Collab-REC: An LLM-based Agentic Framework for Balancing Recommendations in Tourism

ASHMI BANERJEE, Technical University of Munich, Munich, Germany
ADITHI SATISH, Technical University of Munich, Munich, Germany
FITRI NUR AISYAH, Technical University of Munich, Munich, Germany
WOLFGANG WÖRNDL, Technical University of Munich, Munich, Germany
YASHAR DELDJOO, Polytechnic University of Bari, Bari, Italy

We propose Collab-Rec, a multi-agent framework designed to counteract popularity bias and enhance diversity in tourism recommendations. In our setting, three LLM-based agents (*Personalization*, *Popularity*, and *Sustainability*) generate city suggestions from complementary perspectives. A non-LLM moderator then merges and refines these proposals through iterative constrained refinement, ensuring each agent's viewpoint is incorporated while penalizing spurious or repeated responses.

Extensive offline experiments on European city queries using LLMs from different sizes and model families demonstrate that Collab-Rec enhances diversity and overall relevance compared to a single-agent baseline, surfacing lesser-visited locales that are often overlooked. This balanced, context-aware approach better reflects a broader range of user and system-level considerations, highlighting the potential of multi-stakeholder collaboration in LLM-driven recommender systems.

Code, data, and other artifacts are available here: https://github.com/ashmibanerjee/collab-rec while the prompts used are included in the appendix.

CCS Concepts: • **Computing methodologies** → **Artificial intelligence**; • **Information systems** → **Information retrieval**.

Additional Key Words and Phrases: LLMs, Multi-Agent Systems, Tourism Recommender Systems, Multi-Stakeholder Fairness

1 Introduction

Tourism recommender systems (RSs) that suggest travel destinations are increasingly expected to serve *multiple stakeholders* simultaneously. Beyond tailoring suggestions to a traveler's constraints and interests, platforms often optimize for engagement and business objectives (frequently correlated with destination popularity), while destinations and policymakers increasingly require *sustainability-aware demand shaping* to mitigate overtourism, seasonal concentration, and spatial congestion. These requirements naturally induce *competing objectives*: recommendations that are highly personalized can still over-concentrate demand on a small set of iconic hubs; conversely, aggressively pushing long-tail destinations can degrade relevance when user constraints are not respected [2, 9].

Unlike many retail settings, where recommending a popular product mostly affects conversion, tourism recommendations can shape physical flows of visitors across space and time. Repeatedly steering demand toward a few "must-see" cities can exacerbate congestion externalities and reduce the quality of experience for both visitors and residents [21]. At the same time, tourism RSs remain accountable to the individual traveler: a

Authors' Contact Information: Ashmi Banerjee, Technical University of Munich, Munich, Germany; e-mail: ashmi.banerjee@tum.de; Adithi Satish, Technical University of Munich, Munich, Germany; e-mail: adithi.satish@tum.de; Fitri Nur Aisyah, Technical University of Munich, Munich, Germany; e-mail: fitri.aisyah@tum.de; Wolfgang Wörndl, Technical University of Munich, Munich, Germany; e-mail: woerndl@in.tum.de; Yashar Deldjoo, Polytechnic University of Bari, Bari, Italy; e-mail: deldjooy@acm.org.

ACM 2770-6699/2026/8-ART
https://doi.org/10.1145/3837862

recommendation list is only useful if it satisfies hard constraints (e.g., travel dates, budget, seasonal preferences) and aligns with stated interests (e.g., museums, nature, nightlife). This combination makes tourism a prototypical *multi-stakeholder* recommendation setting, where user utility, platform utility, and destination-level sustainability objectives must be balanced rather than optimized in isolation [1, 31].

1.1 LLMs as Travel Recommenders: Capabilities and Shortcomings

Large language models (LLMs) enable conversational travel recommendation, allowing users to express complex, multi-intent requirements in natural language. For example: *"walkable European cities in September, mid-budget, with museums and cultural events, but not overcrowded"*. Recent generative recommenders demonstrate that LLMs can improve interaction quality through dialogue, explanations, and nuanced preference elicitation [26, 37, 39, 57]. However, monolithic LLM recommenders remain brittle when asked to satisfy multiple simultaneous constraints and to balance stakeholder objectives. Two failure modes are especially problematic in tourism: (*i*) **popularity dominance**, where the model repeatedly returns canonical tourist hubs even when users ask for "hidden gems" or when sustainability goals discourage concentration; and (*ii*) **hallucinations**, where the model fabricates destinations or attributes (e.g., incorrect sustainability claims), or returns out-of-catalog entities that cannot be validated in a deployment setting [20, 32, 47, 50]. Since these decisions are produced end-to-end inside the model, it is often unclear *why* a particular trade-off was made, and how the output would change under different stakeholder priorities [35].

1.2 Limitations of Prior LLM-Based Approaches: Why Agentic Design?

A natural baseline for LLM recommenders is single-shot prompt engineering: ask one model to satisfy all constraints and "balance" objectives. In our setting, this approach has three practical limitations. First, mixing multiple objectives in a single prompt often produces unstable behavior: the model may implicitly prioritize popular options or ignore sustainability-oriented constraints. Second, prompt-only specialization is sensitive to surface form and decoding choices; small paraphrases can change which objective dominates. Third, single-shot pipelines offer limited mechanisms for *auditing* and *controlling* how constraints are enforced, especially when recommendations must be grounded in a fixed catalog. Recent work on agentic and multi-agent LLM systems — including dedicated surveys of agentic recommender systems — synthesizes architectures and open challenges, highlighting controllability, trustworthiness, and efficiency as central barriers to deployment [40].

Distributing objectives across specialist agents allows each agent to focus on a single stakeholder dimension, which can reduce the "objective collapse" often observed in monolithic LLM outputs and can surface a broader candidate set before aggregation. However, specialization alone is insufficient: each agent can still overfit to its own biases, and hallucinations remain possible.

We therefore combine role specialization with a *deterministic, non-LLM moderator* that implements two practical desiderata for deployment: (*i*) **catalog grounding** (ensuring outputs map to a known inventory), and (*ii*) **transparent aggregation** (making the trade-off policy explicit and reproducible). The moderator does not reason about proposals; instead, it applies explicit constraint checks and feedback signals that determine whether agents regenerate lists in subsequent rounds. While an LLM could in principle act as a flexible moderator, doing so would introduce additional cost and prompt sensitivity, and would make it harder to disentangle improvements due to role specialization versus improvements due to an additional model. A deterministic moderator provides a strong, auditable control point that supports ablation analysis.

1.3 Our Approach

We propose Collab-Rec, a moderator-mediated, multi-round refinement framework for *grounded* multi-stakeholder tourism recommendation (Figure 1). Given a user query Q, three specialist LLM agents generate candidate destination sets from complementary perspectives: a *Personalization* agent that emphasizes user constraints and interests, a *Popularity* agent that reasons about mainstream appeal and the popularity preference expressed in the query, and a *Sustainability* agent that promotes alternatives that mitigate concentration and seasonality effects. The moderator then (i) grounds candidates to a structured city catalog, (ii) scores candidates using transparent per-objective diagnostics (success, reliability, hallucination penalties), and (iii) computes rejection decisions and broadcasts diagnostic feedback that explicitly constrain agents' subsequent proposals. Agents do not directly reason about each other's outputs; instead, they regenerate lists in response to deterministic moderator signals. This design makes the system modular: additional stakeholder agents (e.g., safety, accessibility) can be integrated without retraining.

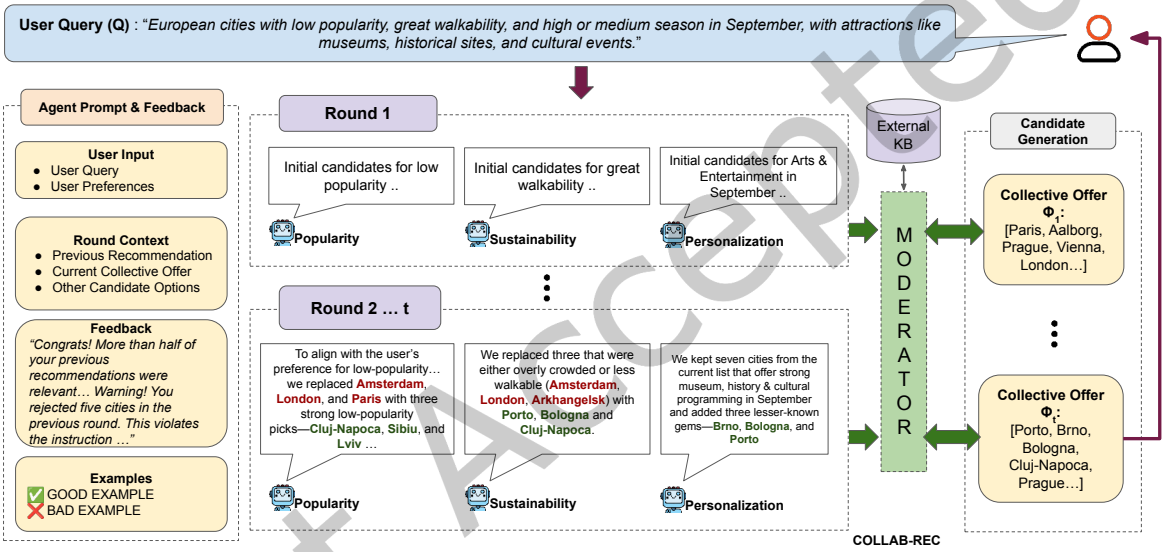


Fig. 1. Overview of the Collab-Rec workflow to generate city trip recommendations using multiple LLM agents. The non-LLM **Moderator** evaluates and combines the agent proposals, iteratively refining the final recommendation set, which is then communicated to the user.

Efficiency-aware multi-round refinement. Multi-round coordination increases latency and cost. To make the framework more practical, we introduce a patience-based early stopping protocol that dynamically terminates coordination when improvements stagnate. Empirically, across models, recommendation quality plateaus by ∼4–5 rounds (Section 5.1); early stopping captures most gains while reducing inference time substantially for API-served models.

Empirical study. We conduct a large-scale offline evaluation on 900 stratified tourism queries derived from SynthTRIPs [11] and a grounded catalog of 200 European cities. We benchmark six LLM backbones spanning proprietary and open-source families (`claude-sonnet-4-5`, `gemini-2.5-flash`, `gpt-oss-20b`, `gemma-3-12b`, `olmo3-7b-instruct`, `gemma-3-4b`), and compare against (i) non-LLM baselines (RandRec, TopPop, and a structured MILP optimizer), (ii) a single-agent baseline (SASI), and (iii) a single-round multi-agent baseline (MASI). We

evaluate *grounded recommendation quality* primarily using moderator success (constraint satisfaction under catalog validation), complemented by diversity and concentration metrics (Gini, entropy, and catalog coverage) and agent behavior metrics (reliability and hallucination tendency). Across model families, Collab-Rec consistently improves grounded success over SASI and MASI, while also reducing recommendation concentration relative to them, leading to more diverse recommendations (Section 5). Although the MILP reference achieves the highest moderator success score, it operates on fully structured catalog inputs with a hand-specified utility function. We therefore treat MILP as a structured-input reference rather than evidence that Collab-Rec outperforms all non-LLM recommenders.

Contributions. Our main contributions are as follows:

- **Problem formulation.** We formalize multi-stakeholder, multi-constraint tourism recommendation as a *grounded* multi-objective ranking problem, where feasibility is defined with respect to an explicit destination catalog and constraint satisfaction (Section 3.1).
- **Framework design.** We introduce Collab-Rec, a modular multi-agent architecture with a transparent, deterministic moderator that scores candidates under multiple objectives and iteratively conditions agent outputs via structured feedback (Section 3.2–3.5).
- **Efficiency-aware moderation.** We propose and empirically validate a patience-based early stopping protocol and analyze the effect of two rejection strategies (*Aggressive* vs. *Majority*) on convergence, stability, and compute cost (Section 3.6).
- **Large-scale evaluation and analysis.** We evaluate 900 queries across six LLM families, report statistical testing and convergence behavior, and analyze the relevance–diversity–cost trade-off induced by multi-round refinement (Section 4–5).
- **Reproducibility.** We release code, prompts, and evaluation artifacts to enable reproduction and extension.

Scope and limitations. Our goal is to provide a reproducible, controllable blueprint for balanced tourism recommendations grounded in catalogs, not a production-ready system. We do not claim to have developed a new foundational multi-agent learning algorithm. Rather, we study how role specialization, multi-round refinement, and grounded moderation affect relevance, diversity, hallucination behavior, and efficiency in a controlled research setting.

Paper organization. Section 2 reviews related work on LLM-based agents, multi-agent recommender systems, and multi-objective recommendation. Section 3 presents the Collab-Rec framework and moderator design. Section 4 describes the experimental setup and evaluation protocol. Section 5 reports results and discussion organized around our research questions, including efficiency and robustness analysis. Finally, Section 6 concludes with limitations and future directions.

2 Related Work

This section reviews (i) LLM-based agents and multi-agent interaction protocols (Section 2.1), (ii) multi-agent recommender systems (Section 2.2), (iii) classical multi-objective and multi-stakeholder recommendation (Section 2.3), and (iv) grounding and hallucination control in LLM recommenders (Section 2.4). We then position Collab-Rec within this landscape and clarify how it complements (rather than replaces) established optimization and reranking approaches in Section 2.5.

2.1 LLM-based Agents and Multi-Agent Interaction

LLM-based agents — LLMs augmented with role instructions, memory, and (optionally) tools—have been studied as a mechanism to decompose complex tasks into interacting components. Recent surveys summarize common agentic architectures and workflows, emphasizing communication protocols, evaluation, and application domains

such as web search and scientific question answering [28, 46, 55, 58, 61]. Empirical studies also highlight limitations: agent success can be brittle, and scaling the number of agents does not automatically improve outcomes unless the interaction protocol is carefully designed [32]. From a recommender-systems perspective, recent work synthesizes definitions and open challenges for agentic RSs and multi-agent RSs, highlighting controllability, trustworthiness, and efficiency as central barriers to deployment [40].

A prominent family of interaction protocols is *multi-agent debate* and round-table consensus, where agents critique each other's responses over multiple rounds to reach an agreement [22, 52]. Such debate-feedback mechanisms can improve performance on reasoning and summarization tasks without additional training data [14–16, 59]. However, forcing explicit consensus can reduce diversity; role differentiation and implicit agreement mechanisms have been proposed as a way to preserve diversity while maintaining broad consistency [56]. These observations are directly relevant to tourism recommendation, where a system must often balance competing objectives rather than optimize a single notion of "correctness."

2.2 Multi-Agent Recommender Systems

Multi-Agent Recommender Systems (MARS) leverage agent specialization to support recommendation-related subtasks (e.g., user preference interpretation, item understanding, search, explanation). Recent LLM-based conversational recommendation frameworks employ agents with memory and external tools to elicit preferences and generate recommendations without large-scale training [53]. Architectures such as MACRS [25] and MACRec [54] adopt a centralized manager coordinating specialized sub-agents, often supported by a search agent querying external sources [43]. MATCHA extends this pattern with safeguard and explanation agents for video game recommendation [30]. More broadly, surveys examine the bidirectional relationship between LLM agents and recommender systems, documenting how agents can enhance recommendation pipelines and how recommendation concepts (e.g., user modeling, feedback loops) can inform agent design [63].

Most existing MARS work, however, focuses on improving personalization or interaction quality in a single objective setting. Tourism-specific multi-agent designs that explicitly balance user preferences, popularity-driven platform incentives, and sustainability constraints remain underexplored. Moreover, many frameworks rely on LLM-based management or open-world retrieval; while powerful, this can make factual grounding and reproducibility harder to guarantee when the system must adhere to a fixed catalog.

2.3 Multi-Objective and Multi-Stakeholder Recommendations

Balancing competing objectives has long been studied in RSs. Multi-objective RSs optimize multiple criteria (e.g., accuracy, diversity, novelty, fairness) via scalarization (weighted sums), constrained optimization, or Pareto-based approaches [19, 62]. Multi-stakeholder recommendation generalizes this view by explicitly modeling utilities for multiple parties (e.g., users, providers, platforms) and analyzing trade-offs among them [4].

A widely adopted practical strategy is *post-processing reranking*: given a candidate set, rerank items to improve diversity or reduce concentration using rank-discounted trade-offs (e.g., MMR [13], xQuAD [48]) or topic-based diversification in recommendation lists [64]. These approaches are highly effective when item features and objective scores are readily available.

In open-ended tourism queries, however, the first challenge is to produce a feasible candidate set that satisfies natural-language constraints and is grounded in an explicit inventory. Our work is therefore complementary: Collab-Rec uses specialist generation plus catalog validation to create feasible candidates, and then applies a transparent scalarization in the moderator to aggregate objectives (Section 3). The resulting pipeline can be seen as an *agentic front-end* that enables multi-objective list construction in a setting where constraint satisfaction and grounding are first-order requirements.

2.4 Grounding and Hallucination Control in LLM Recommenders

Hallucination and factuality concerns have motivated substantial work on grounding LLM outputs, including retrieval-augmented generation, structured output constraints, and post-hoc verification [5, 7, 23, 34, 41]. In recommendation settings, recent research also argues that evaluation should go beyond utility and incorporate aspects such as factual validity, reasoning quality, and robustness, particularly when LLMs are used to generate items or item attributes [20, 32]. Tourism recommendation amplifies these concerns because out-of-catalog or incorrectly attributed destinations can lead to poor user experience and safety risks.

2.5 Positioning Collab-Rec

Collab-Rec is positioned at the intersection of agentic RSs and multi-objective recommendation: it targets open-ended, multi-constraint travel queries; decomposes stakeholder objectives into role-specific generation agents; and enforces catalog grounding and transparent aggregation via a deterministic moderator. Compared to manager-centric multi-agent recommenders (e.g., MACRec [54], MATCHA [30]), Collab-Rec emphasizes *multi-stakeholder balancing* and provides per-objective diagnostics (success, reliability, hallucination penalties) that are directly measurable and ablatable. Compared to classical reranking and multi-objective optimization methods, Collab-Rec addresses the upstream challenge of generating and repairing feasible candidates under natural-language constraints, while still retaining a transparent scalarization layer that can incorporate established multi-objective principles.

Comparison summary. In short, Collab-Rec differs from the closest lines of work along four axes:

- **Problem focus:** multi-stakeholder *tourism* recommendation with explicit attention to popularity concentration and sustainability-oriented balancing;
- **Mechanism:** moderator-mediated *multi-round refinement* with structured feedback, rather than single-shot prompting or a manager-only architecture;
- **Grounding:** explicit catalog validation and hallucination-aware rejection policies to enforce inventory compliance; and
- **Evaluation:** large-scale analysis across six LLM families with statistical testing, convergence/early-stopping behavior, and explicit relevance–diversity–cost trade-offs (Section 4.1–5).

3 Agentic Recommendation Framework

This section formalizes Collab-Rec, a multi-agent recommendation framework for city-trip recommendations that explicitly balances multiple stakeholder objectives. Given a natural-language user query, three specialized LLM agents propose ranked candidate cities from a fixed catalog. A deterministic moderator then grounds, evaluates, and aggregates the proposals through a repeated coordination loop. The loop continues until an online termination criterion indicates convergence or stagnation, after which the final ranked recommendation list is returned.

3.1 Preliminaries and System Goal

Problem setup. We consider a city-trip recommendation task in which the input is a natural-language user query $\mathbb{Q}$. The query encodes (i) textual preferences (for example, "quiet coastal destinations"), and (ii) an explicit set of structured filters $\mathcal{F} = \{f_1, f_2, \ldots, f_m\}$ (for example, budget range, travel month, activity categories, or sustainability preferences). Recommendations are drawn from a closed catalog of candidate cities C, where each city $c \in C$ is associated with structured attributes stored in an external knowledge base (Section 4.1.2).

In our experiments, the natural-language query is accompanied by benchmark-provided structured filters used for validation and scoring. Thus, Collab-Rec should not be interpreted as a fully structure-free raw-text

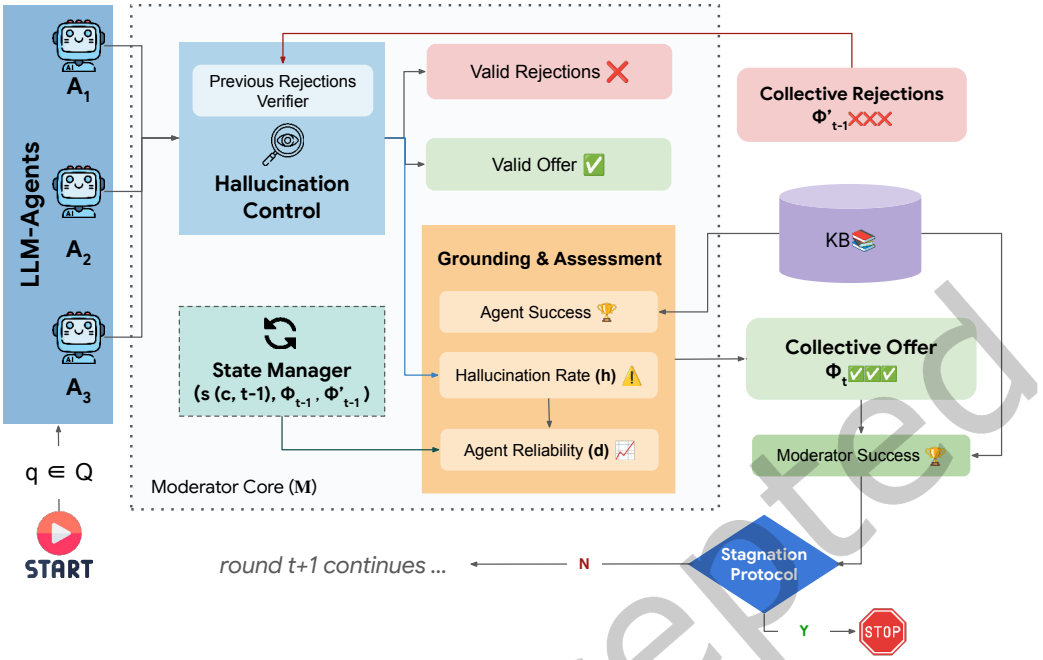


Fig. 2. Overview of Collab-Rec. Three specialist LLM agents independently propose ranked city candidates. A deterministic moderator validates proposals against a knowledge base, computes grounded diagnostics (constraint satisfaction, stability across rounds, and hallucination/invalid-output rate), aggregates candidates into a collective offer, and broadcasts structured feedback and a rejection set. Iteration continues until an online termination protocol is satisfied.

recommender. Its language-model component is responsible for candidate generation and revision conditioned on the natural-language request, while the moderator uses the structured filters and closed catalog for deterministic grounding and constraint checks. The goal is to output a ranked list Φ_t of exactly k cities that balances multiple stakeholder objectives under the constraints expressed by $\mathbb{Q}$ and $\mathcal{F}$. In Collab-Rec, $k = 10$ in all experiments.

Notation. We denote the set of agents by $\mathcal{A} = \{a_1, a_2, a_3\}$, and the round index by $t \in \{1, \ldots, T\}$, where each round corresponds to a step in an iterative constrained refinement process. At round t, agent a_i produces a ranked list of k candidate cities, denoted $L_{a_i,t}$. The moderator maintains two shared state variables: (i) the *collective offer* Φ_t, a ranked list of k cities representing the current best aggregated proposal; and (ii) the *collective rejection set* Φ'_t, containing cities that are disallowed in subsequent rounds under the active rejection strategy (Section 3.3.4).

System goal. Let $s(c, t)$ denote the cumulative moderator score assigned to city c after round t. At termination round T, Collab-Rec returns the size-k ranked list that maximizes the cumulative score:

$$\Phi_T = \arg \max_{L \subseteq C,\, |L|=k} \sum_{c \in L} s(c, T). \tag{1}$$

In practice, Φ_t is constructed online via repeated aggregation and scoring (Section 3.3–3.4) rather than by exhaustive search over $\binom{|C|}{k}$ combinations.

3.2 Architecture and Moderator-Mediated Multi-Round Coordination

3.2.1 System components. The system consists of three specialist agents and a deterministic moderator, as illustrated in Figure 2.

Specialist Agents. Collab-Rec instantiates three specialist agents, each representing a distinct stakeholder objective commonly studied in tourism recommender systems:

- **Personalization Agent** a_1: focuses on the user-centric perspective, prioritizing explicit filters and query-specific preferences such as budget, travel month, and interests.
- **Popularity Agent** a_2: emphasizes the popularity dimension and is configured to mitigate short-head concentration by proposing less popular cities when the query suggests a preference for less crowded destinations.
- **Sustainability Agent** a_3: prioritizes sustainability-related attributes such as walkability, seasonality, and air-quality indicators, and defaults to environmentally preferable cities when the query does not specify sustainability constraints, consistent with multi-stakeholder tourism perspectives [10].

Each agent is powered by an LLM backbone and is constrained to output exactly k catalog city names in a structured schema (Section 3.3.2). To ensure fair comparisons across models, the three agents share the same LLM backbone within each experimental run (Section 4.2).

Moderator. The moderator M is a deterministic (non-LLM) controller (Figure 2) that (i) validates proposals against the catalog and query constraints, (ii) computes grounded diagnostic scores, (iii) aggregates proposals into an updated collective offer Φ_t, and (iv) broadcasts structured feedback and the updated rejection set Φ'_t. The moderator has access to a structured knowledge base that provides attributes for every $c \in C$, enabling catalog grounding, constraint checks, and objective measurements.

3.2.2 What "iterative constrained refinement" means in Collab-Rec. We use *iterative constrained refinement* to denote a moderator-mediated coordination loop, not a debate protocol in which agents directly message, strategize, or bargain. Agents do not exchange messages or coordinate with each other; instead, each agent iteratively refines its own proposals in response to the moderator's structured feedback and the shared evolving state. We use the terms "rounds" and "iterations" interchangeably to refer to steps in the refinement loop.

Each round consists of four steps:

(1) **Proposal:** each agent a_i proposes a ranked list $L_{a_i,t}$ of k destinations aligned with its role.
(2) **Grounding and assessment:** the moderator validates items, computes grounded diagnostics (constraint satisfaction, stability, and invalid-output rate), and updates city scores using a transparent multi-objective policy (Section 3.4–3.5).
(3) **Feedback broadcast:** the moderator publishes the collective offer Φ_t and the collective rejection set Φ'_t, and generates structured feedback that summarizes each agent's behavior (for example, invalid items, excessive churn, or insufficient alignment with role-specific filters).
(4) **Revision:** agents condition on Φ_t and Φ'_t and revise their proposals in the next round by repairing invalid suggestions, exploring alternatives, and adapting to moderator feedback.

Multi-round execution can improve outcomes through three empirically testable mechanisms: (i) *repair* (iterative removal and replacement of invalid or constraint-violating candidates), (ii) *feedback-driven exploration* (moving away from over-recommended short-head destinations when they conflict with the objectives), and (iii) *stabilization* (convergence of the collective offer as marginal improvements diminish), which motivates early stopping (Section 3.6).

3.3 Interaction Protocol and Operational Design

3.3.1 Agent prompting and controlled revision. At $t = 1$, each agent receives the user query $\mathbb{Q}$ and the relevant filters for its role (Section 3.2.1), and returns a ranked list of k catalog cities. For rounds $t > 1$, each agent additionally receives: (i) the current collective offer Φ_{t-1}, (ii) a role-specific feedback message generated by the moderator, and (iii) a set of disallowed cities Φ'_{t-1}. Agents are instructed to produce a *new* ranked list of exactly k cities at every round and to retain at least $k - 3$ cities from the current collective offer Φ_{t-1}, modifying at most three relative to it, which operationalizes limited "offer revision" inspired by iterative offer-and-veto style protocols [24]. The limited replacement budget encourages continuity and makes reliability measurable (Section 5.3).

3.3.2 Hallucination control via structured output constraints. LLM recommenders may produce non-grounded items, including destinations outside the catalog or contradicting explicit constraints [32]. To mitigate this risk, Collab-Rec enforces a structured output schema that requires: (i) exactly k ranked city names, (ii) JSON-serializable fields, and (iii) fixed field names and types for downstream validation. While structured output constraints reduce hallucinations by enforcing syntactic validity, they do not fully guarantee catalog membership. Final grounding is enforced by the moderator through post-generation validation.

Within Collab-Rec, an output is treated as *invalid* if it recommends a city that is either (i) not in the catalog C, or (ii) present in the collective rejection set Φ'_t under the active rejection strategy. We quantify invalidity via the hallucination metric defined in Section 3.5.3.

3.3.3 Constructing the collective offer. After scoring candidate cities at round t, the moderator produces the collective offer Φ_t by selecting the top-k cities under min–max normalized scores [45].

Let

$$\text{norm}(s(c,t)) = \frac{s(c,t) - \min_{c' \in C} s(c',t)}{\max_{c' \in C} s(c',t) - \min_{c' \in C} s(c',t)}.$$

Then the collective offer is defined as:

$$\Phi_t = \arg\text{top}_k \left[\text{norm}(s(c,t))\right], \quad c \in C. \tag{2}$$

where $\arg\text{top}_k$ returns the k cities with the largest normalized scores, ordered from highest to lowest.

3.3.4 Aggregating rejections. The collective rejection set Φ'_t removes cities that are deemed unacceptable under a voting policy based on agent omissions. Let $\mathbb{I}[\cdot]$ be the indicator function, and define the number of "omit votes" for city c at round t as

$$v_t(c) \;=\; \sum_{a_i \in \mathcal{A}} \mathbb{I}\big[\, c \notin L_{a_i,t} \big], \qquad \forall\, c \in \Phi_{t-1}.$$

We consider two rejection strategies:

- ***Aggressive* rejection:** reject c if *any* agent omits it, that is, $v_t(c) \geq 1$.
- ***Majority* rejection:** reject c if *at least two* agents omit it, that is, $v_t(c) \geq 2$.

The collective rejection set is updated as:

$$\Phi'_t \;=\; \Phi'_{t-1} \;\cup\; \{\, c \in \Phi_{t-1} \;:\; v_t(c) \geq \tau \,\}, \tag{3}$$

where $\tau = 1$ for *Aggressive* rejection, and $\tau = 2$ for *Majority* rejection. Cities in Φ'_t are disallowed in subsequent rounds for both the agents and the moderator.

3.4 Scoring and Decision Policy

The moderator's role is to combine individual agent diagnostics into a collective score that reflects the system's multi-objective optimization goals. This aggregation is a canonical example of multi-criteria decision-making [8] and social choice theory [49], where diverse agent evaluations must be fused into a single collective decision. For each agent a_i and round t, the moderator aggregates three grounded diagnostics: (i) agent success (relevance) $r_{a_i,t}$, measuring constraint alignment, (ii) agent reliability $d_{a_i,t}$, capturing temporal stability across rounds, and (iii) hallucination rate $h_{a_i,t}$, indicating the fraction of invalid outputs. These diagnostics are combined via a transparent linear scalarization with inverse-rank discounting, providing both interpretability and the ability to perform ablation analyses (Section 5.5).

Rank-discounted scalarization. For a candidate city c proposed by agent a_i at round t, let $\text{rank}_{a_i,t}(c) \in \{1, \ldots, k\}$ denote its rank position in $L_{a_i,t}$. We define the incremental contribution of agent a_i to city c at round t as:

$$\Delta s_{a_i}(c,t) \;=\; \frac{1}{\text{rank}_{a_i,t}(c)} \Big(\lambda_r \, r_{a_i,t} + \lambda_d \, d_{a_i,t} - \lambda_h \, h_{a_i,t} \Big), \tag{4}$$

where $\lambda_r \geq 0$, $\lambda_d \geq 0$, and $\lambda_h \geq 0$ are weighting coefficients.

Scope of the diagnostic terms. The terms $r_{a_i,t}$, $d_{a_i,t}$, and $h_{a_i,t}$ are agent or list-level diagnostics rather than item-specific utility estimates. They measure, respectively, how well an agent's complete list aligns with its assigned filters, how stable that list is across rounds, and how often the agent produces invalid outputs. Consequently, every city proposed by the same agent in the same round receives the same diagnostic multiplier, modulated only by rank. Item-level differentiation enters through catalog feasibility, the reciprocal-rank discount, repeated endorsement by multiple agents and rounds, and rejection constraints. This design is intentionally transparent and auditable, but it can still assign a diagnostic boost to a weak item appearing in an otherwise strong list. We therefore interpret the score as a heuristic coordination signal rather than an item-level utility model.

The term $1/\text{rank}_{a_i,t}(c)$ is a reciprocal-rank positional weight. It is related to positional aggregation in spirit but is not a standard linear Borda count, which would assign scores such as $k - \text{rank}$ for a list of length k. Reciprocal-rank discounting attenuates lower-ranked items more strongly, thereby concentrating influence on top-ranked proposals. We adopt this parameter-free discount for interpretability; alternative positional functions, including linear Borda-style or logarithmic DCG-style discounts, remain natural variants for future work.

Cumulative scoring across rounds. Scores accumulate across rounds, reflecting repeated endorsement by agents and persistence under moderator grounding:

$$s(c,t) \;=\; s(c,t-1) + \sum_{a_i \in \mathcal{A}} \mathbb{I}[c \in L_{a_i,t}] \; \Delta s_{a_i}(c,t), \tag{5}$$

with $s(c,0) = 0$ for all $c \in C$. Unless stated otherwise, we use $\lambda_r = \lambda_d = \lambda_h = 1$ to preserve interpretability and to avoid tuning on the evaluation set. Ablation analyses over scoring *components* (i.e., removing r, d, or h terms entirely) are reported in Section 5.5.

Positioning of the moderator score. Equation 4 is a transparent heuristic scalarization of grounded diagnostics, not a learned, calibrated, or theoretically optimal objective. We fix $\lambda_r = \lambda_d = \lambda_h = 1$ *a priori* to preserve interpretability and to avoid post-hoc tuning on the evaluation set. The ablations in Section 5.5 remove scoring components under these fixed weights; they should not be interpreted as a sensitivity analysis over the λ parameters.

3.5 Grounding and Assessment

In each round, the moderator computes grounded diagnostics for each agent based on the knowledge base attributes and the catalog constraints.

3.5.1 Agent success. Agent success measures how well an agent aligns its recommendations with the filters assigned to its role. Let $f_{a_i} \subseteq \mathcal{F}$ denote the subset of query filters assigned to agent a_i. Let $\mathcal{M}(c)$ denote the set of filters satisfied by city c under the knowledge base metadata (for example, a city satisfies a budget filter if its cost attribute lies in the requested range). Then the agent's success is:

$$r_{a_i,t} = \frac{1}{|L_{a_i,t}|} \sum_{c \in L_{a_i,t}} \frac{|\mathcal{M}(c) \cap f_{a_i}|}{|f_{a_i}|}, \tag{6}$$

where $r_{a_i,t} \in [0, 1]$.

We distinguish two success metrics: *agent success* (Equation 6) measures constraint satisfaction for an agent's proposed list against its assigned filters f_{a_i}, while *moderator success* (Equation 11) measures average filter satisfaction of the collective offer Φ_t against the full filter set $\mathcal{F}$. We use the terms "success" and "relevance" interchangeably when referring to the agent-level metric.

3.5.2 Agent reliability. Agent reliability quantifies how stable an agent's ranked list is across consecutive rounds. Let $A = L_{a_i,t-1}$ and $B = L_{a_i,t}$. We define a rank-deviation operator:

$$\Delta(A, B) = \sum_{x \in A \cap B} \left|\text{rank}_A(x) - \text{rank}_B(x)\right| + |A \setminus B| \cdot \mu_1 + |B \setminus A| \cdot \mu_2, \tag{7}$$

where μ_1 penalizes dropped candidates and μ_2 penalizes newly introduced candidates.

We set both penalties to the same scalar:

$$\mu_1 = \mu_2 = |L_{a_i,t}|, \tag{8}$$

so that $|L_{a_i,t-1}| \cdot (\mu_1 + \mu_2)$ is a conservative upper bound on $\Delta(A, B)$, guaranteeing $d_{a_i,t} \in [0, 1]$.

Reliability is then:

$$d_{a_i,t} = \max\left(0,\ 1 - \frac{\Delta(L_{a_i,t-1}, L_{a_i,t})}{|L_{a_i,t-1}| \cdot (\mu_1 + \mu_2)}\right), \tag{9}$$

with $d_{a_i,t} \in [0, 1]$.

3.5.3 Hallucination rate. The hallucination rate measures the fraction of invalid recommendations in $L_{a_i,t}$, where "invalid" means either out-of-catalog or rejected by the moderator. Let the currently feasible set be $C \setminus \Phi'_t$. Then:

$$h_{a_i,t} = \frac{1}{k} \sum_{j=1}^{k} \mathbb{I}\left[L_{a_i,t}[j] \notin (C \setminus \Phi'_t)\right], \tag{10}$$

where $L_{a_i,t}[j]$ denotes the city at rank position j in $L_{a_i,t}$. By construction, $h_{a_i,t} \in [0, 1]$, and it is subtracted in the score update through Equation 4.

3.6 Termination Criteria and Complexity

A fixed-round budget can be wasteful when improvements plateau early, but greedy stopping can terminate prematurely when scores fluctuate across rounds. Collab-Rec therefore uses an online termination protocol based on the moderator's success score for the collective offer.

Moderator success for termination. Let $\mathcal{F}$ be the full set of query filters. We define moderator success as the average fraction of satisfied filters over the current collective offer:

$$S(\Phi_t) = \frac{1}{|\Phi_t|} \sum_{c \in \Phi_t} \frac{|\mathcal{M}(c) \cap \mathcal{F}|}{|\mathcal{F}|}, \tag{11}$$

so that $S(\Phi_t) = 1$ indicates that all recommended cities satisfy all query constraints under knowledge-base grounding.

Online termination protocol. The process terminates if either of the following holds:

(1) **Ideal convergence:** if $S(\Phi_t) = 1$, the process stops immediately at round t.
(2) **Patience-based stagnation:** after a minimum exploration phase of $T_{\min}$ rounds, the process stops if improvements are below a threshold ϵ over a sliding patience window of length p:

$$t \geq T_{\min} \ \wedge \ \left(\max_{i \in \{0,\ldots,p\}} S(\Phi_{t-i}) - S(\Phi_{t-p}) < \epsilon \right). \tag{12}$$

Complexity considerations. Each round requires $|\mathcal{A}|$ LLM generations and one deterministic moderator pass over at most $|\mathcal{A}| \cdot k$ proposed items. The dominant cost is LLM inference. In our implementation, agents are executed in parallel, so the wall-clock time per round is close to the slowest agent call plus moderator overhead. The early stopping criterion reduces the expected number of rounds, and Section 5.4 quantifies the resulting latency and token savings.

4 Experiments

This section describes the experimental design used to evaluate Collab-Rec. We detail the dataset and knowledge base, implementation and orchestration settings, baselines, model backbones, evaluation metrics, and the research questions guiding the analysis.

4.1 Setup

4.1.1 Dataset. We evaluate on a stratified sample of 900 queries from SynthTRIPs [11], a knowledge-grounded benchmark for personalized tourism recommendation. The benchmark contains over 4,000 synthetic, natural-language queries designed to reflect diverse user travel intents and sustainability preferences.

To obtain balanced coverage while keeping computational cost tractable, we stratify by two axes provided by the benchmark: (i) *popularity preference level* (low, medium, high) and (ii) *query complexity tier* (medium, hard, sustainable). We uniformly sample 100 queries from each of the $3 \times 3 = 9$ strata, yielding 900 total queries. The resulting queries range from broad requests (for example, planning a short, budget-friendly trip to a less crowded coastal destination) to more constrained prompts that include multiple filters (for example, budget constraints, month constraints, and explicit sustainability requirements). Notably, SynthTRIPs queries stem from LLMs themselves, but we exclude those generated by *Gemini* (a model family included among our evaluated backbones) to reduce the risk of model-specific stylistic advantages. Instead, we rely solely on queries generated using *llama-3.2-90B*, thereby ensuring a clearer separation between query generation and model evaluation.

4.1.2 External knowledge base. Grounding and validation are performed using the SynthTRIPs knowledge base [11], which contains a closed catalog of 200 European cities. Each city is annotated with attributes relevant to the objectives of this paper, including indicators for popularity, budget, seasonality, and sustainability. The moderator uses this knowledge base to: (i) validate that recommended items belong to the catalog, (ii) check satisfaction of structured filters, and (iii) compute the grounded diagnostics used in scoring (Section 3.5).

4.2 Experimental Settings

4.2.1 Implementation details.

Primary configuration. Our main evaluation focuses on the multi-agent, multi-round configuration of Collab-Rec, referred to as the *Multi-Agent Multi-Iteration* (MAMI) configuration. For every query, the system instantiates

the three specialist agents (*Personalization*, *Popularity*, *Sustainability*) and runs the moderator-mediated coordination loop described in Section 3.3. Each agent produces a ranked list of length $k = 10$ at each round.

Decoding parameters and structured output. For all models that expose sampling controls, we set temperature to 0.5 and top-p to 0.95. These parameters are chosen to allow controlled exploration while maintaining stability across rounds. All agents are constrained to produce structured outputs, and the moderator validates outputs against the catalog and the rejection set (Section 3.3.2).

Round budget and termination. The system is allowed up to $T_{\max} = 10$ rounds, but it typically stops earlier via the online termination protocol in Section 3.6. In our experiments, we use: $T_{\min} = 3$ (minimum exploration rounds), $p = 2$ (patience window length), and $\epsilon = 0.005$ (stagnation threshold in moderator success). We report results for both (i) early-stopped runs ($\mathsf{M}_{\text{early}}$) and (ii) full 10-round runs (M_{10}), to separate early-stopped behavior from the full round-budget setting.

Orchestration. We implement Collab-Rec with the Google Agent Development Kit (ADK)[1] to support modular role separation, looping, and parallel agent execution. Agents execute in parallel with an independent local state, while the deterministic moderator maintains global state, applies the rejection strategy, performs grounding checks, and generates structured feedback.

Initialization. At $t = 1$, the first proposals are generated without a prior collective state. For the reliability and hallucination diagnostics, we initialize $d_{a_i,0} = 1$ and $h_{a_i,0} = 0$ and then compute true values from round $t = 1$ onward.

Additional configuration for ablations. To study ablations (RQ5), we run an additional set of experiments on 150 queries with two representative model backbones (*Gemini* and *Olmo-7b*), using the *Aggressive* rejection strategy and a maximum of five rounds under the same early-stopping mechanism (see Section 5.5).

4.2.2 Baselines. We compare Collab-Rec against a diverse set of baselines designed to isolate different aspects of recommendation performance, including popularity bias, single-agent reasoning, and classical multi-objective optimization.

- **Random recommender (RandRec).** A non-language-model baseline that ignores the query and returns a reproducible random set of $k = 10$ cities [38].
- **Top popularity recommender (TopPop).** A non-personalized baseline that returns the globally most popular cities, independent of user constraints [18].
- **Constraint Optimizer (MILP) baseline.** To provide a controlled optimization-based reference, we introduce a Mixed-Integer Linear Programming (MILP) baseline that selects exactly $k = 10$ cities from the catalog:

$$\max_{x_1,\ldots,x_n} \sum_{i=1}^{n} x_i u_i \quad \text{s.t.} \quad x_i \in \{0,1\}, \quad \sum_{i=1}^{n} x_i = k. \tag{13}$$

  The utility of city i is defined as

$$u_i = r_i + s_i + \tilde{p}_i, \tag{14}$$

  where r_i and s_i denote the relevance and sustainability scores computed from the same filter-based signals used by the moderator, and $\tilde{p}_i$ is a conditional popularity score. Specifically, $\tilde{p}_i = p_i$ when the city's categorical popularity level matches the popularity preference expressed in the query, and $\tilde{p}_i = 0$ otherwise. Thus, the term rewards alignment with the requested popularity level; it is not an inverse-popularity penalty and does not uniformly discourage popular cities.

[1] https://google.github.io/adk-docs

This MILP should be interpreted as a structured-input reference rather than a complete non-LLM recommender baseline. It assumes that the query has already been converted into structured filters and that all candidate utilities are available. It also optimizes additive utility under a cardinality constraint and does not model iterative candidate repair, explanation traces, or diversification objectives such as MMR/xQuAD-style coverage, Pareto/weighted reranking, or constrained diversification. Stronger catalog-only baselines of this kind are important future comparators, and we do not claim practical superiority over them in this paper.

- **Single-Agent Single-Iteration (SASI).** A single LLM is prompted with the full query and returns one ranked list of $k = 10$ cities without iterative refinement or a specialist decomposition.
- **Multi-Agent Single-Iteration (MASI).** The three specialist agents each propose an initial list; the moderator grounds and aggregates once, without any additional coordination rounds. For brevity in tables, we also write M_1 for MASI, as a single-round MAMI run is equivalent to MASI.

While open-ended conversational queries do not provide exhaustive ground-truth labels or a verifiable set of globally optimal recommendations, the MILP baseline serves as a controlled and interpretable reference point, complementing RandRec and TopPop. Together, these baselines enable us to quantify how Collab-Rec balances multiple objectives — relevance, diversity, and fairness in practice, even when the true optimal recommendation set is unobservable.

4.2.3 Large language model backbones. We evaluate six reasoning-capable LLMs spanning proprietary and open-source families and a range of parameter scales:

- **Large proprietary models:** Gemini (`gemini-2.5-flash`) [17], and Claude (`claude-sonnet-4-5`) [6].
- **Medium-scale open models:** `gpt-oss-20b` [3], and `gemma-3-12b` [51].
- **Small-scale open models:** `gemma-3-4b` [51], and `olmo3-7b-instruct` [44].

Throughout the paper, we drop version suffixes and refer to these models as *Gemini*, *Claude*, *GPT-OSS*, *Gemma-12b*, *Gemma-4b*, and *Olmo-7b* respectively; these short forms are used consistently in all tables, figures, and body text.

4.3 Evaluation Metrics

We evaluate Collab-Rec from three complementary perspectives: *final recommendation quality*, *per-round agent behavior*, and *computational overhead*.

4.3.1 Final recommendation quality.

Grounded relevance via moderator success (Equation 11). Since open-ended tourism queries lack explicit interaction logs and canonical relevance labels, we measure moderator success as defined in Equation 11, the fraction of recommended cities that satisfy all query constraints under catalog validation. A value of 1 indicates that all recommended cities satisfy all structured constraints under knowledge base validation.

Popularity-bias and diversity. To quantify concentration on short-head destinations, we compute the Gini index [27] and normalized entropy [33] over the distribution of recommended cities aggregated across queries. Lower Gini indicates less concentration, and higher normalized entropy indicates a more even distribution.

Catalog coverage. We additionally report coverage, defined as the fraction of the 200-city catalog that appears at least once in the recommendations. Coverage complements Gini and entropy by directly capturing long-tail breadth.

4.3.2 Agent behavior (per round).

Reliability. We report agent reliability $d_{a_i,t}$ (Equation 9), which measures stability of an agent's ranked list across consecutive rounds.

Invalid-output rate. We report hallucination rate $h_{a_i,t}$ (Equation 10), defined as the fraction of an agent's recommendations that are out-of-catalog or violate the collective rejection constraint.

4.3.3 Computational overhead. We measure (i) wall-clock time per query and (ii) total token usage. Both are aggregated over all agent calls and moderator operations across all executed rounds. These metrics quantify the practical cost of multi-round refinement and motivate early stopping (Section 3.6).

4.4 Research Questions

The experiments are structured around five research questions:

RQ1 Does multi-agent, multi-round refinement improve grounded recommendation quality compared to single-agent prompting, single-round aggregation, and non-language-model baselines?
RQ2 Does multi-round refinement reduce popularity concentration and increase long-tail coverage of destinations?
RQ3 How do specialist agents evolve across rounds in terms of reliability and invalid-output behavior under moderator feedback?
RQ4 What latency and token overhead does multi-round refinement introduce, and how effectively does online early stopping mitigate these costs?
RQ5 How sensitive is the method to its scoring components and design choices, as assessed by ablation studies?

Section 5 reports results and discussion for RQ1–RQ5, with supplementary plots for additional rejection strategies provided in the appendix.

5 Results and Discussion

We evaluate whether enabling multiple specialist agents to coordinate over multiple rounds (MAMI) yields tangible benefits over (i) traditional non-LLM baselines (RandRec, TopPop), (ii) a single-agent single-iteration baseline (SASI), and (iii) a multi-agent single-iteration baseline (MASI). Unless stated otherwise, results are reported over 900 stratified queries (Section 4.1.1) and averaged across queries. For MAMI, we report two operational modes: a *patience-based early-stopped* variant ($\mathsf{M}_{\text{early}}$) and a full 10-round run (M_{10}). We further compare two rejection strategies in the moderator: *Aggressive* (discard any flagged city) and *Majority* (discard only if at least two agents flag a city).

Unless noted otherwise, figures in the main text report MAMI with the *Aggressive* rejection strategy (Figure 6 shows both variants). Tables include results for both *Aggressive* and *Majority*. Detailed results for the *Majority* strategy are provided in the Appendix A–C.

Interpretation of "multi-round iterative constrained refinement" Throughout this section, we use "coordination" or "refinement"in a pragmatic sense: agents do not communicate directly or strategize against each other; instead, they iteratively *adapt* to structured moderator feedback. Mechanistically, MAMI can be understood as an iterative constrained search/optimization loop that alternates between candidate generation (agents) and feasibility-aware aggregation (moderator), with multi-round execution providing additional opportunities to repair constraint violations and escape short-head popularity modes.

5.1 RQ1: System-level impact on grounded recommendation quality

RQ1 asks: *Does the multi-agent, multi-round (MAMI) approach improve final recommendation quality compared to SASI, MASI, and the simple non-LLM baselines (RandRec, TopPop)?* To contextualize the results, we also compare against the MILP reference, which operates over fully structured inputs and is not a direct competitor in the natural-language setting addressed by Collab-Rec. We operationalize system-level quality primarily via

moderator success, i.e., the fraction of recommended cities that satisfy query constraints under catalog validation (Section 4.3).

Overall effectiveness across models and strategies. Table 1 shows that multi-round refinement improves grounded quality for all evaluated model families and for both rejection policies. The improvements are substantial when comparing MAMI to SASI. For example, under *Aggressive* rejection, *Claude* increases from 0.465 (SASI) to 0.657 (M_{early}), and *Olmo-7b* increases from 0.536 to 0.666. Even compared to the stronger single-round multi-agent baseline (MASI), MAMI remains consistently better: e.g., under *Aggressive* rejection, *Gemini* improves from 0.616 (MASI) to 0.648 (M_{early}), and *Gemma-12b* improves from 0.618 to 0.647.

Table 1. Performance comparison across model sizes and rejection strategies. Average moderator success scores are reported for SASI, M_1 (=MASI), M_{early}, and M_{10} (full 10-round run without early stopping). The higher the score, the better. Best scores per model and rejection strategy (Rej. Strat.) are highlighted in **bold**. **Succ. Rate (%)** = fraction of queries where M_{early} moderator success strictly exceeds that of M_1 (=MASI). **Odds Ratio** = $\text{Wins}_{M_{early}\text{ vs }M_1}$ / $\text{Losses}_{M_{early}\text{ vs }M_1}$, i.e., the ratio of queries where M_{early} strictly outperforms M_1 to queries where it strictly underperforms, excluding ties, computed using a Fisher's exact test.

Model Size	Model	Rej. Strat.	$\mathcal{M}$ Success Scores↑				Early-Stopping Metrics			
			LLM-Baselines		Proposed		Succ. Rate (%)	Avg. Conv.	Odds	Ties (%)
			SASI	M_1	M_{early}	M_{10}	$M_{early} > M_1$	Round	Ratio	$M_{early} = M_1$
Big	Claude	A	0.465	0.594	**0.657**	0.620	59.6	3.72	17.54	26.2
		M			**0.647**	0.632	50.7	3.82	16.58	36.9
	Gemini	A	0.520	0.616	**0.648**	0.617	58.4	3.73	15.58	26.8
		M			**0.636**	0.621	49.1	3.96	12.50	37.0
Mid-sized	GPT-OSS-20b	A	0.632	0.640	**0.661**	0.625	51.9	3.73	4.89	24.7
		M			**0.655**	0.634	47.4	3.87	5.62	32.6
	Gemma-12b	A	0.616	0.618	**0.647**	0.617	57.9	3.79	18.85	28.8
		M			**0.637**	0.621	49.4	4.00	13.52	37.1
Small	Olmo-7b	A	0.536	0.597	**0.666**	0.624	70.1	3.84	17.46	13.1
		M			**0.662**	0.628	67.0	3.88	16.82	16.7
	Gemma-4b	A	0.615	0.617	**0.650**	0.618	61.9	3.78	23.05	25.2
		M			**0.636**	0.620	48.3	4.01	16.84	39.9

Non-LLM baselines: RandRec (RR) = 0.501, TopPop (TP) = 0.676, MILP = 0.730; reported for reference and omitted from the table body for clarity.

The non-LLM baselines help contextualize the trade-offs. MILP achieves the highest grounded success (0.73), outperforming all Collab-Rec variants on the moderator metric. This outcome is expected because MILP formulates recommendations as a closed-form optimization problem over fully structured catalog inputs, predefined constraints, and a hand-specified utility function. In contrast, Collab-Rec operates directly on natural-language queries and uses iterative multi-agent refinement with deterministic catalog validation to generate and repair recommendations. Despite this harder setting, Collab-Rec achieves ~90% of MILP's grounded success (0.66 vs. 0.73) while also reducing recommendation concentration and improving diversity (Section 5.2). We therefore treat MILP as a structured-input upper bound on grounded success rather than a direct competitor in a fully conversational recommendation setting.

Collab-Rec's complementary advantages lie in its ability to handle open-ended natural-language requests, iteratively repair constraint violations, expose transparent per-agent diagnostics, and support modular stakeholder

Table 2. Pairwise statistical significance comparisons across models and rejection strategies (MAMI with early stopping, i.e., M_{early}). P-values are Bonferroni-corrected over $m = 3$ planned comparisons ($\alpha_{corr} = 0.05/3 \approx 0.017$); the null hypothesis ($H_0$) assumes equal performance between the compared methods. Significant results ($p_{corr} < 0.05$, equivalently raw $p < 0.017$) are highlighted in green. Each cell reports: t-stat ($p \mid p_{corr}$).

Model	Rejection Strategy	Groups		
		M_{early} vs M_1	M_{early} vs SASI	M_1 vs SASI
Claude	A	3.99 (0.00 | 0.00)	17.10 (0.00 | 0.00)	14.28 (0.00 | 0.00)
	M	2.85 (0.00 | 0.01)	16.04 (0.00 | 0.00)	14.00 (0.00 | 0.00)
Gemini	A	3.63 (0.00 | 0.00)	11.39 (0.00 | 0.00)	8.60 (0.00 | 0.00)
	M	2.22 (0.03 | 0.08)	10.32 (0.00 | 0.00)	8.61 (0.00 | 0.00)
GPT-OSS-20b	A	2.53 (0.01 | 0.03)	3.55 (0.00 | 0.00)	1.08 (0.28 | 0.85)
	M	1.87 (0.06 | 0.19)	2.81 (0.01 | 0.01)	0.97 (0.33 | 0.99)
Gemma-12b	A	3.50 (0.00 | 0.00)	3.71 (0.00 | 0.00)	0.23 (0.82 | 1.00)
	M	2.29 (0.02 | 0.07)	2.56 (0.01 | 0.03)	0.27 (0.78 | 1.00)
Olmo-7b	A	7.07 (0.00 | 0.00)	13.21 (0.00 | 0.00)	7.79 (0.00 | 0.00)
	M	6.47 (0.00 | 0.00)	12.81 (0.00 | 0.00)	7.85 (0.00 | 0.00)
Gemma-4b	A	3.92 (0.00 | 0.00)	4.11 (0.00 | 0.00)	0.22 (0.83 | 1.00)
	M	2.14 (0.03 | 0.10)	2.41 (0.02 | 0.05)	0.29 (0.77 | 1.00)

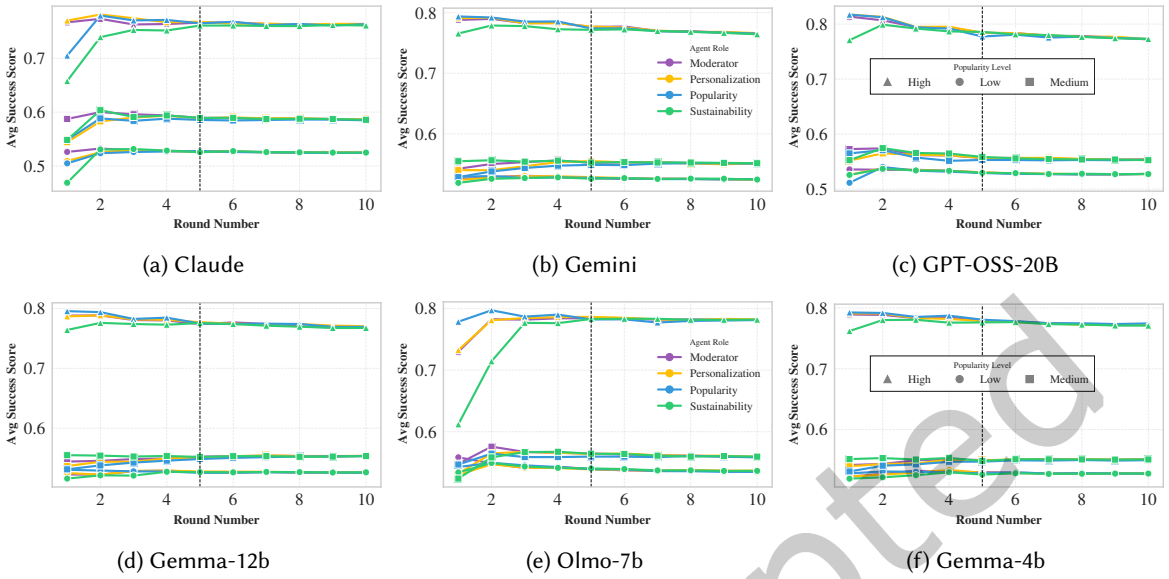
objectives without retraining. By contrast, TopPop attains relatively high grounded success (0.676) largely by repeatedly recommending the most popular cities and ignoring personalization, while RandRec performs substantially worse (0.501), underscoring the difficulty of satisfying multi-constraint queries without reasoning or grounding.

We additionally observe robustness limitations in the SASI pipeline for *Claude* and *GPT-OSS*, which fail to produce valid outputs for ∼157/900 and ∼17/900 queries respectively, further motivating multi-agent moderation.

Early stopping and convergence dynamics. A central practical question is whether multi-round gains require running the full 10-round budget. Table 1 reports that MAMI typically converges by ∼4 rounds (average convergence 3.72–4.01). As shown in Figure 3, which plots average agent success scores over refinement rounds, success rises quickly in the first 3–4 rounds and then reaches a plateau around rounds 4–5. This pattern is consistent across models and indicates diminishing returns for longer runs. Extended runs up to 20 rounds (*Gemma-4b*, *Claude*, *Olmo-7b*; see Figure 10 in Appendix A) confirm convergence within 3–4 rounds, supporting our 10-round cap.

Importantly, early stopping does *not* sacrifice quality: across models, M_{early} is often the best or near-best variant. Under *Aggressive* rejection, M_{early} is higher than M_{10} for every model in Table 1 (e.g., *Claude* 0.657 vs. 0.620, *Gemini* 0.648 vs. 0.617), indicating that later rounds can add churn without improving grounded satisfaction once the system has already settled into a feasible region.

Statistical Significance. Table 2 reports paired comparisons using Bonferroni correction ($\alpha_{corr} = 0.05/3 \approx 0.017$) over query-level distributions. The strongest and most consistent effect is MAMI vs. SASI, which is significant across nearly all models and strategies. The sole borderline case is *Gemma-4b* under *Majority* rejection, where M_{early} vs. SASI lands exactly at the correction threshold ($p_{corr} = 0.05$), leaving this comparison inconclusive under our strict criterion. MAMI vs. MASI is also significant for most settings, with exceptions under *Majority* rejection for *Gemini*, *GPT-OSS*, *Gemma-12b*, and *Gemma-4b* — where the comparison does not reach α_{corr} — indicating that gains over MASI are most reliable when the moderator applies the stricter *Aggressive* rejection strategy.



(a) Claude (b) Gemini (c) GPT-OSS-20B

(d) Gemma-12b (e) Olmo-7b (f) Gemma-4b

Fig. 3. Average agent success scores over refinement rounds under the *Aggressive* rejection strategy. The plots track performance for the *Personalization*, *Popularity*, *Sustainability*, and *Moderator* agents across LLM backbones. Results are stratified by query popularity level (low, medium, high). The dotted black line denotes the convergence plateau typically reached by rounds 4–5. This stabilization validates the patience-based early stopping protocol, as relevance gains generally diminish thereafter.

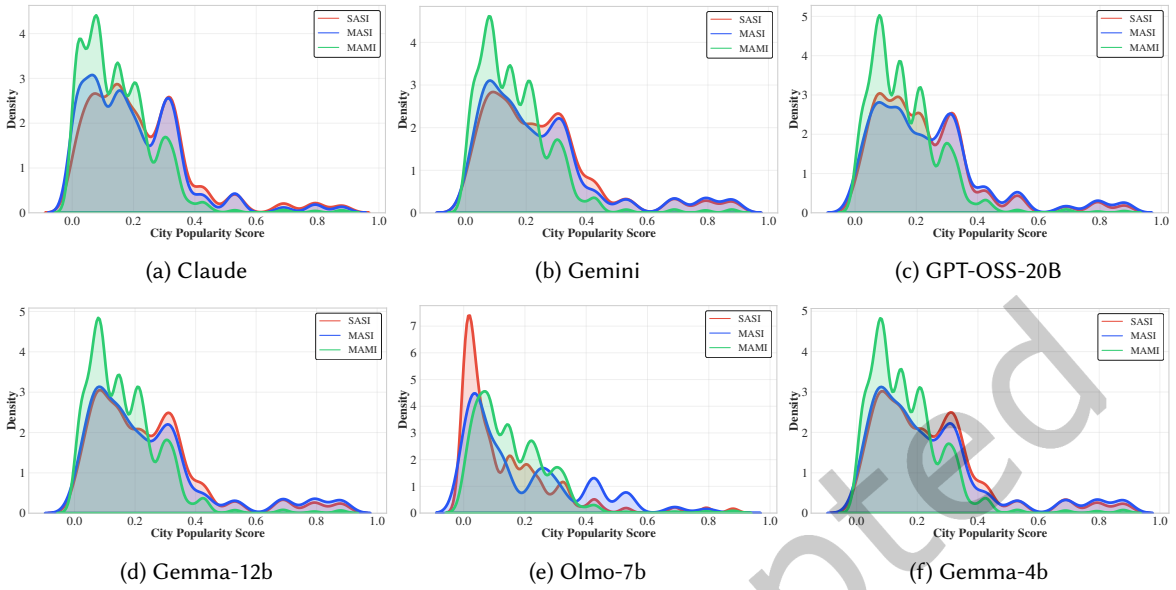
RQ1 Summary. Multi-round, multi-agent refinement improves moderator success over single-agent and single-round baselines across all evaluated LLM families. Quality improves rapidly in the first 3–4 rounds and typically plateaus by rounds 4–5. Patience-based early stopping, therefore, captures most gains while avoiding unnecessary computation.

5.2 RQ2: Popularity Bias and Diversification

RQ2 asks: *Does multi-round refinement reduce popularity concentration and increase long-tail coverage?* We evaluate popularity bias using (i) distributional shifts in recommended popularity scores (KDE in Figure 4), (ii) concentration in the catalog (Lorenz curves in Figure 5), and (iii) summary metrics: Gini and entropy (lower/higher is better) as well as catalog coverage (Table 3).

Distributional shift toward the long tail. Across models, MAMI systematically shifts recommendation mass away from the highest-popularity range and toward the mid/long tail (Figure 4). This effect is strongest under *Aggressive* rejection, where the moderator more readily discards repeated high-traffic suggestions and forces agents to explore alternatives. In contrast, SASI and MASI frequently exhibit sharper peaks in the high-popularity region, reflecting the well-known tendency of LLMs to default to canonical destinations.

Olmo-7b is a notable exception. Its SASI baseline shows pronounced density in very low popularity bins and 100% catalog coverage. *Olmo-7b* routinely emits substantially more than $k = 10$ cities per query in unmoderated mode, trivially inflating both coverage and the apparent long-tail skew in the KDE. The high Avg #recs/city of 242.5, computed after duplicate handling as all metrics in this table, indicates that *Olmo-7b* in unmoderated SASI mode emits far more than $k = 10$ unique cities per query, making the 100% coverage figure an artifact of



(a) Claude (b) Gemini (c) GPT-OSS-20B

(d) Gemma-12b (e) Olmo-7b (f) Gemma-4b

Fig. 4. Kernel Density Estimation (KDE) of city popularity distributions across methods and models for the aggressive rejection strategy. The filled regions illustrate the probability density of recommended cities according to their popularity scores. While SASI and MASI show high-amplitude peaks concentrated on popular hubs, MAMI (Round 10) displays a flatter, broader curve. This visual shift indicates a significant reduction in popularity bias and an increased coverage of lesser-known "long-tail" destinations.

over-generation rather than a signal of genuine diversity. MAMI corrects this by enforcing exactly k cities per round via the moderator, so the apparent drop from 100% to ~77–79% reflects constraint enforcement rather than a coverage regression. On the concentration metrics unaffected by output size (Gini, Lorenz curves in Figure 5), MAMI yields only a marginal improvement (Gini: 0.67 → 0.66 under M_{10}), consistent with *Olmo-7b* not exhibiting strong hub concentration to begin with.

Gini, entropy, and coverage. Table 3 provides a compact numerical view of the same phenomenon. Across most models, multi-round refinement reduces concentration and increases dispersion. For example, for *GPT-OSS* under *Aggressive* rejection, Gini decreases from 0.76 (SASI) to 0.65 (M_{early}/M_{10}), while entropy increases from 0.78 to 0.85. For *Claude* under *Aggressive* rejection, Gini decreases from 0.71 to 0.64 and coverage increases from 66.0% to 81.5%. For *Gemini* under *Aggressive* rejection, Gini decreases from 0.66 to 0.63 and coverage increases from 64.5% to 74.5%.

A key trade-off emerges between relevance and diversity across rounds. While relevance improvements tend to plateau after rounds 4–5 (RQ1: Section 5.1), diversity continues to increase modestly with additional rounds: in Table 3, M_{early} yields the best diversity metrics for five of six models, whereas M_{10} is slightly preferable for Gemini. This suggests that extra rounds can be used as a "diversity budget" when coverage is prioritized, albeit with additional cost (RQ4: Section 5.4).

The non-LLM baselines illustrate the extremes of the relevance–diversity trade-off. RandRec achieves near-perfect equity (Gini 0.08, entropy 0.99, coverage 99.5%) but produces low-quality recommendations, whereas TopPop is highly concentrated (Gini 0.95, entropy 0.43, coverage 5%) due to repeatedly suggesting the most popular cities, yielding only 10 unique cities out of the 200-city catalog.

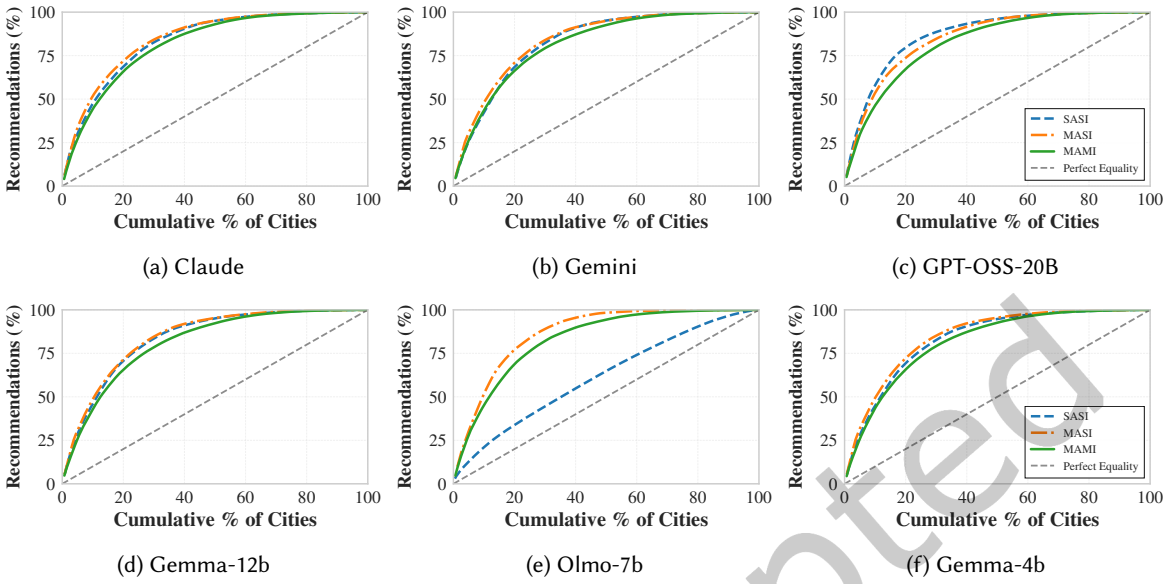
Model	Rej. Strat.	Gini ↓ (Entropy ↑)				Coverage (%) (Avg #recs/city) ↑			
		SASI	M_1	M_{early}	M_{10}	**SASI**	M_1	M_{early}	M_{10}
Claude	A	0.71 (0.82)	0.69 (0.82)	0.65 (0.85)	**0.64** (0.86)	66.0 (49.9)	67.0 (67.2)	**81.5** (55.2)	**81.5** (55.2)
	M			0.67 (0.84)	**0.67** (0.84)		69.0 (65.2)	75.0 (60.0)	**77.0** (58.4)
Gemini	A	0.66 (0.84)	0.68 (0.83)	**0.63** (0.86)	0.64 (0.86)	64.5 (57.9)	65.5 (68.6)	73.5 (61.2)	**74.5** (60.3)
	M			0.67 (0.84)	**0.66** (0.84)		66.5 (67.7)	**73.0** (61.8)	**73.0** (61.6)
GPT-OSS-20b	A	0.76 (0.78)	0.71 (0.81)	0.65 (0.85)	**0.65** (0.85)	81.5 (53.4)	76.0 (59.2)	80.0 (56.2)	79.5 (56.6)
	M			0.68 (0.84)	**0.67** (0.84)		76.5 (58.8)	83.0 (54.2)	**85.0** (52.9)
Gemma-12b	A	0.69 (0.82)	0.69 (0.82)	0.64 (0.86)	**0.63** (0.86)	63.5 (70.5)	67.0 (67.2)	71.5 (62.9)	**72.0** (62.5)
	M			0.67 (0.84)	**0.64** (0.85)		64.5 (69.8)	68.0 (66.2)	**69.0** (65.2)
Olmo-7b	A	0.67 (0.84)	0.73 (0.77)	0.67 (0.84)	**0.66** (0.84)	**100** (242.5)	38.5 (116.9)	77.0 (58.4)	79.5 (56.6)
	M			0.70 (0.82)	**0.68** (0.84)		39.0 (115.4)	77.0 (58.4)	79.0 (57.0)
Gemma-4b	A	0.68 (0.83)	0.70 (0.82)	0.64 (0.86)	**0.63** (0.86)	61.0 (73.3)	69.5 (64.7)	70.0 (64.3)	**74.0** (60.8)
	M			0.66 (0.84)	**0.65** (0.85)		66.0 (68.2)	71.0 (63.4)	**71.5** (62.9)

Note: Non-LLM baselines: Gini (Entropy): **RR** = 0.08 (0.99), **TP** = 0.95 (0.43), MILP = 0.78 (0.77). Coverage: **RR** = 99.5%, **TP** = 5%, MILP = 48.0%.

Table 3. **Gini (Entropy)** and **Coverage metrics** across models and rejection strategies. The table reports concentration bias and long-tail coverage for each LLM model under *Aggressive* **(A)** and *Majority* **(M)** rejection strategies. Lower Gini indicates reduced concentration bias, and higher entropy reflects a more equitable recommendation distribution. **Coverage (%)** is the fraction of the 200-city catalog appearing at least once in validated recommendations; **Avg #recs/city** is the number of valid city appearances divided by the number of unique catalog cities covered. Row-wise maximum coverage is highlighted using **bold**. MAMI variants consistently achieve higher coverage and more balanced distributions than SASI and MASI baselines, except in *Olmo-7b*. All counts are computed after parsing, catalog validation, removal of invalid outputs, and duplicate handling; for methods with failed or invalid outputs, total city appearances may therefore fall below 900×10.

The MILP baseline occupies a middle ground in catalog concentration (Gini 0.78, entropy 0.77, coverage 48%), achieving lower concentration than TopPop while still exhibiting notable popularity bias. This highlights the limitations of static scalarized objectives in capturing nuanced diversity preferences. In contrast, Collab-Rec operates directly on natural-language queries and uses iterative multi-agent constrained refinement to interpret intent and repair constraint violations. Across SynthTRIPs, Collab-Rec variants generally achieve broader catalog coverage and lower recommendation concentration than both single-agent/single-round baselines and the structured MILP reference, surfacing more lesser-visited destinations (Table 3). These results should be interpreted as controlled evidence of diversity and bias mitigation on the SynthTRIPs benchmark, rather than as proof of superiority over stronger real-world constrained-diversification or reranking approaches; validation with user studies and logged interaction data remains future work.

RQ2 Summary. Yes. Multi-round refinement consistently mitigates popularity concentration and increases long-tail coverage. Diversity improves across rounds even after relevance plateaus, exposing an explicit relevance–diversity–cost trade-off: more rounds modestly improve diversity, while early stopping preserves relevance and reduces latency.



(a) Claude (b) Gemini (c) GPT-OSS-20B

(d) Gemma-12b (e) Olmo-7b (f) Gemma-4b

Fig. 5. Lorenz curves showing recommendation concentration across the 200-city catalog. The x-axis represents the cumulative percentage of cities, and the y-axis represents the cumulative percentage of recommendations. The diagonal ($y = x$) indicates perfect equality. Curves that bow further below the diagonal indicate higher concentration, with a few "short-head" cities dominating recommendations. MAMI/M_{10} (solid) consistently bows less than SASI (dashed) and MASI/M_1 (dot-dashed), indicating reduced popularity bias and a more equitable, long-tail distribution of recommended destinations.

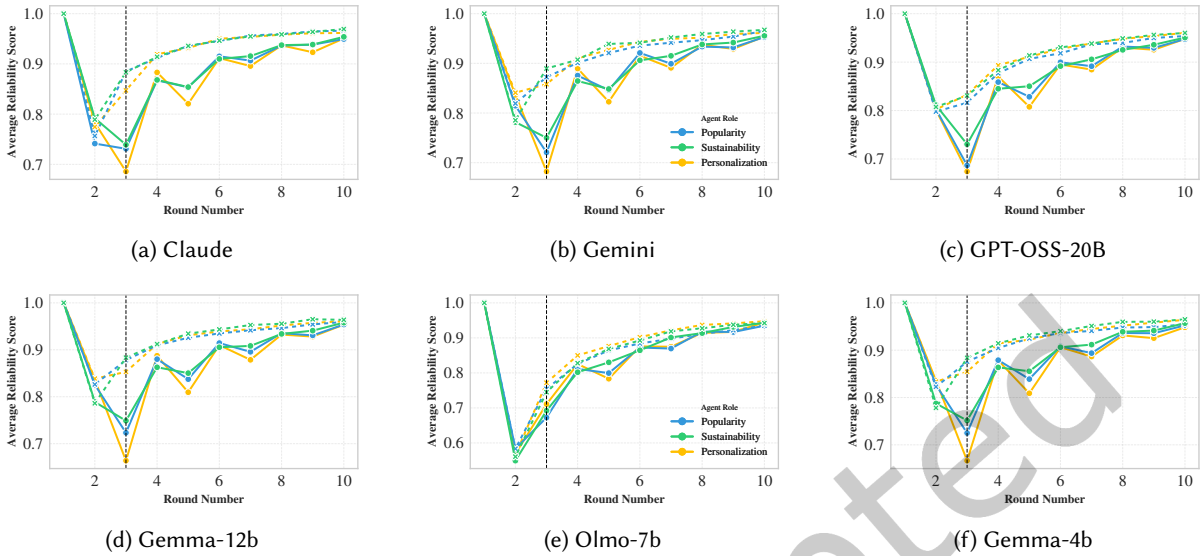
5.3 RQ3: Agent reliability and hallucination behavior

RQ3 asks: *How do specialist agents behave across rounds in terms of stability (reliability) and hallucination tendency?* We focus on two behavioral signals: (i) *reliability* (recommendation stability across rounds) and (ii) *hallucinations*, operationalized as out-of-catalog or repeatedly rejected cities (Section 3.5.3).

Reliability evolves from exploration to stabilization. Figure 6 reveals a consistent pattern across backbones. In early rounds (typically rounds 2–3), reliability dips as agents explore the candidate space and react to moderator feedback (i.e., they replace previously suggested cities that were rejected or penalized). As the collective offer becomes feasible and agents align on a shared set of candidates, reliability increases steadily. By round 10, reliability for all agents typically exceeds 0.90, indicating that the system has stabilized and that additional rounds are unlikely to meaningfully change the final list.

The rejection strategy moderates this stability — exploration trade-off. *Majority* rejection yields smoother and consistently higher reliability than *Aggressive* rejection, because fewer cities are discarded and the candidate set churns less. This improved stability comes at the cost of slightly slower convergence (RQ1: Section 5.1) and often slightly weaker diversity gains (RQ2: Section 5.2).

Hallucinations decrease with grounded feedback. Structured output constraints reduce a large portion of out-of-inventory hallucinations by enforcing syntactic validity; remaining catalog-membership violations are caught by the moderator's post-generation validation. Residual hallucinations can still occur when agents reintroduce cities that were previously rejected by the moderator. Across models, we observe that hallucination tendency decreases



(a) Claude (b) Gemini (c) GPT-OSS-20B

(d) Gemma-12b (e) Olmo-7b (f) Gemma-4b

Fig. 6. Agent behavior metrics showing agents' reliability scores over multiple rounds. Solid lines represent results from the *Aggressive* rejection strategy, while dotted lines represent results from the *Majority* voting strategy. The black-dashed vertical line marks round 3 (T_{min}), after which the patience criterion becomes active. The early-round dip in reliability reflects agents adapting their proposals in response to moderator feedback; as they align on a shared candidate set, reliability scores increase and stabilize in later rounds.

across rounds: after initial exploration, agents increasingly comply with moderator feedback and stop proposing invalid or repeatedly rejected items.

For example, under the *Aggressive* rejection strategy, the average agent-level hallucination rate for *Gemini* is approximately 0.002 in round 1, decreases to 0.000093 by round 2, and converges to 0 in subsequent rounds. Similarly, for *Gemma-4b* under the *Majority* strategy, the rate starts at around 0.0017 in round 1, drops to 0.000056 by round 2, and reaches 0 thereafter. Comparable dynamics are observed across the remaining backbones; we therefore omit the full numerical breakdown for brevity. By the final rounds, hallucination rates converge toward zero across backbones, showing that iterative feedback can enforce grounding even when logit-level constraints are not natively supported.

RQ3 Summary. Agents initially explore aggressively (lower reliability in early rounds) but become increasingly stable and compliant as feedback accumulates. *Majority* rejection promotes smoother stability; *Aggressive* rejection promotes stronger churn and faster correction. Across both policies, hallucination tendency decreases sharply across rounds under catalog grounding and structured feedback.

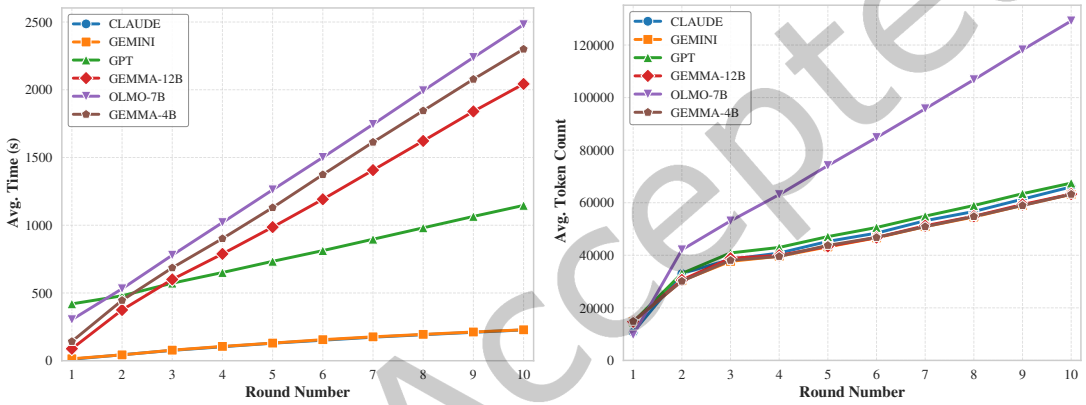
5.4 RQ4: Time and cost complexity

RQ4 asks: *What time and token overheads does multi-round refinement introduce, and how can they be mitigated?* Figure 7 summarizes time and token usage over rounds.

Overhead scales approximately linearly with rounds. As expected, both inference latency and token usage increase with the number of rounds. However, absolute costs vary substantially by deployment mode. For API-served proprietary models, cumulative latency stays below 500 seconds for a full 10-round run: *Gemini*

averages 133.89 s (*Aggressive*) and 211.63 s (*Majority*); *Claude* averages 382.01 s under *Majority* (see Figure 7 for the *Aggressive* result). For locally served open-source models, costs are much higher: under *Aggressive* rejection, *Gemma-12b* reaches 2042.61 seconds and *Gemma-4b* reaches 2299.60 seconds. This gap reflects the absence of endpoint-level batching/parallelization and the sequential execution of multiple agents and moderation on local hardware. *Gemma-4b*'s higher cumulative latency relative to the larger *Gemma-12b* is counterintuitive but reflects greater per-round output-validation overhead: the smaller model produces invalid or out-of-catalog outputs more frequently, triggering additional moderator processing and candidate replacement steps rather than a higher per-token generation cost.

Token usage exhibits a similar pattern under practical deployment. When utilizing early stopping (which averages ~4 rounds), cumulative token counts across all rounds sum to ~44k for *Gemini* (both strategies), while *Claude* (*Majority*) and *Olmo-7b* (*Majority*) accumulate higher totals (68,578.59 and 75,874.79 tokens respectively), consistent with more verbose justifications and longer contextual prompts that grow with each additional round (Figure 13). *Aggressive* rejection typically increases token use by inducing more frequent candidate replacement.



(a) Time Taken (b) Tokens Used

Fig. 7. Average wall-clock time (left) and token usage (right) as a function of the refinement round for all models using the *Aggressive* rejection strategy. Note that the *Avg. Time (s)* axis reflects cumulative wall-clock time across all 10 rounds; for API-served models, the per-round wall-clock time equals the maximum over the 3 parallel agent calls plus moderator overhead, whereas for locally served open-source models, agents execute sequentially on-device without endpoint-level batching. The token count is the total number of tokens used across all agents and the moderator across all rounds.

Early stopping as a practical mitigation. The main operational takeaway is that early stopping captures most of the quality gains (RQ1) while substantially reducing costs. When we stop at round 4, *Gemini* (*Aggressive*) drops to 60.41 seconds, and *Claude* (*Majority*) drops to 87.16 seconds. In the local cluster setting, early stopping reduces *Gemma-12b* (*Aggressive*) to 462.51 seconds and *Gemma-4b* (*Majority*) to 241.51 seconds. Thus, while multi-round refinement is more expensive than single-shot baselines, dynamic termination makes the approach considerably more practical—especially for API-served backbones.

Beyond termination strategies, costs can be further mitigated by utilizing "non-reasoning" or distilled models for specific agent roles, which significantly reduces per-token pricing and inference time without compromising the structural grounding of the recommendation [36]. For production environments, we suggest *agent pruning* [60], i.e., deactivating specific agents once their proposals stabilize, and *semantic caching* [12, 42], which reuses refinement states for semantically similar user queries to avoid redundant computation. However, we leave the

implementation and evaluation of these strategies to future work, as they require additional infrastructure and are not natively supported by all backbones.

Deployment scope. Despite the gains from early stopping, the current implementation remains too slow for interactive real-time recommendation. We therefore position Collab-Rec as a research prototype suitable for offline or high-latency planning scenarios. Techniques such as smaller role-specific models, batching, agent pruning, and semantic caching may reduce cost in future higher-throughput settings, but we do not claim production readiness in the current system.

Reproducibility resources. The Collab-Rec repository[2] provides a complete workflow with dependencies, API / backend setup, example requests, and GPU / cloud instructions to enable replication and experimentation.

RQ4 Summary. Costs scale with the number of rounds and are highly deployment-dependent: API-served models are substantially faster than locally hosted open-source models in our setup. Early stopping at ∼4 rounds is an effective mitigation that preserves most quality gains while reducing latency and token usage.

5.5 RQ5: Ablation Analysis

RQ5 asks: *What is the individual contribution of each moderator scoring component (Relevance, Reliability, Hallucination penalty, and ranking) to overall recommendation quality?*

To assess the contribution of individual moderator components to overall recommendation quality, we conduct an ablation study on two representative models: *Gemini* (a high-capacity closed-source model) and *Olmo-7b* (a representative open-source model). Experiments are performed on 150 stratified queries using the *Aggressive* rejection strategy with five rounds and early stopping, as described in Section 3.6. We systematically ablate the core components of the moderator's scoring function—*Relevance* ($r_{a_i,t}$), *Reliability* ($d_{a_i,t}$), *Hallucination* ($h_{a_i,t}$), and the *ranking mechanism* ($\text{rank}_{a_i,t}(c)$) defined in Equation 4. Each ablated variant is compared against the default $\mathsf{M}_{\text{early}}$ configuration, and we report the mean values of the evaluation metrics (*Moderator Success*, *Diversity (Gini)*, and *Reliability scores*) for each setting. Statistical significance is evaluated using paired t-tests [29] ($p < 0.05$) against the default distribution, with significant differences highlighted using a $*$ on the bar plot in Figure 8.

Across models, we observe distinct sensitivity patterns. While *Gemini* remains largely stable under component removal, exhibiting only marginal fluctuations across *Moderator Success*, diversity, and reliability, *Olmo-7b* shows substantially greater dependence on explicit moderator signals. Removing *Relevance* reduces Moderator Success from 0.65 to 0.63 and lowers reliability from 0.63 to 0.59 (both significant). Eliminating *Reliability* further decreases reliability to 0.48 and slightly increases diversity (Gini 0.54), underscoring the stabilizing role of structured scoring signals for smaller models. In contrast, removing the hallucination penalty results in minimal changes in *Moderator Success* and *diversity*, suggesting that hallucination control is largely enforced through structured output constraints rather than the moderator's scoring term. Notably, diversity metrics remain broadly stable across settings (Gini 0.54–0.60), indicating that popularity bias mitigation persists even under component-level modifications. Since *Gemini* exhibits only minor variation across ablations, we present a detailed visualization of *Olmo-7b* in Figure 8, where the impact of removing individual moderator components is more pronounced and thus more informative.

This ablation study is intentionally limited. It isolates the effect of removing individual scoring components while keeping $\lambda_r = \lambda_d = \lambda_h = 1$ fixed, and therefore it is not a sensitivity analysis over the scalarization weights. It also covers two model families, 150 queries, the *Aggressive* rejection strategy, and a five-round budget. The results should consequently be interpreted as indicative evidence about component behavior rather than as a comprehensive robustness claim. Systematic exploration of alternative λ configurations, learned weight calibration, and broader model/query settings remains future work.

[2]https://github.com/ashmibanerjee/collab-rec

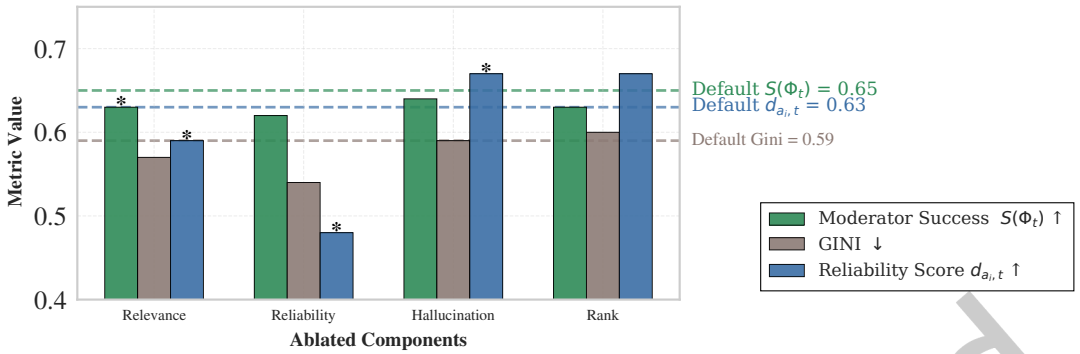


Fig. 8. Olmo-7b ablation results with default values (with all components) shown as horizontal dashed lines. Moderator Success is shown in green, Gini in brown, and Reliability Score in blue. The horizontal lines indicate the default values for each metric, so that the effect of removing each component is clearly visible. The stars indicate significant differences from the default setting, with $*$ for $p < 0.05$. The results show that removing the Relevance and Reliability components leads to significant drops in Moderator Success and Reliability Score, while removing the Hallucination component has a less pronounced effect. Diversity metrics (Gini) remain relatively stable across ablations, suggesting that popularity bias mitigation is not solely dependent on any single scoring component.

RQ5 summary. The ablation study indicates that the framework is robust for higher-capacity models, while the moderator's scoring components are particularly beneficial for stabilizing performance in smaller LLMs. These results are encouraging but remain based on fixed λ values, two model families, and 150 queries. Exploring alternative weight configurations and a broader range of models remains a promising direction for future work.

5.6 Summary of Results

To summarize, our results show that multi-agent multi-round refinement (MAMI) consistently outperforms single-agent (SASI) and single-round (MASI) baselines in moderator success, with most relevance gains saturating after rounds 4–5 (RQ1). Beyond quality improvements, MAMI also mitigates popularity bias by reducing recommendation concentration (lower Gini, higher entropy) and improving long-tail coverage (RQ2). The iterative refinement process further stabilizes agent behavior: structured feedback and rejection signals progressively reduce hallucinations and enhance groundedness (RQ3). While additional rounds increase computational cost, early stopping retains the majority of quality improvements while substantially lowering latency for API-based models, though local execution remains resource-intensive (RQ4). Finally, the ablation analysis confirms that the multi-objective scoring behaves as expected: smaller models benefit strongly from explicit relevance and reliability signals, whereas the hallucination penalty term has only a minor effect, since grounding is mostly enforced by the structured-output constraints (RQ5).

Additional evidence (Appendix). To complement the main RQ1 results, Appendix A reports per-round success trajectories under the *Majority* rejection strategy, stratified by query popularity levels. The trajectories show that improvements concentrate in the early rounds and typically stabilize around rounds 4–5. Appendix A also includes extended twenty-round runs, confirming that additional rounds yield diminishing returns rather than delayed late-stage improvements, which supports the early-stopping criterion in Equation 12.

Appendix B provides distribution-level diagnostics under the *Majority* rejection strategy. The kernel density plots visualize how multi-round refinement shifts recommendations away from the highest-popularity region,

and the Lorenz curves show reduced concentration across the catalog relative to single-shot and single-round baselines. These plots provide an interpretable complement to the aggregate Gini, entropy, and coverage metrics reported in the main text (Section 5.2).

Appendix C reports round-by-round time and token growth under the *Majority* rejection strategy. The plots make explicit that both wall-clock time and token usage increase approximately monotonically with additional rounds, reinforcing the practical value of early stopping once relevance stabilizes.

Reproducibility details (Appendix). The complete prompt templates, role constraints, and iterative constrained refinement context injections used to elicit specialist behavior are documented in Appendix D. This documentation clarifies the exact mechanisms by which moderator feedback constrains revisions, limits churn (to at most 3 replacements), and enforces closed-catalog generation through structured outputs. The ablation experiments in Section 5.5 rely on the same structured-output constraints and revision rules used in the main experiments.

6 Conclusion

This paper presents Collab-Rec, a large-language-model-based agentic recommendation framework for tourism recommendation under competing stakeholder objectives. The method decomposes the recommendation task into three specialist agents that represent personalization, popularity, and sustainability objectives, and coordinates them through a deterministic moderator. The moderator enforces catalog grounding, validates structured constraints, aggregates proposals using a transparent multi-objective scoring policy, and stabilizes computation through an online termination protocol. While Collab-Rec demonstrates the potential of multi-agent coordination for balanced tourism recommendations, it remains a research prototype and is not yet optimized for real-time deployment.

Across a stratified evaluation of 900 benchmark queries and six model backbones, the empirical results show that multi-agent multi-round refinement consistently improves moderator success while simultaneously reducing short-head concentration relative to single-shot and single-round variants. The analyses further show that multi-round execution is most beneficial in the first few rounds: improvements typically plateau after a small number of iterations, which motivates early stopping as a practical mechanism for reducing latency and token usage without sacrificing most of the quality gains.

Limitations and future work. The present evaluation is offline and based on a synthetic benchmark with a closed catalog. Although this design enables controlled catalog grounding and reproducible constraint checks, it does not demonstrate real-world user satisfaction, logged interaction performance, or actual sustainability effects on tourism flows. The moderator score is also a heuristic scalarization of agent- and list-level diagnostics rather than an item-level utility model, and we do not provide a sensitivity analysis over the λ weights. Furthermore, the MILP baseline should be viewed as a structured-input reference, not as a substitute for stronger catalog-grounded reranking and constrained-diversification baselines such as deterministic filter-rankers, MMR/xQuAD-style methods, Pareto/weighted rerankers, or explicit diversity-constrained optimizers. Finally, the current latency is too high for interactive deployment. Future work should evaluate the framework with human preference judgments and logged interactions, calibrate or learn the moderator weights, add item-level utility estimates, compare against stronger catalog-only rerankers, and investigate batching, caching, agent pruning, and smaller role-specific models.

GenAI Usage Disclosure

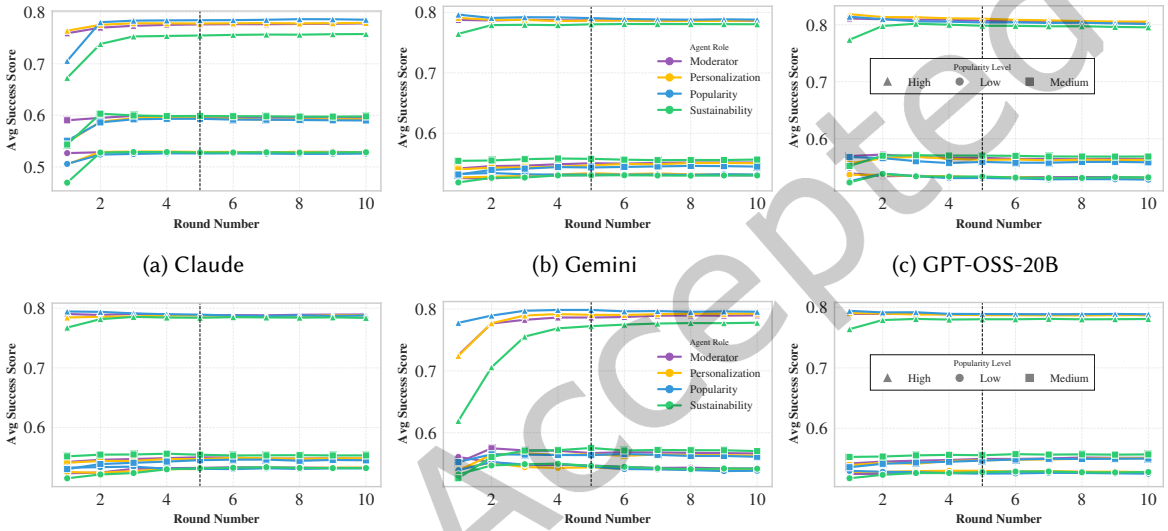
Generative AI tools, including ChatGPT (OpenAI), Claude (Anthropic), and Gemini (Google), were used in supporting roles during this research. These tools assisted with refining code snippets, improving sentence clarity,

identifying grammatical inconsistencies, and enhancing readability. They were not used to generate the scientific contributions, analyses, results, or core intellectual content of this manuscript.

All AI-generated suggestions were carefully reviewed, edited wherever necessary, and validated by the authors. The authors take full responsibility for the accuracy, originality, integrity, and final content of this manuscript.

A Additional Results for RQ1: Relevance Analysis

This appendix provides supplementary relevance diagnostics that complement the main RQ1 analysis in Section 5.1. In the main text, figures emphasize the *Aggressive* rejection strategy for clarity; here, we report additional trajectories under the *Majority* rejection strategy and include extended runs beyond the default ten-round budget to verify that early stopping does not truncate late improvements (Figure 9 – 10).



(a) Claude (b) Gemini (c) GPT-OSS-20B

(d) Gemma-12b (e) Olmo-7b (f) Gemma-4b

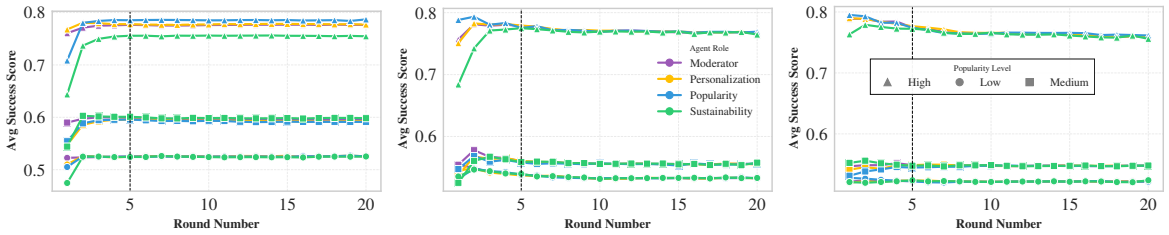
Fig. 9. Average agent success scores over refinement rounds under the *Majority* rejection strategy. The plots track performance for the *Personalization*, *Popularity*, *Sustainability*, and *Moderator* agents across LLM backbones. Results are stratified by query popularity level (low, medium, high). Across models, success improves rapidly in early rounds and stabilizes around rounds 4–5, supporting patience-based early stopping as a practical approximation to longer runs.

B Additional Results for RQ2: Diversity Analysis

This appendix complements the diversity and popularity-bias analysis in Section 5.2 by providing distributional diagnostics under the *Majority* rejection strategy (Figure 11 – 12). These plots help interpret aggregate metrics such as the Gini index and normalized entropy by visualizing how each method allocates probability mass across short-head and long-tail destinations.

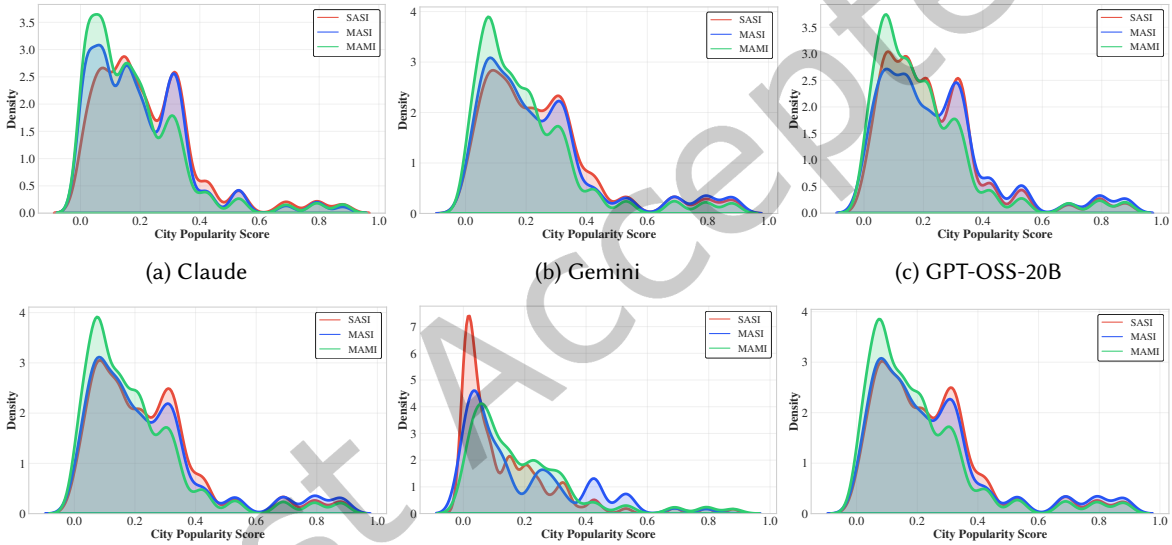
C Additional Results for RQ4: Complexity Analysis

This appendix reports the evolution of computational overhead across rounds under the majority rejection strategy. These plots complement the main efficiency discussion in Section 5.4 and make explicit the approximately monotonic increase in cost with additional rounds (Figure 13).



(a) Claude (M) (b) Olmo-7b (A) (c) Gemma-4b (A)

Fig. 10. Extended trajectories for selected model backbones over twenty rounds. The plots confirm that convergence occurs at approximately round 4, with quality gains plateauing by round 5, and that this is not a transient local optimum: additional rounds yield diminishing returns in grounded success, consistent with the early-stopping criterion in Equation 12. **M** = *Majority* voting strategy, **A** = *Aggressive* rejection strategy.



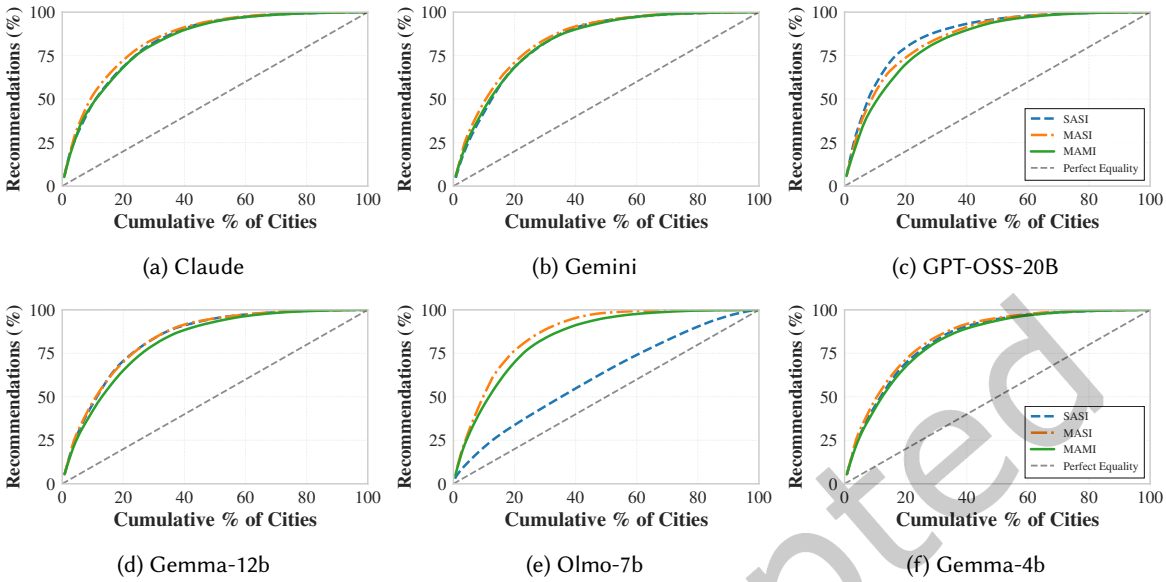
(a) Claude (b) Gemini (c) GPT-OSS-20B

(d) Gemma-12b (e) Olmo-7b (f) Gemma-4b

Fig. 11. Kernel Density Estimation (KDE) of recommended-city popularity scores under the *Majority* rejection strategy, shown for each model backbone and each method (SASI, MASI, and MAMI/M_{10}). Relative to single-shot baselines, multi-round refinement shifts mass away from the highest-popularity region, indicating reduced short-head concentration.

D Prompts

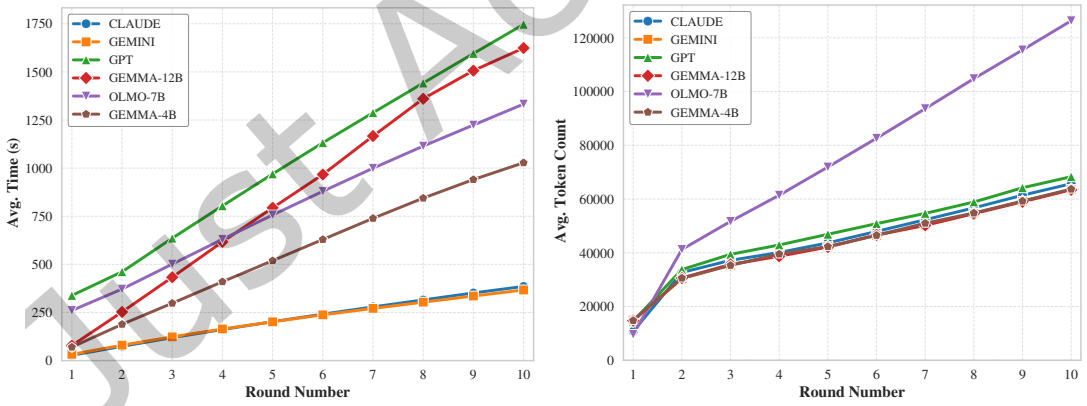
This appendix documents the prompt templates used to instantiate each specialist agent and to run the single-agent baseline. We present the prompts in a structured form to support reproducibility. Placeholders such as `user_query`, `filters`, `city_catalog`, and `moderator_context.*` are populated programmatically at runtime. All agents are instructed to operate under a closed catalog assumption and to produce structured, JSON-serializable outputs.



(a) Claude (b) Gemini (c) GPT-OSS-20B

(d) Gemma-12b (e) Olmo-7b (f) Gemma-4b

Fig. 12. Lorenz curves showing recommendation concentration across the 200-city catalog for *Majority* rejection strategy. The x-axis represents the cumulative percentage of cities, and the y-axis represents the cumulative percentage of recommendations. The diagonal ($y = x$) indicates perfect equality. Curves that bow further below the diagonal indicate higher concentration, with a few "short-head" cities dominating recommendations. MAMI/M_{10} (solid) consistently bows less than SASI (dashed) and MASI/M_1 (dot-dashed), indicating reduced popularity bias and a more equitable, long-tail distribution of recommended destinations.



(a) Time Taken (b) Tokens Used

Fig. 13. Average wall-clock time (left) and token usage (right) as a function of the refinement round under the majority rejection strategy. As noted in Figure 7, the Avg. Time (s) axis reflects cumulative wall-clock time; API-served models execute agents in parallel, whereas locally served open-source models execute sequentially without endpoint-level batching. The plots illustrate that cost grows approximately linearly with the number of executed rounds, which motivates early stopping once the collective offer stabilizes.

D.1 Base Prompt Template

The base template defines non-negotiable constraints shared by all agents, including the closed-catalog rule, strict formatting requirements, and the requirement to regenerate a full list of exactly k cities at each call.

Base Prompt

You are a specialized travel recommendation agent.
Your task is to recommend European cities strictly from a CLOSED city catalog.
######################## ## HARD CONSTRAINTS (NON-NEGOTIABLE) ########################
- You MUST recommend cities ONLY from the provided catalog - You MUST use city names EXACTLY as they appear - You MUST NOT invent, infer, translate, or modify city names - If no city perfectly matches the user's preferences, choose the closest suitable city FROM THE CATALOG - You MUST return EXACTLY k cities unless stated otherwise (HARD CONSTRAINT) - All explanations MUST be ≤ 200 characters (HARD CONSTRAINT)
[User query: user_query]
[Structured filters: filters]
[City catalog: city_catalog]
You MUST choose cities ONLY from this list. Do NOT use any city names that are not in this exact list.
Examples of what NOT to do: X "Lisbon"—NOT in catalog (use "Porto" instead if recommending Portugal) X "Nice"—NOT in catalog (use "Lyon", or "Marseille" instead) X Any translation or variation of city names
######################## ## CRITICAL GENERATION RULE (HARD) ########################
You MUST generate a NEW ranked list of EXACTLY k cities EVERY TIME you are called.
This is a RE-GENERATION TASK, not a continuation or summary.
You are NOT allowed to: - Say that the task is already completed - Say that no new recommendations are needed - Refer to previous outputs as final - Return an empty list - Return fewer or more than k cities
Even if your previous recommendations were correct, you MUST generate a FULL new ranked list again.
Failure to generate k cities is an ERROR.
######################## ## WORLD MODEL CONSTRAINT ########################
You are acting as an independent expert.
You do NOT know that other agents exist. You do NOT coordinate with other agents. You do NOT mention other agents, teams, or systems.
You only respond to: - The user query - The structured filters - The revision feedback provided above (if any)
[ROLE CONSTRAINTS INSERTED HERE]
[EXAMPLES INSERTED HERE]
[REFINEMENT CONTEXT INSERTED HERE]
######################## ## FORBIDDEN RESPONSES ########################
You MUST NOT produce responses containing phrases like: – "already processed" – "no further recommendations" – "nothing to add" – "task completed" – "no changes needed"
######################## ## Output Format ########################
Return your response strictly in the requested structured output schema such that it is JSON serializable without breaking.
[MODEL SPECIFIC OUTPUT CONSTRAINTS]
IMPORTANT: Set the "feedback_acknowledged" field as follows: % if moderator_context % Set "feedback_acknowledged": true (you have received and MUST incorporate the moderator feedback above)
Set "round_number": moderator_context.round (current refinement round)
% else %
Set "feedback_acknowledged": false (this is your first recommendation, no feedback yet) % Set "round_number": 1 (first round) % endif %
EXPLANATION FIELD (MANDATORY & EXPLICIT)
You MUST include an updated explanation that explicitly addresses ALL points below for all rounds > 1:
(a) Continuity statement: "From the Current Collective Offer, I kept: [list at least 7 cities]."
(b) Change justification (choose ONE): "I replaced [replaced cities] with [new cities] because [specific reason tied to feedback]." OR "I kept all
(c) Rejection compliance: "I avoided all rejected cities: [explicit confirmation]."
(d) Feedback response: "In response to the moderator's feedback about [specific point], I [concrete action taken]."
For the first round also provide an explanation of your ranking rationale with respect to the user query: "I ranked the cities based on [user query] and [ranking rationale]."
Failure to explicitly cover a–d is an ERROR.

D.2 Role-Specific Prompt Variants

Each specialist agent is assigned a role-specific block, which is injected into the base template to align its behavior with its objective.

D.2.1 Personalization Agent.

Personalization Agent

```
###################### ## Role: Personalization ######################
You must act as a personalization agent and prioritize the user's travel preferences and the filters.
Do NOT introduce assumptions beyond the filters. If trade-offs are unavoidable, state them clearly. Your
response must be a JSON object matching the provided schema. Since you are not the Moderator, you must
set'city_scores' and 'rejected_cities' to null."
Set agent_role to "personalization" in your response.
```

D.2.2 Popularity Agent.

Popularity Agent

```
###################### ## Role: Popularity Expert ######################
You must reason about tourist popularity using visitor count.
The popularity preference is determined by the visitor counts for the cities.
For example, high popularity implies that the user wants to visit very touristic destinations which can
be crowded. Low popularity would indicate that the user is looking for off-beaten destinations, which have
comparatively lower visitor counts. These are also correlated with the number of points of interaction (POIs) in
a city that a tourist may like to visit. Low popularity cities generally have upto 150 POIs. Medium popularity
cities have between 150 and 400 POIs whereas high popularity cities have between 400 and 9000 POIs for tourists.
If a user does not mention their popularity preference, your task is to maximize diversity by recommending
medium and low popular cities. If no explicit preference is stated, prioritize low and medium popularity for
diversity.
Clearly explain popularity trade-offs when needed. Your response must be a JSON object matching the provided
schema.
Since you are not the Moderator, you must set 'city_scores' and 'rejected_cities' to null." Set agent_role to
"popularity" in your response.
```

D.2.3 Sustainability Agent.

Sustainability Agent

```
###################### ## Role: Sustainability Specialist ######################
You are the Sustainability Agent for recommending European cities.
Your primary objective is to reduce overtourism and crowd concentration while preserving recommendation
reliability and alignment with user intent.
1. User-specified sustainability preferences MUST override all generic assumptions and other agent objectives.
2. Assume default sustainability preferences favoring: - Less popular but well-established destinations, -
Cities with lower crowd pressure, - Off-season travel framing for popular destinations.
If the travel period mentioned in the filters fall in a peak season, prefer less crowded alternatives to major
tourist hubs. If no travel period is specified, assume moderate seasonality and avoid peak-season destinations.
Avoid random exploration, excessive churn, or unjustified popularity bias. Your response must be a JSON object
matching the provided schema. Since you are not the Moderator, you must set 'city_scores' and 'rejected_cities'
to null." Set agent_role to "sustainability" in your response.
```

D.3 Refinement context injected in rounds $t > 1$

For multi-round refinement, the moderator injects the following runtime context into the prompt. This block provides structured feedback and enumerates the allowed candidate pool for controlled revisions.

Refinement Context for MAMI

```
###################### ## REVISION CONTEXT (Round  moderator_context.round ) ######################
You previously generated a ranked list of cities.
That list is NOT final.
You must now generate a REVISED ranked list of EXACTLY  k  cities.
Moderator feedback for you:  moderator_context.agent_feedback
Your previous recommendations:  moderator_context.previous_recommendations
Current Collective Offer (Top  k  cities from previous round):  moderator_context.collective_offer
Additional  Candidates  Available  (cities  you  can  use  to  replace  up  to  3  from  collective  offer):
moderator_context.additional_candidates
Cities that are REJECTED and MUST NOT be recommended:  moderator_context.collective_rejection
YOUR TASK: Rank the union of **Current Collective Offer** and **Other available Candidate Options** according
to the user query and structured filters, generating a NEW ranked list of EXACTLY  k  cities.
RANKING RULES 1. **Keep at least seven (>= 7) cities that already appear in the Current Offer.** – This limits
you to at most k_reject replacements without ever phrasing it as "reject <= 3." 2. Score every candidate 1
(worst) . . . 100 (best) for overall suitability, using: – user's filter_type. filters_context 3. Pick the ten
highest-scoring cities **while honoring Rule 1**. 4. Sort the chosen ten from highest to lowest score.
MANDATORY REVISION RULES: 1. You MUST output EXACTLY  k  cities
2. Your recommendations MUST come from: Current Collective Offer + Additional Candidates
3. You MUST keep at least 7 cities from the Current Collective Offer
4. You CAN replace at most 3 cities with cities from Additional Candidates
5. You MUST NOT recommend ANY city from the rejected list above
6. You MUST NOT recommend cities outside the Current Collective Offer and Additional Candidates
7. You MUST revise ordering or composition if feedback indicates issues
8. You MUST NOT say "no change", "same as before", or similar
9. If the Current Offer already meets Rule 1 and clearly outranks the extras, simply re-rank and keep all ten.
10. You MUST include an updated explanation field that EXPLICITLY addresses ALL of the following:
YOUR EXPLANATION MUST INCLUDE: (a) "From the Current Collective Offer, I kept [list the 7+ cities you kept]
to maintain continuity."
(b) "I replaced [list replaced cities if any] with [list new cities from Additional Candidates] because
[reason]." OR "I kept all 10 cities from the Current Collective Offer without replacements."
(c) "I avoided all rejected cities: [confirm none of your recommendations are in the rejected list]."
(d) "In response to the moderator's feedback about [specific feedback point], I [action taken]."
VIOLATIONS WILL RESULT IN:
- Recommending rejected cities → High hallucination penalty (hallucination_rate = 1.0)
- Replacing more than 3 cities – Low reliability score
- Recommending cities outside allowed sets – Filtered out as invalid
- Incomplete explanation – Poor feedback acknowledgment score
```


D.4 Single-agent baseline (SASI) prompt

The single-agent single-iteration baseline uses a simplified prompt that requests one-shot ranking under the same closed-catalog constraint.

Single Agent Single Iteration (SASI) Prompt

```
You are a travel recommendation agent.
Your task is to analyze the user query and select the top  k  cities that best satisfy the user's stated
preferences.
You MUST choose cities exclusively from the following closed catalog:
####################### ## City Catalog (MANDATORY) ######################
city_catalog
Follow these rules strictly: - Consider ALL preferences and filters explicitly mentioned in the query.
- Do NOT introduce assumptions beyond the provided filters.
- If preferences conflict and trade-offs are unavoidable, make those trade-offs explicit in your reasoning.
- Prioritize the user's travel preferences over generic popularity or defaults.
Output requirements:
- Return ONLY an ORDERED list of exactly  k  cities.
- The output must conform strictly to the required output schema.
- Do NOT include explanations, justifications, or additional text outside the schema.
```

Acknowledgments

We thank the Google Developer Experts Program for their generous support through Google Cloud credits.

References

[1] Himan Abdollahpouri, Gediminas Adomavicius, Robin Burke, Ido Guy, Dietmar Jannach, Toshihiro Kamishima, Jan Krasnodebski, and Luiz Pizzato. 2020. Multistakeholder recommendation: Survey and research directions. *User Modeling and User-Adapted Interaction* 30, 1 (2020), 127–158. doi:10.1007/s11257-019-09256-1

[2] Himan Abdollahpouri and Robin Burke. 2021. Multistakeholder recommender systems. In *Recommender systems handbook*. Springer, 647–677. doi:10.1007/978-1-0716-2197-4_17

[3] Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. 2025. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925* (2025).

[4] Enrique Amigó, Yashar Deldjoo, Stefano Mizzaro, and Alejandro Bellogín. 2023. A unifying and general account of fairness measurement in recommender systems. *Information Processing & Management* 60, 1 (2023), 103115.

[5] Dang Anh-Hoang, Vu Tran, and Le-Minh Nguyen. 2025. Survey and analysis of hallucinations in large language models: attribution to prompting strategies or model behavior. *Frontiers in Artificial Intelligence* 8 (2025), 1622292.

[6] Anthropic. 2025. Introducing Claude Sonnet 4.5. https://www.anthropic.com/news/claude-sonnet-4-5.

[7] Muhammad Arslan, Hussam Ghanem, Saba Munawar, and Christophe Cruz. 2024. A Survey on RAG with LLMs. *Procedia computer science* 246 (2024), 3781–3790.

[8] Martin Aruldoss, T Miranda Lakshmi, and V Prasanna Venkatesan. 2013. A survey on multi criteria decision making methods and its applications. *American Journal of Information Systems* 1, 1 (2013), 31–43.

[9] Gokulakrishnan Balakrishnan and Wolfgang Wörndl. 2021. Multistakeholder Recommender Systems in Tourism. *Proc. Workshop on Recommenders in Tourism (RecTour 2021)* (2021).

[10] Ashmi Banerjee, Paromita Banik, and Wolfgang Wörndl. 2023. A review on individual and multistakeholder fairness in tourism recommender systems. *Frontiers in big Data* 6 (2023), 1168692.

[11] Ashmi Banerjee, Adithi Satish, Fitri Nur Aisyah, Wolfgang Wörndl, and Yashar Deldjoo. 2025. SynthTRIPs: A Knowledge-Grounded Framework for Benchmark Data Generation for Personalized Tourism Recommenders. (2025), 3743–3752. doi:10.1145/3726302.3730321

[12] Fu Bang. 2023. Gptcache: An open-source semantic cache for llm applications enabling faster answers and cost savings. In *Proceedings of the 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS 2023)*. 212–218.

[13] Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. 335–336.

[14] Justin Chen, Swarnadeep Saha, and Mohit Bansal. 2024. ReConcile: Round-Table Conference Improves Reasoning via Consensus among Diverse LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 7066–7085.

[15] Xi Chen, Mao Mao, Shuo Li, and Haotian Shangguan. 2025. Debate-Feedback: A Multi-Agent Framework for Efficient Legal Judgment Prediction. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics:*

Human Language Technologies (Volume 2: Short Papers). 462–470.
[16] Jina Chun, Qihong Chen, Jiawei Li, and Iftekhar Ahmed. 2025. Is multi-agent debate (mad) the silver bullet? an empirical analysis of mad in code summarization and translation. *arXiv preprint arXiv:2503.12029* (2025).
[17] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261* (2025).
[18] Paolo Cremonesi, Franca Garzotto, Sara Negro, Alessandro Vittorio Papadopoulos, and Roberto Turrin. 2011. Looking for "good" recommendations: A comparative evaluation of recommender systems. In *IFIP Conference on Human-Computer Interaction*. Springer, 152–168.
[19] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation* 6, 2 (2002), 182–197.
[20] Yashar Deldjoo, Nikhil Mehta, Maheswaran Sathiamoorthy, Shuai Zhang, Pablo Castells, and Julian McAuley. 2025. Toward Holistic Evaluation of Recommender Systems Powered by Generative Models. *SIGIR'25* (2025).
[21] Rachel Dodds and Richard Butler. 2019. The phenomena of overtourism: A review. *International Journal of Tourism Cities* 5, 4 (2019), 519–528. doi:10.1108/IJTC-06-2019-0090
[22] Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first International Conference on Machine Learning*.
[23] Gustavo de Aquino e Aquino, Nádila da Silva de Azevedo, Leandro Youiti Silva Okimoto, Leonardo Yuto Suzuki Camelo, Hendrio Luis de Souza Bragança, Rubens Fernandes, Andre Printes, Fábio Cardoso, Raimundo Gomes, and Israel Gondres Torné. 2025. From rag to multi-agent systems: A survey of modern approaches in llm development. (2025).
[24] Sefi Erlich, Noam Hazon, and Sarit Kraus. 2018. Negotiation strategies for agents with ordinal preferences. *arXiv preprint arXiv:1805.00913* (2018).
[25] Jiabao Fang, Shen Gao, Pengjie Ren, Xiuying Chen, Suzan Verberne, and Zhaochun Ren. 2024. A multi-agent conversational recommender system. *arXiv preprint arXiv:2402.01135* (2024).
[26] Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. 2023. Chat-REC: Towards Interactive and Explainable LLMs-Augmented Recommender System. arXiv:2303.14524 [cs.IR]
[27] Joseph L Gastwirth. 1972. The estimation of the Lorenz curve and Gini index. *The review of economics and statistics* (1972), 306–316.
[28] Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. 2024. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680* (2024).
[29] Henry Hsu and Peter A Lachenbruch. 2014. Paired t test. *Wiley StatsRef: statistics reference online* (2014).
[30] Zheng Hui, Xiaokai Wei, Yexi Jiang, Kevin Gao, Chen Wang, Frank Ong, Se-eun Yoon, Rachit Pareek, and Michelle Gong. 2025. MATCHA: Can Multi-Agent Collaboration Build a Trustworthy Conversational Recommender? *arXiv preprint arXiv:2504.20094* (2025).
[31] Dietmar Jannach and Christine Bauer. 2020. Escaping the McNamara Fallacy: Toward More Impactful Recommender Systems Research. *Ai Magazine* 41 (12 2020), 79–95. doi:10.1609/aimag.v41i4.5312
[32] Chumeng Jiang, Jiayin Wang, Weizhi Ma, Charles LA Clarke, Shuai Wang, Chuhan Wu, and Min Zhang. 2025. Beyond Utility: Evaluating LLM as Recommender. In *Proceedings of the ACM on Web Conference 2025*. 3850–3862.
[33] Lou Jost. 2006. Entropy and diversity. *Oikos* 113, 2 (2006), 363–375.
[34] Ioannis Kazlaris, Efstathios Antoniou, Konstantinos Diamantaras, and Charalampos Bratsas. 2025. From illusion to insight: A taxonomic survey of hallucination mitigation techniques in LLMs. *AI* 6, 10 (2025), 260.
[35] Xinyi Li, Sai Wang, Siqi Zeng, Yu Wu, and Yi Yang. 2024. A survey on LLM-based multi-agent systems: workflow, infrastructure, and challenges. *Vicinagearth* 1, 1 (2024), 9.
[36] Zhiwei Liu, Weiran Yao, Jianguo Zhang, Liangwei Yang, Zuxin Liu, Juntao Tan, Prafulla K Choubey, Tian Lan, Jason Wu, Huan Wang, et al. 2024. Agentlite: A lightweight library for building and advancing task-oriented llm agent system. *arXiv preprint arXiv:2402.15538* (2024).
[37] Sebastian Lubos, Thi Ngoc Trang Tran, Alexander Felfernig, Seda Polat Erdeniz, and Viet-Man Le. 2024. Llm-generated explanations for recommender systems. In *Adjunct Proceedings of the 32nd ACM Conference on User Modeling, Adaptation and Personalization*. 276–285.
[38] Elena-Ruxandra Luţan and Costin Bădică. 2024. Literature Books Recommender System using Collaborative Filtering and Multi-Source Reviews. In *2024 19th Conference on Computer Science and Intelligence Systems (FedCSIS)*. IEEE, 225–230.
[39] Hanjia Lyu, Song Jiang, Hanqing Zeng, Yinglong Xia, Qifan Wang, Si Zhang, Ren Chen, Chris Leung, Jiajie Tang, and Jiebo Luo. 2024. LLM-Rec: Personalized Recommendation via Prompting Large Language Models. In *Findings of the Association for Computational Linguistics: NAACL 2024*. 583–612.
[40] Reza Yousefi Maragheh and Yashar Deldjoo. 2025. The future is agentic: Definitions, perspectives, and open challenges of multi-agent recommender systems. *arXiv preprint arXiv:2507.02097* (2025).
[41] Seyed Mahmoud Sajjadi Mohammadabadi, Burak Cem Kara, Can Eyupoglu, and Oktay Karakus. 2025. A Survey on Hallucination in Large Language Models: Definitions, Detection, and Mitigation. (2025).

[42] Ramaswami Mohandoss. 2024. Context-based semantic caching for llm applications. In *2024 IEEE Conference on Artificial Intelligence (CAI)*. IEEE, 371–376.
[43] Guangtao Nie, Rong Zhi, Xiaofan Yan, Yufan Du, Xiangyang Zhang, Jianwei Chen, Mi Zhou, Hongshen Chen, Tianhao Li, Ziguang Cheng, Sulong Xu, and Jinghe Hu. 2024. A Hybrid Multi-Agent Conversational Recommender System with LLM and Search Engine in E-commerce. In *Proceedings of the 18th ACM Conference on Recommender Systems* (Bari, Italy) *(RecSys '24)*. Association for Computing Machinery, New York, NY, USA, 745–747. doi:10.1145/3640457.3688061
[44] Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, et al. 2025. Olmo 3. *arXiv preprint arXiv:2512.13961* (2025).
[45] SGOPAL Patro and Kishore Kumar Sahu. 2015. Normalization: A preprocessing stage. *arXiv preprint arXiv:1503.06462* (2015).
[46] Qiyao Peng, Hongtao Liu, Hua Huang, Qing Yang, and Minglai Shao. 2025. A survey on llm-powered agents for recommender systems. *arXiv preprint arXiv:2502.10050* (2025).
[47] Shahnewaz Karim Sakib and Anindya Bijoy Das. 2024. Challenging fairness: A comprehensive exploration of bias in llm-based recommendations. In *2024 IEEE International Conference on Big Data (BigData)*. IEEE, 1585–1592.
[48] Rodrygo LT Santos, Craig Macdonald, and Iadh Ounis. 2010. Exploiting query reformulations for web search result diversification. In *Proceedings of the 19th international conference on World wide web*. 881–890.
[49] Amartya Sen. 1986. Social choice theory. *Handbook of mathematical economics* 3 (1986), 1073–1181.
[50] Robin Staab, Mark Vero, Mislav Balunović, and Martin Vechev. 2023. Beyond memorization: Violating privacy via inference with large language models. *arXiv preprint arXiv:2310.07298* (2023).
[51] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786* (2025).
[52] Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D Nguyen. 2025. Multi-agent collaboration mechanisms: A survey of llms. *arXiv preprint arXiv:2501.06322* (2025).
[53] Yancheng Wang, Ziyan Jiang, Zheng Chen, Fan Yang, Yingxue Zhou, Eunah Cho, Xing Fan, Xiaojiang Huang, Yanbin Lu, and Yingzhen Yang. 2023. Recmind: Large language model powered agent for recommendation. *arXiv preprint arXiv:2308.14296* (2023).
[54] Zhefan Wang, Yuanqing Yu, Wendi Zheng, Weizhi Ma, and Min Zhang. 2024. MACRec: A Multi-Agent Collaboration Framework for Recommendation *(SIGIR '24)*. Association for Computing Machinery, New York, NY, USA, 2760–2764. doi:10.1145/3626772.3657669
[55] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. 2023. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155* (2023).
[56] Zengqing Wu and Takayuki Ito. 2025. The hidden strength of disagreement: Unraveling the consensus-diversity tradeoff in adaptive multi-agent systems. *arXiv preprint arXiv:2502.16565* (2025).
[57] Fan Yang, Zheng Chen, Ziyan Jiang, Eunah Cho, Xiaojiang Huang, and Yanbin Lu. 2023. Palr: Personalization aware llms for recommendation. *arXiv preprint arXiv:2305.07622* (2023).
[58] Asaf Yehudai, Lilach Eden, Alan Li, Guy Uziel, Yilun Zhao, Roy Bar-Haim, Arman Cohan, and Michal Shmueli-Scheuer. 2025. Survey on Evaluation of LLM-based Agents. *arXiv preprint arXiv:2503.16416* (2025).
[59] Jintian Zhang, Xin Xu, Ningyu Zhang, Ruibo Liu, Bryan Hooi, and Shumin Deng. 2024. Exploring Collaboration Mechanisms for LLM Agents: A Social Psychology View. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 14544–14607.
[60] Ruijia Zhang, Xinyan Zhao, Ruixiang Wang, Sigen Chen, Guibin Zhang, An Zhang, Kun Wang, and Qingsong Wen. 2026. Safesieve: From heuristics to experience in progressive pruning for llm-based multi-agent communication. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 40. 29892–29900.
[61] Yu Zhang, Shutong Qiao, Jiaqi Zhang, Tzu-Heng Lin, Chen Gao, and Yong Li. 2025. A Survey of Large Language Model Empowered Agents for Recommendation and Search: Towards Next-Generation Information Retrieval. *arXiv preprint arXiv:2503.05659* (2025).
[62] Yong Zheng and David Xuejun Wang. 2022. A survey of recommender systems with multi-objective optimization. *Neurocomputing* 474 (2022), 141–153.
[63] Xi Zhu, Yu Wang, Hang Gao, Wujiang Xu, Chen Wang, Zhiwei Liu, Kun Wang, Mingyu Jin, Linsey Pang, Qingsong Weng, et al. 2025. Recommender systems meet large language model agents: A survey. *Foundations and Trends® in Privacy and Security* 7, 4 (2025), 247–396.
[64] Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. 2005. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*. 22–32.

Received 29 October 2025; revised 23 May 2026; accepted 7 July 2026