \frac{1}{\sqrt{m_{ij} + 1}} \cdot \frac{1}{1 + g_{ij}/\tau_c}. \end{aligned} \quad (14)$$

The system then constructs the next view by selecting the candidates with the highest values:

$$V_{q_j}^{(t+1)} = \mathrm{TopM}_{c_i \in \mathcal{C}_{q_j}} \; a_{ij}. \quad (15)$$

3.3 Entropy-Based Query Scheduling

At the query level, AMBER dynamically allocates the remaining global budget based on the expected

utility of the *next* VLM call. Rather than assessing query difficulty in isolation, we condition the scheduling score U_{q_j} on the specific view $V_{q_j}^{(t+1)}$ that would be evaluated.

For each eligible query q_j, we simulate its next view $V_{q_j}^{(t+1)}$ and estimate the pairwise uncertainty within it. For every pair (c_i, c_k) in this view, let $p_{ik}^{(j)} := P(c_i \succ c_k \mid q_j)$ denote their Elo winning probability. The normalised binary entropy derived from $p_{ik}^{(j)}$ is:

$$H\left(p_{ik}^{(j)}\right) = -\frac{1}{\log 2}\Big[p_{ik}^{(j)} \log p_{ik}^{(j)} + \left(1 - p_{ik}^{(j)}\right) \log\left(1 - p_{ik}^{(j)}\right)\Big]. \tag{16}$$

The view-level pairwise entropy is the average uncertainty over all comparisons in the view:

$$H_{\text{pair}}\left(V_{q_j}^{(t+1)}\right) = \frac{2}{M(M-1)} \times \sum_{\substack{c_i,\, c_k \in V_{q_j}^{(t+1)} \\ i<k}} H\left(p_{ik}^{(j)}\right). \tag{17}$$

To ensure the budget serves top-ranking improvements, we scale this entropy by the view's average head importance:

$$w_{\text{head}}\left(V_{q_j}^{(t+1)}\right) = \left(\frac{1}{M} \sum_{c_i \in V_{q_j}^{(t+1)}} \frac{1}{\log_2(r_{ij}+1)}\right)^{1/2} \tag{18}$$

The final query scheduling score is formulated as:

$$U_{q_j} = H_{\text{pair}}\left(V_{q_j}^{(t+1)}\right) \cdot w_{\text{head}}\left(V_{q_j}^{(t+1)}\right). \tag{19}$$

This ensures the system strictly prioritises queries where the designated VLM call will resolve the most critical and uncertain head comparisons.

3.4 Overall Algorithm

The overall AMBER framework operates in two distinct phases, summarised in Algorithm 1.

Phase 1 (Initial Reranking). To establish baseline observations, every query $q_j \in \mathcal{Q}$ receives exactly one VLM call. Local rankings are obtained and translated into initial Elo updates, initialising $\mathcal{S}_{q_j}$ for all j.

Phase 2 (Adaptive Allocation). The framework enters an iterative budget allocation loop. While the total spent budget S is less than B, AMBER computes the query scheduling score U_{q_j} for all eligible queries. The queries with the highest U_{q_j} are granted a VLM call for their respective $V_{q_j}^{(t+1)}$. Elo states $\mathcal{S}_{q_j}$ are continuously updated until the budget is exhausted, at which point the final hybrid rankings $\hat{\sigma}_{q_j}$ are returned.

3.5 Theoretical Analysis

We provide theoretical grounding for AMBER's three core mechanisms under a generative assumption on VLM outputs. All proofs are deferred to Appendix C.

Assumption 1 (Bradley–Terry Model)**.** *The pairwise preferences produced by the VLM follow a Bradley–Terry (BT) model: there exist true latent scores* $\mathbf{s}^* = \{s_i^*\}_{i=1}^N$ *such that for any pair* (c_i, c_k),

$$P^*(c_i \succ c_k) = \frac{1}{1 + 10^{(s_k^* - s_i^*)/\tau}}, \tag{20}$$

and VLM outputs for distinct pairs are conditionally independent given $\mathbf{s}^*$.

Proposition 1 (Elo Updates as Online MLE)**.** *Under Assumption 1, each Elo update* (Eqs. (7)–(8)) *applied to a single observed outcome* $c_i \succ c_k$ *is equivalent to a stochastic gradient ascent step on the BT log-likelihood with effective learning rate* $\eta = K\tau/\ln 10$.

Corollary 1 (Convergence Rate)**.** *Under Assumption 1 and a weak* ℓ_2*-regularized BT objective, with decaying step size* $K_t = K_0/\sqrt{t}$*, the Elo estimates satisfy*

$$\mathbb{E}\big[\ell(\mathbf{s}^{\text{MLE}}) - \ell(\hat{\mathbf{s}}_T)\big] = O\left(\frac{1}{\sqrt{T}}\right), \tag{21}$$

where T *is the total number of processed pairwise outcomes. With a fixed step size the rate improves to* $O(1/T)$.

Proposition 2 (Sample Complexity for Top-K Recovery)**.** *Under Assumption 1, suppose the true scores satisfy* $s_1^* > s_2^* > \cdots > s_N^*$*, and let* $\Delta_K = s_K^* - s_{K+1}^*$ *denote the score gap at the* K*-th boundary. If every straddling pair* (c_i, c_k) *with* $i \le K < k$ *is observed at least*

$$m \;\ge\; \frac{C\,\tau^2 \ln(N/\delta)}{\Delta_K^2} \tag{22}$$

times for an absolute constant $C > 0$*, then with probability at least* $1 - \delta$ *the top-*K *set induced by the Elo scores equals the true top-*K *set.*

Algorithm 1 AMBER Reranking

Require: Query set $\mathcal{Q} = \{q_1, \ldots, q_m\}$, candidate pools $\{\mathcal{C}_{q_j}\}_{j=1}^m$, view size M, total budget B
Ensure: Final rankings $\{\hat{\sigma}_{q_j}\}_{j=1}^m$

```
for each query q_j ∈ Q do
    Initialise Elo scores {s_ij} from embedding ranks
    Initialise observation counts m_ij ← 0, observed set O_{q_j} ← ∅
    Initialise global ranks {r_ij} by embedding order
end for
Phase 1: Initial Reranking
for each query q_j ∈ Q do
    Select view V_{q_j}^(1) by candidate sampling
    Obtain local ranking π_{q_j}^(1) from the VLM
    Convert π_{q_j}^(1) into pairwise outcomes; update {s_ij}, {m_ij}, O_{q_j}, and {r_ij}
end for
S ← m
Phase 2: Adaptive Allocation
while S < B do
    for each eligible query q_j do
        Build next view V_{q_j}^(t+1) via Eq. (15)
        Compute scheduling score U_{q_j} = H_pair(V_{q_j}^(t+1)) · w_head(V_{q_j}^(t+1))
    end for
    Select queries with the largest U_{q_j}
    for each selected query q_j do
        Rerank V_{q_j}^(t+1) with the VLM; obtain π_{q_j}^(t+1)
        Update {s_ij} from pairwise outcomes
        Update {m_ij}, O_{q_j}, and {r_ij}
    end for
    Update spent budget S
end while
for each query q_j ∈ Q do
    Sort observed candidates by Elo scores
    Append unobserved candidates by embedding order
    Return final ranking σ̂_{q_j}
end for
```

Proposition 2 directly motivates the three factors in the candidate sampling value a_{ij} (Eq. (14)):

- **Head importance** h_{ij}: the required observation count concerns only pairs straddling the top-K boundary. Candidates near the head of the ranking are far more likely to be involved in such pairs, justifying $h_{ij} = 1/\log_2(r_{ij} + 1)$ as a proximity-to-boundary weight.
- **Observation penalty** o_{ij}: once a candidate has been observed m_{ij} times, further comparisons yield diminishing reductions in estimation error (the $O(1/\sqrt{m})$ bound). The factor $o_{ij} = 1/\sqrt{m_{ij} + 1}$ implements exactly this diminishing-returns discount, preventing redundant allocation.
- **Margin ambiguity** u_{ij}: Eq. (22) shows that the required observation count scales as $1/\Delta_K^2$. Candidates whose current Elo gap g_{ij} to their nearest neighbour is small have an estimated $\Delta_K \approx g_{ij}$ and therefore require the most additional observations. The factor $u_{ij} = 1/(1 + g_{ij}/\tau_c)$ assigns higher priority to precisely these ambiguous candidates.

Definition 1. *For query q_j, let $\mathcal{A}_{q_j}$ denote the set of VLM calls already made. The* uncertainty reduction *achieved by $\mathcal{A}_{q_j}$ is*

$$f_{q_j}(\mathcal{A}_{q_j}) = H_{\text{pair}}\left(V_{q_j}^{(t+1)} \mid \emptyset\right) - H_{\text{pair}}\left(V_{q_j}^{(t+1)} \mid \mathcal{A}_{q_j}\right). \quad (23)$$

Proposition 3 (Submodularity of Uncertainty Reduction)**.** *Under Assumption 1, f_{q_j} is a non-negative, monotone, submodular function of $\mathcal{A}_{q_j}$.*

Corollary 2 (Greedy Approximation Guarantee)**.** *Let $\mathcal{A}^*$ be the optimal allocation of the remaining budget after Phase 1 that maximises $\sum_j f_{q_j}(\mathcal{A}_{q_j})$. The greedy policy of AMBER—selecting $q^* = \arg\max_j U_{q_j}$ at each step—achieves*

$$\sum_j f_{q_j}(\hat{\mathcal{A}}_{q_j}) \;\geq\; \left(1 - \frac{1}{e}\right) \sum_j f_{q_j}(\mathcal{A}^*_{q_j}), \quad (24)$$

where $\hat{\mathcal{A}}_{q_j}$ is the set of calls allocated by AMBER. The scheduling score U_{q_j} (Eq. (19)) serves as a tractable, head-weighted proxy for the marginal gain $f_{q_j}(\mathcal{A}_{q_j} \cup \{x\}) - f_{q_j}(\mathcal{A}_{q_j})$.

Limitations. The results above rest on Assumption 1. In practice, VLM outputs may exhibit positional bias and context sensitivity, and Proposition 3 assumes conditional independence across calls, whereas AMBER's Elo state couples successive observations. These gaps motivate the empirical evaluation in Section 4.

Table 1: Summary of dataset statistics, scales, and evaluation characteristics.

Dataset	# Images	# Queries	Targets / Query	Language
CIRR	36,000+	36,554	Single	English
CIRCO	123,403	1,020	Multiple (Avg. 4.53)	English
PhotoBench	3,582	1,188	Intent-driven	EN / CN

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate our method on three multimodal retrieval benchmarks: CIRR (Liu et al., 2021), CIRCO (Baldrati et al., 2023), and PhotoBench (Xu et al., 2026). CIRR is a composed image retrieval benchmark where each query consists of a reference image and a relative modification text, and each query has one target image. This makes the evaluation especially sensitive to the exact position of the target image in the ranked list. CIRCO is a more challenging composed image retrieval benchmark where each query may have multiple valid target images. Therefore, in addition to Recall@K, mAP@K is an important metric for evaluating the quality of the ranked list. PhotoBench evaluates retrieval over personal photo albums. We use the merged-album setting and report results separately for Chinese and English queries.

Evaluation Metrics. For CIRR, we report Recall@1, Recall@2, Recall@5, and Recall@10. For CIRCO, we report Recall@5, Recall@10, Recall@25, Recall@50, and mAP@5, mAP@10, mAP@25, and mAP@50. For PhotoBench, we report Recall@K and NDCG@K. Some PhotoBench queries have empty ground-truth sets. We keep these queries in evaluation and assign zero to all their metrics, rather than skipping them. All PhotoBench numbers are multiplied by 100.

Implementation Details. All experiments use Qwen3-VL-Embedding-8B as the first-stage retriever and Qwen3-VL-8B-Instruct as the VLM reranker. The first-stage retriever produces the initial candidate pool, and all reranking methods are evaluated on the same recalled candidates for fair comparison. Unless otherwise specified, the candidate pool size is set to 50 and the VLM local view size is set to 10. A VLM call denotes one listwise reranking call over one local candidate view. All experiments are conducted without chain-of-thought prompting or supervised fine-tuning.

Our method, denoted as **AMBER**, performs online Elo aggregation to update candidate scores from local VLM listwise outputs. Unless otherwise specified, AMBER uses view-conditioned pairwise entropy for query-level budget allocation and head-aware observation-margin sampling for candidate-level view selection. The total VLM-call budget is set to approximately 24 calls per query on average.

Baselines. We compare AMBER with embedding-only retrieval baselines and several multi-call VLM reranking baselines. The embedding-only baselines include QQMM-Embed-v2 (Xue et al., 2025), RzenEmbed-v2-7B (Jian et al., 2025), and Qwen3-VL-Embedding-8B (Li et al., 2026). For VLM reranking, we compare with the following baselines:

- **AcuRank** (Yoon et al., 2025): an uncertainty-aware adaptive reranking baseline.
- **Sliding Window**: applies listwise reranking over fixed local windows of the candidate list.
- **TourRank** (Chen et al., 2025): a tournament-style reranking baseline that compares candidates through grouped local competitions.
- **UniRank**: a non-adaptive uniform-budget baseline that assigns the same number of VLM calls to each query and repeatedly reranks the current top-10 candidates.

4.2 Main Results

Table 2 summarizes the main results across CIRR, CIRCO, and PhotoBench. AMBER achieves the strongest overall performance among the compared reranking baselines under the same top-50 candidate pools and comparable VLM-call budgets.

On CIRR, AMBER achieves the best R@1 and R@2 among all methods in the table, and the strongest overall performance among VLM reranking baselines. Compared with TourRank, the strongest reranking baseline on R@1, AMBER improves R@1 from 51.37 to 54.07. This indicates that online Elo aggregation is particularly effective for improving the very top position in single-target composed image retrieval. Although QQMM-Embed-v2 obtains higher R@5 and R@10 as an embedding-only reference, AMBER consistently improves over the Qwen3-VL-Embedding-8B candidate pool used by all reranking methods.

On CIRCO, AMBER achieves the best mAP across all reported cutoffs. The gains are more pronounced on mAP@25 and mAP@50, indicating

Table 2: Main reranking results across datasets. R@K denotes Recall@K, and N@K denotes NDCG@K. QQMM-Embed-v2, RzenEmbed-v2-7B, and Qwen3-VL-Embedding-8B are embedding-only retrieval baselines; all VLM reranking methods are evaluated on the same top-50 candidate pools retrieved by Qwen3-VL-Embedding-8B. For CIRCO, we report R@5 and mAP metrics, while for PhotoBench, Avg. N@10 averages Chinese and English NDCG@10 scores. All PhotoBench values are multiplied by 100. Avg. Rank is computed over the displayed metrics within each dataset; lower is better. Best and second-best results are shown in bold and underlined, respectively. Full results are reported in Appendix A.

	Method	**Calls**	**R@1**	**R@2**	**R@5**	**R@10**	**R@50**	**Avg. Rank**
CIRR	QQMM-Embed-v2	–	38.43	59.80	**80.36**	**89.23**	**97.59**	3.40
	RzenEmbed-v2-7B	–	19.30	44.96	70.36	83.08	96.77	8.00
	Qwen3-VL-Embedding-8B	–	48.80	63.81	78.39	86.55	96.94	4.20
	AcuRank	∼24	50.63	63.78	77.90	85.06	96.94	5.00
	Sliding Window	24	51.11	63.86	76.05	86.55	96.94	4.20
	TourRank	24	51.37	64.63	77.13	85.52	96.94	4.00
	UniRank	24	53.42	66.58	80.19	86.55	96.94	2.20
	AMBER	24	**54.07**	**66.60**	79.30	87.64	96.94	**1.80**
	Method	**Calls**	**R@5**	**mAP@5**	**mAP@10**	**mAP@25**	**mAP@50**	**Avg. Rank**
CIRCO	QQMM-Embed-v2	–	71.12	45.92	47.13	50.39	51.45	6.00
	RzenEmbed-v2-7B	–	60.75	32.20	34.19	37.41	38.61	8.00
	Qwen3-VL-Embedding-8B	–	63.38	44.60	45.96	48.96	50.08	7.00
	AcuRank	∼24	73.75	56.28	56.88	58.49	59.52	2.80
	Sliding Window	24	71.50	54.52	55.64	58.33	58.86	4.40
	TourRank	24	74.00	56.45	56.74	59.30	59.95	2.20
	UniRank	24	71.62	55.09	53.09	56.00	57.13	4.60
	AMBER	24	**74.75**	**58.80**	**58.73**	**61.17**	**61.78**	**1.00**
	Method	**Calls**	**CN N@1**	**CN N@10**	**EN N@1**	**EN N@10**	**Avg. N@10**	**Avg. Rank**
PhotoBench	QQMM-Embed-v2	–	46.69	49.65	44.38	47.47	48.56	6.00
	RzenEmbed-v2-7B	–	48.54	50.07	45.64	48.24	49.16	5.00
	Qwen3-VL-Embedding-8B	–	44.52	45.54	41.95	44.13	44.84	7.00
	AcuRank	∼24	51.97	52.27	49.33	50.37	51.32	3.20
	Sliding Window	24	51.38	52.35	49.08	50.75	51.55	2.80
	TourRank	24	51.30	51.98	50.50	50.93	51.46	3.00
	AMBER	24	**54.06**	**53.89**	**52.60**	**52.27**	**53.08**	**1.00**

that AMBER improves not only whether relevant images appear in the candidate head, but also their relative ordering. This is important for CIRCO, where each query may have multiple valid targets.

On PhotoBench, AMBER consistently improves ranking quality for both Chinese and English queries. It obtains the best NDCG@1 and NDCG@10 for both languages, suggesting that the proposed budgeted multi-view aggregation is also effective in personal photo retrieval scenarios.

4.3 Ablation Study

Table 3 studies the contribution of each component in the candidate sampling value under the same fixed reranking budget. The full sampling value achieves the best overall ranked-list quality, obtaining the highest mAP@10, mAP@25, and mAP@50. Removing the head factor causes the largest degradation in mAP, confirming that prioritizing candidates near the ranking head is important for budgeted VLM reranking. Removing the observation factor also leads to clear performance drops, especially on R@10 and the mAP metrics, suggesting that repeatedly evaluating already-observed candidates wastes limited VLM calls. Removing the margin factor is less harmful and even slightly improves R@5 and mAP@5, but it underperforms the full variant on deeper mAP metrics. This indicates that local margin ambiguity mainly helps refine the ordering of multiple relevant candidates beyond the very top positions.

4.4 Sensitivity Analysis

We further analyze the sensitivity of AMBER to the strength of the embedding-prior Elo initialization. Table 4 reports CIRCO results under a fixed budget of 8 VLM calls per query. The performance remains stable across rating ranges from 100 to 800, with the best overall results obtained around 200–400. This indicates that AMBER does not rely on a finely tuned initialization scale: the embedding prior provides a useful starting point, while online Elo updates quickly incorporate VLM preference evidence.

5 Related Works

5.1 Multimodal Image Retrieval

Recent advancements leverage Multimodal Large Language Models (MLLMs) to create unified rep-

Table 3: Ablation study of candidate sampling factors on CIRCO under a fixed budget of 8 VLM calls per query. *Full* combines the head-importance factor, the observation-penalty factor, and the local-margin ambiguity factor. Each ablated variant removes one factor while keeping the other two unchanged. Best results are shown in bold.

Variant	Head	Obs.	Margin	Calls	R@5	R@10	R@25	R@50	mAP@5	mAP@10	mAP@25	mAP@50
Full	✓	✓	✓	8	73.38	**82.38**	89.25	93.25	57.34	**57.37**	**59.40**	**60.30**
w/o Head	–	✓	✓	8	71.12	81.88	**90.00**	93.25	53.34	53.82	56.59	57.37
w/o Obs.	✓	–	✓	8	73.00	79.00	87.88	93.25	56.20	54.67	56.75	57.87
w/o Margin	✓	✓	–	8	**73.62**	81.50	89.12	93.25	**57.42**	56.92	59.02	59.97

Table 4: Sensitivity analysis of the Elo initialization range on CIRCO under a fixed budget of 8 VLM calls per query. The rating range controls the strength of the embedding-prior initialization: the top-ranked candidate from the first-stage retriever receives the highest initial Elo score, while the last candidate receives the base score.

Rating Range	R@5	R@10	R@25	R@50	mAP@5	mAP@10	mAP@25	mAP@50
100	73.00	82.25	89.38	93.25	56.36	56.39	58.52	59.42
200	**73.25**	**82.38**	89.38	93.25	**56.87**	**56.84**	**58.88**	**59.76**
400	73.12	82.25	**89.50**	93.25	56.85	56.79	58.87	59.75
800	73.12	82.25	89.38	93.25	56.69	56.74	58.79	59.68

resentation and ranking architectures (Lin et al., 2025; Li et al., 2026), including training-free adaptations (Su and Wang, 2026). Beyond similarity matching, methods have evolved to incorporate generative memorization (Li et al., 2024), conversational query rewriting (Hu et al., 2026), and Chain-of-Thought reasoning (Wu et al., 2025). In the complex domain of Composed Image Retrieval (CIR), approaches range from tailored LLMs (Huynh et al., 2025) and zero-shot visual-semantic integration (Ye et al., 2026) to autonomous cross-modal agents (Yang et al., 2026). While these advancements significantly enhance single-query reasoning, scaling their heavy architectures across massive candidate pools remains a systemic bottleneck.

5.2 LLMs for Listwise Ranking

The viability of LLMs as listwise rerankers was established by proprietary models (Sun et al., 2024; Ma et al., 2023) and democratized by robust open-source alternatives like RankVicuna and RankZephyr (Pradeep et al., 2023a,b). Research quickly shifted toward optimizing multi-call strategies, such as tournament-inspired schedules (Chen et al., 2025), while analyzing inherent limitations like positional bias (Qiao et al., 2026).

To mitigate severe computational costs, a wave of efficiency-driven innovations emerged, including ranking-aware passage compression (Zhi et al., 2026), minimal reasoning frameworks (Liu and Li, 2026), embedding-repurposed rerankers (Liu et al., 2025), and early-stopping strategies like EcoRank and AcuRank (Rashid et al., 2024; Yoon et al., 2025). Recently, this efficient listwise paradigm has successfully expanded to handle visually rich, multimodal tasks (Cai, 2026; Sun et al., 2026).

Crucially, existing multi-call strategies either apply rigid computation uniformly (e.g., TourRank) or confine adaptivity to single queries (e.g., AcuRank, EcoRank), failing to dynamically redistribute budgets globally. AMBER overcomes this by converting local listwise outputs into continuous Elo updates to maintain a lightweight global ranking state. This enables a dual-level submodular allocation that directs every VLM call to the most informative query-candidate pair globally. Extended related works are detailed in the Appendix B.

6 Conclusion

In this paper, we present AMBER, an online, budgeted multi-view reranking framework that dynamically optimizes global resource allocation for listwise Vision-Language Models (VLMs). To overcome the severe computational bottleneck of VLMs, AMBER converts fragmented local rankings into continuous Elo updates, maintaining a lightweight global ranking state without an explicit pairwise preference matrix. Building on this, we introduce a dual-level allocation strategy: dynamically constructing candidate views based on score ambiguity and scheduling queries to maximize expected information gain. We theoretically establish AMBER's equivalence to online Maximum Likelihood Estimation under the Bradley-Terry model and prove its submodular approximation guarantees. Extensive evaluations on CIRR, CIRCO, and PhotoBench confirm that AMBER achieves state-of-the-art precision using significantly fewer VLM calls than existing rigid baselines. Future work will extend this budget-aware framework to broader domains, including text and text-to-image retrieval.

7 Limitations

First, while our evaluation focuses on image retrieval, our modality-agnostic framework can naturally generalize to other domains like video or audio retrieval. Second, we employ standard listwise prompting without Chain-of-Thought (CoT) reasoning. Notably, AMBER consistently outperforms strong baselines even without CoT's added computational overhead, emphasizing the efficiency of our core allocation strategy. Finally, while our sensitivity analysis confirms robustness, the current hyperparameters are static; automatically tuning these parameters during the reranking loop remains a promising direction for future work.

References

Alberto Baldrati, Lorenzo Agnolucci, Marco Bertini, and Alberto Del Bimbo. 2023. Zero-shot composed image retrieval with textual inversion. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 15338–15347.

Hongyi Cai. 2026. When vision meets texts in listwise reranking. *Preprint*, arXiv:2601.20623.

Yiqun Chen, Qi Liu, Yi Zhang, Weiwei Sun, Xinyu Ma, Wei Yang, Daiting Shi, Jiaxin Mao, and Dawei Yin. 2025. Tourrank: Utilizing large language models for documents ranking with a tournament-inspired strategy. *Preprint*, arXiv:2406.11678.

Yuan Hu, ZhiYu Cao, PeiFeng Li, and QiaoMing Zhu. 2026. Recqr: Incorporating conversational query rewriting to improve multimodal image retrieval. *Preprint*, arXiv:2603.26669.

Chuong Huynh, Jinyu Yang, Ashish Tawari, Mubarak Shah, Son Tran, Raffay Hamid, Trishul Chilimbi, and Abhinav Shrivastava. 2025. Collm: A large language model for composed image retrieval. *Preprint*, arXiv:2503.19910.

Weijian Jian, Yajun Zhang, Dawei Liang, Chunyu Xie, Yixiao He, Dawei Leng, and Yuhui Yin. 2025. Rzenembed: Towards comprehensive multimodal retrieval. *Preprint*, arXiv:2510.27350.

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

Yongqi Li, Wenjie Wang, Leigang Qu, Liqiang Nie, Wenjie Li, and Tat-Seng Chua. 2024. Generative cross-modal retrieval: Memorizing images in multimodal language models for retrieval and beyond. *Preprint*, arXiv:2402.10805.

Sheng-Chieh Lin, Chankyu Lee, Mohammad Shoeybi, Jimmy Lin, Bryan Catanzaro, and Wei Ping. 2025. Mm-embed: Universal multimodal retrieval with multimodal llms. *Preprint*, arXiv:2411.02571.

Danyang Liu and Kan Li. 2026. Stop overthinking: Unlocking efficient listwise reranking with minimal reasoning. *Preprint*, arXiv:2605.14450.

Qi Liu, Yanzhao Zhang, Mingxin Li, Dingkun Long, Pengjun Xie, and Jiaxin Mao. 2025. E2rank: Your text embedding can also be an effective and efficient listwise reranker. *Preprint*, arXiv:2510.22733.

Zheyuan Liu, Cristian Rodriguez-Opazo, Damien Teney, and Stephen Gould. 2021. Image retrieval on real-life images with pre-trained vision-and-language models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2125–2134.

Xueguang Ma, Xinyu Zhang, Ronak Pradeep, and Jimmy Lin. 2023. Zero-shot listwise document reranking with a large language model. *Preprint*, arXiv:2305.02156.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294.

Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. 2009. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609.

Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023a. Rankvicuna: Zero-shot listwise document reranking with open-source large language models. *Preprint*, arXiv:2309.15088.

Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023b. Rankzephyr: Effective and robust zero-shot listwise reranking is a breeze! *Preprint*, arXiv:2312.02724.

Jingfen Qiao, Jin Huang, Xinyu Ma, Shuaiqiang Wang, Dawei Yin, Evangelos Kanoulas, and Andrew Yates. 2026. Llm-based listwise reranking under the effect of positional bias. *Preprint*, arXiv:2604.03642.

Alexander Rakhlin, Ohad Shamir, and Karthik Sridharan. 2012. Making gradient descent optimal for strongly convex stochastic optimization. In *ICML*.

Muhammad Shihab Rashid, Jannat Ara Meem, Yue Dong, and Vagelis Hristidis. 2024. Ecorank: Budget-constrained text re-ranking using large language models. *Preprint*, arXiv:2402.10866.

Herbert Robbins and Sutton Monro. 1951. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

Dawei Su and Dongsheng Wang. 2026. Retllm: Training and data-free mllms for multimodal information retrieval. *Preprint*, arXiv:2602.22278.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2024. Is chatgpt good at search? investigating large language models as re-ranking agents. *Preprint*, arXiv:2304.09542.

Yiqun Sun, Pengfei Wei, and Lawrence B. Hsieh. 2026. Very efficient listwise multimodal reranking for long documents. *Preprint*, arXiv:2605.11864.

Shangrong Wu, Yanghong Zhou, Yang Chen, Feng Zhang, and P. Y. Mok. 2025. Chain-of-thought re-ranking for image retrieval tasks. *Preprint*, arXiv:2509.14746.

Tianyi Xu, Rong Shan, Junjie Wu, Jiadeng Huang, Teng Wang, Jiachen Zhu, Wenteng Chen, Minxin Tu, Quantao Dou, Zhaoxiang Wang, and 1 others. 2026. Photobench: Beyond visual matching towards personalized intent-driven photo retrieval. *arXiv preprint arXiv:2603.01493*.

Youze Xue, Dian Li, and Gang Liu. 2025. Improve multi-modal embedding learning via explicit hard negative gradient amplifying. *Preprint*, arXiv:2506.02020.

Zhongyu Yang, Wei Pang, and Yingfang Yuan. 2026. X ^r : Cross-modal agents for composed image retrieval. In *Proceedings of the ACM Web Conference 2026*, page 2071–2082. ACM.

Yongcong Ye, Kai Zhang, Yanghai Zhang, Enhong Chen, Longfei Li, and Jun Zhou. 2026. Fine-grained zero-shot composed image retrieval with complementary visual-semantic integration. *Preprint*, arXiv:2601.14060.

Soyoung Yoon, Gyuwan Kim, Gyu-Hwung Cho, and Seung won Hwang. 2025. Acurank: Uncertainty-aware adaptive computation for listwise reranking. *Preprint*, arXiv:2505.18512.

Zhewei Zhi, Yingyi Zhang, Yizhen Jing, Xianneng Li, Jianing Liu, Huajie Liu, and Yongliang Ding. 2026. Compress-then-rank: Faster and better listwise reranking with large language models via ranking-aware passage compression. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(41):35059–35067.

A Full Experimental Results

This appendix reports additional full-metric results for CIRCO and PhotoBench. CIRR results are already fully reported in Table 2.

B Extended Related Work

B.1 Multimodal Image Retrieval

Recent advancements in multimodal image retrieval have heavily leveraged Multimodal Large Language Models (MLLMs) to create unified and versatile representations. For instance, MM-Embed leverages MLLMs to establish a universal framework capable of handling diverse multimodal retrieval tasks through cohesive semantic alignment (Lin et al., 2025). Building on this push for unified architectures, the Qwen3-VL framework provides a comprehensive state-of-the-art pipeline that harmonizes both the multimodal embedding generation and the subsequent ranking stages (Li et al., 2026). To further reduce deployment overhead, RETLLM proposes a training-free and data-free approach, adapting existing MLLMs directly for multimodal information retrieval without the need for extensive fine-tuning (Su and Wang, 2026).

Beyond traditional similarity matching, new paradigms are incorporating generative strategies and sophisticated query interpretation. Generative cross-modal retrieval methods have explored memorizing images directly within the parameters of language models, fundamentally shifting how visual retrieval and downstream generation tasks are executed (Li et al., 2024). Furthermore, to bridge the semantic gap between ambiguous user inputs and system representations, ReCQR incorporates conversational query rewriting to clarify user intent and substantially improve retrieval accuracy (Hu et al., 2026). To refine initial retrieval results, advanced reasoning techniques have been integrated into the concluding ranking phases. Notably, the application of Chain-of-Thought (CoT) prompting has been shown to effectively guide models through step-by-step logical evaluation, thereby improving the accuracy of image retrieval re-ranking tasks (Wu et al., 2025).

A particularly prominent sub-field driving recent innovation is Composed Image Retrieval (CIR), where queries consist of both a reference image and textual modification instructions. CoLLM addresses this complex task by introducing a large language model specifically tailored to process and synthesize these compound multimodal queries (Huynh et al., 2025). Tackling CIR in resource-constrained environments, recent work has integrated complementary visual-semantic features to achieve fine-grained, zero-shot retrieval without requiring paired training data (Ye et al., 2026).

Table 5: Full results on CIRCO. Best results are shown in bold and second-best results are underlined.

Method	Calls	R@5	R@10	R@25	R@50	mAP@5	mAP@10	mAP@25	mAP@50
QQMM-Embed-v2	–	71.12	79.75	90.25	**95.26**	45.92	47.13	50.39	51.45
RZENEmbed-v2-7B	–	60.75	72.38	85.75	92.25	32.20	34.19	37.41	38.61
Qwen3-VL-Embedding-8B	–	63.38	76.50	87.88	93.25	44.60	45.96	48.96	50.08
AcuRank	23.28	73.75	83.38	87.38	93.25	56.28	56.88	58.49	59.52
Sliding Window	24	71.50	**84.50**	**91.38**	93.25	54.52	55.64	58.33	58.86
TourRank	24	74.00	82.38	91.12	93.25	56.45	56.74	59.30	59.95
UniRank	24	71.62	76.50	87.88	93.25	55.09	53.09	56.00	57.13
AMBER	24	**74.75**	83.50	90.75	93.25	**58.80**	**58.73**	**61.17**	**61.78**

Table 6: Full PhotoBench results in the merged-album setting under our evaluation protocol. Queries with empty ground-truth sets are retained and assigned zero metric values instead of being skipped. All metric values are multiplied by 100. Best results are shown in bold and second-best results are underlined.

Lang.	Method	Calls	R@1	R@5	R@10	R@20	NDCG@1	NDCG@10	NDCG@20
CN	QQMM-Embed-v2	–	23.56	44.53	51.63	58.51	46.69	49.65	51.62
CN	RZENEmbed-v2-7B	–	24.02	44.30	51.59	58.05	48.54	50.07	51.78
CN	Qwen3-VL-Embedding-8B	–	22.04	41.05	47.26	52.65	44.52	45.54	46.94
CN	AcuRank	24.3	25.95	46.48	52.25	54.51	51.97	52.27	52.18
CN	Sliding Window	24	25.85	46.14	52.76	56.58	51.38	52.35	52.89
CN	TourRank	24	24.89	46.13	52.65	**56.67**	51.30	51.98	52.60
CN	**AMBER**	24	**26.91**	**48.00**	**53.24**	56.62	**54.06**	**53.89**	**54.22**
EN	QQMM-Embed-v2	–	22.53	43.05	50.49	57.40	44.38	47.47	49.62
EN	RZENEmbed-v2-7B	–	22.86	43.42	50.89	57.68	45.64	48.24	50.20
EN	Qwen3-VL-Embedding-8B	–	21.20	40.64	46.57	52.32	41.95	44.13	45.63
EN	AcuRank	24.2	24.62	45.45	51.00	54.21	49.33	50.37	50.68
EN	Sliding Window	24	24.67	45.17	51.92	56.61	49.08	50.75	51.72
EN	TourRank	24	24.84	45.81	51.64	55.75	50.50	50.93	51.76
EN	**AMBER**	24	**25.94**	**46.64**	**52.26**	**56.72**	**52.60**	**52.27**	**53.08**

Adding an autonomous dimension to this process, X^R introduces intelligent cross-modal agents that actively reason and take steps to resolve intricate composed image retrieval queries (Yang et al., 2026).

B.2 LLMs for Listwise Ranking

The application of Large Language Models (LLMs) to listwise ranking has garnered significant attention, beginning with early investigations into ChatGPT's capabilities as a re-ranking agent (Sun et al., 2024). Building on this foundation, researchers demonstrated the viability of zero-shot listwise document reranking using proprietary LLMs (Ma et al., 2023). This approach was quickly democratized by the development of open-source alternatives like RankVicuna, which proved that open-weight LLMs could also perform effective zero-shot listwise reranking (Pradeep et al., 2023a). Subsequent models, such as RankZephyr, further refined this paradigm to offer more robust zero-shot listwise reranking capabilities (Pradeep et al., 2023b).

As LLM-based rerankers matured, research shifted toward optimizing ranking strategies and addressing inherent model limitations. For instance, TourRank introduced a tournament-inspired strategy to evaluate and structure the document ranking process more effectively (Chen et al., 2025). Concurrently, other work highlighted the persistent challenge of positional bias in LLMs, thoroughly analyzing its effects on listwise reranking performance (Qiao et al., 2026).

Efficiency and computational costs have also emerged as critical bottlenecks in deploying listwise ranking systems. To mitigate these issues, EcoRank proposed a budget-constrained approach for text re-ranking that balances cost and performance (Rashid et al., 2024). Similarly, AcuRank implemented an uncertainty-aware adaptive computation method to dynamically allocate processing resources during reranking (Yoon et al., 2025). Other efficiency-driven innovations include the Compress-then-Rank framework, which accelerates listwise reranking through ranking-aware passage compression (Zhi et al., 2026). Researchers have also explored unlocking efficient reranking

with minimal reasoning to prevent computational overthinking (Liu and Li, 2026), while E2Rank demonstrated that standard text embeddings can be repurposed as highly efficient listwise rerankers (Liu et al., 2025).

Most recently, the scope of listwise reranking has expanded beyond pure text to incorporate multimodal information. Pioneering work has begun exploring the intersection of vision and text within listwise reranking frameworks to seamlessly handle diverse data types (Cai, 2026). Building upon this, highly efficient listwise multimodal reranking techniques have been developed specifically to process and rank long, visually rich documents (Sun et al., 2026).

Despite the growing emphasis on efficiency, existing multi-call and budget-aware strategies share critical limitations. Fixed-schedule traversal methods such as TourRank apply uniform computation across queries, which can waste resources on easy queries or uninformative comparisons. Conversely, adaptive methods such as AcuRank and EcoRank mitigate this issue through early stopping, but their adaptivity is primarily confined within individual queries and does not explicitly redistribute a shared global budget across queries.

AMBER departs from both rigid schedules and isolated intra-query heuristics. By converting local listwise outputs into continuous Elo-style updates, AMBER maintains a lightweight cross-view ranking state without an explicit preference matrix. Building on this state, it couples candidate-level view construction with query-level budget allocation, directing VLM calls toward uncertain comparisons that are more likely to affect head-ranking quality.

C Theoretical Analysis

We provide theoretical grounding for AMBER's three core mechanisms under Assumption 1 (Bradley–Terry model) introduced in Section 3.5. All results below are stated under this assumption.

C.1 Elo Aggregation as Online Maximum Likelihood Estimation

We first establish that AMBER's update rule corresponds exactly to stochastic gradient ascent (SGA) on the BT log-likelihood.

Proof of Proposition 1. The BT log-likelihood for a single observation $c_i \succ c_k$ is

$$\begin{aligned}\ell(s_{ij}, s_{kj}) &= \log P(c_i \succ c_k \mid q_j) \\ &= -\log\Big(1 + 10^{(s_{kj}-s_{ij})/\tau}\Big)\,.\end{aligned} \tag{25}$$

Computing partial derivatives:

$$\frac{\partial \ell}{\partial s_{ij}} = \frac{\ln 10}{\tau}\big(1 - P(c_i \succ c_k \mid q_j)\big), \tag{26}$$

$$\frac{\partial \ell}{\partial s_{kj}} = -\frac{\ln 10}{\tau}\big(1 - P(c_i \succ c_k \mid q_j)\big). \tag{27}$$

The Elo updates in Eqs. (7)–(8) can then be written as

$$\begin{aligned}s_{ij} &\leftarrow s_{ij} + K\big(1 - P(c_i \succ c_k \mid q_j)\big) \\ &= s_{ij} + \frac{K\tau}{\ln 10} \cdot \frac{\partial \ell}{\partial s_{ij}},\end{aligned} \tag{28}$$

$$\begin{aligned}s_{kj} &\leftarrow s_{kj} - K\big(1 - P(c_i \succ c_k \mid q_j)\big) \\ &= s_{kj} + \frac{K\tau}{\ln 10} \cdot \frac{\partial \ell}{\partial s_{kj}},\end{aligned} \tag{29}$$

which is precisely gradient ascent with step size $\eta = K\tau/\ln 10$. □

Remark 1. *The BT log-likelihood is globally concave but not strongly concave, due to translation invariance (adding a constant to all scores leaves every pairwise probability unchanged). A standard remedy is to add a weak ℓ_2 regularizer $-\lambda\|\mathbf{s}\|^2$, which breaks this degeneracy without affecting the induced ranking. Under this regularized objective, classical SGA theory (Robbins and Monro, 1951) guarantees convergence to the unique MLE. In AMBER, the embedding-prior initialisation (Eq. (9)) further anchors the score scale and plays a role similar to such regularisation without requiring an explicit penalty.*

Corollary 3 (Convergence Rate)**.** *Under Assumption 1 and the regularized BT objective, with decaying step size $K_t = K_0/\sqrt{t}$, the Elo estimates satisfy*

$$\mathbb{E}\big[\ell(\mathbf{s}^{\mathrm{MLE}}) - \ell(\hat{\mathbf{s}}_T)\big] = O\left(\frac{1}{\sqrt{T}}\right), \tag{30}$$

where T is the total number of processed pairwise outcomes and $\hat{\mathbf{s}}_T$ denotes the Elo state after T updates. Under stronger smoothness and strong-concavity assumptions, faster rates may be obtained with appropriate step-size schedules.

Corollary 3 confirms that AMBER's Elo aggregation provides a statistically consistent estimate of the true candidate ordering, with error decreasing as more VLM calls are processed.

C.2 Sample Complexity and the Rationale for Head-Aware Observation-Margin Sampling

Proposition 1 shows that Elo scores converge asymptotically. Under a finite budget B, however, not all candidate pairs can be observed. We now ask: *which pairs must be observed, and how many times, to guarantee correct top-K recovery?*

Proof Sketch of Proposition 2. By Proposition 1, each Elo score $\hat{s}_{ij}$ is an SGA iterate on the BT log-likelihood. By standard concentration results for SGA on concave objectives (Nemirovski et al., 2009; Rakhlin et al., 2012), after m observations of pair (c_i, c_k), the estimation error satisfies $|\hat{s}_{ij} - s_i^*| \leq O(\tau/\sqrt{m})$ with high probability. For the top-K set to be correctly recovered, it suffices that for every straddling pair the accumulated error is smaller than the gap:

$$|\hat{s}_{ij} - s_i^*| + |\hat{s}_{kj} - s_k^*| < \Delta_K. \tag{31}$$

Setting $O(\tau/\sqrt{m}) < \Delta_K/2$ and applying a union bound over all $O(N^2)$ pairs yields Eq. (22). □

Proposition 2 directly motivates the three factors in the candidate sampling value a_{ij} (Eq. (14)):

- **Head importance** h_{ij}: the required observation count concerns only pairs straddling the top-K boundary. Candidates near the head of the ranking are far more likely to be involved in such pairs, justifying $h_{ij} = 1/\log_2(r_{ij}+1)$ as a proximity-to-boundary weight.
- **Observation penalty** o_{ij}: once a candidate has been observed m_{ij} times, further comparisons yield diminishing reductions in estimation error (the $O(1/\sqrt{m})$ bound). The factor $o_{ij} = 1/\sqrt{m_{ij}+1}$ implements exactly this diminishing-returns discount, preventing redundant allocation.
- **Margin ambiguity** u_{ij}: Eq. (22) shows that the required observation count scales as $1/\Delta_K^2$. Candidates whose current Elo gap g_{ij} to their nearest neighbour is small have an estimated $\Delta_K \approx g_{ij}$ and therefore require the most additional observations. The factor $u_{ij} = 1/(1 + g_{ij}/\tau_c)$ assigns higher priority to precisely these ambiguous candidates.

Together, the product $a_{ij} = h_{ij} \cdot o_{ij} \cdot u_{ij}$ approximates a greedy policy for minimising the total observation shortfall implied by Proposition 2.

C.3 Submodular Motivation for Query Scheduling

We further provide an information-theoretic motivation for the view-conditioned query scheduler. The practical scheduling score in Eq. (19) is a lightweight proxy for the marginal value of one additional VLM call. To clarify this connection, we first consider an idealized uncertainty-reduction objective.

Definition 2 (Idealized uncertainty reduction)**.** *For query q_j, let $\mathcal{A}_{q_j}$ denote the set of VLM calls already made. Let $\mathbf{s}^*_{q_j}$ be the latent BT scores for candidates in $\mathcal{C}_{q_j}$. Define the idealized information gain as*

$$F_{q_j}(\mathcal{A}_{q_j}) = H(\mathbf{s}^*_{q_j}) - H(\mathbf{s}^*_{q_j} \mid \mathcal{A}_{q_j}). \tag{32}$$

Proposition 4 (Submodularity of idealized information gain)**.** *Under Assumption 1, if VLM observations are conditionally independent given $\mathbf{s}^*_{q_j}$, then $F_{q_j}(\mathcal{A}_{q_j})$ is a non-negative, monotone, submodular set function. That is, for any $\mathcal{A}_{q_j} \subseteq \mathcal{B}_{q_j}$ and any additional call $x \notin \mathcal{B}_{q_j}$,*

$$\begin{aligned} &F_{q_j}(\mathcal{A}_{q_j} \cup \{x\}) - F_{q_j}(\mathcal{A}_{q_j}) \\ &\quad \geq F_{q_j}(\mathcal{B}_{q_j} \cup \{x\}) - F_{q_j}(\mathcal{B}_{q_j}). \end{aligned} \tag{33}$$

Proof Sketch. The marginal gain of adding a call x is the conditional mutual information

$$\begin{aligned} &I(\mathbf{s}^*_{q_j}; x \mid \mathcal{A}_{q_j}) \\ &\quad = H(\mathbf{s}^*_{q_j} \mid \mathcal{A}_{q_j}) - H(\mathbf{s}^*_{q_j} \mid \mathcal{A}_{q_j} \cup \{x\}). \end{aligned} \tag{34}$$

Under conditional independence of observations given $\mathbf{s}^*_{q_j}$, conditional mutual information exhibits diminishing returns as the conditioning set grows. Therefore, the information-gain objective is monotone and submodular. □

For the idealized objective $\sum_j F_{q_j}(\mathcal{A}_{q_j})$, the classical result of Nemhauser et al. (1978) implies that greedy maximisation under a cardinality budget achieves a $(1 - 1/e)$ approximation to the optimal allocation. Directly computing the marginal information gain, however, requires maintaining a posterior distribution over latent BT scores, which is impractical in our online VLM reranking setting.

AMBER therefore uses the view-conditioned scheduling score U_{q_j} in Eq. (19) as a tractable proxy for this marginal gain. Specifically, H_{pair} estimates the uncertainty of the exact next view that would be reranked, while w_{head} biases the

allocation toward comparisons that are more consequential for head-ranking metrics. The function f_{q_j} in Definition 1 can be viewed as a view-restricted, head-weighted instantiation of F_{q_j}, inheriting submodularity under the same conditional independence assumption. Thus, the submodular analysis motivates the greedy scheduling design, while the empirical results validate the effectiveness of the practical proxy.

Limitations of the theoretical analysis. The results above rest on Assumption 1. In practice, VLM outputs may exhibit positional bias and context sensitivity, meaning the true pairwise distributions deviate from the BT model. Furthermore, Proposition 4 assumes conditional independence across calls, whereas AMBER's Elo state explicitly couples successive observations. These gaps motivate the empirical evaluation in Section 4, which provides complementary evidence for AMBER's effectiveness under realistic conditions.