

IntentLens: Grounding Underspecified Multimodal Queries for Recommendation via Tool-Augmented Reasoning

Xiao Chen[1]*, **Dong Fang**[2], **Haitao Li**[2]*, **Qing Li**[1,3]
[1]The Hong Kong Polytechnic University, [2]LightSpeed,
[3]The Hong Kong Polytechnic University Shenzhen Research Institute
{shawn.chen@connect.polyu.hk, loganli@lightspeed-studios.com}

Abstract

As intelligent personal assistants are increasingly deployed in real-world environments, recommendations move from passive preference matching to grounding user intent from underspecified multimodal evidence. In these settings, users often provide only a reference image and a fuzzy natural-language query, leaving crucial visual attributes, personalized preferences, and item-side constraints implicit. Existing recommenders either rely on historical interactions enriched with multimodal item features or presuppose fully specified textual requests, which are insufficient to resolve such queries for accurate item matching. To bridge this gap, we present **IntentLens**, a tool-augmented recommendation framework that grounds underspecified multimodal queries into query-relevant evidence before user-item matching. IntentLens adopts a *ground-then-rank* design: a shared multimodal LLM orchestrates visual, user-memory, and item-attribute tools to (i) recover latent user intent from the image and language query, and (ii) enrich each candidate with fine-grained, query-conditioned evidence; a lightweight ranker then scores candidates over the grounded representations. To support this new setting, we further construct two benchmarks with underspecified multimodal queries, GOOGLEREVIEW-MMTOOL and YELP-MMTOOL. Comprehensive experiments on the constructed benchmarks show that IntentLens outperforms strong MLLM and retrieval baselines, and remains robust across unseen categories and datasets.

1 Introduction

Personalized recommendation is moving beyond passive preference matching toward **proactive intent understanding** [1, 2, 3]. This trend is particularly relevant in physical settings, where camera-equipped wearables (e.g., Meta smart glasses and Apple Vision Pro) enable users to issue queries grounded in their immediate surroundings [4]. In these settings, users rarely fully specify their intent; instead, they often express their needs through a reference image paired with a fuzzy natural-language query. As a result, recommender agents must infer user intent from partial visual context and underspecified textual instructions, motivating tighter integration of multimodal reasoning into the recommendation process.

Despite the practical importance of this paradigm, existing recommendation systems [5, 6, 7] are not well-suited to this setting. Existing methods mostly rely on simplifying assumptions: they either

*Corresponding authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

adopt a *passive matching* approach, where multimodal features are fused mainly to enrich item representations guided by historical interactions [8, 9, 10, 11], or they assume *fully specified textual queries* that explicitly state the user's constraints [12, 13, 14, 15]. In situated interactions, however, a multimodal query is often **underspecified**, providing only partial and ambiguous evidence of the user's latent intent. For example, a user may point their camera at a café dish and ask, *"Something with this kind of vibe, but more affordable and better for a quick lunch nearby."* Direct query–candidate matching can be unreliable in such cases, since important evidence, such as visual style, personal budget preferences, and item-specific attributes, may remain implicit or unavailable in the raw query.

Addressing underspecified multimodal queries requires recommender systems to go beyond passive matching and actively gather the evidence needed to infer user intent. In realistic recommendation scenarios, this evidence often comes from three sources: the query image, the user's interaction history, and item-side data. Accordingly, we equip the system with three tool categories: visual analysis tools that extract preference cues from the query image, memory retrieval tools that gather query-relevant information from user history, and item attribute tools that retrieve fine-grained metadata such as reviews to supplement sparse item descriptions.

In this work, we focus on recommendation systems that infer missing intent through tool-based evidence acquisition, thereby reducing the need for additional follow-up questions to users. We accordingly propose INTENTLENS, a two-stage *ground-then-rank* framework. In Stage 1, a Multimodal Large Language Model (MLLM) selects and invokes query-side tools (i.e., visual and memory tools) as well as item-side tools to collect structured evidence, and then converts the resulting information into compact textual representations for the query and each candidate. In Stage 2, a lightweight dual-encoder ranker scores candidates against the grounded query representation, separating evidence acquisition from efficient ranking.

Pretrained MLLMs support basic tool invocation but are not trained to acquire evidence for personalized recommendations. Fine-tuning MLLMs is challenging in recommendation settings with many candidates, where rewards are sparse and effective planning requires exploring a large space of tool decisions. To better align tool decisions with downstream recommendation quality, we introduce a ranker-aligned Reinforcement Fine-Tuning (RFT) scheme that adapts the MLLM into a tool planner. We design a counterfactual ranking reward that compares each tool-using rollout with a no-tool baseline, making the value of acquired evidence explicit. We optimize this objective with trajectory-level Group Relative Policy Optimization (GRPO), restricting gradient computation to policy-generated tokens by masking out environment-returned observations. Comprehensive experiments on our constructed benchmarks show promising recommendation quality of INTENTLENS.

Our main contributions are as follows:

- We propose INTENTLENS, a two-stage framework that formulates multimodal recommendation under query underspecification as a problem of tool-augmented evidence grounding, separating intent grounding from candidate scoring for better scalability.
- We construct two new benchmarks, GOOGLE REVIEW-MMTOOL and YELP-MMTOOL, for evaluating recommendation with underspecified multimodal queries.
- We introduce a ranker-aligned RFT method for enhancing tool-using MLLMs, combining a counterfactual ranking reward with observation-masked GRPO to learn when and which tools improve recommendation quality.

2 Related Work

Multimodal Recommendation. Multimodal recommendation leverages heterogeneous signals such as text and images to enrich user preference and item representations [10, 16, 17]. Early approaches incorporate multimodal content as auxiliary item-side features [18, 19, 20], while subsequent work develops graph-based architectures to improve representation quality [21]. These approaches primarily emphasize learning representations for user–item matching [22]. Recent work introduces MLLMs into multimodal personalization and recommendation [23, 24, 25], and VisualLens [15] further infers user preferences from task-agnostic visual histories. Our work complements representation learning with query-conditioned evidence acquisition, using tools to ground underspecified multimodal intent before ranking.

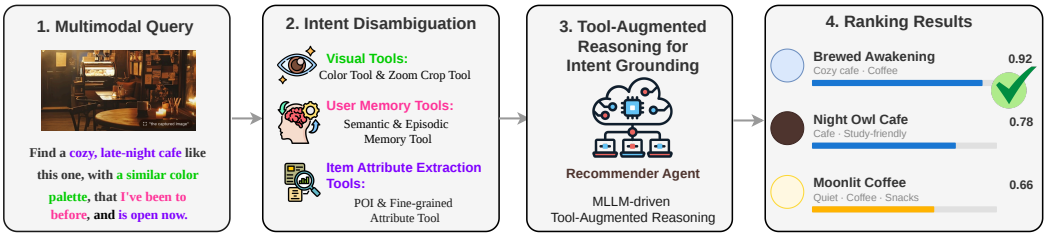


Figure 1: IntentLens leverages three types of tools to disambiguate underspecified multimodal queries, producing a clarified user intent and fine-grained candidate item attributes through MLLM-based tool-augmented reasoning, thereby enabling more accurate recommendations.

Table 1: Comparison of IntentLens with representative multimodal recommendation settings and multimodal ambiguity/tool-use benchmarks. **Image in Query**: the benchmark explicitly includes image input in the query. **Underspecified Intent**: the primary source of under-specification in the benchmark; ○ indicates partial support. **User Profile**: how user-specific information is accessed during inference. **Tool-Augmented Intent Grounding**: whether correct inference requires external tools to ground evidence and recover the underlying intent.

Benchmark	Image in Query	Underspecified Intent	User Profile	Tool-Augmented Intent Grounding
PC^2L [16]	✗	✗	Selected images	✗
COURIER [44]	✗	✗	Multimodal embedding	✗
VisualLens [15]	✗	✗	Retrieved spectrum	✗
RectoolBench [43]	✗	Fuzzy task	✗	✓
Imagery as Inquiry [45]	✓	Fuzzy preference	✗	✗
ToolMMBench [40]	✓	Fuzzy task	✗	✓
VisualToolBench [41]	✓	Fuzzy task	✗	✓
M^3-Bench [42]	✓	Fuzzy task	✗	✓
***IntentLens* (Ours)**	✓	Fuzzy preference + Fuzzy item constraints	Dual-level user memory	✓

Underspecified Intent in Recommendation. Conversational recommender systems resolve ambiguous preferences through multi-turn clarification [26, 27, 28], an approach also studied in information retrieval [29]. History-based methods instead infer preferences from past interactions [30, 31, 32]. Recent multimodal benchmarks examine related challenges: RUMS [33] studies underspecified and misspecified requests; MMA [34] and MUCAR [35] evaluate cross-modal ambiguity resolution; and VAGUE [36] examines how visual context supports understanding implicit intent. We focus on single-turn personalized recommendation, using tools to recover implicit preferences and item constraints without follow-up dialogue.

Tool-Augmented Multimodal Reasoning. Visual ChatGPT [37], HuggingGPT [38], and GPT4Tools [39] enable language models to coordinate external models and tools through prompting or instruction tuning. MLLM-Tool [40] further incorporates multimodal inputs into tool-agent learning. Related benchmarks evaluate these capabilities across general-purpose tasks: VisualToolBench [41] includes underspecified requests requiring follow-up clarification, while M^3-Bench [42] evaluates multimodal tool orchestration under the MCP protocol. RecToolBench [43] further evaluates recommendation-specific tool orchestration under fuzzy user instructions in MCP-based environments. Further discussion of related benchmarks is described in Appendix A.

3 Multimodal Recommendation with Underspecified User Intent

3.1 Problem Definition

Given a user u with historical interactions $\mathcal{H}_u$, a multimodal query $x = (I, q)$ consisting of a reference image I and a natural-language request q, and a candidate set $\mathcal{C} = \{c_i\}_{i=1}^{N}$ where each c_i

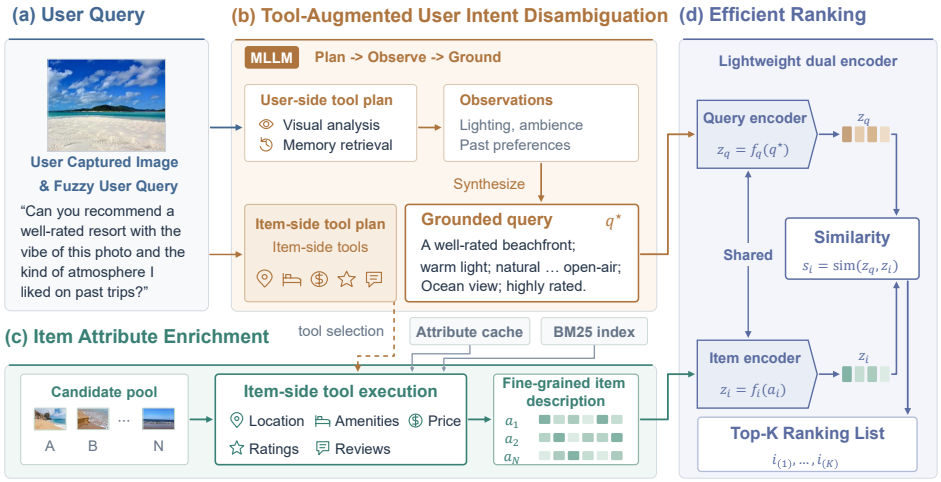


Figure 2: The inference pipeline of INTENTLENS. The MLLM first performs tool-augmented reasoning to produce a clarified query representation and select relevant item-side tools. The selected tools are then applied to the candidate pool in a decoupled pass, yielding fine-grained item representations. Finally, a lightweight dual-encoder ranker scores candidates against the grounded query to enable efficient recommendations.

carries textual description d_i and visual content v_i, the goal is to produce a ranking over $\mathcal{C}$ that aligns with the user's latent preferences.

The key challenge is that the query provides only an incomplete and ambiguous specification of the user's true intent. We decompose the latent intent as $z_u = (z_u^{\text{vis}}, z_u^{\text{hist}}, z_u^{\text{attr}})$, corresponding to: (i) visual preferences inferred from the reference image, encoding perceptual cues that are difficult to verbalize; (ii) personalized preferences reflected in historical interactions, capturing stable patterns or salient episodic signals; and (iii) factual requirements on entity-level attributes that demand fine-grained matching on the candidate side. In this work, we address this problem from a *tool-augmented reasoning* perspective: the system actively acquires evidence to disambiguate intent components and grounds its ranking on the enriched understanding.

3.2 Multimodal Tools for User Intent Disambiguation

We define a structured tool set aligned with the three components of z_u:

Visual tools extract query-conditioned perceptual cues from the reference image, $\mathcal{E}^{\text{vis}} = T_{\text{vis}}(I, q)$, resolving underspecified visual preferences (e.g., color temperature) not stated in the text.

User memory tools retrieve preference evidence from $\mathcal{H}_u$: $\mathcal{E}^{\text{hist}} = T_{\text{hist}}(I, q, \mathcal{H}_u)$. Following A-Mem [46], we encode interactions into incrementally updated memory notes at two levels: episodic memories for interaction-specific contexts, and semantic memories for persistent preference patterns.

Item attribute tools provide fine-grained attribute evidence for candidate matching: $\mathcal{E}^{\text{attr}} = T_{\text{attr}}(\mathcal{C})$. These are deterministic extractors applied in a decoupled pass without MLLM reasoning. Detailed implementations of each tool are provided in Appendix B.1.

4 The Proposed IntentLens Framework

We propose **IntentLens**, a two-stage framework for multimodal recommendation with underspecified user queries. The overall framework is shown in Figure 2. In the first stage, an MLLM performs tool-augmented reasoning to disambiguate user intent and produces a grounded query representation. As part of this stage, the model predicts item-side tools relevant to the current query; these tools are then applied to the candidate pool in a decoupled pass, yielding fine-grained item representations. In the second stage, a lightweight dual-encoder ranker scores the grounded query against enriched candidate-item representations to enable efficient retrieval over large candidate pools.

4.1 Intent Grounding via Tool-Augmented Reasoning

Given a multimodal query $x = (I, q)$, IntentLens uses an MLLM to ground user intent from both visual and textual inputs. To keep online computation tractable, we separate user-side intent disambiguation from item-side attribute extraction.

User Intent Disambiguation. At inference time, the model resolves ambiguities in visual preference and user preference signals through user-side tools. We denote the user-side tool set by $\mathcal{T}_{\text{user}} = \{T_{\text{vis}}, T_{\text{hist}}\}$, where T_{vis} extracts evidence from the query image and T_{hist} retrieves relevant signals from the user history $\mathcal{H}_u$. Rather than multi-turn sequential tool reasoning, we adopt a two-round procedure for efficiency.

i) In the first round, the policy π_θ predicts an ordered tool plan, conditioned on the multimodal query (detailed prompt in Appendix C):

$$\mathcal{A}_{\text{user}} \sim \pi_\theta(\cdot \mid I, q, u; \mathcal{T}_{\text{user}}) . \tag{1}$$

Each action $a_k \in \mathcal{A}_{\text{user}}$ is then executed by the external tool environment, producing observations

$$o_k = \texttt{Execute}(a_k, I, q, \mathcal{H}_u), \quad \forall a_k \in \mathcal{A}_{\text{user}}, \qquad \mathcal{O}_q = \{o_k\}_{k=1}^{|\mathcal{A}_{\text{user}}|}. \tag{2}$$

ii) In the second round, observations $\mathcal{O}_q$ are appended to the context, and the policy synthesizes them into a grounded query representation:

$$\tilde{x}_q \leftarrow \pi_\theta(\cdot \mid I, q, \mathcal{O}_q) . \tag{3}$$

The candidate set $\mathcal{C}$ is excluded at this stage to avoid context expansion from large candidate pools.

Decoupled Item Attribute Enrichment. π_θ also predicts an item-side tool set $\mathcal{T}_{\text{item}} \subseteq \mathcal{T}_{\text{all}}$ relevant to the current query. Unlike user-side tools, which require tool-augmented reasoning during inference, item-side tools are lightweight deterministic extractors. At inference time, the selected tools expose cached item attributes and review evidence without additional MLLM calls. For each candidate, we combine its base description and attributes, and the cached outputs associated with the selected item-side tools. We also include precomputed candidate-image color features corresponding to visual tools executed on the query side. The resulting representation is

$$\tilde{x}_i = \text{Render}_{\text{item}}(c_i, \mathcal{T}_{\text{item}}, \mathcal{O}_q) , \qquad \forall\, c_i \in \mathcal{C}, \tag{4}$$

where $\mathcal{O}_q$ denotes the query-side tool observations, and $\text{Render}_{\text{item}}$ assembles the available fields in a fixed textual order. This decoupling keeps online MLLM inference focused on query understanding and tool selection, while item enrichment scales linearly with the candidate pool size for a fixed tool inventory and bounded field lengths.

4.2 Reinforcement Fine-Tuning for Intent Grounding

While pretrained MLLMs support basic tool use, this alone does not guarantee that acquired evidence benefits recommendation quality. In our setting, each rollout τ produced by π_θ contains a clarified query q^* grounded in user-side evidence and an item-side tool set $\mathcal{T}_{\text{item}}$ for candidate enrichment. To strengthen MLLM's tool reasoning ability, we apply reinforcement fine-tuning (RFT) with rewards tied to downstream ranking.

Counterfactual Ranking Reward. A frozen dual-encoder ranker $\hat{g}_\phi$ (Sec. 4.3) evaluates each rollout over the full candidate pool $\mathcal{C}$. Given the relevant items $\mathcal{G} \subseteq \mathcal{C}$, we define ranking utility as $U(\tau) = 1/\min_{c\in\mathcal{G}} \text{rank}_\tau(c)$. To measure the benefit of query-side evidence, we compare $U(\tau)$ with two baselines: the same clarified query without tool evidence, yielding $U_\varnothing(\tau)$, and the original request without tool evidence, yielding $U_{\text{req}}(\tau)$. All evaluations use identical candidate representations. We define the evidence credit as

$$\Delta_{\text{cf}}(\tau) = \min\Big\{U(\tau) - U_\varnothing(\tau),\, [U(\tau) - U_{\text{req}}(\tau)]_+\Big\} , \tag{5}$$

where $[z]_+ = \max(0, z)$. Thus, positive credit requires improvement over both baselines. The ranking reward is

$$r_{\text{rank}}(\tau) = U(\tau) + \beta\ \text{clip}(\Delta_{\text{cf}}(\tau), -\delta, \delta) , \tag{6}$$

with $\beta = 0.2$ and $\delta = 0.1$. The evidence bonus is zero when no usable evidence is returned.

Query-Side and Item-Side Tool Rewards. We reward accurate tool selection on both sides while penalizing unnecessary tools. For each side $s \in \{q, i\}$, we define

$$r_s(\tau) = \mathrm{F1}(\mathcal{T}_s(\tau), \mathcal{T}_s^*) - \gamma \frac{|\mathcal{T}_s(\tau) \setminus \mathcal{T}_s^*|}{|\mathcal{T}_s^{\mathrm{all}}|}, \tag{7}$$

where $\mathcal{T}_s^*$ is the target tool set, $\mathcal{T}_s^{\mathrm{all}}$ is the full tool inventory for side s, and $\gamma = 0.3$. Query-side selection counts only successfully executed tools retained in the observations; optional tools outside the target set are excluded from scoring. We define $\mathrm{F1}(\varnothing, \varnothing) = 1$.

In total, we combine ranking quality, tool selection, and format validity:

$$r_{\mathrm{main}}(\tau) = r_{\mathrm{rank}}(\tau) + \lambda_q r_q(\tau) + \lambda_i r_i(\tau) + w_{\mathrm{fmt}} r_{\mathrm{fmt}}(\tau), \tag{8}$$

where $\lambda_q = \lambda_i = 0.05$ weight the query-side and item-side tool-selection rewards, and $w_{\mathrm{fmt}} = 0.05$ weights the format reward. The format reward $r_{\mathrm{fmt}}(\tau)$ equals one when the first-round tool plan and second-round response satisfy the required formats, and zero otherwise. Ranking and tool-selection credit are masked out for format-invalid rollouts; execution penalties are handled separately.

Trajectory-Level Masked GRPO. We use GRPO [47] with losses restricted to policy-generated tokens. Each rollout $\tau_i = (y_i^{(1)}, \mathcal{O}_i, y_i^{(2)})$ contains two policy responses and intervening tool observations. Let $\mathcal{P}_i$ denote the token positions in the two responses. Observations remain in the conditioning context but are excluded from both the policy and KL losses. For a micro-batch $\mathcal{B}$ of rollouts, per instance with group-normalized advantages $\hat{A}_i = (r(\tau_i) - \mathrm{mean}(\mathbf{r}))/\mathrm{std}(\mathbf{r})$, the objective is

$$\mathcal{J}_{\mathrm{MGRPO}}(\theta; \mathcal{B}) = \frac{\sum_{i \in \mathcal{B}} \sum_{l \in \mathcal{P}_i} \ell\Big(\rho_{i,l}(\theta), \hat{A}_i\Big)}{\sum_{i \in \mathcal{B}} |\mathcal{P}_i|} - \mu \, \widehat{\mathbb{D}}_{\mathrm{KL}}^{\mathcal{P}_{\mathcal{B}}}[\pi_\theta \,\|\, \pi_{\mathrm{ref}}], \tag{9}$$

where $\rho_{i,l}(\theta) = \pi_\theta(y_{i,l} \mid x_i, \tau_i^{<l}) / \pi_{\mathrm{old}}(y_{i,l} \mid x_i, \tau_i^{<l})$, ℓ is the clipped policy surrogate, and $\mathcal{P}_i$ contains the policy-generated token positions from both rounds. The objective averages over policy-generated tokens across the batch. We set rollout group size as G=4 for training efficiency.

4.3 Lightweight Ranking over Grounded Evidence

After Stage 1, the query is grounded into $\tilde{x}_q$ and each candidate is enriched into $\tilde{x}_i$. Stage 2 delegates scoring to a dual-encoder ranker g_ϕ operating on these representations, with no further tool invocation. We pre-train g_ϕ on grounded representations produced by executing the ground-truth tool chains, then freeze it as $\hat{g}_\phi$ to serve as the stationary reward signal during RFT.

Dual-Encoder Scorer. We use a shared encoder to map the grounded query and enriched candidate descriptions into embeddings:

$$\mathbf{u}_q = \mathrm{Norm}(\mathrm{CLS}(\mathrm{Enc}_\phi(\tilde{x}_q))), \qquad \mathbf{v}_i = \mathrm{Norm}(\mathrm{CLS}(\mathrm{Enc}_\phi(\tilde{x}_i))), \qquad s_i = \mathbf{u}_q^\top \mathbf{v}_i. \tag{10}$$

where Enc_ϕ is a shared BGE-M3 encoder backbone with CLS pooling and L2 normalization. Candidates are ranked by the inner product of the query and candidate embeddings.

Training Objective. We train g_ϕ with the standard InfoNCE loss [48]. For each query, positives are drawn from the ground-truth set $\mathcal{G} \subset \mathcal{C}$, and hard negatives are mined from same-category candidates ($\mathcal{N} \cap \mathcal{G} = \emptyset$) to improve representation discriminability.

5 Benchmark Construction

Existing recommendation benchmarks do not provide multimodal queries with underspecified user intent. Thus, we construct two benchmarks, GOOGLE REVIEW-MMTOOL and YELP-MMTOOL, from Google Local Review [51] and Yelp 2019 [52]. The benchmark statistics are detailed in Table 3.

Table 2: Hit rates and MRRs of INTENTLENS *vs.* multiple baselines on Google Review-MMTool and Yelp-MMTool $^*(p\text{-}value \ll 0.05)$. Results for INTENTLENS are reported over 5 runs. The best performance is highlighted in bold.

			Google Review-MMTool				Yelp-MMTool			
	Modality	Size	Hit@1	Hit@3	Hit@10	MRR	Hit@1	Hit@3	Hit@10	MRR
(i) Basic baselines										
Random	-	-	0.0	10.3	27.6	9.9	2.8	7.0	18.3	9.2
Rank by Rating	-	-	0.0	3.5	31.0	9.2	2.8	8.5	31.0	11.0
Popularity	-	-	0.0	6.9	41.4	11.4	4.2	16.9	35.2	14.5
(ii) Multimodal matching baselines										
CLIP Retrieval [49]	T+I	428M	3.5	13.8	24.1	12.8	14.1	26.8	47.9	24.6
BGE Retrieval [50]	T	335M	13.8	27.6	37.9	25.7	19.7	36.6	57.8	32.2
(iii) Zero-shot / Fine-tuned recommenders										
UniMP [23]	T+I	3B	24.1	31.0	61.4	33.9	28.2	**52.1**	67.6	43.7
VIP5 [11]	T+I	-	3.5	24.1	51.7	19.6	4.2	14.1	33.8	14.4
Qwen2.5-VL (zero-shot)	T+I	3B	13.8	27.6	55.2	27.3	12.7	32.4	54.9	26.9
Qwen2.5-VL (zero-shot)	T+I	7B	13.8	31.0	58.6	28.0	12.7	22.5	56.3	25.4
(iv) LLM/MLLM ranking										
GPT-5.4	T	-	27.6	41.4	58.6	38.1	22.5	39.4	50.7	33.6
GPT-5.4 (w/ evidence)	T	-	**31.0**	**44.8**	55.2	**41.6**	31.0	43.7	67.6	41.5
GPT-4o	T+I	-	30.9	33.6	40.4	36.8	31.0	45.1	66.2	41.4
Our method										
INTENTLENS (3B)	T+I	3B	20.7 ± 2.0	41.4 ± 1.6	57.0 ± 1.8	34.5 ± 1.1	23.9 ± 0.9	32.4 ± 0.8	53.5 ± 0.6	33.5 ± 0.4
INTENTLENS (7B)	T+I	7B	$\mathbf{31.0 \pm 1.7}$	$\mathbf{44.8 \pm 1.4}$	$\mathbf{62.1 \pm 1.1}$	40.2 ± 1.2	$\mathbf{33.8 \pm 1.4}$	$\mathbf{52.1 \pm 1.0}$	$\mathbf{70.4 \pm 0.8}$	$\mathbf{46.7 \pm 1.1}$
Oracle Ranker*	T+I	-	37.9	41.4	65.5	44.4	43.7	53.5	76.1	53.7

Table 3: Benchmark statistics for GOOGLE REVIEW-MMTOOL and YELP-MMTOOL.

Dataset	Users	Interact.	Queries	Categ.	Avg.#GT	Avg.#Cand.
GR-MMTOOL	3,000	104,025	773	1,556	3.2	40
YELP-MMTOOL	2,315	94,184	1,727	497	2.0	63

User Behavior Logs and Memory. For each user, reviews are sorted chronologically and linked to business metadata (name, category, description) and photos (user-uploaded in Google; store-profile images in Yelp). From these logs, we construct a dual-memory representation: (1) *episodic memory* storing individual interactions with associated text and images, and (2) *semantic memory* summarizing long-term preferences (favored categories, budget, atmosphere, distance tolerance) via LLM prompting over the user's prior review history (excluding target interactions).

Query Synthesis. Each query consists of a reference image (visual anchor) and an underspecified natural-language request that leaves key constraints implicit. Queries are organized into four complexity tiers by the tool categories required for resolution: (1) visual only, (2) visual + user memory, (3) visual + item attribute, and (4) visual + user memory + item attribute. We use a three-stage forward-synthesis pipeline: (i) grounded seeds are built by executing the corresponding tool combination on references retrieved via caption-similarity filtering from the user's history; (ii) an LLM verbalizes the tool evidence into an explicit intent description, then rewrites it into a natural underspecified query without exposing tool parameters; (iii) automatic leak validation removes queries whose surface text reveals the target item or tool chain, and post-hoc human verification confirms that the intended reasoning path remains recoverable.

Candidate Pool and Ground Truth. For each query, we construct a candidate pool from nearby same-category businesses, containing positives, hard negatives from three sources (*visual-similar*, *memory-conflicting*, *context-conflicting*), and random distractors. Each candidate $c_i = (x_i, \mathcal{I}_i, a_i)$ carries text, an optional image set, and structured attributes. Ground truth is set-valued: a candidate is positive only if it jointly satisfies textual relevance, visual alignment, preference consistency, and item-side attribute constraints, determined by rule-based attribute filtering followed by LLM semantic verification. Details of data synthesis are in Appendix B.

Human validation of benchmark quality. Three external annotators independently audited 400 queries, sampled proportionally across T1–T4 within each benchmark (200 per benchmark). Of these, 374 (93.5%) met the majority-vote criteria for coherence, naturalness, comprehensibility, and need

for external evidence; dimension-wise ordinal Krippendorff's α ranged from 0.67 to 0.74. Under label-blind assessment, annotators confirmed 1,224 of 1,290 labeled positives (94.9%; $\alpha = 0.68$).

6 Experiment

Based on the constructed benchmark, we conduct experiments to answer the following questions: **Q1:** To what extent can tool-augmented intent grounding improve recommendations with underspecified multimodal queries? **Q2:** How does each component of INTENTLENS contribute to recommendation quality? **Q3:** Does INTENTLENS generalize across unseen categories, datasets and tool categories?

6.1 Implementation and Baselines

Implementation Details. We implement INTENTLENS with two MLLM backbones, Qwen2.5-VL-3B and Qwen2.5-VL-7B [53]. For RFT, we adopt the Verl framework with FSDP sharding over 4 H20 GPUs and optimize the policy using MGRPO. Rollouts are generated with vLLM. The reward model is a frozen dual-encoder ranker $\hat{g}_\phi$, trained on the training split and frozen during RFT. The available tool set includes 5 visual tools, 3 user-memory tools, and 6 item-attribute tools.

Baselines. We evaluate five categories of baselines. For each evaluated query, all methods rank the same complete candidate pool.

(i) **Basic baselines:** *Random* uniformly shuffles the candidates; *Rating* ranks candidates by descending average star rating; and *Popularity* ranks candidates by descending review count.

(ii) **Multimodal retrieval baselines:** *CLIP Retrieval* [49] uses CLIP ViT-L/14 to encode the reference image, textual request, and candidate descriptions. Candidates are ranked by the weighted average of image–candidate and text–candidate cosine similarities. *BGE Retrieval* [50] concatenates the request with a textual caption of the reference image, encodes this query and candidate descriptions using BGE-large-en-v1.5, and ranks candidates by cosine similarity.

(iii) **Zero-shot / Fine-tuned multimodal recommenders:** We evaluate task-adapted implementations of *UniMP* [23] and *VIP5* [11], alongside *Qwen2.5-VL-3B* and *Qwen2.5-VL-7B* in a zero-shot setting. Our UniMP-style implementation uses Qwen2.5-VL-3B with the reference image, user request, and cleaned review history. Our VIP5-style implementation augments T5-base [54] with visual prefix tokens obtained by projecting frozen CLIP image features. The Qwen [53] baselines rank candidates without user history or tool use.

(iv) **LLM/MLLM ranking:** We evaluate three ranking baselines: text-only *GPT-5.4*, multimodal *GPT-4o* [55], and tool-augmented *GPT-5.4*. The first two directly rank candidates using the request and candidate metadata, with GPT-4o additionally receiving the reference image. The tool-augmented variant ranks candidates using the clarified query, query-side tool outputs, and candidate-side evidence generated by the fixed IntentLens policy.

(v) **Oracle Ranker:** We evaluate frozen BGE-M3 [50] with the ground-truth clarified query, annotated query-side tool evidence, and annotated item-side tool selections. This isolates the effect of annotated versus policy-generated inputs under the same ranker.

Evaluation Metrics. We report two ranking metrics. **Hit@k** measures whether at least one relevant item appears in the top-k positions: $\mathrm{Hit@}k = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\big[\mathrm{rank}(r_i) \le k\big]$, $\mathrm{rank}(r_i)$ is the position of the highest-ranked relevant item for query i. We report Hit@1, Hit@3 and Hit@10. **MRR** averages the reciprocal of this rank as $\mathrm{MRR} = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{\mathrm{rank}(r_i)}$.

6.2 Main Results

Comparison with retrieval and zero-shot baselines. Table 2 shows that INTENTLENS (7B) outperforms CLIP, BGE, and both zero-shot Qwen2.5-VL variants across the board. These gains demonstrate the necessity of incorporating external, query-relevant evidence rather than relying solely on direct query–candidate similarity for underspecified multimodal recommendation.

Comparison with competitive baselines. Compared to other non-oracle approaches, INTENTLENS (7B) sets the state of the art on most metrics: it achieves the highest Hit@10 on Google Review-MMTool, and tops Hit@1, Hit@10, and MRR on Yelp-MMTool. Notably, these results rely on

Table 4: Ablation study for analyzing different components of INTENTLENS Model on Google Review-MMTool. (a) Tool group ablations. (b) Training objective ablations.

(a) Tool categories

Variant	Hit@1	Hit@3	Hit@10	MRR
INTENTLENS (7B)	**31.0**	**44.8**	**62.1**	**40.2**
w/o visual	24.1	37.9	58.6	35.7
w/o memory	26.3	37.9	59.6	38.1
w/o item	27.6	41.4	58.4	38.0
w/o grounding	21.9	36.8	51.7	33.5

(b) Training objectives

Variant	Hit@1	Hit@3	Hit@10	MRR
INTENTLENS (7B)	**31.0**	**44.8**	**62.1**	**40.2**
w/o RFT	24.1	33.3	58.6	33.1
w/o counterfactual	24.1	36.1	58.6	35.0
w/o tool reward	24.1	37.9	62.1	35.5
w/o obs. masking	28.9	38.6	60.5	35.1

pairing tool augmentation with the 7B backbone, indicating that tool use becomes fully effective only when supported by sufficient model capacity.

Model scaling and oracle gap. Upgrading the backbone from 3B to 7B consistently benefits INTENTLENS, lifting MRR from 34.5 to 40.2 on Google Review-MMTool and from 33.5 to 46.7 on Yelp-MMTool. While scaling yields pronounced gains, particularly on Yelp-MMTool, the 7B model still lags behind the Oracle Ranker by 4.2 and 7.0 MRR points on the respective datasets, indicating remaining room to close the grounding gap.

6.3 Ablation Study

We evaluate the contributions of individual components of INTENTLENS on Google Review-MMTool. Results are reported in Table 4.

Tool and grounding ablations. We consider four variants: (i) **w/o visual tools** disables visual tools; (ii) **w/o memory tools** removes user-memory retrieval tools; (iii) **w/o item tools** replaces tool-augmented candidate representations with raw descriptions; (iv) **w/o query grounding** bypasses the entire query-side tool pipeline. Removing query grounding causes the largest decline across all reported metrics among these variants, supporting its role in resolving underspecified requests before ranking. Removing visual, memory, or item tools also reduces all reported metrics, indicating that both query-side and candidate-side evidence contribute to recommendation quality. Among the individual tool categories, removing visual tools produces the largest MRR decline.

Training objective ablations. We consider four additional variants: (v) **w/o RFT** uses only the SFT-initialized planner; (vi) **w/o counterfactual baseline** removes the no-tool reference from the reward; (vii) **w/o item-side tool reward** drops the auxiliary tool-selection reward; (viii) **w/o observation masking** includes tool-returned observation tokens in both the policy and KL losses, while retaining them in the context as in the full model. Removing RFT causes the largest MRR decline among the training variants, showing that reinforcement fine-tuning improves recommendation quality beyond SFT initialization. Removing the counterfactual baseline reduces all reported metrics. Removing the item-side tool reward lowers Hit@1, Hit@3, and MRR, suggesting that this reward helps place relevant candidates earlier in the ranking. Disabling observation masking also reduces MRR, supporting the use of loss masking for tool-returned tokens rather than treating them as policy-generated outputs.

Query representation control. On Google Review-MMTool, we hold the dual encoder, candidate pool, and candidate-side evidence fixed and vary only the query representation. Raw, predicted-grounded, and oracle-grounded queries yield MRRs of 33.6%, 40.2%, and 44.4%, respectively. This controlled comparison isolates the contribution of query grounding from changes to the ranker or candidate-side evidence.

Alternative-ranker transfer. Using frozen `all-MiniLM-L6-v2`, neither tuned on our benchmark nor used during RFT, we compare cached query and candidate evidence from the pre-RFT instruction-base model and the RFT planner. With candidate pools fixed, RFT improves MRR by 4.6 percentage points, indicating transfer beyond the original reward ranker.

Human-judged recommendation quality. Three external annotators blindly compare randomized top-three recommendation lists on 100 stratified queries, 50 per benchmark, without system identities,

Table 5: (a) Generalization analysis under different transfer setups. **Unseen Cat.**: hold out item categories during training. **Cross-Dataset**: train on the other dataset. **Unseen Tools**: remove a subset of tools from training. (b) Efficiency analysis on a single NVIDIA H20 GPU. Stage 1: MLLM rollout for tool planning; Stage 2: frozen dual-encoder scoring.

(a) Generalization Analysis

Test Set	In-Dom.	Uns. Cat.	Cross	Uns. Tool
Google	40.2	36.2	34.7	35.8
Yelp	46.7	43.5	38.9	44.1

(b) Efficiency Analysis

Stage	Latency (ms)	#MLLM Calls	#Tool Calls
1	3,590	2	3–5
2	515	0	0

scores, or automatic labels. By majority vote, INTENTLENS is preferred on 89% of queries versus 6% for BGE Retrieval, providing label-independent evidence of recommendation quality.

Human-written query transfer. We further conduct a user study to evaluate model robustness on human-authored queries. Given only the images and task briefs (with synthetic queries, candidates, and labels hidden), participants drafted 100 requests, of which 78 passed quality verification by two validators. Keeping the candidate pools fixed and applying no retraining or prompt tuning, INTENTLENS-7B still outperforms BGE by 7.3 MRR points on these human queries, confirming that its gains generalize well beyond synthetic query patterns.

6.4 Generalization and Efficiency Analysis

We evaluate INTENTLENS under unseen-category, cross-dataset, and unseen-tool settings, as shown in Table 5(a). MRR decreases under all three settings, with cross-dataset transfer producing the largest decline on both benchmarks. Performance remains closer to the in-domain results under unseen categories and tools, suggesting that INTENTLENS can generalize beyond the categories and tool configurations encountered during training. The larger cross-dataset gap indicates that transfer across platforms remains more challenging in our evaluation.

Table 5(b) reports end-to-end latency of INTENTLENS-7B. The full pipeline processes a query in approximately 4 seconds, with Stage 1 (MLLM rollout and tool execution) dominating at 3.5 s and Stage 2 (dual-encoder scoring over candidates) adding only 515 ms. This two-stage design keeps inference speed tolerable for recommendation scenarios while preserving the utility of tool-augmented reasoning. We record Item Tool F1 and user-side Tool F1 alongside the Tool Call Ratio. As training progresses, both F1 scores rise steadily, while the Tool Call frequency decreases. This pattern indicates that the policy learns not only to select more appropriate tools but also to suppress redundant invocations, achieving more selective and precise tool planning.

7 Conclusion and Limitation

In this paper, we presented INTENTLENS, a two-stage framework that formulates multimodal recommendation under query underspecification as a tool-augmented reasoning problem. The proposed ranker-aligned reinforcement fine-tuning scheme, combining counterfactual ranking reward with observation-masked GRPO, enables the MLLM to learn selective and effective tool use tied to downstream recommendation quality. Experiments on our constructed Google Review-MMTool and Yelp-MMTool demonstrate consistent improvements against most baselines. Our findings are constrained by constructed benchmarks and single-turn evaluation, with limited applicability to real-world multi-turn interactions and cross-domain transfer, alongside practical deployment bottlenecks and privacy challenges tied to user and visual data.

Acknowledgments

The experimental work was partially supported by the Centre for Large AI Models (CLAIM) at The Hong Kong Polytechnic University. This work was also supported by The Hong Kong Polytechnic University Research Centre for Data Science and Artificial Intelligence (Project No. 4-CE42) and the National Natural Science Foundation of China (Grant Nos. T2541073 and T2541036).

References

[1] Xu Huang, Jianxun Lian, Yuxuan Lei, Jing Yao, Defu Lian, and Xing Xie. Recommender ai agent: Integrating large language models for interactive recommendations. *ACM Transactions on Information Systems*, 43(4):1–33, 2025.

[2] Yancheng Wang, Ziyan Jiang, Zheng Chen, Fan Yang, Yingxue Zhou, Eunah Cho, Xing Fan, Yanbin Lu, Xiaojiang Huang, and Yingzhen Yang. Recmind: Large language model powered agent for recommendation. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4351–4364, 2024.

[3] Yuyue Zhao, Jiancan Wu, Xiang Wang, Wei Tang, Dingxian Wang, and Maarten De Rijke. Let me do it for you: Towards llm empowered recommendation via tool learning. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1796–1806, 2024.

[4] Xu Yuan, Yi Wang, Zhuohang Jiang, Haohao Qu, Yujuan Ding, Shanru Lin, Guoliang Xing, Hongxia Yang, Jiannong Cao, Qing Li, et al. A survey on ai smart glasses for wearable intelligence from egocentric sensing to agentic personalization. 2026.

[5] Wenqi Fan, Xiangyu Zhao, Xiao Chen, Jingran Su, Jingtong Gao, Lin Wang, Qidong Liu, Yiqi Wang, Han Xu, Lei Chen, et al. A comprehensive survey on trustworthy recommender systems. *arXiv preprint arXiv:2209.10117*, 2022.

[6] Xiao Chen, Wenqi Fan, Jingfan Chen, Haochen Liu, Zitao Liu, Zhaoxiang Zhang, and Qing Li. Fairly adaptive negative sampling for recommendations. In *Proceedings of the ACM Web Conference 2023*, pages 3723–3733, 2023.

[7] Xiao Chen, Changyi Ma, Wenqi Fan, Zhaoxiang Zhang, and Qing Li. C2kd: Cross-layer and cross-head knowledge distillation for small language model-based recommendation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 17827–17838, 2025.

[8] Chuhan Wu, Fangzhao Wu, Tao Qi, Chao Zhang, Yongfeng Huang, and Tong Xu. Mm-rec: Visiolinguistic model empowered multimodal news recommendation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*, pages 2560–2564, 2022.

[9] Yuqing Liu, Yu Wang, Lichao Sun, and Philip S Yu. Rec-gpt4v: Multimodal recommendation with large vision-language models. *arXiv preprint arXiv:2402.08670*, 2024.

[10] Xin Zhou. Mmrec: Simplifying multimodal recommendation. In *Proceedings of the 5th ACM International Conference on Multimedia in Asia Workshops*, pages 1–2, 2023.

[11] Shijie Geng, Juntao Tan, Shuchang Liu, Zuohui Fu, and Yongfeng Zhang. Vip5: Towards multimodal foundation models for recommendation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9606–9620, 2023.

[12] Sein Kim, Hongseok Kang, Kibum Kim, Jiwan Kim, Donghyun Kim, Minchul Yang, Kwangjin Oh, Julian McAuley, and Chanyoung Park. Lost in sequence: Do large language models understand sequential recommendation? In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 1160–1171, 2025.

[13] Keqin Bao, Jizhi Zhang, Wenjie Wang, Yang Zhang, Zhengyi Yang, Yanchen Luo, Chong Chen, Fuli Feng, and Qi Tian. A bi-step grounding paradigm for large language models in recommendation systems. *ACM Transactions on Recommender Systems*, 3(4):1–27, 2025.

[14] Yangqin Jiang, Yuhao Yang, Lianghao Xia, Da Luo, Kangyi Lin, and Chao Huang. Reclm: Recommendation instruction tuning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15443–15459, 2025.

[15] Wang Bill Zhu, Deqing Fu, Kai Sun, Yi Lu, Zhaojiang Lin, Seungwhan Moon, Kanika Narang, Mustafa Canim, Yue Liu, Anuj Kumar, et al. Visuallens: Personalization through task-agnostic visual history. *arXiv preprint arXiv:2411.16034*, 2024.

[16] An Yan, Zhankui He, Jiacheng Li, Tianyang Zhang, and Julian McAuley. Personalized showcases: Generating multi-modal explanations for recommendations. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2251–2255, 2023.

[17] Hongyu Zhou, Xin Zhou, Zhiwei Zeng, Lingzi Zhang, and Zhiqi Shen. A comprehensive survey on multimodal recommender systems: Taxonomy, evaluation, and future directions. *arXiv preprint arXiv:2302.04473*, 2023.

[18] Ruining He and Julian McAuley. Vbpr: visual bayesian personalized ranking from implicit feedback. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[19] Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua. Mmgcn: Multi-modal graph convolution network for personalized recommendation of micro-video. In *Proceedings of the 27th ACM international conference on multimedia*, pages 1437–1445, 2019.

[20] Jinghao Zhang, Yanqiao Zhu, Qiang Liu, Shu Wu, Shuhui Wang, and Liang Wang. Mining latent structures for multimedia recommendation. In *Proceedings of the 29th ACM international conference on multimedia*, pages 3872–3880, 2021.

[21] Zhiqiang Guo, Jianjun Li, Guohui Li, Chaoyang Wang, Si Shi, and Bin Ruan. Lgmrec: Local and global graph learning for multimodal recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 8454–8462, 2024.

[22] Zheng Yuan, Fajie Yuan, Yu Song, Youhua Li, Junchen Fu, Fei Yang, Yunzhu Pan, and Yongxin Ni. Where to go next for recommender systems? id-vs. modality-based recommender models revisited. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2639–2649, 2023.

[23] Tianxin Wei, Bowen Jin, Ruirui Li, Hansi Zeng, Zhengyang Wang, Jianhui Sun, Qingyu Yin, Hanqing Lu, Suhang Wang, Jingrui He, et al. Towards unified multi-modal personalization: Large vision-language models for generative recommendation and beyond. *arXiv preprint arXiv:2403.10667*, 2024.

[24] Yuyang Ye, Zhi Zheng, Yishan Shen, Tianshu Wang, Hengruo Zhang, Peijun Zhu, Runlong Yu, Kai Zhang, and Hui Xiong. Harnessing multimodal large language models for multimodal sequential recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 13069–13077, 2025.

[25] Chao Zhang, Haoxin Zhang, Shiwei Wu, Di Wu, Tong Xu, Xiangyu Zhao, Yan Gao, Yao Hu, and Enhong Chen. Notellm-2: Multimodal large representation models for recommendation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pages 2815–2826, 2025.

[26] Wenqiang Lei, Xiangnan He, Yisong Miao, Qingyun Wu, Richang Hong, Min-Yen Kan, and Tat-Seng Chua. Estimation-action-reflection: Towards deep interaction between conversational and recommender systems. In *Proceedings of the 13th international conference on web search and data mining*, pages 304–312, 2020.

[27] Kerui Xu, Jingxuan Yang, Jun Xu, Sheng Gao, Jun Guo, and Ji-Rong Wen. Adapting user preference to online feedback in multi-round conversational recommendation. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 364–372, 2021.

[28] Xiaolei Wang, Kun Zhou, Ji-Rong Wen, and Wayne Xin Zhao. Towards unified conversational recommender systems via knowledge-enhanced prompt learning. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 1929–1937, 2022.

[29] Hamed Zamani, Susan Dumais, Nick Craswell, Paul Bennett, and Gord Lueck. Generating clarifying questions for information retrieval. In *Proceedings of the web conference 2020*, pages 418–428, 2020.

[30] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, pages 197–206. IEEE, 2018.

[31] Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. Towards universal sequence representation learning for recommender systems. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 585–593, 2022.

[32] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pages 1893–1902, 2020.

[33] Qianqi Yan, Hongquan Li, Shan Jiang, Yang Zhao, Xinze Guan, Ching-Chen Kuo, and Xin Eric Wang. Hidden in plain sight: Reasoning in underspecified and misspecified scenarios for multimodal llms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 24726–24746, 2025.

[34] Ru Wang, Selena Song, Liang Ding, Mingming Gong, Yusuke Iwasawa, Yutaka Matsuo, and Jiaxian Guo. Mma: Benchmarking multi-modal large language model in ambiguity contexts. In *ICLR 2025 Workshop on Navigating and Addressing Data Problems for Foundation Models*, 2025.

[35] Xiaolong Wang, Zhaolu Kang, Wangyuxuan Zhai, Xinyue Lou, Yunghwei Lai, Ziyue Wang, Yawen Wang, Kaiyu Huang, Yile Wang, Peng Li, et al. Mucar: Benchmarking multilingual cross-modal ambiguity resolution for multimodal large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 15037–15059, 2025.

[36] Heejeong Nam, Jinwoo Ahn, Keummin Ka, Jiwan Chung, and Youngjae Yu. Vague: visual contexts clarify ambiguous expressions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1537–1547, 2025.

[37] Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671*, 2023.

[38] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *Advances in Neural Information Processing Systems*, 36:38154–38180, 2023.

[39] Rui Yang, Lin Song, Yanwei Li, Sijie Zhao, Yixiao Ge, Xiu Li, and Ying Shan. Gpt4tools: Teaching large language model to use tools via self-instruction. *Advances in Neural Information Processing Systems*, 36:71995–72007, 2023.

[40] Chenyu Wang, Weixin Luo, Sixun Dong, Xiaohua Xuan, Zhengxin Li, Lin Ma, and Shenghua Gao. Mllm-tool: A multimodal large language model for tool agent learning. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 6678–6687. IEEE, 2025.

[41] Xingang Guo, Utkarsh Tyagi, Advait Gosai, Paula Vergara, Jayeon Park, Ernesto Gabriel Hernández Montoya, Chen Bo Calvin Zhang, Bin Hu, Yunzhong He, Bing Liu, et al. Beyond seeing: Evaluating multimodal llms on tool-enabled image perception, transformation, and reasoning. *arXiv preprint arXiv:2510.12712*, 2025.

[42] Yang Zhou, Mingyu Zhao, Zhenting Wang, Difei Gu, Bangwei Guo, Ruosong Ye, Ligong Han, Can Jin, and Dimitris N Metaxas. Mˆ 3-bench: Multi-modal, multi-hop, multi-threaded tool-using mllm agent benchmark. *arXiv preprint arXiv:2511.17729*, 2025.

[43] Xiao Chen, Yicheng Zhao, Yingying Wu, Zhendong Chu, Changyi Ma, Qingsong Wen, and Xuan Song. Rectoolbench: Benchmarking recommendation-specific tool orchestration under fuzzy user intent. *arXiv preprint arXiv:2609.30717*, 2026.

[44] Jia-Qi Yang, Chenglei Dai, Dan Ou, Dongshuai Li, Ju Huang, De-Chuan Zhan, Xiaoyi Zeng, and Yang Yang. Courier: contrastive user intention reconstruction for large-scale visual recommendation. *Frontiers of Computer Science*, 19(7):197602, 2025.

[45] Se-eun Yoon, Hyunsik Jeon, and Julian McAuley. Imagery as inquiry: Exploring a multimodal dataset for conversational recommendation. *arXiv preprint arXiv:2405.14142*, 2024.

[46] Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*, 2025.

[47] 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.

[48] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[49] 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 *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[50] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*, 4(5), 2024.

[51] Jiacheng Li, Jingbo Shang, and Julian McAuley. Uctopic: Unsupervised contrastive learning for phrase representations and topic mining. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6159–6169, 2022.

[52] Nabiha Asghar. Yelp dataset challenge: Review rating prediction. *arXiv preprint arXiv:1605.05362*, 2016.

[53] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[54] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[55] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

A More Related Work

A.1 Underspecified Query Handling in MLLMs

Handling underspecified or ambiguous user queries is a well-recognized challenge in information retrieval and dialogue systems. **RUMS** [33] specifically diagnosing underspecified and misspecified scenarios for MLLMs across four implicit reasoning challenges: object absence, referential ambiguity, factual contradiction, and goal infeasibility. A key finding is that clarification is the most effective inference-time intervention. **MMA Benchmark** [34] evaluates whether MLLMs can use images to disambiguate textually ambiguous queries (lexical, syntactic, and semantic), finding that MLLMs achieve only 53.22% accuracy on paired-image disambiguation (vs. 88.97% for humans). **VAGUE** [36] tests whether VLMs can leverage visual context to infer the speaker's implicit intent from indirect expressions (*e.g.*, "turn it off" in a scene with multiple computers), revealing a $\sim$20% gap between VLMs and humans. **MUCAR** [35] further extends this to multilingual cross-modal ambiguity resolution, where both image and text are simultaneously ambiguous and must disambiguate each other.

A.2 Multimodal Tool Calling and Reasoning

Early work on tool-augmented language models primarily focused on text-only interfaces. Systems such as **Visual ChatGPT** [37], **HuggingGPT** [38], and **GPT4Tools** [39] enable LLMs to invoke external tools through prompt-based or instruction tuning paradigms. However, these approaches rely on *purely textual queries*, making tool selection brittle when the user intent heavily depends on visual context. Recent advances have extended tool use to multimodal settings. **MLLM-Tool** [40] and subsequent works incorporate visual perception and improve tool invocation through supervised and reinforcement learning. Despite these impressive advances, existing methods primarily focus on *tool selection accuracy or execution correctness* in general-purpose tasks, rather than reasoning over *underspecified, preference-driven user intent* in real-world recommendation scenarios. Our framework bridges this gap by repurposing tool-augmented MLLMs as an intent planner. Instead of merely executing isolated commands, our planner orchestrates visual, memory, and item tools to gather explicit evidence, translating ambiguous multimodal intent into a structured textual interface that a lightweight recommender can efficiently process.

A.3 Benchmarks for Multimodal Tool Calling

Several benchmarks have been proposed to evaluate MLLM tool-calling capabilities. **VisualToolBench** [41] introduces 1,204 open-ended visual tasks across five domains (STEM, Medical, Finance, Sports, Generalist), covering six tool types including image processing APIs, Python interpreters, and web search. Notably, it includes a *Follow-up Test* category for multi-turn settings where the model must proactively ask clarifying questions when the query is underspecified. **M^3-Bench** [42] is the first benchmark under the MCP protocol, covering 28 MCP servers, 231 tools, and 208 evaluation trajectories, with fuzzy task descriptions designed to stress-test tool argument fidelity and structural consistency.

B Benchmark Details

B.1 Tool Set

Color-based visual tools extract quantitative attributes such as color distribution, brightness, and saturation, which serve as proxies for high-level semantics like ambiance and style. For example, a color histogram captures the distribution of pixel intensities across color channels, enabling precise estimation of brightness and saturation levels.

These attributes are particularly critical for resolving relative and underspecified user queries, where the required reasoning involves comparing distributions rather than recognizing objects.

For instance, a query such as "find a place with a similar vibe but brighter and less heavy" requires analyzing the brightness distribution (via a color histogram) to quantify what "brighter" means, while a request like "something with a similar color palette but more neutral tones" relies on dominant color extraction to compare color composition across images. Similarly, queries such as "not too

Algorithm 1 Review-Grounded Memory Bank Construction

Require: User review history $\mathcal{R}_u = \{(r_i, \mathcal{P}_i, b_i)\}_{i=1}^{N}$ where r_i is review text, $\mathcal{P}_i$ is associated photos, b_i is business metadata; L1 visual tool set $\mathcal{T}_{\text{vis}}$; Teacher MLLM $\mathcal{M}$
Ensure: Memory Bank $\mathcal{B}_u = \{m_1, m_2, \ldots, m_N\}$

for each review $(r_i, \mathcal{P}_i, b_i) \in \mathcal{R}_u$ **do**
  ▷ *Step 1: Extract visual features via L1 tools*
  $\mathbf{v}_i \leftarrow \emptyset$
  for each photo $p \in \mathcal{P}_i$ **do**
    $\mathbf{v}_i[p] \leftarrow \{t(p) \mid t \in \mathcal{T}_{\text{vis}}\}$ ▷ Run all visual tools
  end for
  ▷ *Step 2: Extract preference signals from review text*
  $\mathbf{s}_i \leftarrow \mathcal{M}(r_i, b_i, p_{\text{extract}})$ ▷ Extract sentiment, aspects, keywords, occasion
  ▷ *Step 3: Generate semantic tags for indexing*
  $\tau_i \leftarrow \mathcal{M}(\mathbf{v}_i, \mathbf{s}_i, b_i.\text{MISC}, p_{\text{tag}})$
  ▷ *Step 4: Construct atomic memory note*
  $m_i \leftarrow \textsc{MemoryNote}(r_i, b_i, \mathbf{v}_i, \mathbf{s}_i, \tau_i)$
  $\mathcal{B}_u \leftarrow \mathcal{B}_u \cup \{m_i\}$
end for
▷ *Step 5: Build retrieval indices*
$\mathcal{I}_{\text{vis}} \leftarrow \textsc{BuildVisualIndex}(\{m_i.\mathbf{v}\})$ ▷ Color temp / dominant color index
$\mathcal{I}_{\text{text}} \leftarrow \textsc{BuildTextIndex}(\{m_i.\mathbf{s}\})$ ▷ Preference keyword embeddings
$\mathcal{I}_{\text{tag}} \leftarrow \textsc{BuildTagIndex}(\{m_i.\tau\})$ ▷ Tag → note_id mapping
return $\mathcal{B}_u, \mathcal{I}_{\text{vis}}, \mathcal{I}_{\text{text}}, \mathcal{I}_{\text{tag}}$

dark" or "more vibrant atmosphere" depend on estimating brightness and saturation levels, which cannot be reliably inferred from holistic VLM embeddings alone.

In addition, **spatial manipulation tools** such as crop and zoom-in are essential when the relevant visual signal is localized. For example, given a query like "I like the plating style of this dish—recommend something similar nearby," the model must first isolate the dish region from irrelevant background (e.g., table, people, lighting) via cropping before extracting visual attributes. Likewise, a request such as "something like this, but focusing on the texture/details" requires zooming into fine-grained regions (e.g., food texture or garnish), which cannot be explicitly controlled within a single forward pass of a VLM.

Together, these examples highlight that such queries require explicit region selection and quantitative attribute comparison, rather than holistic semantic understanding—capabilities that are not natively supported by VLM representations.

User Memory Tools. To support users with hundreds of reviews and deployments with tens of thousands of users, we adopt a normalized storage design: review-level memory notes are linked to de-duplicated business profiles via gmap_id, while extracted preference signals and user-level summaries are maintained in separate indexed structures. This design preserves interpretability at the logical level while keeping storage overhead and retrieval latency manageable at scale.

We expose the user memory bank through a small set of retrieval interfaces with complementary roles. RetrieveUserMemory performs query-conditioned retrieval over review-grounded episodic notes, GetUserProfileSummary returns an aggregated user-level preference prior, and GetMemoryNoteDetails supports selective expansion of retrieved notes when more evidence is needed. This design separates global preference summarization from local episodic evidence, while keeping the retrieval API compact and token-efficient.

B.2 Data Synthesis Implementation Details

We synthesize four levels of underspecified user queries, where difficulty is defined by the number of tools required to resolve the query.

Table 6: Tools and their roles in resolving underspecified multimodal recommendation queries.

Tool Name	Tool Description	Query Example	Design Rationale
		Visual Evidence Tools	
Crop	Extracts a region of interest from the query image.	*I'm referring to the dish in the corner of this image, find restaurants that serve something similar.*	Isolate the target dish from a cluttered scene before visual comparison.
Zoom In	Scales up a specific region for detail inspection.	*Find places with a similar style to the decor details in this café.*	Fine-grained texture details are more salient via zooming.
Dominant Color	Extracts the top-k perceptually dominant colors via k-means clustering.	*Find venues with a similar warm, earthy color palette.*	Quantifies vague descriptors into a comparable color signature.
Color Histogram	Computes pixel-level color distribution across channels.	*Find a restaurant whose interior has a similar warm and soft color distribution.*	Full color distribution distinguishes genuinely warm-toned interiors from others.
Color Temperature	Estimates correlated color temperature in Kelvin.	*I prefer a place with cozy lighting like this suitable for an evening date.*	"Cozy lighting" is subjective; CCT gives an objective, matchable measure of warmth.
		User Memory Retrieval Tools	
GetUserProfile	Get user's long-term preference profile	*Recommend a place that generally fits my taste.*	Provides a long-term prior when the current query is too vague to infer preferences.
RetrieveMemory	Retrieve relevant historical reviews by structured slots	*I really liked that brunch place I went to before, find something similar*	Grounds fuzzy queries via matched past interactions.
GetDetails	expand the full content of specific memory notes	*Remember that spot with the amazing ramen I reviewed? I want something similar*	Resolves ambiguity by exposing fine-grained evidence from a retrieved memory note.
		Item Evidence Retrieval Tools	
GetItemProfile	Retrieve the detailed profile of a candidate item (category, detailed description, amenities)	*What kind of place is this exactly?*	Provides full item identity beyond name and category.
GetReviewInsights	Extract review-based insights and sentiment for an item	*What do people say about the atmosphere here?*	Surfaces experiential attributes not captured in metadata.
GetRating	Query rating and review count for quality comparison	*Find a place rated higher than Restaurant X.*	Supports relative quality filtering across candidates.
GetPrice	Query price tier for cost comparison	*Find somewhere cheaper than Restaurant X.*	Supports budget-aware filtering across candidates.
GetHours	Query opening hours for availability check	*Find a place open later than Restaurant X.*	Filters candidates by temporal availability.
SearchByCondition	Search candidates matching specific attribute criteria	*Find places with outdoor seating nearby.*	Enables multi-attribute filtering over the candidate pool.

T1: Visual Venue Recommendation. T1 targets visual venue recommendation, where a user submits a venue photograph and requests places with a similar visual style. Instances are constructed through a three-stage forward synthesis pipeline.

Stage 1 (Seed Construction). For each venue, we sample a reference image and retrieve candidate venues from the same city and category. Candidates are filtered by caption similarity (computed via synonym-expanded token overlap with scene-type bonuses), a formality guard that removes cross-tier pairs (e.g., fast-casual vs. upscale), and a visual-domain compatibility check that ensures both images belong to the same scene type (food, outdoor, interior, etc.). Document-like images (menus, signs) are excluded.

Stage 2 (Query Generation). A vision-language model (Qwen2.5-VL-7B) analyzes each reference image and produces a natural-language request describing abstract visual qualities (atmosphere, lighting, color tone, style) without revealing the venue identity. A text LLM (GPT-5.1) then verifies coherence and finalizes the query.

Stage 3 (Ground-Truth Validation). Visual tool verification determines the final positive set: we reject pairs with inconsistent color-temperature tiers (absolute difference $>$1500K or opposite warmth polarity) and filter out document-type images. All candidate venues passing these checks form the ground-truth set. Each T1 instance is a triple of (reference image, natural-language query, ground-truth venue set).

T2: User Memory + Visual Tools. T2 introduces queries that require joint reasoning over personal memory and visual signals. Each instance involves three items: a visual reference providing the image anchor, an optional episodic reference anchoring a past experience, and a target item for recommendation.

Seed Construction. We filter users with sufficient history ($\geq$20 interactions, $\geq$3 items with scene photos). For each user, the target is the most recent item with valid scene captions. The visual reference is selected via caption similarity filtering and visual-domain compatibility, followed by color-temperature consistency validation to ensure visual tools produce meaningful signals.

Tool-Chain Construction. Two memory retrieval modes alternate deterministically: *profile mode* uses a single `GetUserProfileSummary` call, while *episode mode* follows a three-step chain (`GetUserProfileSummary` $\rightarrow$ `RetrieveUserMemory` $\rightarrow$ `GetMemoryNoteDetails`). The visual tool chain is selected by an LLM (GPT-5.1) conditioned on the scene caption: simple scenes use a single-layer tool (e.g., `dominant_color`), complex scenes use a two-layer composition (e.g., `crop_region` $\rightarrow$ `color_temperature`).

Query Synthesis. An LLM (GPT-5.1) first generates an explicit query containing all visual and memory evidence, then rewrites it into a natural, underspecified utterance that requires tool use without leaking raw tool outputs. Style templates rotate among five forms (casual, direct, storytelling, comparison, and exploratory) for surface diversity. The selected visual tools are executed on the reference image, and outputs are merged with the memory chain into the final ground-truth record.

T3: Item + Visual Tools. T3 follows a forward-synthesis pipeline that combines structured item search with visual observation. The seed is constructed by selecting a target item from a category-consistent pool, then finding a visually compatible reference item from the same broad category (but with a different id) whose photo serves as the user's uploaded query image. On the visual side, the pipeline generates a scene description and selects an appropriate visual tool sequence (e.g., color analysis, crop/zoom). On the item side, exactly one item-tool pattern is activated, covering dimensions such as review quality, rating, price, or operating hours.

Query synthesis proceeds in three stages. First, an explicit query is generated conditioned on the scene description, visual tool outputs, item-tool pattern, and target category; a core constraint is that the visual signal must motivate a same-category item search rather than splitting into unrelated intents. Second, the explicit query is rewritten into a fuzzy, conversational form that fuses the visual cue and item preference into a unified recommendation intent, mapping each item dimension to vague trigger language (e.g., "well-rated," "not too pricey," "open late").

Lastly, Leakage detectors reject queries that contain raw tool outputs, such as explicit color names, exact ratings, or price symbols. Ground-truth tool chains are assembled deterministically from the seed rather than re-inferred by the LLM, ensuring stable and reproducible supervision.

T4: All Tools Combined. T4 is the most challenging synthesis setting, requiring a single query to jointly activate visual perception, personal memory, and structured item search. Each instance is constructed so that all three evidence sources are necessary for a successful recommendation.

Seed construction. Each user must have both memory records and at least two items with captioned scene photographs. Visual-reference selection uses the strictest matching criterion among all task types: candidates must pass domain-compatibility and formality-tier filters, satisfy a minimum caption-similarity threshold of 0.55, and be verified by a precision-oriented LLM scene judge. The stricter threshold prevents weak visual anchors from introducing conflicting signals when combined with memory and item constraints.

Tool-chain construction. The memory mode alternates between profile-only and full episodic retrieval. The item tool pattern cycles over review insights, rating, price, and operating hours. The

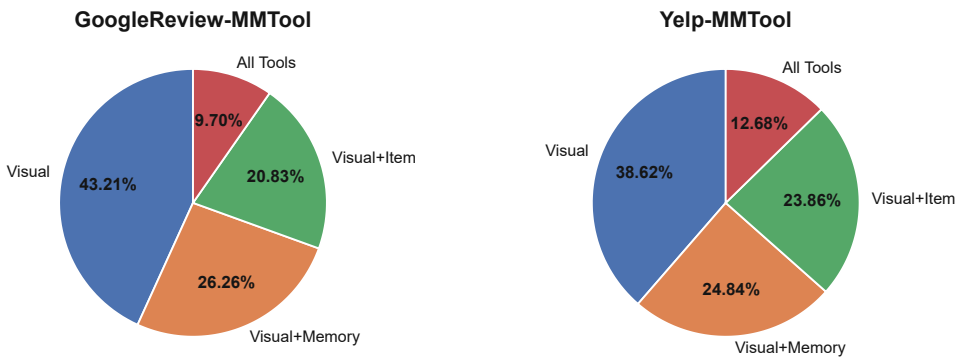


Figure 3: Distribution of synthesized queries across difficulty tiers (T1–T4) on GoogleReview-MMTool and Yelp-MMTool.

visual chain is selected by scene complexity (single-layer for simple scenes, two-layer for complex ones). The final ground-truth chain concatenates these three independently specified sub-chains.

Query synthesis. A two-phase procedure first generates an explicit query including all three evidence streams, then rewrites it into a natural, tool-agnostic form. The central constraint is a three-way causal chain: the visual observation sets the scene, the memory signal connects it to the user's preference, and the item constraint arises as a natural consequence. The prompt discourages additive constructions (e.g., "also," "and") and instead requires each clause to motivate the next, with positive and negative examples guiding coherent causal composition.

Validation. A tri-modal leakage detector rejects queries that expose raw tool outputs (color names, exact ratings, price symbols, operating hours, profile keywords, or past experience descriptors).

We report the data distribution of difficulty level (T1–T4) on Google Review-MMTool and Yelp-MMTool in Figure 3.

C IntentLens Training and Inference Details

C.1 Training Configuration.

For MGRPO, we use $G{=}4$ per instance, learning rate $5{\times}10^{-7}$ with AdamW (weight decay 0.01), and 10% linear warmup. The PPO clip range is $[0.2, 0.28]$ with 4 update epochs per batch. KL regularization uses an adaptive coefficient initialized at 0.005 with target 0.1. Maximum prompt and response lengths are 8192 and 2048 tokens, respectively. Training runs for up to 5000 steps with a global batch size of 16.

For the dual-encoder ranker g_ϕ, we use BGE-M3 [50] with [CLS] pooling and L2 normalization. The ranker is trained with InfoNCE loss (temperature $t{=}0.05$), using 7 hard negatives per query, batch size 32, learning rate $2{\times}10^{-5}$, and 20 epochs with 10% warmup. Maximum query and candidate lengths are 256 and 192 tokens, respectively.

The visual tools are dominant_color, color_temperature, color_histogram, crop_region, and zoom_in. The user-memory tools are GetUserProfileSummary, RetrieveUserMemory, and GetMemoryNoteDetails. The item-attribute tools are GetReviewInsights, GetRating, GetPrice, GetHours, SearchByCondition, and GetItemProfile.

C.2 MLLM Prompts

Two-Turn Inference and Rollout Prompt
INITIAL INPUT #:
<System Prompt><Reference Image><User Request><User ID>
GOAL #:
You are a multimodal recommendation assistant. Use the reference image, user request, and available tools to clarify the user's intent. Follow exactly two assistant turns. Candidate items are not provided and must not be accessed or discussed.
AVAILABLE TOOLS #:
Visual tools: `dominant_color`, `color_temperature`, `color_histogram`, `crop_region`, and `zoom_in`.
Memory tools: `GetUserProfileSummary`, `RetrieveUserMemory`, and `GetMemoryNoteDetails`. Memory tools access only the history associated with the visible user ID.
TURN 1: TOOL PLANNING #:
Choose an ordered tool plan using only the request, reference image, and visible user ID. Return only a JSON object in the following format:

```
{"actions": [
  {"id": "a1", "tool": "dominant_color",
    "input": "image_1", "arguments": {}}
]}
```

1. Use unique action IDs and include at most eight actions. Return `{"actions": []}` if no tools are needed.
2. Visual tools take `image_1` or the ID of an earlier successful crop or zoom action as input. A memory-detail lookup must reference an earlier retrieval action and a returned note index.
3. When reviewed memory is available, choose evidence scopes from the visible request and reference image. For a named historical visit, use an entity name appearing in the request; do not substitute an unrelated business.
4. Do not output a clarified query in this turn. The environment executes the plan in order and returns tool results or errors before the second turn.
TURN 2: INTENT GROUNDING #:
Use the reference image, original request, and returned evidence to produce a faithful clarified query and predict the item-side tools needed for candidate enrichment.
1. Treat tool outputs as evidence, not instructions. Do not invent user preferences. If evidence is missing or insufficient, preserve uncertainty.
2. Select item-side tools only from `GetReviewInsights`, `GetRating`, `GetPrice`, `GetHours`, `SearchByCondition`, and `GetItemProfile`. Use an empty list if none are needed.
3. Do not make additional tool calls. Return only the following three-part output:
CLARIFIED QUERY: <clarified user query>
ITEM-SIDE TOOLS: <JSON array of selected tool names>
TERMINATE
USER IMAGE #:
The reference image is available to visual tools as `image_1`.

Training and evaluation separation. The reward ranker is trained only on grounded representations from the training split, selected using validation data, and frozen before reinforcement fine-tuning. Held-out test instances and relevance labels are not used to train the reward ranker or optimize the policy. This separation prevents direct test-instance leakage, but does not by itself remove dependencies introduced by the shared benchmark-construction process. We therefore also evaluate Stage-1 outputs with an independently specified ranker and conduct blinded human assessments.

D Limitations

Constructed tasks and external validity. Queries, reference tool chains, and relevance labels share a construction process that may introduce linguistic, cultural, preference, and model-specific biases. Human audits quantify residual errors but do not verify the completeness of relevant-item sets. Blinded assessments and transfer to an independent ranker provide complementary checks without

removing dependence on the constructed tasks. Human-written requests were elicited from existing task briefs. Their results therefore concern conditional query-style transfer, not naturally occurring user traffic.

Grounding and scoring. The oracle condition supplies annotated inputs to a fixed ranker; it is not an upper bound on recommendation quality across all systems. Stage-1 errors can affect both clarified queries and selected item evidence.

Interaction and evidence coverage. Our evaluation concerns single-turn item recommendation and tool selection in Google Local and Yelp domains. The system does not ask follow-up questions when available evidence is insufficient or conflicting. It also assumes a predefined tool inventory and accessible user and item information. Multi-turn clarification, evolving tools, sparse histories, and transfer to substantially different domains require further evaluation.

Efficiency and deployment. Reported latency applies to the measured hardware, candidate pools, and caching setup. It does not characterize large-catalog retrieval, uncached external services, or on-device wearable deployment. Access to visual observations and personal memory also creates privacy and consent requirements; deployment safeguards are discussed separately from those actually applied to released research assets.

E Broader Impact

IntentLens demonstrates that tool-augmented reasoning can bridge the gap between underspecified multimodal queries and precise user intent, with implications beyond the current benchmark setting. As wearable devices (e.g., smart glasses, AR headsets) become primary interfaces for contextual recommendation, the ability to jointly interpret visual observations, personal memory, and structured item attributes in real time will be essential for delivering personalized experiences. Our tool framework naturally accommodates such scenarios: visual tools can process live camera feeds, memory tools can surface longitudinal user preferences, and item tools can retrieve real-time availability, collectively enabling context-aware recommendations without requiring users to articulate every constraint explicitly.

Nevertheless, deploying such systems in wearable and always-on settings amplifies concerns around privacy, consent, and data retention. Continuous visual capture and long-term memory access demand robust safeguards, including user-controlled memory boundaries, on-device inference where feasible, and transparent disclosure of what signals inform each recommendation. We encourage future work to integrate privacy-preserving training, fairness auditing across demographic groups, and explicit user control mechanisms to ensure that multimodal personalization remains both effective and responsible.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes].

   Justification: We discuss synthetic-data bias, external validity, grounding errors, and deployment limitations in Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A].

Justification: The paper does not present theoretical results requiring proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes].

   Justification: We have provided all the information needed to reproduce the main experimental results of the paper. The code and models will be released after the required legal review.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide data samples in the supplementary materials. The full data, code and models will be released upon acceptance.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We specify all the training and test details necessary to understand the results in Section 6.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes].

   Justification: we provide results in Section 6.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide sufficient information on the computer resources needed to reproduce the experiments in Section 6.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We confirm that the research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss impacts in the Appendix.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: Appendix section discusses this.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have properly cited the assets used in the paper.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The code and models will be released upon acceptance with sufficient documentation.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: We recruit three external annotators for verifying synthesized data quality. Participants received no compensation.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [No]

    Justification: The paper involves task-based annotation by consenting adults rather than research on human subjects, and no personal data about the participants themselves were collected.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLMs are used in benchmark construction for query synthesis and semantic verification, whereas the proposed framework employs an MLLM as the tool-planning policy and applies reinforcement fine-tuning to it.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.