SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Data Engineering and Analytics

Towards Sustainable and Diverse City Trips: Leveraging Large Language Models to Address Data Scarcity and Popularity Bias in Tourism Recommender Systems

Adithi Satish

SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Data Engineering and Analytics

Towards Sustainable and Diverse City Trips: Leveraging Large Language Models to Address Data Scarcity and Popularity Bias in Tourism Recommender Systems

Für nachhaltige und vielfältige Städtereisen: Nutzung großer Sprachmodelle zur Behebung von Datenknappheit und Popularitätsverzerrungen in touristischen Empfehlungssystemen

Author:	Adithi Satish
Supervisor:	Dr. -Ing. Jörg Ott, Dr. Wolfgang Wörndl
Advisor:	Ashmi Banerjee
Submission Date:	05 June 2025

I confirm that this master's thesis in data engineering and analytics is my own work and I have documented all sources and material used.

Munich, 05 June 2025 Adithi Satish

Acknowledgments

First and foremost, I would like to thank my advisor, Ashmi Banerjee, for all the invaluable support, guidance and encouragement throughout the research process. It was truly an incredible experience working with her in the Chair of Connected Mobility under Dr. Jörg Ott, further enriched with crucial inputs and encouragement from Dr. Wolfgang Wörndl and Dr. Yashar Deldjoo during this thesis. Additionally, this thesis and its composite publications would not have been possible without Fitri Nur Aisyah, who has been a fantastic teammate through the brainstorming and subsequent implementation sessions, making it an absolute pleasure to delve into the interdisciplinary world of natural language processing and tourism recommender systems.

I would further like to thank my parents and grandmother, whose encouragement and confidence in my abilities have been a steadfast motivator throughout my degree. Thank you for always helping me see the light at the end of the tunnel. Finally, I would be remiss not to thank all my wonderful friends, who have been a constant source of joy this past year, reminding me that it is alright to take a breather every once in a while. It is easy to get caught up in the throes of a thesis, leaving few opportunities for anything else, so thank you for our travels, stupidity, laughter and all the memories I will always cherish.

Publications Included in this Thesis

1. SynthTRIPS: A Knowledge-Grounded Framework for Benchmark Query Generation for Personalized Tourism Recommenders, Ashmi Banerjee, *Adithi Satish*, Fitri Nur Aisyah, Wolfgang Wörndl, Yashar Deldjoo, The 48th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2025 [1]

2. Collab-REC: An LLM-based Agentic Framework for Balancing Recommendations in Tourism, Ashmi Banerjee, Fitri Nur Aisyah, *Adithi Satish*, Wolfgang Wörndl, Yashar Deldjoo, *under review as of 05 June 2025*

Abstract

Tourism Recommender Systems (TRS) play a vital role in personalizing travel experiences by tailoring suggestions to users' preferences, constraints, and contextual factors. Conversational Recommender Systems (CRS), in particular, leverage natural language to deliver more intuitive and adaptive recommendations. However, two persistent challenges have hindered progress in this area: (i) the lack of publicly available datasets and benchmarks that capture realistic user travel requests, and (ii) the prevalence of popularity bias, which leads to over-recommendation of well-known destinations at the expense of emerging or underexplored locales, aggravating overtourism.

This thesis explores the potential of Large Language Models (LLMs) to address both challenges. First, we introduce SynthTRIPS, a framework for generating synthetic travel queries that reflect diverse user personas and include structured filters such as budget and sustainability preferences. By grounding LLM outputs in a curated Knowledge Base (KB), we ensure factual consistency and reduce hallucinations. The query generation process is formalized with evaluation metrics to assess both groundedness and overall fit. Empirical results, supported by both human and automatic evaluations, confirm that SynthTRIPS produces high-quality queries that reflect multi-faceted and nuanced travel preferences.

To further tackle popularity bias in recommendation, we propose Collab-Rec, a multi-agent framework comprising three specialized LLM-based agents—**Personalization**, **Popularity**, and **Sustainability**. Each agent, instructed to adhere to its own objective, provides city recommendations from its distinct perspective, while a non-LLM moderator fuses their outputs through a heuristic-based scoring procedure, facilitating multi-round negotiation, and promoting diverse, relevant, and well-balanced recommendations. Experiments using SynthTRIPS queries demonstrate that Collab-Rec outperforms single-agent baselines by surfacing lesser-known destinations and aligning more closely with user constraints.

Together, these contributions highlight the potential of combining synthetic data generation and multi-agent collaboration to build more equitable, sustainable, and personalized tourism recommender systems.

Kurzfassung

Empfehlungssysteme für den Tourismus (TRS) spielen eine wichtige Rolle bei der Personalisierung von Reiseerlebnissen, indem sie die Vorschläge auf die Vorlieben, Einschränkungen und Kontextfaktoren der Nutzer abstimmen. Insbesondere Conversational Recommender Systems (CRS) nutzen die natürliche Sprache, um intuitivere und anpassungsfähigere Empfehlungen zu geben. Zwei anhaltende Probleme haben jedoch den Fortschritt in diesem Bereich behindert: (i) der Mangel an öffentlich zugänglichen Datensätzen und Benchmarks, die realistische Reisewünsche der Nutzer erfassen, und (ii) die weit verbreitete Beliebtheit, die dazu führt, dass bekannte Reiseziele auf Kosten neuer oder wenig erforschter Orte übermäßig empfohlen werden, was den Übertourismus verschärft.

In dieser Masterarbeit wird das Potenzial von Large Language Models (LLMs) zur Bewältigung beider Herausforderungen untersucht. Zunächst stellen wir SynthTRIPS vor, ein Framework zur Generierung synthetischer Reiseanfragen, die verschiedene Nutzerpersönlichkeiten widerspiegeln und strukturierte Filter wie Budget und Nachhaltigkeitspräferenzen enthalten. Durch die Verankerung der LLM-Ausgaben in einer kuratierten Wissensdatenbank (KB) stellen wir die faktische Konsistenz sicher und reduzieren Halluzinationen. Der Prozess der Abfragegenerierung wird mit Bewertungsmetriken formalisiert, um sowohl die Fundiertheit als auch die Gesamtübereinstimmung zu bewerten. Empirische Ergebnisse, die sowohl durch menschliche als auch durch automatische Auswertungen gestützt werden, bestätigen, dass SynthTRIPS qualitativ hochwertige Abfragen erzeugt, die vielschichtige und nuancierte Reisepräferenzen widerspiegeln.

Zur weiteren Bekämpfung von Popularitätsverzerrungen bei Empfehlungen schlagen wir Collab-Rec vor, ein Multi-Agenten-System, das drei spezialisierte LLM-basierte Agenten umfasst - **Personalisierung**, **Popularität** und **Nachhaltigkeit**. Jeder Agent, der angewiesen ist, sein eigenes Ziel zu verfolgen, liefert Stadtempfehlungen aus seiner eigenen Perspektive, während ein Nicht-LLM-Moderator die Ergebnisse der Agenten durch ein heuristisches Bewertungsverfahren zusammenführt, was die Verhandlung über mehrere Runden erleichtert und vielfältige, relevante und ausgewogene Empfehlungen fördert. Experimente mit SynthTRIPS-Abfragen zeigen, dass Collab-Rec besser abschneidet als Single-Agent-Baselines, indem es weniger bekannte Reiseziele aufzeigt und sich besser an die Einschränkungen der Nutzer anpasst.

Zusammengenommen zeigen diese Beiträge das Potenzial der Kombination von synthetischer Datengenerierung und Multi-Agenten-Kollaboration auf, um gerechtere, nachhaltigere und personalisierte touristische Empfehlungssysteme zu entwickeln.

Contents

1. Introduction

Tourism is a dynamic global industry that fuels cultural exchange, drives economic development, and offers opportunities for personal growth and discovery. As international travel becomes more accessible, today's travelers are presented with an abundance of choices in terms of destinations, experiences, and itineraries that span every corner of the globe. From popular destinations like *Paris*, *London*, and *Amsterdam* to hidden gems such as *Malatya*, *Sibiu*, or *Thessaloniki*, the sheer variety of options can transform the excitement of planning a trip into a daunting task. Moreover, tourists have a wide range of interests - historical landmarks, immersive cultural experiences, natural landscapes, or local culinary traditions - making it difficult to curate the ideal holiday. Ironically, what should be a relaxing and enjoyable experience can quickly become overwhelming, underscoring the need for tools that help travelers navigate this complex decision-making process.

In this context, Tourism Recommender Systems, or TRS, have emerged as essential tools to aid travellers in navigating this multitude of travel choices. In general, Recommender Systems (RS) are indispensable in helping users sift through massive amounts of information in domains such as e-commerce, media streaming, and travel [2]. Within the travel domain, these systems leverage data-driven algorithms to provide personalized suggestions and can also curate detailed itineraries that align with users' preferences and travel goals.

In essence, city trip recommender systems have traditionally aimed to emulate the role of travel agents by assisting users in the decision-making process, optimizing satisfaction, and ultimately enriching their travel experience. These systems typically rely on analyzing users' historical interactions, such as explicit travel preferences, previously visited destinations, or even implicit factors like geotagged photos and check-in behavior [3, 4, 5, 6, 7], leveraging them to predict and suggest destinations to users.

However, in recent years, the emergence of powerful *Large Language Models (LLMs)* has introduced a paradigm shift in how recommendations can be delivered through Conversational Recommender Systems (CRS). With their ability to understand and generate natural language, LLMs, like ChatGPT, enable RS to move beyond static data and fixed attributes, facilitating more dynamic and conversational interactions. This development is especially impactful in tourism, where users often express complex and evolving preferences. LLM-based recommenders (RecLLMs) can engage with the users in nuanced dialogue, clarify vague preferences and adapt to real-time feedback, providing natural, human-like conversations [8, 9].

For example, asking ChatGPT the following question: *"Do you have some suggestions for cities in Europe where I can go hiking and skiing in the winter for a holiday? I plan to travel in January."* results in the LLM recommending *Innsbruck, Chamonix, Zermatt, Cortina d'Ampezzo, and Garmisch-Partenkirchen*, along with relevant information about each of these places, as shown in Figure 1.1. In this example, the LLM can easily pick up on multiple user preferences ({*month*

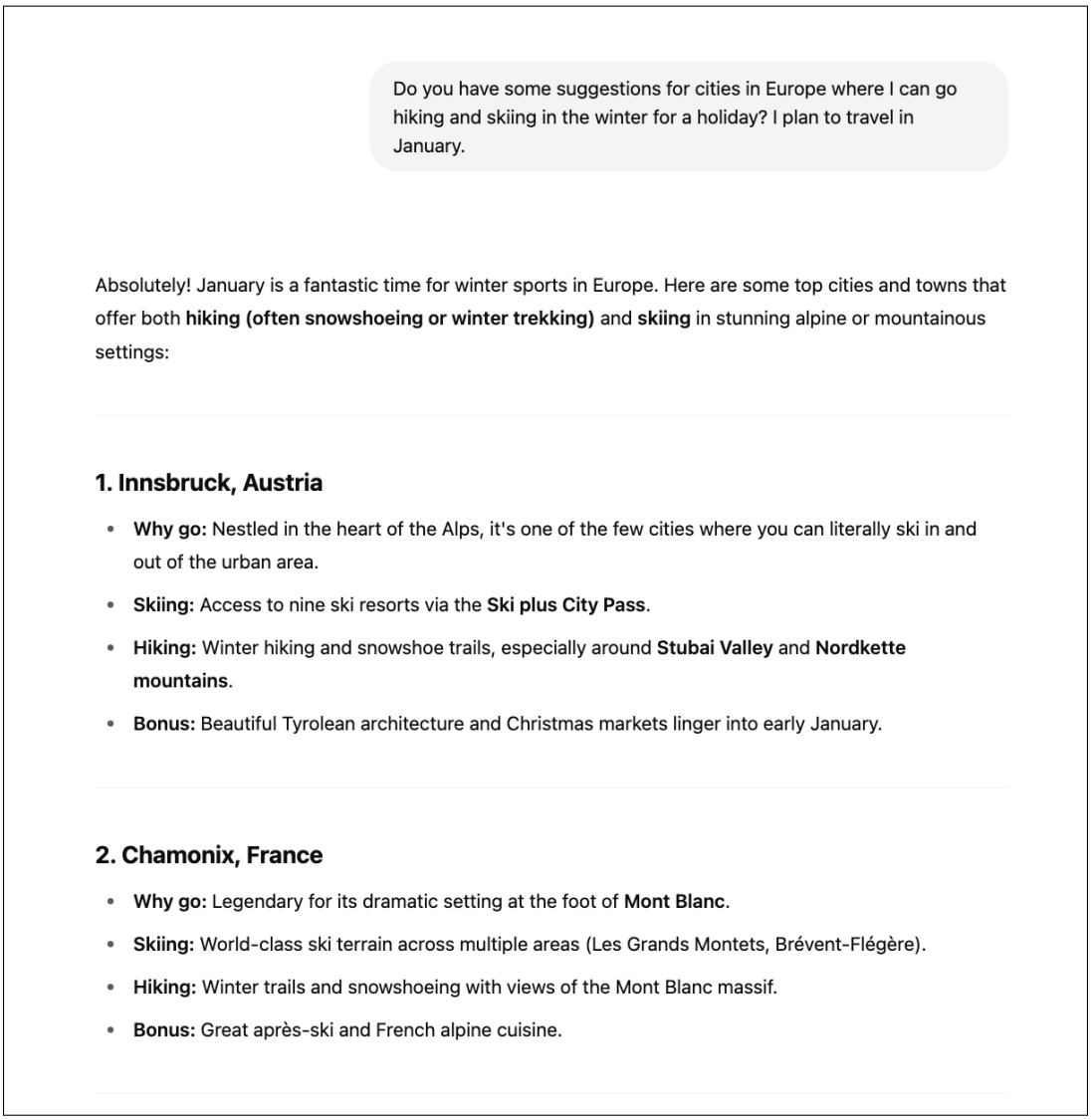


Figure 1.1.: An example of querying ChatGPT to provide travel recommendations. The screenshot only shows the top two options for brevity.

of travel: January; interests: hiking, skiing}) to recommend appropriate cities with justifications, thereby indicating how LLMs can use only their inherent knowledge to understand nuances from natural language and function as recommenders, even without access to any external database.

1.1. Problem Statement & Scope

While TRS offers several advantages in alleviating the stress of planning a holiday, they are not without challenges. Both traditional and LLM-based recommenders fall prey to fairness and bias issues, where popular destinations are more likely to be recommended, reducing the system's diversity and unintentionally ignoring cities that are off the beaten track.

For example, in Figure 1.2, the user asks the LLM to recommend cities that are *"not very popular or crowded"* amongst other preferences. However, the LLM's recommendations are *Ljubljana, Bratislava, Timisoara, Lublin, and Kaunas*. While the latter three recommendations are indeed not popular cities, the LLM's top two recommendations, *Ljubljana* and *Bratislava*, still have high popularity, indicating that these systems can often falter in such situations and further exacerbate this popularity bias.

Although this is a common issue in recommender systems across domains, increasing popularity bias can adversely affect tourism, amplifying **overtourism**, which further leads to environmental degradation and strain on local infrastructure.

In order to combat this, a **multi-stakeholder** perspective to TRS is paramount, which, in addition to user satisfaction, considers item fairness and the societal impact of these recommendations [10]. This is especially important in RecLLMs, which can often overlook the other stakeholders to maximize personalization.

For the scope of this thesis, we define popularity in terms of the number of Points of Interest (POIs) a city contains, which has been shown to correlate with the number of visitors [11]. Low popularity cities generally contain upto 150 POIs whereas Medium popularity cities have between 150 and 400 POIs. High popularity cities like *London* and *Paris* contain 400-9000 POIs.

Research Gap

Synthetic Data Generation: The precursor to designing any recommender system is the availability of a dataset that can be used during a potential training phase and, subsequently, for validation and evaluation. In CRS for tourism, however, the availability of datasets, particularly for city trip recommendations, is sparse. Synthetic benchmarks like TravelPlanner [12] and TravelDest [13] either have different objectives like curating entire itineraries, or deal with broad and ambiguous queries. Furthermore, they do not account for factors like sustainability and consist of predominantly popular cities, which can exacerbate the aforementioned bias in TRS trained on this data.

Therefore, the first part of this thesis focuses on designing and evaluating a synthetic data generation framework, which includes multi-faceted aspects of travel, from user pref-

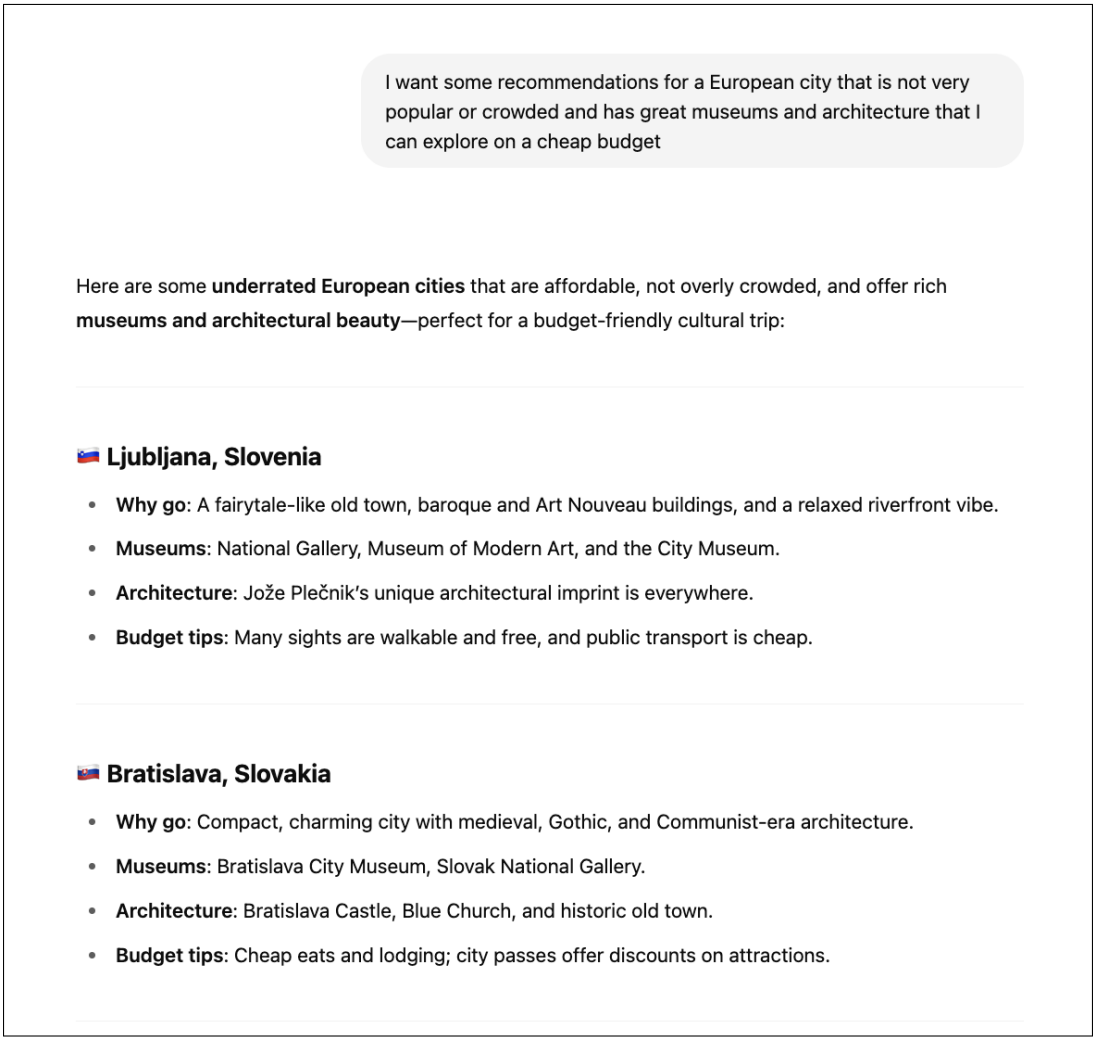


Figure 1.2.: An example of querying ChatGPT to provide travel recommendations. Although the user specifically asks for less-crowded destinations, the top two choices are *Ljubljana* and *Bratislava,* both of which have high popularity. The screenshot only shows the top two options for brevity.

erences to sustainability and popularity factors of destinations. The contributions include a comprehensive *knowledge base*, consisting of travel-related information about 200 European cities, in addition to the framework and dataset. The framework, SynthTRIPS, uses LLMs as user simulators, evaluating the resulting queries performed on multiple dimensions and measuring their groundedness, diversity, personalization, clarity, and overall fit.

Multi-Agent Recommender System: Figure 1.2 provides a glimpse of how a RecLLM, when provided with different user preferences (or *aspects*), can potentially compromise on less-straightforward aspects like sustainability and popularity to instead hyper-focus on aspects that are easier to dissect, like budget, month and travel interests.

An emerging research direction involves multi-agent systems [14], where multiple LLMs, each with a distinct objective, collaborate to fulfill user requests. These systems aim to achieve a common goal by leveraging the strengths of different LLMs. In recommendation, these agents might specialize in web search, user profiling, item analysis, or explanation generation tasks.

Inspired by the multi-stakeholder view, the second part of this thesis presents Collab-Rec, a multi-agent system designed to balance personalization and item fairness. Each agent represents a stakeholder with a specific goal - for instance, a *Popularity Agent* focuses on popularity preferences. In contrast, a *Personalization Agent* handles aspects like budget or travel interests. These agents negotiate via a moderator, which fuses their outputs into a revised recommendation list using heuristics designed to evaluate the relevance and reliability of the agents. Using SynthTRIPS, we evaluate Collab-Rec to explore whether a multi-agent setup can enhance diversity while preserving user relevance.

Contributions

Broadly, the main contributions of this thesis are: (i) SynthTRIPS: LLM-based user simulation to generate synthetic travel queries, and (ii) Collab-Rec: balancing personalization and popularity bias in travel recommendations using a multi-agent system. Further details about the motivation and specific contributions can be found in their respective chapters.

1.2. Structure of the Thesis

The thesis is structured as follows: in Chapter 2 we explore the previous research in TRS, specifically targeted towards using LLMs to alleviate the challenges within this domain. Chapter 3 further introduces SynthTRIPS, the eponymous dataset and query generation framework, subsequently used by Collab-Rec in Chapter 4 to design and evaluate a multi-agent recommender system with the aim of balancing diversity and relevance in TRS. Finally, Chapter 5 provides a concise summary of the results and discussion from the previous chapters and an outlook into the limitations and future scope of how LLMs can improve the quality of, and encourage sustainable recommendations in CRS for tourism.

2. Related Work

In this chapter, we discuss the recent advancement as well as current state-of-the-art in Conversational Recommender Systems (CRS) using LLMs, structured as follows: Section 2.1 contains an overview of the different architectures, SOTA and problems in Conversational Recommender Systems, followed by Section 2.2, which provides the list of existing datasets for offline evaluation of Conversational Recommender Systems as well as illustrating the need to create a personalized dataset for the tourism domain. Section 2.3, Section 2.4, Section 2.5 and Section 2.6 discuss different phases in the recommender system pipeline, and the potential bottlenecks therein, which the introduction of LLMs could assuage, such as Preference Elicitation, Integration of Sustainability, Ensemble LLMs, and Multi-Agent Systems.

2.1. Conversational Recommender Systems

Conversational Recommender Systems (CRS) are systems that integrate recommender techniques like collaborative/content-based filtering with dialog systems, in order to increase personalization through natural language in a session-based format [15]. CRS typically comprise of an Natural Language Understanding (NLU) module which interacts with the user to understand their interests and preferences, a State Manager, which decides what action to take at any point during the session, the Recommendation module with its relevant paradigms, and, the Natural Language Generation (NLG) module, responsible for generating the final response to the user in a clear and conscise manner.

An earlier survey conducted by Jannach, Manzoor, Cai, and L. Chen [16] examines the research in CRS, analyzing each module, their interaction protocols, datasets, module-specific objectives, and their evaluation. The challenges identified by the authors, as well as in successive research [17, 18, 19], include CRS as a group-decision-maker, generating convincing explanations of the recommendation, and whether traditional metrics of evaluation suffice in quantifying success in CRS.

However, over the past few years, with the rapid developments in the NLU & NLG abilities of language models, or rather *large language models*, research in LLM-based recommenders (or RecLLMs) has exploded with LLMs helping to alleviate some of the challenges mentioned above [20, 21]. Friedman, Ahuja, Allen, et al. [22] study the development of RecLLMs, which include using LLMs to extract preferences, generate explanations, and introduce implementations to enhance personalization by adapting natural language user profiles. Furthermore, incorporating external knowledge sources can help nudge the LLMs towards the best recommendations, instead of relying on their own internal knowledge [23].

However, these systems are not without their challenges - LLMs can often contain biases

Table 2.1.: An overview of the existing datasets and benchmarks as well as SynthTRIPS.

Dataset	Domain	Goal/Task	Description	Size (#queries)	LLMs/Model Used	Data Sources
reddit-travel-QA-finetuning [26]	Travel	Question Answering + Finetuning	Real-time travel discussions (posts and top comments) on top 100 travel-related subreddits	10,000	N/A	Reddit
TravelPlanner [12]	Travel	Travel Plan Generation	Synthetically generated queries and travel itineraries using LLMs, including constraints such as cuisine, accommodation, and transport	1,225	GPT-4	Scraped from the Internet
TravelDest [13]	Travel	City Trip Recommendations	Broad and indirect queries spanning over 700 cities	50	Manually Generated	Wikivoyage
FeB4RAG [27]	General	Federated Search	Information requests collected from the BEIR subcollections across multiple domains	790	GPT-4	BEIR Datasets [28]
Recipe-MPR [29]	Food	Multi-Aspect Recommendation	Manually annotated preference-based queries that contain multi-item descriptions to benchmark conversational recommender systems	500	Manually Generated	FoodKG [30], Recipe1M+ [31]
SynthTRIPS	**Travel**	**City Trip Recommendations**	**Personalized travel queries generated synthetically covering travel filters such as budget, interests integrated with additional sustainability features**	**4,604**	**Llama-3.2-90B, Gemini-1.5-Pro**	**Wikivoyage, Nomadlist, Where And When, Tripadvisor**

learned from their training data, which can influence their responses, leading to bias and unfairness [24]. Furthermore, they have a tendency to hallucinate and produce non-factual information [25]. Identifying these issues is particularly hard, given their complex algorithms.

The following subsections provide a detailed outlook into the different phases of designing recommender systems and how LLMs can be used to address long-standing challenges in these areas.

2.2. Synthetic Data Generation

Early travel recommenders primarily relied on historical booking logs and user reviews [5, 4, 32]. While these sources offer insights into popular destinations, they rarely include explicit data for *sustainability preferences* or off-peak travel. Subsequent works have integrated *knowledge graphs* to enrich semantic understanding of destination attributes [33], yet they often lack *diverse user persona data*. A popular and effective way of considering the nuances of user preferences is through Conversational Recommender Systems, which provide more

interactive recommendations, as compared to traditional recommenders [34]. Table 2.1 provides an overview of some of the datasets in this field, across domains like travel [26, 12, 13] and food [29]. However, the dearth of data in order to evaluate this systems is a persistent problem, especially with the time-consuming and resource intensive nature of manual data collection and annotation.

Meanwhile, with the recent advancements in Large Language Models (LLMs) and their ability to generate human-like conversations, there is a growing field of study that explores their potential in simulating users and their behavior [35, 36, 37, 38, 39, 40, 41]. LLM-based synthetic data generation has gained traction in domains such as healthcare [42] and education [43], demonstrating the potential to produce large-scale, high-quality text data automatically.

Nonetheless, existing approaches either (1) do not specifically tailor synthetic data to *travel* use cases, like FeB4RAG [27], or (2) do not incorporate robust *sustainability and personalization* constraints. Furthermore, naive LLM-based query generation may lead to factual errors unless carefully grounded.

Some studies have used LLMs to generate data for TRS, but they are often broad queries that do not necessarily involve a user asking for city trip recommendations. For example, the Reddit Q&A dataset introduced by Meyer, Singh, Tam, et al. [26] consists of 10,000 posts and comments taken from various travel-related subreddits, covering a broad spectrum of travel-related queries, including but not limited to visas, required vaccines, and itinerary suggestions. TravelDest, a benchmark introduced [13], consists of 50 broad and indirect textual queries, which are not sufficient to train algorithms. In tourism, Xie, K. Zhang, J. Chen, et al. [12] uses LLMs to generate synthetic travel queries for origins and destinations in the United States to annotate and benchmark travel itineraries. TravelPlanner only focuses on testing the planning capabilities, so it doesn't consider the persona aspect of the query. All the queries sound similar and are in the same template, so we can't test preference extraction capabilities on this dataset.

2.3. Preference Elicitation

In TRS, there are often multiple aspects present in the user's query, like when they intend to travel, the purpose of the trip, and preferences about what to do during the trip, to name a few. Korikov, Saad, Baron, et al. [44] formulate the Multi-Aspect Reviewed-Item Retrieval (MA-RIR) problem as an extension of the RIR problem and the subsequent Late Fusion (LF) technique proposed by Abdollah Pour, Farinneya, Toroghi, et al. [45]. They further propose Aspect Fusion (AF), a method that is able to identify different aspects present in a user's query, perform separate retrievals, and subsequently fuse the results using LF to obtain the final list of recommended items. Other studies have also focused on developing clarifying question pipelines which enable the CRS with the means to find out more information about the user's preferences through a dialogue [46].

2.3.1. Large Language Models for Preference Elicitation

Wen, Y. Liu, J. Zhang, et al. [13] use LLMs in order to perform Query Reformulation (QR) on broad and indirect travel queries to clarify user intents. The authors introduce a novel prompting technique, which identifies keyphrases, or subtopics, from the user's broad query and provides elaborations using city examples.

In contrast, Kemper, Cui, Dicarlantonio, et al. [47] use few-shot learning to enhance LLMs to extract user preferences as hard and soft constraints. This enables state updates, allowing the system to keep track of the user's old preferences and account for any changes. In addition to the preferences, the authors also instruct the LLM to keep track of the rejected items, increasing personalization.

While this thesis assumes a predefined set of travel preferences - budget, travel month, and interests - are already available and assigned to each LLM agent alongside the user's query, this subsection lays the groundwork for incorporating a future prerequisite step of preference elicitation directly from the user input.

2.4. Sustainable TRS

Recent work highlighted by Banerjee, Banik, and Wörndl [10] points to the rise of a promising new research direction that recognizes society or the environment as key stakeholders in the recommendation process. For instance, Merinov [48] integrates sustainability into a multi-stakeholder recommender system by formulating an objective to redistribute tourist flows and mitigate overcrowding. Other sustainability goals explored in earlier studies include economic sustainability[49], food security [50], air pollution reduction [51], promotion of eco-friendly accommodations [52].

Furthermore, Banerjee, Mahmudov, Adler, et al. [11] develop a scoring mechanism to weight cities based on multiple sustainability factors, including carbon emissions, popularity, and seasonality. This S-Fairness score is utilized in Sustainability Augmented Reranking (SAR), introduced by Banerjee, Satish, and Wörndl [53] during the workflow of a Retrieval Augmented Generation (RAG) pipeline to recommend sustainable city trips to users using LLMs. However, neither traditional nor LLM-based recommenders possess the inherent sustainability knowledge, with the latter even ignoring the SAR procedure to revert to recommending non-sustainable cities.

2.5. Popularity Bias in Recommender Systems

2.5.1. Fairness of Recommendations

Fairness of Recommender Systems has received an increasing amount of attention from the community over the past few years. However, the definition of fairness when it comes to recommendations is multi-faceted and can differ from different perspectives - user, item, system, and legal and ethical perspectives as well [54]. On the *user's* side, an imbalance between very active users and inactive, or new users, can result in unfairness of recommendations towards

the latter [55]. From the *item provider's* perspective, sticking to the most popular items can exacerbate the popularity bias, leading to long-tail and niche items not being recommended to users. Finally, fairness for the entire *system* implies maximising its long-term goals, in terms of user feedback and metrics of success. In the travel domain, where item-side unfairness has a direct link to overtourism and worse sustainability, *multi-stakeholder* approaches have been studied to analyze biases and balance recommendations between the user, items, and society [11, 10].

Identifying unfairness of recommendations is particularly challenging when the systems are black-box, like LLM-based recommenders (RecLLMs), which often consist of an immense amount of training information and complex internal algorithms, making it harder to identify the origin of the bias. Furthermore, due to its differences in architecture with traditional recommendation systems, previously established benchmarks of fairness may not always be applicable to evaluate RecLLMs [56], often requiring specific prompt tuning or fine-tuning-based strategies for evaluation [57].

For example, M. Jiang, K. Bao, J. Zhang, et al. [58] investigate the item-side fairness problem in LLM-based Recommender Systems, finding that these systems are significantly affected by the popularity bias of items, and introduce a reweighting and re-ranking-based framework used to finetune LLMs and improve their item-side fairness.

2.5.2. Popularity Bias in LLM-based Recommenders

Lichtenberg, Buchholz, and Schwöbel [59] use the definition of popularity bias in recommender systems as defined by Klimashevskaia, Jannach, Elahi, and Trattner [60] as *"when the recommendations provided by the system focus on popular items to the extent that they limit the value of the system or create harm for some of the involved stakeholders"*. In the case of LLM-based recommenders, the authors posit that it is not unexpected to observe popularity bias, as the models are more likely to be trained on popular items as opposed to unpopular items.

The authors introduce a popularity bias metric, *log popularity difference*, based on desiderata such as well-behaved, zero-centered, anti-symmetric, long tail sensitivity, and monotonicity. The log popularity metric uses a log transformation along with computing the mean to aggregate the values, and finally computes the difference between the aggregated and transformed popularities of the recommended list as well as the user's base popularity.

In order to measure the popularity bias of standard recommenders as well as an LLM-based recommender (WOK, or WOrld Knowledge recommender), the MovieLens dataset is used [61]. The authors instruct the LLM to use only the knowledge gathered during training and provide it with the user's watch history. The results show that LLM-based recommenders, like Claude and GPT, show reduced popularity bias, even without explicit bias mitigation instructions. Adding mitigation and minimization instructions in the prompts can further reduce popularity bias; however, they also show decreased recommendation performance, measured by the hit rates HR@5 and HR@10, indicating a trade-off between bias mitigation and recommendation relevance.

2.6. Ensemble LLMs and Multi-Agent Systems

Although LLMs are powerful tools with impressive generative capabilities, they often fall short and struggle with complicated tasks involving multiple criteria to adhere to. Recent studies have found that instead of a single LLM, using multiple LLMs in an ensemble setting can improve their performance [62]. Guo, X. Chen, Y. Wang, et al. [63] survey the current research into multi-agent systems in various domains, studying the different frameworks and applications. Furthermore, J. Li, Q. Zhang, Y. Yu, et al. [64] analyze the scaling effects of LLM agents and find that simple sampling-and-voting strategies are able to enhance the collective performance to provide improved overall results for problems of varying complexities. In recommender systems, Y. Zhang, Qiao, J. Zhang, et al. [65] survey the state-of-the-art paradigms and architectures that develop and apply LLM agents to enhance search and recommendation capabilities.

2.6.1. Ensemble LLM Agents

D. Jiang, X. Ren, and B. Y. Lin [66] introduce LLM-Blender, a post-hoc framework that ranks and subsequently fuses the results of an ensemble of LLMs. The ranking is done with a PairRanker, which uses a Cross-Attention Transformer to compute the pairwise comparison between a pair of LLMs and the input. These scores are then aggregated for every pair of candidates to get the top-K "best" candidates. The second step involves fusing these K candidates using a seq2seq approach [67] by concatenating the inputs with the candidates and fine-tuning a generative model like Flan-T5 [68] to generate the final output. This approach allows them to take advantage of the diverse strengths of different LLMs and exploit their abilities to compute superior results.

Furthermore, B. Lv, C. Tang, Y. Zhang, et al. [69] build on the LLM-Blender method to introduce the Uniform Ranking and Generation (URG) approach, which incorporates cross attention with KL-loss as well as a noise mitigation training stage to train against badly ranked candidates, thereby enhancing the robustness of the framework and achieving State-of-the-Art (SOTA) performance with both automatic and human evaluation metrics.

2.6.2. Multi-Agent Systems

LLM-based Multi-Agent Systems

An emerging area of research involves testing the applicability of LLMs as agents in a multi-agent setting to tackle problems in various domains, from mathematical reasoning to decision-making [63, 70]. These settings often involve autonomous LLM agents with designated roles collaborating through interactions and sharing information. Surveys by Yehudai, Eden, A. Li, et al. [71] and Q. Peng, H. Liu, H. Huang, et al. [14] review current algorithms and workflows, focusing on communication, evaluation, and applications in areas like web search and scientific Q&A. K.-H. Huang, Prabhakar, Dhawan, et al. [72] design an agentic system and benchmarks for customer resource management and find that LLMs tend to struggle with this task and achieve limited success.

Facilitating Interaction between Agents

As each agent in a multi-agent system is autonomous with its own designated tasks and instructions, efficient information sharing and collaboration between multiple agents are paramount to help the agents benefit from other perspectives and negotiate with each other [73, 74]. One prominent strategy explored is Multi-Agent Debate, where agents are provided with each other's responses and asked to debate over multiple rounds to arrive at a consensus [75]. X. Chen, Mao, S. Li, and Shangguan [76] apply a similar debate-feedback algorithm to legal documents and find that multi-agent systems, enhanced with reliability analysis of the agents, can significantly outperform state-of-the-art models in legal judgment prediction while minimizing the need for extensive training data. In ReConcile, the authors use round table conferences between agents with confidence-based weighted voting protocols to reach consensus [77]. Further studies by Wan, J. Chen, Stengel-Eskin, and M. Bansal [78] and Shridhar, Sinha, Cohen, et al. [79] also show how iterative refinement by asking subquestions and providing critiques can improve the performance of LLM agents in summarization and Q&A. However, explicit consensus via majority voting can often lead to a trade-off with diversity, as explored by Z. Wu and Ito [80], who posit that distinctive roles for agents coupled with implicit consensus mechanisms can bolster diversity while maintaining broad consensus.

Multi-Agent Recommender Systems

LLM-based Multi-Agent Systems have also been explored in a conversational recommendation setting. These LLM agents, equipped with memory and knowledge of historical states as well as access to external APIs, collaborate with each other to effectively understand user preferences and provide tailored recommendations without the need for large datasets or fine-tuning [81]. These systems generally consist of a centralized Manager agent, as introduced in MACRS [82] and MACREC [83], along with agents instructed to analyze the user and item spaces. A Search agent is also commonly included, consisting of a search engine and access to external sources of information, which could be the Web or a database [84]. With MATCHA [85], this setup is further extended to incorporate safeguard agents against jailbreaking and dangerous content generation, and a dedicated agent for improving the explainability of the recommendation in the video game domain. These systems provide a good foundation for the various roles agents can play and how they can interact with each other to provide high-quality recommendations.

However, to our knowledge, multi-agent systems have not yet been explored for tourism recommenders. This thesis aims to address this research gap by designing an agentic recommendation framework to provide personalized recommendations given a user query. Furthermore, by incorporating the multi-stakeholder approach to tourism in our system design [10], we aim to study if a multi-agent system with distinct agent roles and objectives can provide recommendations that are not only customized to user preferences but are also diverse and balanced.

3. SynthTRIPS: Synthetic Query Generation

This chapter introduces and discusses SynthTRIPS [1] - the synthetic query generation framework, as depicted in Figure 3.1, customized to enhance personalization and sustainability in Tourism Recommender Systems.

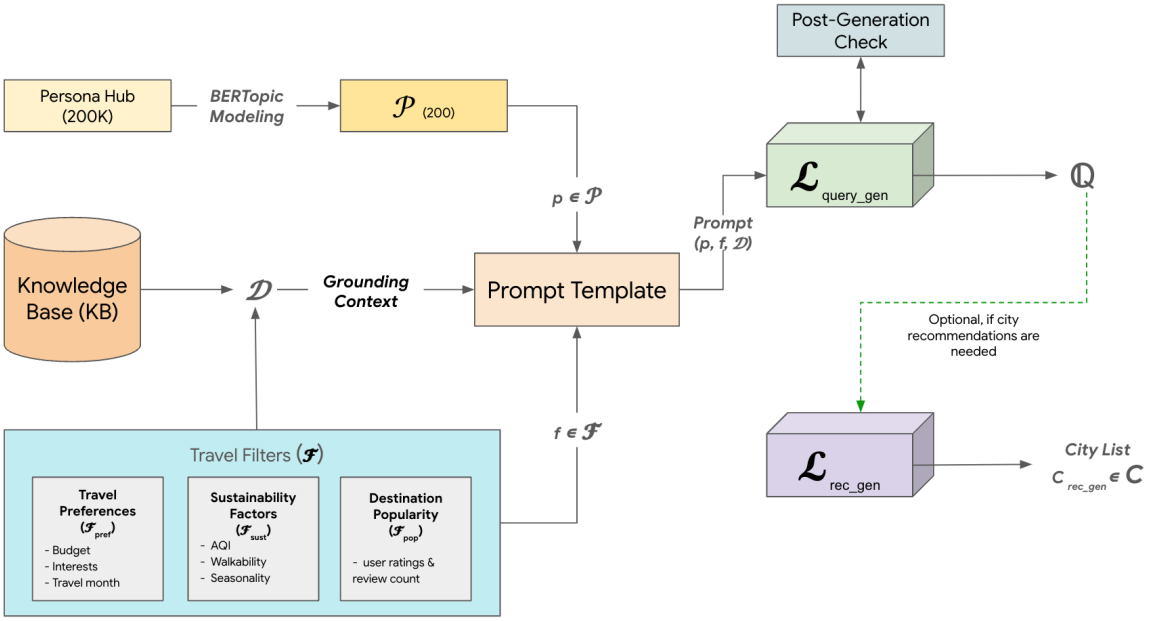


Figure 3.1.: The SynthTRIPS framework for generating synthetic data using LLMs for personalized, sustainable city trips.

Motivation

In the travel domain, personalization is especially valuable: user choices depend on individual preferences (e.g., budget, travel style, sustainability concerns) and contextual factors (destination popularity, local activities, etc.) [86]. Consequently, **high-quality training data** that reflects real-world travel preferences is essential for building effective and robust Tourism Recommender Systems.

However, publicly available travel datasets are often limited in **breadth** (e.g., focusing only on popular cities) and **depth** (e.g., missing nuanced preferences like climate consciousness, walkability, or budget constraints). The lack of richly annotated data hinders research on

advanced personalization strategies incorporating specialized filters — such as sustainable travel or off-peak tourism.

Large Language Models (LLMs) — such as GPT-4, PaLM, and LLaMA — provide a powerful way to generate human-like text, which can be used to emulate users for recommender systems [87]. In the travel context, LLMs can:

1. Emulate diverse *user personas* (e.g., a budget-conscious student vs. a sustainability-focused family).
2. Incorporate *structured filters* (e.g., budget limits and environmental impact considerations).
3. Generate *natural-sounding requests* that integrate factual knowledge (e.g., city highlights, monthly visitor footfall).

However, LLMs are prone to *hallucination* when asked for domain-specific details. To mitigate this, we propose *grounding* LLM responses in curated knowledge bases (KB) to ensure factual correctness regarding destinations, costs, or sustainability metrics.

Given that travel data with rich user and contextual features is notoriously difficult, our goal is to design *(i)* a synthetic dataset that captures realistic travel constraints; *(ii)* a generation pipeline that enables flexible updates (new personas, destinations, or sustainability constraints); and *(iii)* reproducible evaluation protocols, in order to address the pressing need for better coverage, deeper personalization, and a more robust domain grounding in tourism recommendations.

Below are some example user queries for a sustainable city trip, ensuring factual accuracy by grounding in a verified KB.

- **Q1.** *"Find a low-budget, walkable city in Europe with unusual museums or a hidden, alternative nightlife scene."*
- **Q2.** *"Quiet European coastal city with good air quality, affordable, not touristy, with interesting nightlife options."*
- **Q3.** *"Best European cities for unique, artistic experiences and independent cinema, avoiding mainstream tourist attractions?"*

Such queries capture complex **personalization** aspects (budget, interests, sustainability) that are typically underrepresented in publicly available travel datasets [12, 13]. *As evident from Figure 3.1, our resource generation framework,* SynthTRIPS*, automates the creation of such queries at scale, pairing persona definitions with domain-specific constraints to produce a rich corpus for training and benchmarking TRS models.*

Contributions

SynthTRIPS is a novel resource for building sustainable, persona-aware travel data via LLMs, which addresses a critical gap in the tourism recommendation community by offering:

- **A reproducible pipeline for large-scale synthetic data generation** rooted in a curated knowledge base (KB) that ensures factual grounding. By focusing on European city trips and integrating sustainability metrics such as walkability, off-peak travel, and popularity constraints, SynthTRIPS provides a versatile blueprint adaptable to diverse travel contexts.
- **Open-source datasets and code**: We publicly release all components, accessible here, including:
  - **Data**: (1) A structured Knowledge Base (KB) containing real-world attributes of European cities, (2) Generated queries that integrate filters (budget, timing, popularity, sustainability)
  - **Code for Query Generation**: (1) Prompt templates leveraging In-Context Learning (ICL) with various persona definitions, (2) Colab notebooks for straightforward replication of the entire pipeline.
  - **Code for Evaluation**: (1) Automated LLM-based evaluation measuring groundedness, factual correctness, persona alignment, and sustainability compliance, (2) Tools for expert (human) evaluation for clarity, diversity, and overall rating.
- **A flexible framework for future expansions**: Researchers can easily adapt SynthTRIPS to other regions or thematic focuses (beyond sustainability) by updating the underlying KB and modifying persona definitions or filter sets.

By offering a complete methodology for generating data and robust evaluation metrics, SynthTRIPS opens new avenues for more **personalized, sustainability-focused** travel recommendations.

This chapter is organized as follows: Section 3.2 provides a detailed formal description and methodology of our query generation framework. Subsection 3.1.1 subsequently presents our primary resources along with their key statistics and Section 3.3 outlines the various validation dimensions utilized to evaluate the quality of our generated queries.

3.1. Dataset

3.1.1. Knowledge Base

Our *KB* construction is inspired by the data sources and methodology outlined in Banerjee, Satish, and Wörndl [88]. We utilize Wikivoyage's *Listings*[1] data for detailed tabular information on POIs, including places to visit, accommodations, and dining options in 141 unique European cities. The POIs in this dataset are categorized into different activity categories such as *'see'*, *'do'*, *'eat'*, *'drink'*, *'sleep'*, and *'go'*.

To map the wikivoyage listings POIs to the five travel preferences in Section 3.2, we use `bart-large-mnli`, a BART [89] variant trained on the MNLI dataset [90], for zero-shot classification. This pre-trained Natural Language Inference (NLI) model enables sequence

[1] https://github.com/baturin/wikivoyage-listings

classification without additional training [91] and assigns a probability score indicating how well each POI aligns with the defined travel filters. The top three most probable items in each activity sub-category per city are selected to maintain consistency, reduce anomalies, and remain within the LLM's context window.

Publicly available data from Tripadvisor[2] and Where and When (W2W)[3] is utilized to incorporate both popularity and seasonality into the sustainability features of our KG. Tripadvisor reviews and opinions represent city popularity, while W2W's monthly visitor footfall data indicates seasonality [88]. Additional cost labels, AQI levels, and walkability data are sourced from Nomadlist[4].

Basic Statistics

The final dataset comprises 200 cities from 43 European countries, formed by combining the cities from the aforementioned four datasets. Analyzing the distribution of popularity in our *KB* shows that 59.16% of the cities are categorized as high popularity, with 29.66% and 11.18% categorized as medium and low popularity respectively. To ensure that this skew in the data does not reflect in our generated queries, we use a popularity-based stratification while generating the configurations such that there is an equal representation of low, medium, and popular cities.

Furthermore, looking into the seasonality reveals that the peak seasons across cities are the summer months in Europe, from May – September. December also sees a slight increase in seasonality, potentially due to the Christmas holiday season, which precedes the decrease in seasonality over the winter. January – March is the low-season months across cities.

Following the classification of the interests into the 5 categories as described in Subsection 3.2.1, we observe that the distribution of travel interests in our *KB* is predominantly related to *Arts & Entertainment* and *Food*, comprising of almost 70% of the dataset, followed by *Outdoors & Recreation*, *Nightlife Spot* and *Shops & Services*. These interest categories cover a wide range of points of interest, from museums, galleries, and theaters to restaurants, bars, and natural sights, ensuring diversity of travel interests during query generation.

3.1.2. Generated Queries

Our framework results in a total of 4,604 diverse queries generated from 2,302 key functions in 3 experimental settings (q_{v}, q_{p0}, and q_{p1}) for each model namely *gemini-1.5-pro* and *llama-3.2-90b* . The dataset consists of the mapping between key functions, retrieved-context, and generated queries. The configuration setting includes the selected persona, city popularity, complexity, and travel filters. The retrieved context includes the information used to ground the query ($f \in \mathcal{F}$) and the set of cities from our knowledge base that align with the travel filters ($\mathscr{D} \subset \mathcal{D}$). Additionally, the dataset contains three types of queries, corresponding to each experimental setting.

[2] `https://tripadvisor.com`
[3] `https://www.whereandwhen.net`
[4] `https://nomadlist.com`

Basic Statistics

Each query in our generated queries dataset is mapped to an un-ranked list of ground truth cities. To ensure balanced exposure for each city, we stratify our key functions based on the city category described in Subsection 3.1.1. This results in a well-proportioned distribution of generated queries: 739 low-popularity cities (32%), 784 medium-popularity cities (34%), and 779 high-popularity cities (33%). In this dataset, we have, on average, 11 cities from our *KB* mapped to a generated query.

3.2. SynthTRIPS Query Generation Pipeline

We now describe the formal design of SynthTRIPS and illustrate how LLMs are leveraged to generate rich, realistic user queries. Our system is organized around three core modules: *(1) Persona Hub*, *(2) Travel Filters*, and *(3) Contextual Prompting with KB Grounding*, as shown in Figure 3.1.

3.2.1. Formal Description

Personas: Prior research has shown that persona-guided LLMs can generate more diverse annotations than standard LLMs, improving the variability of LLM-based data annotation [92, 93]. Hence, to include the personalization aspect in our setup, we use personas from the PersonaHub dataset, containing 200,000 unique personas. This dataset is a repository of diverse user profiles (e.g., student, family, digital nomad), where each profile includes attributes such as age, interests, and environmental concerns.

To ensure broader coverage and reduce redundancy, we apply **BERTopic modeling** [94] using `all-MiniLM-L6-v2` embeddings on 196,693 English personas, clustering them into 201 topics (with one outlier group). We then select the highest probability persona from each cluster, resulting in a refined set of 200 diverse and representative personas.

Let $p \in \mathcal{P}$, where $\mathcal{P} = \{p_1, p_2, \ldots, p_{200}\}$ denotes this refined set of 200 personas, and let $f \in \mathcal{F}$ be a set of travel filters (explained below).

Knowledge Base (KB) We denote our knowledge base of European cities and their attributes as:

$$KB = (C, A, V),$$

where:

- C is the set of *city entities* (e.g., "Amsterdam," "Berlin," "Budapest"),
- A is the set of *attribute types* relevant for city travel (e.g., "walkability index," "average cost," "peak season," "AQI"),
- $V : C \times A \to \mathcal{D}$ is a partial function mapping city-attribute pairs to a value domain $\mathcal{D}$ (e.g., real numbers or categorical labels).

We refer to *KB* as containing all factual knowledge used to ground the LLM generations.

Travel Filters: These filters $\mathcal{F}$ are used to retrieve the relevant domain information from *KB* using structured queries. We define the set of travel filters $\mathcal{F}$ as a combination of **travel filters**, **sustainability factors**, and **destination popularity**:

$$\mathcal{F} = \mathcal{F}_{\text{pref}} \cup \mathcal{F}_{\text{sust}} \cup \mathcal{F}_{\text{pop}} \tag{3.1}$$

Travel filters ($\mathcal{F}_{\text{pref}}$):

$$\mathcal{F}_{\text{pref}} = \{\mathcal{F}_{\text{budget}}, \mathcal{F}_{\text{month}}, \mathcal{F}_{\text{interests}}\} \tag{3.2}$$

where:

- $\mathcal{F}_{\text{budget}}$: (low, medium, high),
- $\mathcal{F}_{\text{month}}$: (January–December),
- $\mathcal{F}_{\text{interests}}$: from five categories [95]: *Arts & Entertainment*, *Outdoors & Recreation*, *Food*, *Nightlife Spots*, *Shops & Services*.

Sustainability Factors ($\mathcal{F}_{\text{sust}}$):

$$\mathcal{F}_{\text{sust}} = \{\mathcal{F}_{\text{seasonality}}, \mathcal{F}_{\text{walkability}}, \mathcal{F}_{\text{AQI}}\} \tag{3.3}$$

Here, $\mathcal{F}_{\text{seasonality}}$ focuses on less crowded (off-peak) months, $\mathcal{F}_{\text{walkability}}$ imposes a minimum walkability index, and $\mathcal{F}_{\text{AQI}}$ checks city air quality thresholds. Sustainable destinations typically prioritize low seasonality, high walkability, and excellent AQI.
Destination Popularity ($\mathcal{F}_{\text{pop}}$): We estimate popularity from the normalized number of Tripadvisor reviews [11]. For a city c, let $R(c)$ be the number of reviews. We define:

$$\text{popularity}(c) = \frac{R(c) - \min\limits_{c' \in C} R(c')}{\max\limits_{c' \in C} R(c') - \min\limits_{c' \in C} R(c')} \tag{3.4}$$

We then split the normalized scores into three tiers (*low*, *medium*, *high*). We separate popularity from sustainability constraints to ensure a balanced dataset covering all popularity ranges.

Complexity Levels of Filters. Inspired by TravelPlanner [12], we define four levels of complexity:

$$\begin{aligned}
\textbf{Easy:} \quad & f_{\text{easy}} = \{f\}, \quad f \in_R \mathcal{F}_{\text{pref}} \\
\textbf{Medium:} \quad & f_{\text{med}} = \{f_1, f_2\}, \quad f_1, f_2 \in_R \mathcal{F}_{\text{pref}}, f_1 \neq f_2 \\
\textbf{Hard:} \quad & f_{\text{hard}} = \mathcal{F}_{\text{pref}} \\
\textbf{Sustainable:} \quad & f_{\text{sust}} = \{f_1, f_2\} \cup \{s\}, \quad f_1, f_2 \in_R \mathcal{F}_{\text{pref}}, s \in_R \mathcal{F}_{\text{sust}}
\end{aligned}$$

This ensures a mix of simple and more complex queries. Formally, let

$$\mathcal{F}_{\text{complexity}} = \{f_{\text{easy}}, f_{\text{med}}, f_{\text{hard}}, f_{\text{sust}}\}.$$

Overall Key Functions. For each persona $p \in \mathcal{P}$, we generate queries across all four complexity levels and three city popularity tiers (*low*, *medium*, *high*). This yields:

$$|\mathcal{P}| \times |\mathcal{F}_{\text{complexity}}| \times |\mathcal{F}_{\text{pop}}| = 200 \times 4 \times 3 = 2{,}400$$

possible key functions, denoted $\ell(p,f)$.

Retrieval of Grounding Context. When a key function $\ell(p,f)$ is chosen, we run a structured query on *KB*:

$$\mathscr{D} \subset \mathcal{D} \;=\; \text{Retrieve}(KB,\, f),$$

thus obtaining only the relevant facts (e.g., cities $c_{\mathscr{D}}$ and their corresponding attributes satisfying the filter constraints). If $\text{Retrieve}(KB,f) = \emptyset$, the filter is deemed invalid. This results in 2302 valid key functions for our use case.

Prompt Construction. The context $\mathscr{D}$, along with the persona p and the filter set f, is assembled into a structured prompt:

$$\text{Prompt}\ (p,f,\mathscr{D}).$$

We feed this prompt to a *Query Generator LLM*, denoted $\mathcal{L}_{\text{query\_gen}}$. The resulting final user query is:

$$\mathbb{Q} = \mathcal{L}_{\text{query\_gen}}\left(\text{Prompt}(p,f,\mathscr{D})\right).$$

City Recommendation (Optional). If a subsequent step uses an LLM or any recommender model $\mathcal{L}_{\text{rec\_gen}}$ for city recommendations (given the newly generated user query), we denote that:

$$c_{rec\_gen} \;=\; \mathcal{L}_{\text{rec\_gen}}\ (\mathbb{Q}),$$

where c_{rec_gen} is the list of recommended cities provided by $\mathcal{L}_{\text{rec\_gen}}$. However, this step is optional for creating our *synthetic user query* dataset, since we primarily focus on generating queries in **SynthTRIPS**. Still, the framework naturally extends to a multi-turn pipeline.

3.2.2. Goal

The goal of SYNTHTRIPS is to produce a dataset $\mathbb{Q} = \{q_1, q_2, \dots\}$ and accompanying cities in the context $c_{\mathscr{D}}$, ensuring the queries are realistic, personalized, and grounded in factual knowledge.

Example

Consider a configuration where:

- **Persona (p)**: *"A wanderlust-filled trader who appreciates and sells the artisan's creations in different corners of the world"*

- **Filters (f)**:
  - Popularity: Low
  - Interests: Nightlife Spot

Given this configuration, the pipeline generates a variety of queries:

- **Vanilla Query ($q_{\mathbf{v}}$)**: *"Recommend off-the-beaten-path European cities with low popularity for a nightlife-focused trip with a mix of bars, clubs, and live music venues."*
- **Persona-Specific Query 1 ($q_{\mathbf{p0}}$)**: *"Unique nightlife and cultural experiences in off-the-beaten-path European cities for a budget-conscious traveler interested in local artisans."*
- **Persona-Specific Query 2 ($q_{\mathbf{p1}}$)**: *"Which European cities offer a rich cultural heritage, historic centers, and local artisan markets to explore?"*

This example highlights how the combination of personas and filters shapes the generated queries, providing both general (vanilla) and tailored (persona-specific) outputs. We explain the different outputs (vanilla, and personalized) in Subsection 3.2.4.

3.2.3. In-Context Learning and Prompt Templates

SYNTHTRIPS employs few-shot or single-shot **In-Context Learning (ICL)** to guide the LLM to produce queries that are *stylistically aligned* with the persona and *faithful* to the filters. Our prompt template [96] typically includes:

1. **Task Description**: Instruct the LLM to generate user queries given a specific persona, set of travel filters, and retrieved KB data.
2. **In-Context Examples**: One or more example queries illustrating the desired style and structure.
3. **Persona, Filters, & Context**: The explicit persona traits, the chosen filter constraints, and relevant city attributes from the KB.
4. **Output Requirements**: The final request to produce a single-sentence or multi-sentence user query, possibly with formatting constraints (e.g., no direct city names).

3.2.4. Response Generation

For each of the 2,302 valid key functions, we experiment with three conditions using *llama-3.2-90b* and *gemini-1.5-pro* as the $\mathcal{L}_{\text{query\_gen}}$:

- **$q_{\mathbf{v}}$: Vanilla (Non-Personalized)**: Persona information is omitted.
- **$q_{\mathbf{p0}}$: Personalized Zero-Shot**: Persona information is included, but without an example.
- **$q_{\mathbf{p1}}$: Personalized Single-Shot**: Persona information is included along with an example.

Hallucination Mitigation via Grounding

LLMs can produce *hallucinated* statements when lacking access to factual references [97]. To address this in SynthTRIPS, we explicitly **ground** the LLM output by:

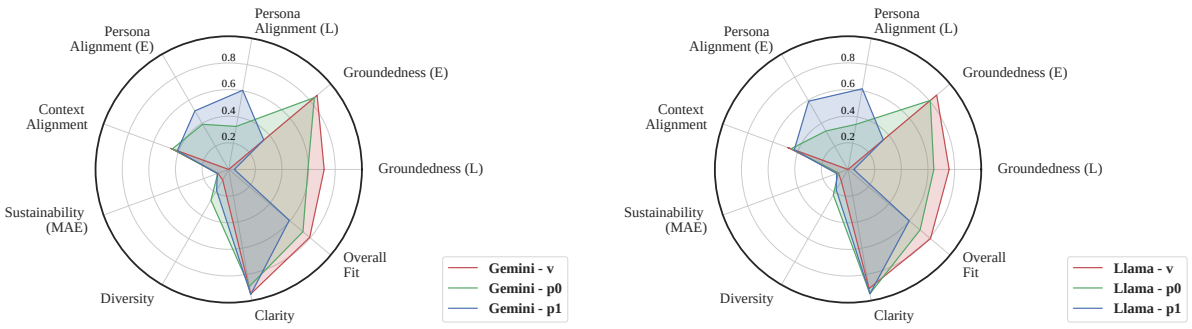
- **Strict retrieval from** *KB*: Only validated attributes from *KB* that meet the filter constraints are provided in the prompt.
- **Post-generation checks**: To ensure properly formatted outputs, we implement an output parser. When the LLM generates overly verbose responses, we first apply regex-based extraction. If unsuccessful, we use an LLM-based parser using *gemini-1.5-pro* , followed by manual verification.

SynthTRIPS significantly reduces inaccuracies and fosters factual consistency through prompt guidance and post-generation verifications.

3.3. Evaluation

In this section, we present the validation of the generated queries, focusing primarily on assessing their quality. Additionally, we demonstrate their practical utility by leveraging a recommender LLM, denoted as $\mathcal{L}_{\text{rec\_gen}}$, to generate travel city recommendations.

We validate the generated queries across the following dimensions: groundedness with travel filters (Subsection 3.3.2), alignment with context (Subsection 3.3.4), alignment with personas (Subsection 3.3.3), adherence to sustainability filters (Subsection 3.3.5), diversity of generated queries and overall clarity and quality assessment (Subsection 3.3.6). Additionally, Subsection 3.3.1 provides an overview of our experimental setup. Figure 3.2 shows a radar chart showing the different dimensions of validation and performance of queries generated by both the models.



(a) *gemini-2.5-flash* (b) *llama-3.2-90b*

Figure 3.2.: Radar Charts comparing the different dimensions of validation and performance of queries generated by *gemini-2.5-flash*and *llama-3.2-90b* . L (E) denotes LLM (Expert) validations.

3.3.1. Experimental Setup

We use a combination of offline experiments and expert (human) evaluations to assess the quality of the generated queries. The subsequent discussion provides a detailed overview of our experimental setup across these validation criteria.

Offline Experiments

We assess the quality of the generated queries through the following offline experiments: (1) JudgeLLM, or J-LLM (*gpt-4o*) to evaluate alignment with travel filters and persona, (2) semantic similarity for contextual alignment, (3) Mean Average Error (MAE) to measure sustainability alignment, and (4) Self-BLEU scores to assess query diversity.

Expert Evaluation

To align JudgeLLM outputs with human judgment, we conduct expert evaluations on 60 queries per model. Two domain experts independently assess the queries using a custom Streamlit-based tool, available on Hugging Face Spaces[5]. Figure 3.3 shows a screenshot of the interface.

Experts review queries from q_{v}, q_{p0}, and q_{p1} blindly, with access to relevant personas and filters to reduce bias. The tool includes collapsible instructions, supports incremental submissions, and saves data in a Firebase real-time database [98]. Evaluations focus on dimensions LLMs often struggle with: persona alignment, travel filters, clarity, and overall fit.

The following sections provide a detailed analysis of the evaluation outcomes for each query dimension.

3.3.2. Groundedness with Travel Filters

The following subsections elaborate on quantifying the coverage of the generated queries, or rather, how well they are grounded with respect to the input filters. Coverage indicates how many of the input filters are included in the query. For example, asking for a popular destination when the input filter is "low popularity" should result in a lower groundedness and is an indication that the LLMs did not listen to instructions.

Finding Number of Matched Filters using JudgeLLM

To measure how well the generated queries represent the input constraints or the travel filters, we measure the groundedness using LLMs as an evaluator (JudgeLLM) [99]. We use *gpt-4o* as our JudgeLLM to evaluate the queries generated by *llama-3.2-90b* and *gemini-1.5-pro* by providing the generated query and the travel filters and asking it to find the number of filters present in the query, along with an explanation. We then use these matches to compute the Mean Recall (MR) for each query setting as follows:

[5]https://huggingface.co/spaces/ashmib/user-feedback

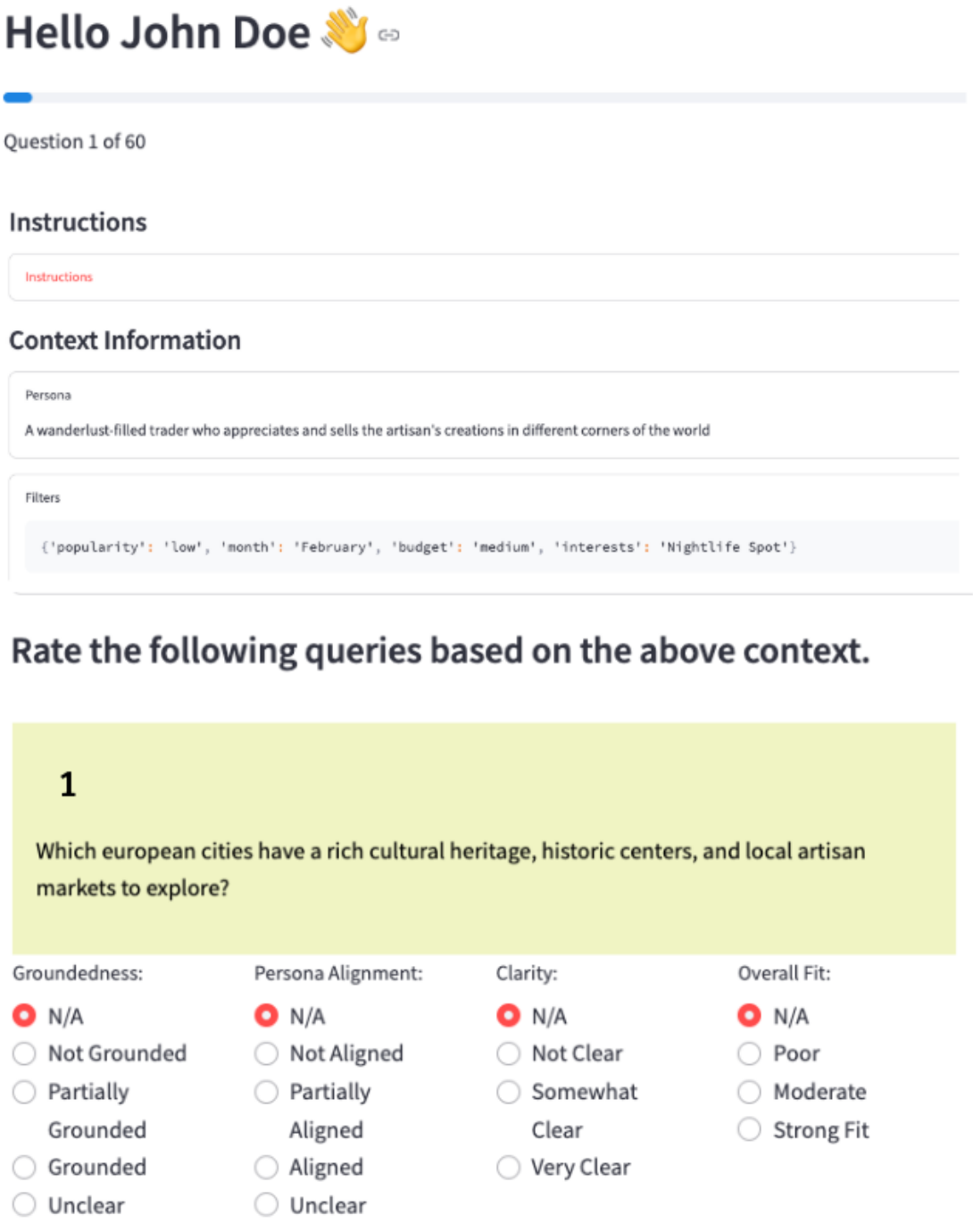


Figure 3.3.: Screenshot showing a part of the evaluation tool developed for the expert study. The full version can be found here.

$$MR = \sum_{i=1}^{N} \frac{len(matches)_i}{len(filters)_i} \tag{3.5}$$

The results of the MR scores can be found in Table 3.1. We can see that the vanilla setting (q_{v}) consistently outperforms the other two settings across models, with the personalized zero-shot (q_{p0}) setting showing the second-best performance. However, the personalized single-shot (q_{p1}) setting shows extremely low recall values across models. We theorize that this is due to the models overfitting on the persona and the ICL example provided in the prompts, which also led to lower diversity scores for this setting.

Expert evaluations align with the observed trends of JudgeLLM in Table 3.1, with evaluators categorizing queries into four levels: 0 for *Unclear*, 1 for *Not grounded*, 2 for *Partially grounded* and 3 for *Grounded*, based on their alignment with travel filters. The percentage of queries rated as *Grounded* (level 3) is reported in Table 3.1. To further analyze the reliability of expert judgments, we measured inter-evaluator agreement, where we obtained a Mean Absolute Error (MAE) of 0.083, indicating a high level of consistency between the two evaluators.

Table 3.1.: Validation results assessing the alignment of generated queries with travel filters ($\mathcal{F}$) and personas (p), as evaluated by both the LLM as Judge (J-LLM) and domain experts. The values in **bold** (*italics*) denote the maximum (minimum) values for each model across all the settings – q_{v}, q_{p0}, and q_{p1}. ↑(↓) means higher (lower) is better.

Model	Query Setting	Alignment (%) ↑			
		$\mathcal{F}$		p	
		J-LLM	Expert	J-LLM	Expert
Llama	q_{v}	**0.759**	**0.869**	N/A	N/A
	q_{p0}	0.644	0.806	*34.348*	*33.333*
	q_{p1}	*0.043*	*0.346*	**61.538**	**59.167**
Gemini-1.5-Pro	q_{v}	**0.717**	**0.867**	N/A	N/A
	q_{p0}	0.595	0.840	*32.841*	*39.167*
	q_{p1}	*0.042*	*0.344*	**60.382**	**50.833**

Further Metrics of Groudedness

Cosine Similarity: A simple and straightforward metric to measure groundedness without having to rely on a JudgeLLM is to compute the cosine similarity of the queries with the input filters. Here, we use *all-miniLM-L6-v2*[6][100] to compute the sentence embeddings of the queries and the filters. We then compute the mean cosine similarity between the query and each input filter. Table 3.2 shows the results of this cosine similarity score, averaged

[6]https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

Table 3.2.: Filter Groundedness Results using Cosine Similarity and JudgeLLM (D+F). The values in **bold** denote the maximum value for each model across the settings. ↑means higher is better.

Model	Query Setting	Cosine Similarity ↑	JudgeLLM (D+F) Recall ↑
Llama	q_{v}	**0.405**	**0.801**
	q_{p0}	0.360	0.676
	q_{p1}	0.279	0.093
Gemini-1.5-Pro	q_{v}	**0.407**	**0.727**
	q_{p0}	0.373	0.611
	q_{p1}	0.296	0.056

over the number of queries. We observe that q_{v} shows the best similarity with filters across *llama-3.2-90b* and *gemini-2.5-flash*, outperforming q_{p0} and q_{p1}, indicating that providing a persona to the LLM may cause it to overlook some travel filters.

Decomposition and Faithfulness with a JudgeLLM: While the earlier evaluation with the JudgeLLM only asked the judge to count the number of matches, in order to bolster this evaluation with further evidence, we experiment with a two-phase evaluation approach:

- *Decomposition (D)*, where we prompt the JudgeLLM (with an example) to decompose the given query into several standalone statements, where each statement ideally encompasses a filter. This is done in order to reduce confusion by separating filters from each other and evaluating them independently.
- *Faithfulness (F)*, where the LLM is asked to map each of the decomposed sentences with the travel filters.

The system and user prompts for both stages can be found in Subsubsection A.1.1. In Table 3.2, we report the mean recall of the mapping done during the JudgeLLM (D+F) evaluation, revealing yet again, that vanilla queries tend to have the best coverage with respect to the travel filters. We also notice the sharp decline in recall from q_{p0} to q_{p1}, lending credence to the assumption that providing an example in addition to the persona only confuses the LLMs and leads to worse coverage.

3.3.3. Persona Alignment

We assess the alignment of generated queries with personas using expert evaluation and JudgeLLM (*gpt-4o*), following a two-fold approach similar to that described in Subsection 3.3.2. Both LLM-based and expert evaluators are provided with the persona description, the query, and four rating options: *Not Aligned*, *Partially Aligned*, *Aligned*, and *Unclear*. They are asked to determine how likely the persona would formulate the given query, focusing on tone and phrasing rather than inferred filters.

To mitigate potential biases associated with specific personas, the evaluation prompts explicitly instruct the LLMs and expert evaluators to assess only the tone and linguistic style of the query rather than making assumptions about the persona's domain expertise or preferences. This ensures that alignment ratings reflect the stylistic coherence between the query and persona rather than pre-existing biases.

Table 3.1 presents the results of the persona alignment for each model and query setting, comparing JudgeLLM and expert evaluations. We measure the persona alignment as the percentage of queries rated as *Aligned*. Although q_{p1} shows weaker groundedness with the travel filters, it achieves a higher persona alignment score than q_{p0}, according to JudgeLLM and expert evaluations. This suggests a potential trade-off between persona alignment and query groundedness.

To quantify inter-evaluator agreement, we calculate the MAE score between expert ratings, obtaining a moderate agreement level with an MAE of 0.3. The relatively high MAE suggests that persona interpretation is inherently subjective and can vary across individuals, highlighting the challenges of achieving consistent persona alignment in query generation.

3.3.4. Contextual Alignment

While Subsection 3.3.2 and Subsection 3.3.3 focus on evaluating the alignment of the query with respect to the filters and persona respectively, in the following subsections we assess query groundedness within the context to minimize LLM hallucinations and ensure factual accuracy from our knowledge base (*KB*).

Context Groundedness with MiniCheck

The groundedness with respect to the retrieved context is computed using MiniCheck, introduced by L. Tang, Laban, and Durrett [101], an LLM-based factchecker. Provided with a source document (here, the retrieved context) and a target claim (here, our queries), MiniCheck uses Flan-T5 [102] to predict entailment between the claim and source, i.e., the probability that the claim is validated by the source.

The results of the evaluation using MiniCheck are presented in Table 3.3. The average MiniCheck probabilities across *llama-3.2-90b* and *gemini-1.5-pro* are the highest for the q_{v}, followed by a sharp decrease for for the personalized variants, with q_{p1} showing the worst results. This provides further evidence to establish the tradeoff between personas and groundedness, suggesting that the LLMs may choose to focus on the persona, leading to a compromise in contextual alignment.

Semantic Search with the Vector KB

Another evaluation method used to quantify the contextual alignment is a semantic search for the query q using our *KB* (now represented as a vector database) to retrieve the query context and cities, respectively. *Our assumption follows that if the queries are factually grounded in the KB, then a subsequent semantic retrieval using these queries from the same KB should result in the same*

Table 3.3.: Context Groundedness Results using MiniCheck. The values in **bold** denote the maximum value for each model across the settings. ↑means higher is better.

Model	Query Setting	Average MiniCheck Probability ↑
Llama	q_{v}	**0.206**
	q_{p0}	0.067
	q_{p1}	0.015
Gemini-1.5-Pro	q_{v}	**0.237**
	q_{p0}	0.089
	q_{p1}	0.014

Table 3.4.: Evaluation of context groundedness, sustainability features, and diversity in the generated queries. The values in **bold** (*italics*) denote the maximum (minimum) values for each model across all the settings – q_{v}, q_{p0}, and q_{p1}. ↑(↓) means higher (lower) is better.

Model	Query Setting	Contextual Alignment ↑	Sustainability		Diversity ↑				
			Similarity ↑	MAE ↓	Overall	Query Level			
						f_{easy}	f_{med}	f_{hard}	f_{sust}
N/A	**Baseline**	0.073	N/A	N/A	-	-	-	-	-
Llama	q_{v}	**0.482**	**0.659**	*0.077*	*0.101*	*0.138*	0.136	*0.169*	*0.179*
	q_{p0}	0.455	0.624	0.078	**0.221**	**0.352**	**0.274**	0.299	0.319
	q_{p1}	*0.430*	*0.605*	**0.091**	0.177	0.307	*0.241*	**0.315**	**0.327**
Gemini-1.5-Pro	q_{v}	**0.463**	**0.648**	*0.086*	*0.091*	*0.140*	*0.119*	*0.113*	*0.167*
	q_{p0}	0.451	0.589	0.087	**0.267**	**0.415**	**0.321**	0.327	0.360
	q_{p1}	*0.411*	*0.526*	**0.091**	0.182	0.346	0.251	**0.342**	**0.378**

information. This retrieved context is compared with the original context used during query generation by computing the cosine similarity between the respective contextual embeddings generated using the *all-mpnet-base-v2*[7] model.

From Table 3.4, we observe that the models and query settings show similar performance across metrics, with the vanilla setting barely outperforming the other two settings. However, there is a sizeable increase in performance when compared to the baseline, which is computed by comparing a paragraph of Lorem Ipsum text with the original context, indicating that the addition of the *KB* in the query generation leads to queries that are factually grounded.

3.3.5. Sustainability Alignment

The ***sustainable*** query complexity level f_{sust} contains two non-sustainable travel filters and one sustainable filter, along with the destination popularity information. While we want to introduce sustainable travel into our dataset in a seamless manner, we also need to verify that the sustainable queries do not show substantial differences semantically when compared to the other queries. We measure the cosine similarity scores using the *all-mpnet-base-v2* model between the sustainable queries and non-sustainable queries, as shown in Table 3.4.

The results show moderately high scores between the sustainable and non-sustainable queries across model and query settings, indicating semantic similarity between the two sets. However, this does not indicate whether the sustainable filter is well represented within each sustainable query or if it is compromised in favor of the non-sustainable travel filters.

To evaluate how well the generated queries consider the sustainability constraints, we compute the cosine similarities between each travel filter and the query belonging to the sustainable complexity (q_i). We further compute the MAE between the similarities of the sustainability and the average similarity of the non-sustainability filters (represented by j) for each query setting as the following:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}\left| sim_{\text{sust},q_i} - \frac{1}{M}\sum_{j=1}^{M} sim_{j,q_i} \right| \tag{3.6}$$

The results in Table 3.4 show low MAE scores across query settings and models, suggesting that the LLMs do not overlook sustainability in favor of the other travel filters, ensuring a balanced representation of sustainability during query generation.

3.3.6. Diversity and Expert Evaluation

Diversity

We measure the diversity of the generated queries using Self-BLEU [103], which calculates sentence similarity between a hypothesis and a set of reference texts using n-gram matching. For each generated query q_{i}, we measure Self-BLEU against the other queries as the reference

[7] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

set using 4-grams. We define the diversity score as

$$Diversity = 1 - \frac{(\Sigma_{i=1}^{|G|} Self\text{-}BLEU(q_i))}{|G|}, \quad q \in G, G \subseteq \mathbb{Q} \tag{3.7}$$

A higher diversity score signals better diversity.

Table 3.4 shows that personalization improves query diversity, with q_{p0} and q_{p1} outperforming q_{v}. However, queries from the same persona often sound similar across key functions, lowering overall diversity. At the query level—especially for sustainable topics—more contextual dimensions increase diversity. Yet, q_{p1} may overfit, reducing diversity by closely mimicking prompt examples. This highlights the need for careful prompt design to balance personalization and diversity.

Clarity and Overall Fit

While the JudgeLLM captures many aspects of query quality, it often misses subtle language and semantic nuances. For example, a query like *"Low-cost European city with untapped market potential for retail"* receives perfect JudgeLLM scores for persona alignment and groundedness, yet experts note that "untapped market potential" doesn't fully align with the stated low-popularity, low-budget preference—resulting in a moderate overall fit score, underscoring the need for expert review.

To address this, we include two expert-assessed metrics: *clarity* and *overall fit* (holistic relevance to persona). Clarity scores are consistently high across models, with q_{p1} scoring highest (0.947 for *llama-3.2-90b* , 0.953 for *gemini-1.5-pro*) and a very low inter-rater MAE (0.011), indicating strong agreement.

For overall fit, scores are stable across evaluators (avg. 0.7), but show an inverse trend: q_{v} outperforms q_{p1}, likely due to overfitting in the latter. The low MAE (0.16) still reflects reliable consistency. These findings highlight the value of expert evaluations in capturing subtleties missed by automated metrics.

3.4. Preliminary Benchmarking using a RecLLM

As a preliminary analysis, we evaluate the queries generated by Gemini as $\mathcal{L}_{\text{query\_gen}}$ using the same Gemini model (*gemini-1.5-pro*), now serving as our recommender LLM (RecLLM, $\mathcal{L}_{\text{rec\_gen}}$), grounded with web search [104].

While both query generation and recommendation are performed using *gemini-1.5-pro* in our setup, other LLMs can also function as $\mathcal{L}_{\text{rec\_gen}}$, as their generative capabilities are inherently task- and prompt-dependent [105].

We instruct the RecLLM to recommend cities from the predefined 200 cities in our *KB*, in zero- and few-shot settings. Grounding it with web search allows it to provide citations from web searches if available.

On average, $\mathcal{L}_{\text{rec\_gen}}$ recommends 10 cities per query, with 80% found in our knowledge base (KB). To evaluate popularity bias, we measure the proportion of recommended cities labeled as "high-popularity" in the KB.

Across all queries, 79% of recommendations are highly popular. Even for queries specifically requesting less-visited places, $\mathcal{L}_{\text{rec\_gen}}$ still suggests popular cities 77% of the time.

For example, the c_{rec_gen} for the query *"Trip to a less-visited European city with historical sites and museums in July. High budget."* include popular destinations like *Budapest (Hungary)* and *Prague (Czechia)*, contrasting with the $c_{\mathscr{D}}$, which contains less popular cities such as *Syktyvkar (Russia)*, *Malatya* and *Kars (Türkiye)*, and *Ioannina (Greece)*. This indicates that $\mathcal{L}_{\text{rec\_gen}}$ struggles with nuanced queries and defaults to common destinations.

4. Collab-Rec: Balancing Recommendations using Multi-Agent Systems

In this chapter, we delve into the details of Collab-Rec, an LLM-based agentic framework designed to balance recommendations in TRS, emphasizing on popularity of destinations and sustainability factors, in addition to user personalization.

Motivation

Alongside ensuring the **personal choice of users** (personalization aspect), modern TRS are increasingly called upon to account for other critical demands, notably the **popularity of destinations**, and **sustainability** considerations such as reduced crowding, lower environmental impacts, and eco-friendly travel opportunities [10]. Balancing these three aspects — relevance, popularity, and sustainability poses a substantial challenge for any single recommender algorithm, especially when implemented via LLMs.

As discussed in Chapter 1 and Chapter 2, recent advances in generative recommenders show that LLMs can enhance user experiences with natural explanations and dialogue [106, 107]. While they offer deeper personalization and interpretability, they also risk hallucinations, bias amplification, and privacy leaks [108, 109, 25, 110]. These capabilities blur lines between retrieval, ranking, and explanation, making single-LLM recommenders prone to inexplicable decisions, weak factual grounding, and skewed trade-offs (e.g., favoring popular destinations over sustainability).

Why Agentic Design?

Distributing different objectives across multiple specialized agents offers a promising solution. Instead of a monolithic LLM that tries (and often fails) to optimize for relevance, popularity, and sustainability all at once, an agentic design delegates each task to a distinct LLM agent with a clear mission. A Personalization Agent can focus on personal factors like budget and interests, a Popularity Agent can ensure that less-exposed or regionally diverse locations are not perpetually overshadowed, and a Sustainability Agent can foreground eco-friendly or off-peak destinations. By having these agents propose complementary sets of city recommendations, we harness their "collective intelligence" while reducing the risk that any single objective, especially popularity, dominates system outputs.

Yet merely assembling specialized agents is not enough. Each LLM may harbor its drawbacks:

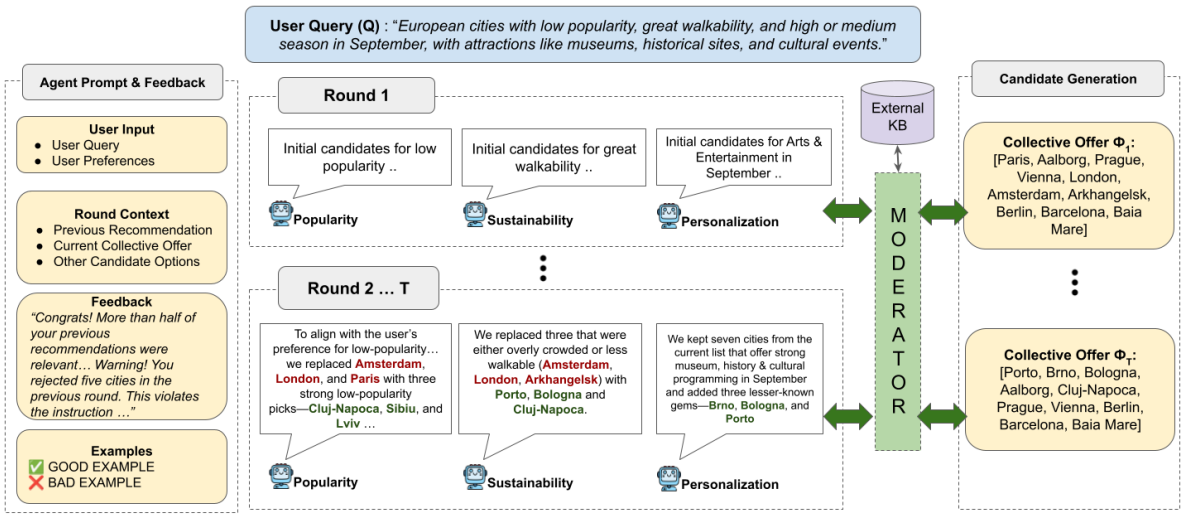


Figure 4.1.: Overview of the Collab-Rec workflow to generate city trip recommendations using multiple LLM agents.

- A popularity-focused model might still over-rely on training corpora that favor major European capitals.
- A sustainability-minded model may hallucinate data about walkability or air quality.
- A personalization agent might confound budget constraints with other factors or inadvertently overlook real user preferences if the prompt design is inadequate.

To address these multifaceted requirements, we propose Collab-Rec, an agentic framework in which multiple LLM-based agents focus on different stakeholder objectives — consumer personalization, popularity, and sustainability, and negotiate in multiple rounds to produce city trip recommendations. As depicted in Figure 4.1, Collab-Rec encompasses three specialized agents, each focusing on distinct stakeholder objectives:

- **Popularity Agent:** promotes lesser-exposed destinations;
- **Personalization Agent:** enforces strict filters (e.g., budget, travel dates, interests);
- **Sustainability Agent:** prioritizes eco-centric criteria such as air quality, seasonality, and walkability.

A non-LLM **moderator** then combines and scores the candidates proposed by these agents. During each round of iterative refinement, the moderator:

- Grounds recommendations in an external knowledge base of 200 European cities to avert hallucination;

- Computes composite scores balancing relevance, reliability, and penalties for invalid or repeated (*hallucinated*) proposals; and
- Releases a *Collective Offer* that becomes the basis for the next negotiation round.

Overall, research on multi-agent LLM systems shows that specialist agents, each with a narrow remit, can negotiate or debate their way to better answers on complex tasks [74, 76]. In recommendation, prototypes such as MACRec [83] and MATCHA [85] already split the work between a manager and various sub-agents, yet none have been tailored to tourism or to the three-way tension among relevance, popularity, and sustainability. We argue that an agentic architecture is particularly apt here for four reasons:

- **Objective isolation:** Each agent can optimize a single stakeholder goal. A *Popularity Agent* deliberately searches the long tail; a *Personalization Agent* enforces hard filters; and a *Sustainability Agent* promotes eco-friendly choices.
- **Transparent negotiation:** Since proposals are exposed in natural language, a non-LLM moderator can audit, penalize, or reward each agent's behavior — curbing hallucinations and revealing implicit trade-offs.
- **Mitigation of model-internal bias:** By design, the three agents pull in different directions. Their iterative compromise, orchestrated by the moderator, naturally dampens the popularity bias that a single LLM would otherwise reinforce.
- **Graceful extensibility:** New stakeholder roles (e.g., safety, accessibility) can be plugged in as additional agents without re-training the whole system, echoing calls for holistic evaluation and modular guardrails in the next-generation Gen-RecSys [25].

Limitations of Existing Approaches

Most current multi-agent LLM frameworks either focus on domain-agnostic tasks (e.g., Q&A) or treat recommendation from a purely *single-objective* lens (e.g., maximizing personalization alone). Meanwhile, existing multi-stakeholder tourism recommenders have not leveraged LLM-based multi-agent negotiation. This gap leaves open the question of how to *jointly* optimize relevance, popularity, and sustainability *within* a single, integrated conversation pipeline. Our work aims to fill this void, exploring the feasibility and benefits of an iterative, penalty-aware approach that explicitly balances different stakeholder goals.

Contributions

We introduce Collab-Rec, a collaborative, multi-agent framework for LLM-based tourism recommendations that explicitly balances user constraints, popularity, and sustainability. Our main contributions are as follows:

- **Agentic Multi-Stakeholder Design:** By delegating the tasks of personalization, popularity, and sustainability to specialized LLM agents, Collab-Rec achieves richer, more balanced recommendations compared to single-agent pipelines.

- **Multi-Round Negotiation:** Agents iteratively propose and refine city candidates under the guidance of a *non-LLM* moderator, which penalizes repeated or hallucinated suggestions. This process fosters iterative compromise and significantly broadens recommendation diversity.
- **Scoring & Moderation for Bias Mitigation:** A custom scoring function that integrates agent success ($r_{a_i,t}$), reliability ($d_{a_i,t}$), and hallucination penalty ($h_{a_i,t}$) helps curb the popularity bias often embedded in large language models.
- **Empirical Evaluation:** Using synthetic and real-world travel queries, we show that Collab-Rec yields systematically higher *relevance* and *diversity* than single-round or single-agent baselines, thus promising a more equitable and eco-conscious travel recommender system.

In the remainder of this chapter, we describe our system design (Section 4.1), experimental setup (Section 4.2), and evaluation results (Section 4.3).

4.1. Collab-Rec Agentic Recommendation Framework

Figure 4.1 illustrates Collab-Rec, an iterative multi-agent framework for city trip recommendations. Given a complex user query $\mathbb{Q}$, specialized agents (e.g., Popularity, Sustainability, Constraints) handle subtasks reflecting different stakeholders in a multi-stakeholder tourism recommendation scenario. In each iteration, agents propose city candidates based on their criteria. A moderator module (non-LLM) evaluates and integrates these proposals using feedback, scoring metrics, and external knowledge, refining the collective recommendation until termination criteria are met.

4.1.1. Preliminaries & System Goal

Problem Setup A user query $\mathbb{Q}$ is an input in natural language that includes the user's preferences and requirements for a city trip recommendation. It also includes a set of structured *filters* denoted by $\mathcal{F} = \{f_1, f_2, \ldots, f_m\}$ (e.g., budget, month, interests, etc.), where each filter f denotes an explicit constraint on the city attributes. Let $\mathcal{C}$ represent the catalog of all candidate cities.

Notation Given a query q with filters $\mathcal{F}$, our recommendation system uses multiple LLM-based agents and a non-LLM moderator in a multi-round interaction process. We denote rounds by $t = 1, \ldots, T$, agents by $a_i \in \mathcal{A}$, and cities by $c \in \mathcal{C}$. The candidate list generated by agent a_i at round t is denoted by $L_{a_i,t}$.

System Goal The goal of the Collab-Rec framework is to select an ordered list of cities Φ_T at the final round T that maximizes the cumulative evaluation score $s(c, t)$:

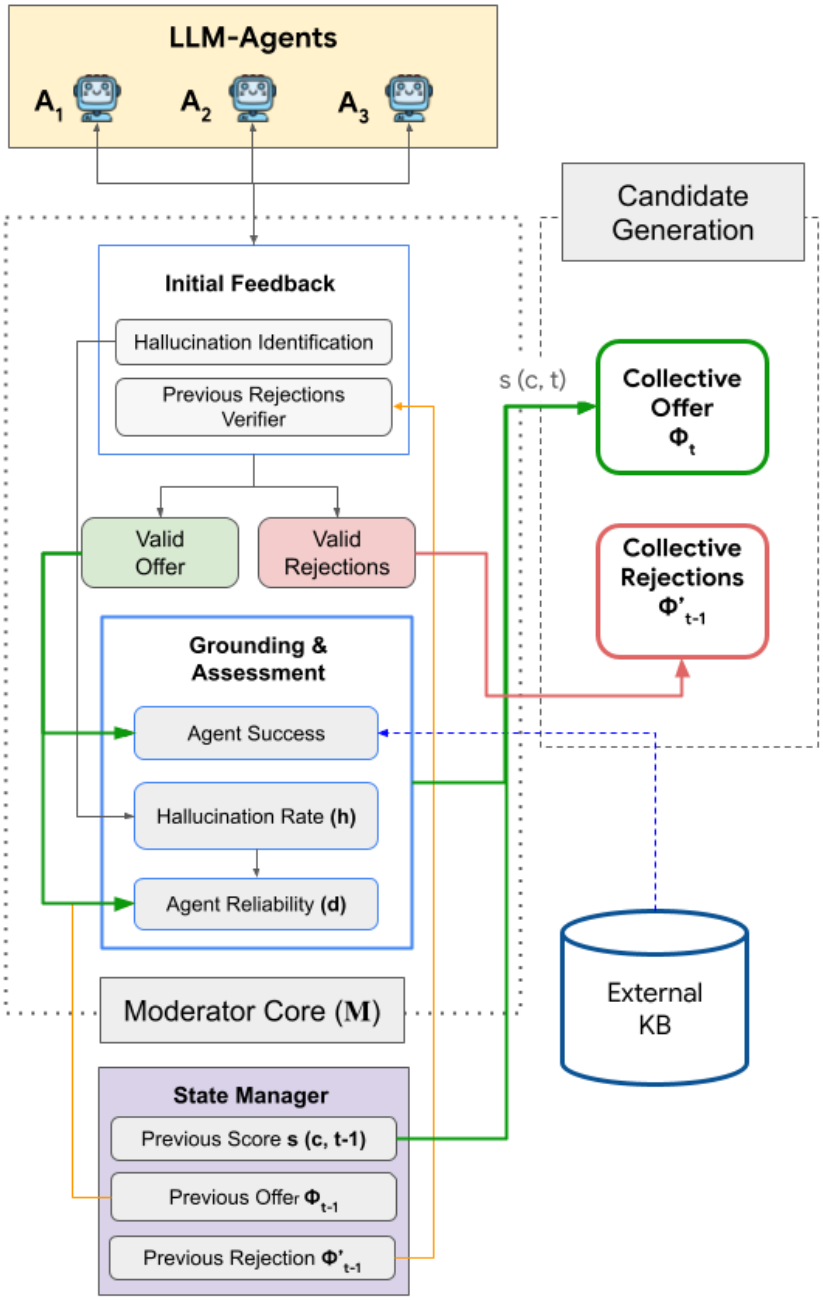


Figure 4.2.: Overview of the Collab-Rec Moderator core to generate city trip recommendations using multiple LLM agents.

$$\Phi_T = \arg\max_{\substack{L_T \\ |L_T|=k}} \left[\sum_{c \in L_T} s(c, T)\right], \tag{4.1}$$

where L_T ranges over all possible city combinations of size k, and $s(c, T)$ denotes the cumulative evaluation score of city c at round T. We define $s(c, T)$ as a combination of the hallucination penalty $h_{a_i,t}$, agent success score $r_{a_i,t}$, and reliability score $d_{a_i,t}$. Each component is further described in Subsection 4.1.4.

4.1.2. Architecture & Components

Recommender Agents

Each agent operates on a natural language query, a specified filter to focus on, the current collective offer, and a set of previously rejected candidates. Inspired by the multi-stakeholder perspective in TRS [10], we instantiate three distinct agents, each representing a key stakeholder group. Each agent produces a recommendation list $L_{a_i,t}$ of candidate cities:

1. a_1*: Item Popularity Agent* — aims to promote less popular items by considering general popularity preferences inferred from the query, thus mitigating popularity bias and item unfairness.
2. a_2*: Personalization Agent* — focuses on user-specified preferences and travel filters (consumer-centric) such as budget, preferred month, and interest categories.
3. a_3*: Sustainability Agent* — prioritizes sustainable travel recommendations, considering factors such as air quality index (AQI), seasonality, and walkability. If no explicit sustainability preferences are provided, this agent defaults to recommending the most sustainable cities available, which focuses on the environment or society stakeholder as identified in [10].

Moderator Core

As seen in Figure 4.2 the moderator component, denoted as $\mathcal{M}$, governs the interaction between agents and orchestrates the multi-round negotiation process. It is a non-LLM decision module responsible for the following tasks: *detecting hallucinations, computing evaluation scores, and aggregating candidate lists* into a final recommendation.

At each round t, the moderator receives the candidate lists $L_{a_i,t}$ from all agents, evaluates each city using the scoring function $s(c, t)$ (Equation 4.6), and constructs the *Collective Offer* ϕ_t by selecting the top-k ranked cities. Cities not accepted in previous rounds form the *Collective Rejection* set ϕ'_t. This updated context is passed back to the agents to guide the next round of recommendations.

These states are logged at each round, along with agent-specific success and reliability metrics, enabling longitudinal analysis of negotiation dynamics and agent behavior. Additionally, it has access to an external knowledge base (KB), which it leverages to ground agent responses and compute evaluation scores with factual consistency.

4.1.3. Interaction Protocol

Our approach follows a multi-round interaction process in which three agents, each representing a distinct stakeholder perspective, collaborate iteratively to reach a consensus that maximizes overall relevance. Each interaction loop consists of the following components:

Agent Instruction

Collab-Rec starts by prompting each agent to generate an initial top-k recommendation list based on the user query $\mathbb{Q}$ and a set of travel-related filters $\mathcal{F}$ such as city popularity, sustainability preferences, and user travel preferences. Each agent tailors its recommendations to the stakeholders it represents — popularity (item-provider-centric), sustainability (society-centric), or personalization (consumer-centric).

In each subsequent iteration, agents receive the current *Collective Offer* ϕ_t from the moderator $\mathcal{M}$ and are instructed to revise their candidate lists accordingly. Inspired by the Voting by Alternating Offers and Vetoes (VAOV) protocol used by Erlich, Hazon, and Kraus [111], we permit the Agents to replace up to three items and must justify any changes with supporting reasoning. We use in-context learning with few-shot prompting during the iteration phase with good and bad examples of recommendations, tailored to each agent's respective preferences. The prompt provided to each agent at a round t consists of its own previous recommendation, the collective offer generated at the end of round $t-1$, and specific feedback about its behavior intended to provide constructive criticism about the agent's performance, similar to the strategy used by Wan, J. Chen, Stengel-Eskin, and M. Bansal [78]. This feedback is generated based on (a) how many of the candidates previously recommended by the agent are present in the collective offer and (b) the number of replacements.

Hallucination Identification

Previous research has shown that LLMs can run the risk of fabricating out-of-catalog items, leading to faulty recommendations [25]. To mitigate this, we incorporate a grounding mechanism into the moderation process. The moderator has access to a finite knowledge base (KB) of 200 European cities, each annotated with relevant metadata, used to ground agent responses. Any candidate city proposed by an agent that is not present in the KB or has already been rejected in a previous round is flagged as a hallucinated or non-grounded response. During this phase, the moderator iteratively checks each candidate in the agent's list $L_{a_i,t}$ for validity. If a hallucinated or previously rejected city is found, the agent is prompted to substitute it with a valid alternative. To avoid infinite feedback loops, a maximum number of iterations is enforced.

In our prototype implementation, this hallucination identification loop is executed only once per round to limit API usage and computational cost. If the agent continues to return to hallucinated cities after the first correction attempt, it is penalized during the evaluation phase (Subsection 4.1.4). However, our framework supports a configurable number of hallucination correction cycles to ensure all candidates ultimately conform to the query constraints.

Generating Collective Offer

We define the Collective Offer (ϕ_t) as a ranked list of cities that will be presented to each agent by the moderator in the next round of iterations $t+1$. This list serves as a shared negotiation baseline for the specific round and reflects the most promising candidates according to the moderator's evaluation.

The collective offer is obtained by applying min-max normalization [112] to the evaluation scores (explained in Subsection 4.1.4) of all candidate cities in the catalog $\mathcal{C}$, scaling them to the $[0, 1]$ range for comparability across rounds. The moderator then selects the top-k highest-scoring cities, which constitute the collective recommendation set returned to the agents for the next round of refinement.

Aggregation of Rejections

In this phase, we identify newly rejected cities by comparing each agent's candidate list with the previous collective offer. A city added to the Collective Rejection (ϕ'_t) is excluded from future recommendations by both the agents and the moderator. We explore two rejection, or voting strategies (we use the terms "rejection strategy" and "voting strategy" interchangeably in the thesis): **Majority rejection (M)**, where a city is discarded if at least two agents omit it in their revised lists; and **Aggressive rejection, (A)**, where omission by even a single agent leads to rejection.

4.1.4. Scoring Functions & Decision Rules

Our framework relies on a set of scoring functions and decision rules implemented by the moderator to guide the multi-agent recommendation process. Specifically, the moderator evaluates each agent's candidate list along three key dimensions — groundedness with respect to the query filters (*agent success*), *agent reliability*, and *hallucination rate*. These evaluation criteria are detailed in Subsubsection 4.1.4, while Subsubsection 4.1.4 outlines the termination conditions and additional operational considerations of the framework.

Grounding & Assessment

In each round, the moderator evaluates the agents and their recommended candidates across three key dimensions — *Agent Success*, *Agent Reliability*, and *Hallucination Rate*.

Agent Success ($r_{a_i,t} \in [0, 1]$) quantifies how well the agent's recommendations align with its assigned filters, as defined in Equation 4.2. It is calculated as the average proportion of filters matched per candidate, based on the filters specific to that agent.

$$r_{a_i,t} = \frac{1}{|L_{a_i,t}|} \sum_{c \in L_{a_i,t}} \frac{|matched\_filters(c_{a_i,t})|}{|f_{a_i}|} \tag{4.2}$$

Agent Reliability ($d_{a_i,t} \in [0,1]$) measures the consistency of an agent's recommendations by quantifying changes in candidate rankings between two consecutive rounds, $t-1$ and t. A high-reliability score indicates that an agent's recommendations remain stable over time.

We compute the reliability score $d_{a_i,t}$ using three key components: (i) the cumulative change in rank positions for candidates appearing in both rounds, (ii) a drop penalty (μ_1) for candidates that were dropped between rounds, and (iii) an added penalty (μ_2) for newly introduced candidates. The drop penalty μ_1 is set to the length of the candidate list, while the add penalty μ_2 is computed based on the minimum rank deviation between the city's position in the moderator's collective offer and its position in the current candidate list, capped by μ_1. Specifically, μ_2 is set to the minimum of the absolute rank difference and the base drop penalty, thereby assigning a lower penalty for cities selected from the collective offer. However, if a city is newly introduced by the agent and was not present in the previous collective offer, it is penalized with the maximum value, μ_1, to discourage hallucinated or ungrounded additions.

$$d_{a_i,t} = \max\left(0,\ 1 - \frac{\Delta_{\mathcal{L}_{a_i,t-1},L_{a_i,t}}}{|\mathcal{L}_{a_i,t-1}| \cdot (\mu_1 + \mu_2)}\right) \tag{4.3}$$

Here, $\Delta_{\mathcal{L}_{a_i,t-1},L_{a_i,t}}$ denotes the rank deviation between the candidate lists from two consecutive rounds for an agent a_i and is defined as:

$$\Delta_{A,B} = \sum_{x \in A \cap B} |\text{rank}_A(x) - \text{rank}_B(x)| + |A \setminus B| \cdot \mu_1 + |B \setminus A| \cdot \mu_2 \tag{4.4}$$

Hallucination Rate ($h_{a_i,t} \in [-1,0]$) While the Hallucination Identification stage (Subsubsection 4.1.3) provides an initial check and asks the agent to replace candidates that are either (i) out-of-catalog or (ii) rejected, the LLMs can still fail to obey. In order to account for this scenario and penalize an agent that continues to hallucinate even beyond the initial check, we introduce the *Hallucination Rate*, or $h_{a_i,t}$, as a penalty during the assessment phase. We define $h_{a_i,t}$ as the negated hit rate between the agent's candidate list $L_{a_i,t}$ and the available cities ($\mathcal{C} \setminus \phi'_t$) in the catalog.

$$h_{a_i,t} = 1 - \left(\frac{1}{K}\sum_{k=1}^{K} \mathbb{I}\left[(L_{a_i,t})_k \in (\mathcal{C} \setminus \phi'_t)\right]\right) \tag{4.5}$$

Evaluation Score We compute the evaluation score (Equation 4.6), $s(c,t)$, for each recommended city c for round t as the sum of the *agent success* ($r_{a_i,t}$) and *agent reliability* ($d_{a_i,t}$) negated by *agent hallucination* ($h_{a_i,t}$) for the current round added to the evaluation score from the previous round $t-1$.

$$s(c,t) = s(c,t-1) + \sum_{a_i \in \mathcal{A}} \left(\frac{1}{rank(c_{a_i})} \cdot (-h_{a_i,t} + r_{a_i,t} + d_{a_i,t})\right) \tag{4.6}$$

The new collective offer, ϕ_t for round t, is the top-k cities, ranked according to their corresponding normalized evaluation scores.

Termination Criteria & Complexities

The termination criteria for agent interactions are defined in two ways. First, we introduce a metric-based condition termed *Moderator Success*, computed analogously to *Agent Success* (Subsection 4.1.4) but evaluated over the collective offer with respect to all travel filters, rather than only the filters relevant to a specific agent. If the *moderator success* score reaches its maximum value of 1 or achieves an improvement of τ% over the score at round 0, the process is terminated via *early stopping*. However, since this ideal score may not always be attainable, we enforce a maximum of T interaction rounds to ensure termination and avoid potential infinite loops. Additionally, we enforce a minimum number of conversation rounds (set empirically at 5 rounds) to ensure interaction between agents and avoid premature termination.

4.2. Experimental Setup

4.2.1. Setup

Dataset - SynthTRIPS

To evaluate Collab-Rec, we use a *stratified sample* of 45 queries from the SynthTRIPS dataset [1] (see Chapter 3). We specifically select queries across three *popularity* levels (low, medium, and high) and three *complexity* tiers (*medium*, *hard*, and *sustainable*), yielding 5 queries per (popularity, complexity) combination. The chosen 45 queries reflect tasks such as *"Plan a budget-friendly 3-day trip to a less crowded coastal city in Europe"* or *"Suggest a moderately priced metropolis known for art galleries,"* each containing explicit filters (e.g., budget, month) and implicit constraints (e.g., local culture, sustainability). Overall, this yields a balanced set of user prompts with realistic constraints (e.g., budget, travel dates, sustainability concerns) while controlling computational overhead. Notably, SynthTRIPS queries stem from large language models themselves, but we exclude those generated by *Gemini* (one of our tested LLMs) to avoid any overfitting or bias in evaluating our approach.

External Knowledge Base

To validate agent outputs and detect hallucinations, Collab-Rec leverages the SynthTRIPS **knowledge base (KB)** [1], containing 200 European cities. Each city has attributes relevant to *popularity, budget, seasonality,* and *sustainability*, among others. During each negotiation round, the moderator uses this KB to:

1. *Compute Agent Success* ($r_{a_i,t}$) by matching filters against the metadata of the city,
2. *Identify Hallucinations* if an agent recommends a city not found in the KB or already rejected in a prior round.

4.2.2. Experimental Settings

Implementation Details

We focus primarily on the **Multi-Agent Multi-Iteration (MAMI)** setup, running it on all 45 queries. The three specialized agents (Popularity, Sustainability, Personalization) each use the *same* LLM backbone for fairness. Negotiation proceeds up to $T = 10$ rounds (unless early-stopped). In each round t:

- Each agent proposes a *top-k* list of cities ($k = 10$), with newly introduced or replaced items clearly justified.
- The moderator detects and attempts to correct any *hallucinated* city by prompting the agent for a valid alternative.
- Final proposals are scored and combined into a *Collective Offer*, which is returned to all agents to guide the next round.

Initialization. At round 0, we assign each agent the *ideal* starting values of $d_{a_i,t} = 1$, $h_{a_i,t} = 0$, and no penalty from previous offers. In practice, the first actual proposals from the agents are generated in round 1, at which point their real reliability and hallucination rates begin to diverge.

Baselines: SASI and MASI

To highlight the effect of multi-round interaction, we compare MAMI against two simpler baselines:

- **SASI (Single-Agent Single-Iteration):** A single LLM is prompted with the entire query (including filters). It returns a ranked list of $k = 10$ cities in one shot, without any negotiation.
- **MASI (Multi-Agent Single-Iteration):** All three agents produce their initial $k = 10$ proposals. The moderator fuses them (after a single check for hallucinations) into a final recommendation, with *no iterative refinement*.

Models

All experiments use two *reasoning* LLMs:

1. ***gpt-o4-mini*** [113]
2. ***gemini-2.5-flash*** [114]

We also tested non-reasoning variants (*claude-3.5-sonnet*, *gemini-2.0-flash*, *deepseek-chat-v3*) but found they consistently ignored feedback and failed to adapt across rounds. Hence, we exclude them from final reporting.

4.2.3. Evaluation Metrics

We measure performance from two perspectives:

1. **Final Recommendation Quality**
   - *Relevance.* We capture how well the *final* Collective Offer matches all user filters. Concretely, we define *Moderator Success* as an analog to Agent Success, but scored over *all* user constraints in the final offer. A score of 1 indicates that every recommended city fully matches the user filters.
   - *Diversity.* To assess whether the system avoids over-concentrating on top-tier tourist hubs, we compute the **GINI Index** [115] (lower = more evenly distributed) and **Normalized Entropy** [116] (higher = more variety) over the final recommended set of cities.
2. **Agent Behavior** (per round)
   - *Reliability* ($d_{a_i,t}$) measures how much each agent's candidate list changes from one round to the next.
   - *Hallucination Rate* ($h_{a_i,t}$) is a negative penalty in $[-1, 0]$ that tracks out-of-catalog or previously rejected cities that persist despite the moderator's warnings.

Summary of Experimental Goals. Ultimately, building on Section 4.1, we seek to answer four key research questions:

RQ1: Do multiple agents and multiple rounds (**MAMI**) improve *final* recommendation quality over single-agent (**SASI**) or single-round (**MASI**) approaches?

RQ2: Does **MAMI** help reduce *popularity bias* and increase coverage of lesser-known destinations?

RQ3: How do the specialized agents evolve *internally* across repeated negotiation, in terms of reliability and hallucinations?

RQ4: What time and cost overheads does **MAMI** introduce, and how might these overheads be mitigated?

In the next Section 4.3, we evaluate the outcomes of these experiments, discussing system-level impact (RQ1, RQ2) as well as agent behavior (RQ3) and computational costs (RQ4).

4.3. Results & Discussion

At a high level, throughout the subsequent RQ analysis, we aim to investigate whether enabling multiple agents to negotiate over several rounds (**MAMI**) yields tangible benefits compared to:

- Multi-agent single-iteration (**MASI**), and

- Single-agent single-iteration (**SASI**).

We evaluate these three configurations (**MAMI**, **MASI**, **SASI**) using a suite of 45 representative queries, stratified equally across three popularity levels—low, medium, and high (each with 15 representative queries). These queries encompass various user constraints (e.g., budget, travel dates), popularity preferences, and sustainability concerns. To ensure robust analysis, we employ **statistical significance testing** with multiple comparison tests across the distributions of results obtained from the queries.

4.3.1. System-Level Impact

First, we begin by analyzing **RQ1:** *"Do Multiple Agents and Multiple Rounds Improve Outcomes?"*, where success is measured in terms of the overall effectiveness of the collective offer (i.e., how well the combined recommendations meet user constraints and agent-specific objectives).

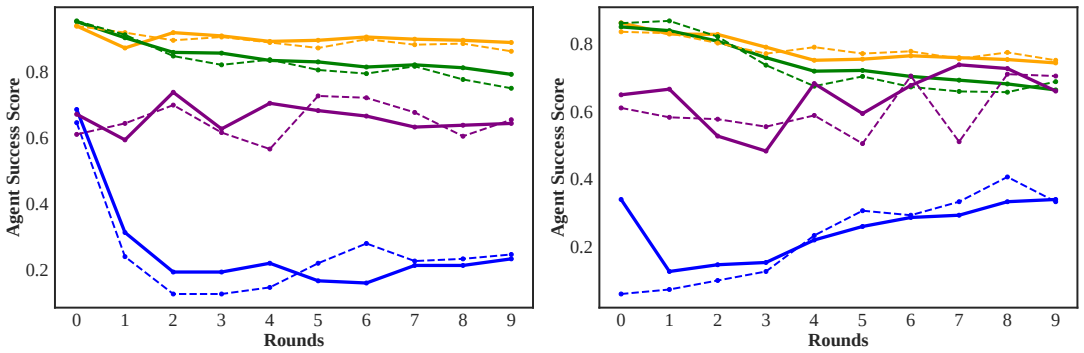
Dominance & Negotiation Dynamics. Figure 4.3 shows each agent's *success score* across negotiation rounds, split into three query types: (a) Low popularity, (b) Medium popularity, and (c) High popularity. Here, a higher success score means that the agent's proposed cities more closely satisfy both user constraints (e.g., budget, travel time) *and* the agent's own objectives (e.g., popularity, sustainability).

We see that for **high-popularity** queries (panel c), the Popularity agent tends to dominate early (blue lines climbing above 0.95 by round 3). For example, if a user specifically asks about "Paris" or "Rome," the Popularity agent can easily fulfill the constraints using abundant travel data. Conversely, for **low-popularity** queries (panel a), we observe that the Popularity agent struggles (success score often below 0.3 in the first few rounds), while the Sustainability and Personalization agents maintain higher scores ($\sim$0.7–0.8). This dynamic suggests that multi-round negotiation allows each agent to "shine" in scenarios where it is most relevant. Ultimately, the collective offer (purple line) reflects a compromise of all three agents' proposals, typically converging around 0.7–0.8 after several rounds.

Beyond these per-agent curves, Table 4.1 and Table 4.2 summarize overall system-level outcomes:

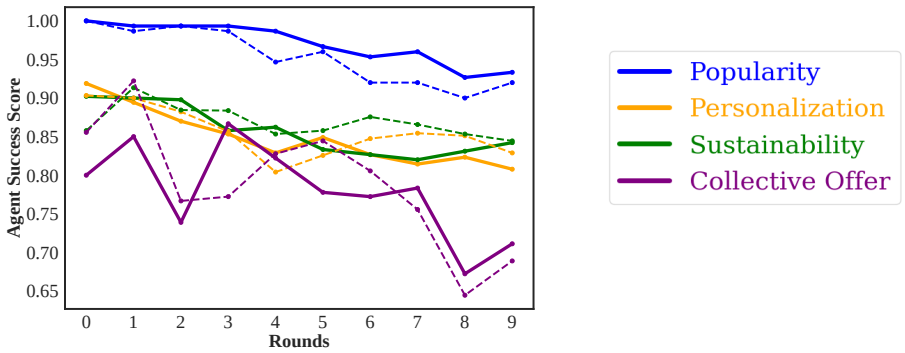
- **Table 4.1** reports the *final* collective-offer success scores under the three system configurations — **SASI**, **MASI**, and **MAMI**. We find that **MAMI** consistently achieves higher scores (around 0.75–0.80) compared to **SASI** ($\sim$ 0.55–0.60) and **MASI** ($\sim$ 0.65–0.70) across all popularity levels.

- **Table 4.2** shows the results of pairwise *statistical significance* tests (ANOVA with post-hoc Bonferroni correction). We see that **MAMI** outperforms both **SASI** and **MASI** with p-values under 0.01, confirming that *multiple agents plus multiple rounds* lead to significantly better final recommendations. Meanwhile, there is no statistically significant difference between **SASI** and **MASI** (corrected p-value > 0.05), suggesting that *merely adding agents without iterative negotiation* brings only limited gains.



(a) Low

(b) Medium



(c) High

Figure 4.3.: Agent Dominance when split by the popularity levels of the queries. Sustainability and personalization agents tend to dominate for medium and low popularity queries, but the popularity agent takes a substantial lead for high popularity queries, signaling a potential popularity bias inherent in pre-trained models. —— (dotted line) represents the results from *gemini-2.5-flash* while — (continuous line) represents the results from *gpt-o4-mini* .

From a practical perspective, these findings indicate that *repeated refinement* across agents —where each agent proposes, critiques, and updates its suggestions — is key to generating higher-quality city recommendations. For instance, in one sample query:

> *"Find me a mid-sized European city that's child-friendly and not too expensive."*

SASI often returns well-known but moderately priced capitals (e.g., *Budapest*). **MASI** introduces some variety but is still skewed toward popular tourist centers. In contrast, the multi-agent *multi-round* (**MAMI**) negotiation yields final lists with additional mid-tier cities that are more aligned to the user's constraints (e.g., *Ghent*, *Liège*), demonstrating the benefits of iterative agent interplay.

Answer to RQ1. *Multiple agents and multiple negotiation rounds (**MAMI**) outperform single-agent or single-round baselines in terms of final success scores, as shown by Table 4.1 and significance tests in Table 4.2. The iterative interaction allows each agent to better address the user's constraints and prevents any single agent (e.g., Popularity) from dominating unless truly appropriate.*

Table 4.1.: Overall system performance (*Moderator Success* ↑) and comparison with baselines at various early stopping thresholds. Each entry is denoted as v, where v represents the moderator success value. The values in bold denote significant differences to SASI, while values in **bold blue** denote significant differences between SASI and MASI. The significance test is conducted using multiple comparison t-tests with Bonferroni correction $\alpha = 0.017$. Early stopping threshold `None` indicates that the negotiation continued until $T = 10$ for MAMI and $T = 1$ for MASI and SASI.

Model	Rejection Strategy	Approach				
		SASI	**MASI**	**MAMI (20%)**	**MAMI (60%)**	**MAMI (None)**
GPT4	A	0.237	0.707	**0.802**	**0.807**	**0.672**
	M	-	0.717	**0.811**	**0.772**	**0.672**
Gemini	A	0.727	0.693	**0.859**	**0.843**	0.683
	M	-	0.730	0.783	0.772	0.685

4.3.2. The Impact of Negotiation on Popularity Bias and Diversification

We now specifically focus on how well each approach (**SASI**, **MASI**, **MAMI**) balances *popularity* versus *lesser-known* destinations. While Subsection 4.3.1 established that multi-round negotiation improves overall outcomes, here we examine **RQ2**: *which cities are ultimately recommended and does agent interplay curb the tendency to over-rely on famous locales?*

Table 4.2.: Pairwise statistical comparison results for *gemini-2.5-flash* at $\tau = 20\%$. The corrected p-values are adjusted using Bonferroni correction $\alpha = 0.017$.

Group 1	Group 2	Stat.	p-value	Corrected p-value	Reject H_0
MAMI	MASI	3.6661	0.0004	0.0013	True
MAMI	SASI	3.1637	0.0021	0.0064	True
MASI	SASI	-0.7555	0.452	1.0000	False

Mitigating Popularity Bias. Figure 4.4 and Table 4.3 (GINI and Entropy metrics) reveal that:

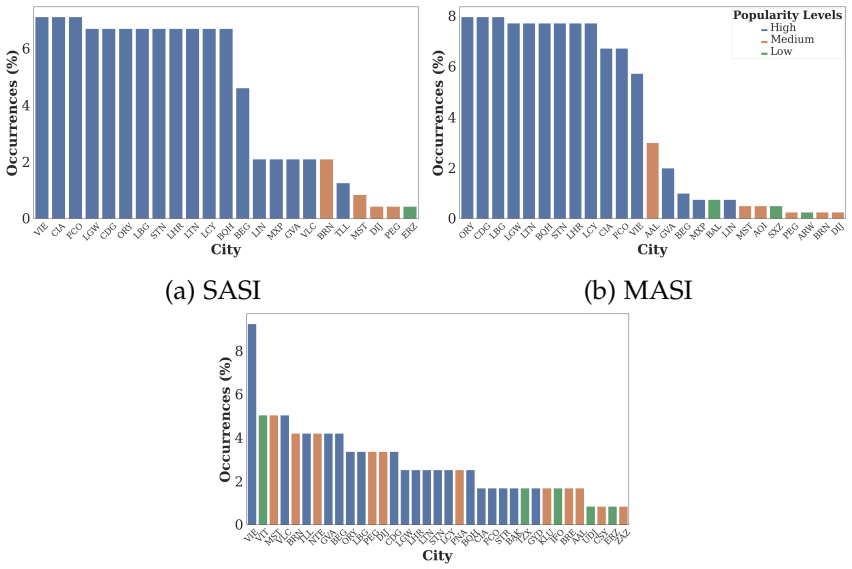
- **SASI** skews heavily towards well-known hubs such as *Paris*, *Barcelona*, or *Berlin*. For instance, across low-popularity queries, **SASI** re-suggests popular capitals in roughly 60% of its final outputs.
- **MASI** improves slightly but still exhibits notable concentration on top-tier tourist cities (GINI around 0.44–0.50).
- **MAMI** shows the most diverse outcomes: GINI drops to as low as 0.28 (and normalized entropy rises above 0.80) when the moderator enforces iterative corrections. Concretely, **MAMI** frequently surfaces smaller or mid-sized European destinations (e.g., *Trento*, *Malaga*, and *Cluj-Napoca*) that were almost never mentioned in **SASI's** final lists.

For a practical example, suppose a user requests *"an affordable coastal city in Europe, less crowded, with strong local culture." Single-agent* systems often default to major (cheaper) coastal spots like *Sofia* or *Belgrade*. However, *multi-agent multi-round* negotiation lets the *Sustainability* and *Personalization* agents argue for under-the-radar sites (e.g., *Thessaloniki*, *Varna*). By round 3 or 4, the *Popularity* agent partially concedes to these suggestions, so that the final recommendation set incorporates both recognized cities and lesser-known coastal gems.

Answer to RQ2. *Multi-agent negotiation demonstrably reduces popularity bias and boosts recommendation diversity. Successive rounds of agent interplay ensure popular cities do not automatically dominate when less-mainstream destinations better match user constraints. Measured by GINI & Entropy (Table 4.3),* ***MAMI*** *achieves significantly broader coverage across popularity tiers, addressing a key concern in travel recommender systems.*

4.3.3. Agent Reliability and Hallucinations

Having established the advantages of our proposed multi-agent, multi-round setup (**MAMI**) — notably in terms of improved personalization and greater recommendation diversity, we now turn our attention to less-explored aspects of agentic recommender systems. In particular, we



(a) SASI

(b) MASI

(c) MAMI

Figure 4.4.: City Distributions when split by the popularity levels of the recommended cities for 50 randomly sampled cities. For brevity, the x-axis represents the IATA codes of the respective cities. MAMI tends to provide lesser-known cities as a recommendation when compared to SASI and MASI.

seek to answer **RQ3**: *How do specialized agents behave over multiple negotiation rounds, especially with respect to their reliability and susceptibility to hallucination?*

Reliability Trends. Figure 4.5 (figures **(a)** and **(b)**) compares the *reliability score* of each agent over multiple rounds, under both Aggressive and Majority voting strategies. Recall that reliability indicates how consistently city proposals by an agent follow its own prior round, i.e., how *stable* its suggestions remain once it receives feedback. We initialize the reliability of each agent at 1.0 (round 0), then observe an immediate drop in round 1—- (red circle in Figure 4.5 top row) once negotiation begins and the collective offer by the moderator forces agents to reassess their candidates. In subsequent rounds, reliability steadily *increases* because the agents converge on stable, negotiation-driven recommendations. Majority voting (right side) tends to yield higher final reliability ($\gtrsim 0.8$) than Aggressive rejection, which is stricter and drives more frequent changes in agent proposals.

Hallucination Rate. Figure 4.5 (figures **(c)** and **(d)**) tracks each *hallucination rate* by each agent — the fraction of city proposals that are invalid (not in the knowledge catalog) or incorrectly grounded. Although the moderator attempts to correct such invalid suggestions, the agents can still hallucinate. Overall, we see modest improvements over the rounds: e.g., the hallucination rate for *gpt-4o* under Aggressive rejection drops from $\sim$0.25 in round 1 to $\sim$0.15 in round 6. Meanwhile, the Majority approach usually shows lower hallucination overall, since it constrains the agents less harshly, making it easier for them to refine — rather than replace — their candidate lists.

Table 4.3.: GINI Index and Normalized Entropy of final candidates across popularity. The values in **bold** denote the optimum values for each model across all settings - for GINI (Entropy), the optimum value is the minimum (maximum).

Model	Rejection Strategy	Scoring	Approach				
			SASI	MASI	MAMI (20%)	MAMI (60%)	MAMI (None)
GPT4	A	GINI ↓	0.548	0.441	**0.329**	0.367	0.367
		Entropy ↑	0.451	0.634	**0.823**	0.732	0.732
	M	GINI ↓	0.548	0.505	**0.427**	0.447	0.447
		Entropy ↑	0.451	0.534	**0.692**	0.732	0.732
Gemini	A	GINI ↓	0.553	0.495	**0.287**	0.342	0.342
		Entropy ↑	0.431	0.548	**0.865**	0.761	0.761
	M	GINI ↓	0.553	0.490	0.372	0.372	**0.355**
		Entropy ↑	0.431	0.546	0.764	0.764	**0.782**

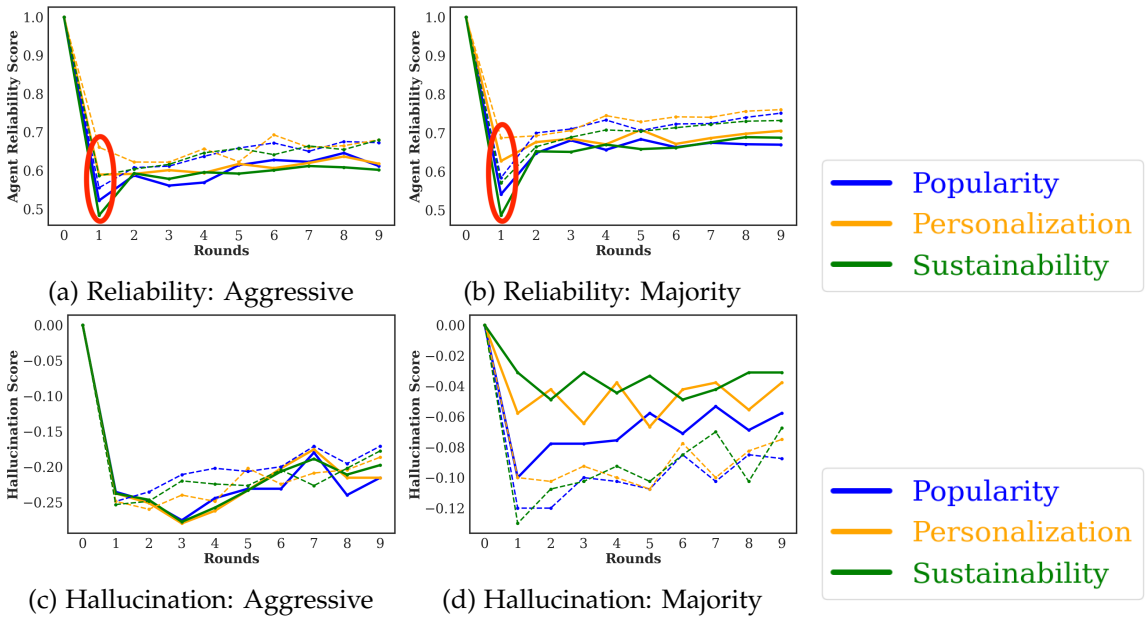
For example, when asked to recommend cultural city hidden gems, the *Sustainability* agent suggests *Poznań*. While this is a suitable choice, it is not present in the item catalog. The moderator promptly flags it as invalid, prompting the agent to adhere better to the instruction. In the next round, the *Sustainability* agent proposes *Košice*, a similarly sustainable city that is however, included in the context. These feedback loops help prevent out-of-catalog recommendations while ensuring a comparable alternative is suggested.

Answer to RQ3. *Agents become more reliable and gradually reduce hallucinations across multiple negotiation rounds. The iterative moderator feedback and scoring scheme effectively penalizes invalid proposals, guiding all agents toward more stable, factually valid cities.*

4.3.4. Time & Cost Complexity of Multi-Agent Negotiation

Having analyzed the qualitative benefits of multi-agent negotiation, we now turn to its practical implications. Specifically, in this RQ we ask: **RQ4:** *How does the multi-round, multi-agent negotiation protocol (**MAMI**) affect inference time and token usage compared to single-round baselines?*

Increased Overhead. Figure 4.6 shows the average time per round for *gpt-o4-mini* and *gemini-2.5-flash* under Aggressive rejection. By round 10, the system takes over 100 seconds per round for each query (totaling $\sim$1000 seconds per query), as each of the three agents issues an updated set of proposals and the moderator processes them. Over 45 queries, **MAMI** consumes about 7.4 million tokens and 2700 API calls per model, nearly 60$\times$ the cost of **SASI** — which calls a single agent in a single iteration.



(a) Reliability: Aggressive (b) Reliability: Majority

(c) Hallucination: Aggressive (d) Hallucination: Majority

Figure 4.5.: Agent Behavior metrics showing the agents' reliability score and hallucination rate over multiple rounds. —— (dotted line) represents the results from *gemini-2.5-flash* while — (continuous line) represents the results from *gpt-o4-mini* .

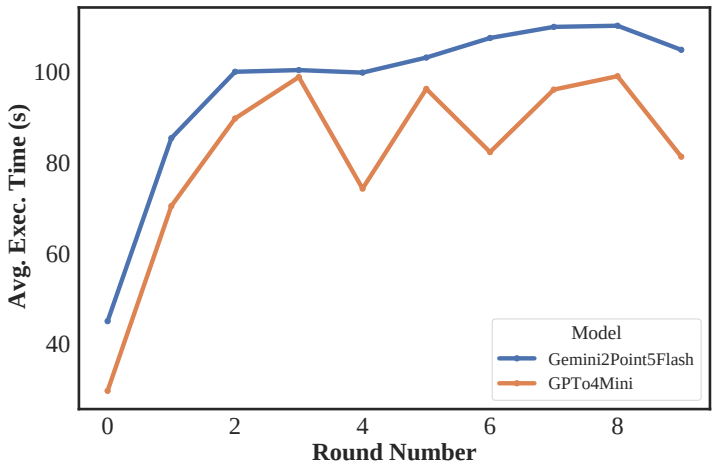


Figure 4.6.: Average time taken for Collab-Rec for 10 rounds for two models using Aggressive strategy.

Trade-Off: Quality vs. Cost. MASI partially mitigates this overhead but lacks the iterative refinement that boosts relevance and diversity. Meanwhile, **MAMI** achieves superior recommendation quality at the expense of lengthy run times and higher token usage. This underscores a practical tension in real-world deployment: iterative negotiation fosters better results but raises concerns over latency and carbon footprint. Methods like early stopping,

caching, or agent pruning (e.g., removing an agent once its score stabilizes) could alleviate these costs in future work.

Answer to RQ4. *Multi-round approach by **MAMI** substantially outperforms single-round baselines at the cost of higher computational overhead. For large-scale or real-time systems, this trade-off may require optimizations (e.g., early stopping or partial agent involvement) to balance quality with efficiency.*

5. Conclusion

As the preferences and expectations of travelers become increasingly diverse and context-dependent, the need for personalized, adaptive, and conversational recommendation systems has grown significantly. This thesis examines the evolving landscape of Tourism Recommender Systems (TRS), with a particular emphasis on leveraging the natural language understanding and generation capabilities of Large Language Models (LLMs) to address persistent challenges in the field, for example, the dearth of natural language datasets that can be used to train and evaluate these systems. Additionally, we investigate the potential of LLM-based recommenders, or RecLLMs, to balance personalization and diversity in travel recommendations in an effort to address the issue of overtourism. By explicitly incorporating sustainability-related factors such as AQI and walkability into the query generation and recommendation process, this work also explores another dimension in TRS research: the broader societal and environmental impacts of tourism.

Specifically, this work addresses two key research questions: (i) how effectively can LLMs function as user simulators when grounded in travel-related context, and (ii) whether a multi-agent framework can achieve a more effective balance between relevance and diversity of recommendations compared to a single RecLLM.

Chapter 3 introduces SynthTRIPS, a novel framework leveraging LLMs and a curated knowledge base for generating synthetic travel queries, addressing the need for more personalized and sustainable travel recommendation datasets. Through persona-based preferences, sustainability filters, and grounding techniques, SynthTRIPS effectively creates realistic and diverse queries, as validated by human experts and automatic LLM-based assessments. The framework's reproducible pipeline, open-source resources, and flexible design offer valuable tools for researchers in the tourism recommendation community.

Furthermore, to reduce popularity bias in RecLLMs, as observed during the preliminary analysis done with SynthTRIPS, we introduce Collab-Rec in Chapter 4, a multi-agent LLM-based framework that balances different stakeholder objectives, combining personalization, sustainability, and popularity perspectives through iterative agent negotiation. Our results show that Collab-Rec improves relevance and diversity over single-agent baselines while reducing hallucinations via grounded moderation. Further detailed analysis shows that the agents become more reliable over multiple iterations while the moderator's fusion mechanism can effectively penalize hallucinations, increasing the system's stability.

5.1. Limitations & Future Scope

However, there are limitations with both SynthTRIPS and Collab-Rec. Both rely on a fixed city catalog, restricting adaptability to emerging destinations. With the former, there is a potential for subjective interpretation of personas and the observed trade-off between personalization and groundedness, where highly personalized queries may exhibit lower factual accuracy or alignment with their constraints. Furthermore, the personas in SynthTRIPS are not always domain-aligned. Since PersonaHub consists of synthetically generated data, it often lacks diversity and realism. While clustering them by topic marginally improves diversity, the issue of unrealism in the personas persists and remains evident in our dataset.

Future work could extend SynthTRIPS by incorporating more user filters (e.g., traveling with families, solo travels, accessibility, etc.), expanding beyond European cities to enhance global applicability, and developing strategies to maintain the knowledge base as real-world travel data evolves. Concerning the evaluation, comparing the synthetic queries with manually generated user queries would provide a more comprehensive view of whether LLMs can truly mimic human behavior.

While Collab-Rec attempts to address the popularity bias observed during the preliminary RecLLM analysis with SynthTRIPS, the multi-agent framework has its own limitations. Although agent specialization improves balance, popularity bias resurfaces when users request highly popular destinations. Future work could extend the number of negotiation rounds to study longer-term convergence trends and introduce sampling strategies to reduce the candidate pool presented to agents. This may help lower hallucination rates by narrowing the decision space, especially in high-complexity scenarios. Despite these challenges, Collab-Rec offers a promising step toward more balanced recommendation systems and highlights the potential of collaborative LLM agents in complex decision-making tasks.

Environmental Impact: Despite the growth of LLMs and their proven efficiency at various tasks, they also impose significant environmental costs, particularly regarding energy consumption and carbon emissions. Jegham, Abdelatti, Elmoubarki, and Hendawi [117] benchmark the water, energy, and carbon footprints of various LLMs, highlighting that as AI systems become more accessible and efficient, the strain they cause on the environment continues to grow. Consequently, although promoting sustainable travel through recommendations is a valuable objective, doing so with RecLLMs may offset the intended benefits by causing an environmental burden from excessive use of computational resources. Therefore, identifying and addressing the unintended environmental costs of LLM-based systems is essential, particularly when the overarching goal is to promote sustainability.

A. Appendix

A.1. Prompts

This appendix contains the prompts used for the models for synthetic query generation in SynthTRIPS (Chapter 3) and for each agent in Collab-Rec (Chapter 4).

A.1.1. SynthTRIPS Prompts

Query Generation

System Prompt

```
You are a query generator for a travel recommender system for European city
trips.
You are provided with a list of cities, some information about them
(context), and certain travel filters (for example, regarding the budget
or points of interest to visit).
You should include this information in your response to ensure a city
travel recommender system suggests the provided cities as an answer to your
generated query (response).
You should not include the names of the cities in the response.  The query
should not exceed 200 characters and should be only relevant for European
cities.
```

User Prompt - Vanilla Query

```
Here are the following for the new user:
City Information:  {{ new_subgraph }}
Travel filters:  {{ new_travel_filters }}

Your task:  Guess a query that a user may ask you to generate:
```

User Prompt - Personalized with Zero-Shot

```
Here are the following for the user who is a {{ persona }}:
City Information: {{ new_subgraph }}
Travel filters: {{ new_travel_filters }}

Your task: Guess a query that the following person may ask you to generate:
{{ persona }}
```

User Prompt - Personalized with In Context Learning

```
Here is an example of a good query for a user who is a {{ sample_persona }},
the provided city information and associated travel filters.
City Information: {{ sample_subgraph }}
Travel filters: {{ sample_travel_filters }}
Generated Query: {{ sample_generated_query }}

Here are the following for the new user who is a {{ persona }}:
City Information: {{ new_subgraph }}
Travel filters: {{ new_travel_filters }}

Your task: Guess a query, similar to the example above, that the following
person may ask you to generate: {{ persona }}
```

JudgeLLM

System Prompt - Persona Alignment

```
You will be given a user and a travel query. Your task is to provide a
'rating' scoring how likely the user will ask the travel query.
You have three options:
A: Not Aligned - The user is not likely at all to ask this query.
B: Partially Aligned - The user is quite likely to ask this query.
C: Aligned - The user is very likely to ask this query.
D: Unclear - It is unclear whether the user will ask this query.

Provide your feedback as follows:

Feedback:::
Evaluation: (your rationale for the rating, as a text)
Rating: (the option you chose)

You MUST provide values for 'Evaluation:' and 'Rating:' in your answer.
```

User Prompt - Persona Alignment

```
Now here are the user and query.

User: {{persona}}
Travel Query: {{query}}

Provide your feedback.
Feedback:::
Evaluation:
```

System Prompt - Filter Groundedness

```
You are given a query and a list of filters. Your task is to identify how
many filters are present in the query. Only return the number of matches
and your explanation for this in the following format:

Matches: (number of matches found)
Explanation: (which parts of the query matched with the filters)

If there are no matches, return 0.
------

Here is an example of filters and a query that contains all the filters:
Query: {{good_query}}
Filters: {{filters}}
In this example, the result would be 4 matches, as all four filters are
present in the query.

Here is another example of a query which does not contain all the filters:
Query: {{bad_query}}
Filters: {{filters}}

In this case, popularity is missing from the query. Therefore the number
of matches returned should be 3.
```

User Prompt - Filter Groundedness

```
Here is your query and list of filters:

Query: {{query}}
Filters: {{filters}}

How many matches do you find?
```

System Prompt - Filter Groundedness (Decomposition)

```
Decompose and break down each of the input sentences into one or more
standalone statements. Each statement should be a standalone claim that
can be independently verified. Follow the level of atomicity and coverage
as shown in the example. Return only a list of the decomposed sentences.
----
Example:
Sentence: {{sample_query}}
Result: {{sample_result}}
```

User Prompt - Filter Groundedness (Decomposition)

```
Sentence: {{query}}
Result:
```

System Prompt - Filter Groundedness (Faithfulness)

```
You are given a list of independent sentences called Queries and a list of
filters called Filters.
Your task is to match each sentence in Queries with a filter in Filters.
Each sentence in Queries can only be mapped once and only to one filter in
Filters.
Return your result in the format: ‘‘sentence: filter’’. Only return the
result, do not return any other text with it.
If there is no match, return ‘‘None’’ for that sentence.

Here’s an example:

Queries: {{sample_results}}
Filters: {{filters}}
Example result: {{mapping}}
```

User Prompt - Filter Groundedness (Faithfulness)

```
Here are your lists of sentences and filters.
Queries: {{query_decomposed_list}}
Filters: {{filter_list}}

What is your result?
```

A.1.2. Collab-Rec Prompts

Single Agent

System Prompt

```
You are a travel recommender system. Your objective is to consider the
user's query and provide the top {{k_offer}} cities that best match the
user's preferences, out of the list of 200 cities provided to you. Make
sure you consider ALL of the user's preferences mentioned in the query.

Here's the list of available cities: {{cities}}

Your only output should be an ORDERED list of {{k_offer}} cities as a comma
separated list in the JSON format:

{
"City" : [your list of cities]
"Reasoning": you reasoning for your recommendation in 4-5 sentences
}
```

User Prompt

```
Here is the user's query: {{query}}

What is your recommendation?
```

Multi Agent

System Prompt - Initialize (MASI)

```
You are a {{role}} for a travel recommender system, advocating for
{{filter_type}} preferences.
{{filter_context}}
Given a user query, your objective is to ONLY consider the user's
{{filter_type}} preferences and provide the top {{k_offer}} cities that
best match the preferences, out of the list of 200 cities provided to you.
Here's the list of available cities: {{cities}}

Your only output should be an ORDERED list of {{k_offer}} cities as a comma
separated list in valid JSON format:

{
"City" : [your list of cities]
}
```

User Prompt - Initialize (MASI)

```
Here is the user's query: {{query}}
Given this {{filter_type}} preference of the user: {{filters}}

What is your recommendation?
```

System Prompt - Negotiate (MAMI)

```
You are **Relevancy-Ranker-Agent**, collaborating with another agent to
refine a list of ten city recommendations.

Mission
- Produce an **ordered top-{{k_offer}} list** of unique cities that best
satisfy the user's preferences.
- Start from the union of:  Current Offer + Other Candidate Options.
- Return your list and reasoning in valid JSON.

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}}.  {{filter_context}}

3.  Pick the ten highest-scoring cities **while honoring Rule 1**.
4.  Sort the chosen ten from highest to lowest score.

Output format (exactly):
{
"City":  ["city 1", ..., "city 10"],
"Scores":  [97, ..., 70], # same order as "City"
"Reasoning":  "concise justification of swaps, scores, and ordering"
}

Policy
- Never invent cities outside the provided candidate pools.
- The recommended list must contain 10 unique cities.
- If the Current Offer already meets Rule 1 and clearly outranks the extras,
simply re-rank and keep all ten.
- Do **not** reveal internal scores or Rule 1 in `Reasoning`; keep it
user-friendly.
```

User Prompt - Negotiate (MAMI)

```
USER QUERY: {{query}}

USER PREFERENCES: {{filters}}

ROUND CONTEXT:
• Previous Recommendation: {{prev_output}}
• Current Offer: {{curr_offer}}
• Other Candidate Options: {{candidate_cities}}
• Feedback (optional): {{agent_feedback}}
What is your recommendation? Return in the valid JSON format mentioned in
the instructions.

TASK:
Rank the union of **Current Offer** and **Other Candidate Options**
according to the System-Prompt rules.
Return ONLY the JSON object in the required format (City, Scores,
Reasoning).
```

System Prompt - Initial Feedback

```
You are a {{role}} for a travel recommender system, advocating for
{{filter_type}} preferences.
{{filter_context}} Your objective is to consider the user's query and their
{{filter_type}} preferences and provide the top {{k_offer}} cities that best
match the preferences, out of the available options provided to you.
In your earlier recommendation, you recommended {{k_invalid}} invalid
candidates out of {{k_offer}}. Please substitute ONLY the invalid
candidates with other available choices. Do not replace any of the valid
candidates. Note that if you further recommend an invalid candidate, it
will negatively impact the user's trust in you.

Your only output should be a dictionary containing the substitution for
each invalid city in the following valid format:

{
''<invalid city name>'' : ''<your subtitution>''
}
```

User Prompt - Initial Feedback

```
Here is the user's query: {{query}}
Given this {{filter_type}} preference of the user: {{filters}}
Your earlier recommendation: {{valid_candidates}}
Invalid candidates: {{invalid_candidates}}

Available options: {{available_candidates}}

Please provide your substitutions, in the instructed format.
```

A.2. Generative AI Usage Disclosure

We used ChatGPT (OpenAI) for code snippet suggestions during the development of this work. We also used Grammarly to check for grammar inconsistencies in the thesis and to polish the text for better clarity. We have critically reviewed and revised all GenAI outputs to ensure that accuracy and originality are maintained, and we accept full responsibility for the content presented in this draft.

List of Figures

List of Tables

Bibliography

[1] A. Banerjee, A. Satish, F. N. Aisyah, W. Wörndl, and Y. Deldjoo. "SynthTRIPs: A Knowledge-Grounded Framework for Benchmark Query Generation for Personalized Tourism Recommenders". In: *arXiv preprint arXiv:2504.09277* (2025).

[2] F. O. Isinkaye, Y. O. Folajimi, and B. A. Ojokoh. "Recommendation systems: Principles, methods and evaluation". In: *Egyptian informatics journal* 16.3 (2015), pp. 261–273.

[3] Y. Takeuchi and M. Sugimoto. "CityVoyager: An outdoor recommendation system based on user location history". In: *International Conference on Ubiquitous Intelligence and Computing*. Springer. 2006, pp. 625–636.

[4] K. H. Lim, J. Chan, C. Leckie, and S. Karunasekera. "Personalized trip recommendation for tourists based on user interests, points of interest visit durations and visit recency". In: *Knowledge and Information Systems* 54 (2018), pp. 375–406.

[5] E. H.-C. Lu, C.-Y. Chen, and V. S. Tseng. "Personalized trip recommendation with multiple constraints by mining user check-in behaviors". In: *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*. 2012, pp. 209–218.

[6] A. Majid, L. Chen, G. Chen, H. T. Mirza, I. Hussain, and J. Woodward. "A context-aware personalized travel recommendation system based on geotagged social media data mining". In: *International Journal of Geographical Information Science* 27.4 (2013), pp. 662–684.

[7] A.-J. Cheng, Y.-Y. Chen, Y.-T. Huang, W. H. Hsu, and H.-Y. M. Liao. "Personalized travel recommendation by mining people attributes from community-contributed photos". In: *Proceedings of the 19th ACM international conference on Multimedia*. 2011, pp. 83–92.

[8] S. Dai, N. Shao, H. Zhao, W. Yu, Z. Si, C. Xu, Z. Sun, X. Zhang, and J. Xu. "Uncovering ChatGPT's Capabilities in Recommender Systems". In: *Proceedings of the 17th ACM Conference on Recommender Systems*. RecSys '23. New York, NY, USA: Association for Computing Machinery, Sept. 2023, pp. 1126–1132. ISBN: 9798400702419. DOI: 10.1145/3604915.3610646. URL: https://doi.org/10.1145/3604915.3610646 (visited on 04/15/2024).

[9] J. Lin, X. Dai, Y. Xi, W. Liu, B. Chen, H. Zhang, Y. Liu, C. Wu, X. Li, C. Zhu, et al. "How can recommender systems benefit from large language models: A survey". In: *arXiv preprint arXiv:2306.05817* (2023).

[10] A. Banerjee, P. Banik, and W. Wörndl. "A review on individual and multistakeholder fairness in tourism recommender systems". en. In: *Frontiers in Big Data* 6 (May 2023), p. 1168692. ISSN: 2624-909X. DOI: 10.3389/fdata.2023.1168692. URL: https://www.frontiersin.org/articles/10.3389/fdata.2023.1168692/full (visited on 04/15/2024).

[11] A. Banerjee, T. Mahmudov, E. Adler, F. N. Aisyah, and W. Wörndl. "Modeling sustainable city trips: integrating CO 2 e emissions, popularity, and seasonality into tourism recommender systems". In: *Information Technology & Tourism* (2025), pp. 1–38.

[12] J. Xie, K. Zhang, J. Chen, T. Zhu, R. Lou, Y. Tian, Y. Xiao, and Y. Su. "Travelplanner: A benchmark for real-world planning with language agents". In: *arXiv preprint arXiv:2402.01622* (2024).

[13] Q. Wen, Y. Liu, J. Zhang, G. Saad, A. Korikov, Y. Sambale, and S. Sanner. *Elaborative Subtopic Query Reformulation for Broad and Indirect Queries in Travel Destination Recommendation*. en. arXiv:2410.01598 [cs]. Oct. 2024. URL: http://arxiv.org/abs/2410.01598 (visited on 10/29/2024).

[14] Q. Peng, H. Liu, H. Huang, Q. Yang, and M. Shao. "A survey on llm-powered agents for recommender systems". In: *arXiv preprint arXiv:2502.10050* (2025).

[15] Y. Sun and Y. Zhang. "Conversational recommender system". In: *The 41st international acm sigir conference on research & development in information retrieval*. 2018, pp. 235–244.

[16] D. Jannach, A. Manzoor, W. Cai, and L. Chen. "A survey on conversational recommender systems". In: *ACM Computing Surveys (CSUR)* 54.5 (2021), pp. 1–36.

[17] C. Gao, W. Lei, X. He, M. De Rijke, and T.-S. Chua. "Advances and challenges in conversational recommender systems: A survey". In: *AI open* 2 (2021), pp. 100–126.

[18] D. Jannach. "Evaluating conversational recommender systems: A landscape of research". In: *Artificial Intelligence Review* 56.3 (2023), pp. 2365–2400.

[19] D. Pramod and P. Bafna. "Conversational recommender systems techniques, tools, acceptance, and adoption: a state of the art review". In: *Expert Systems with Applications* 203 (2022), p. 117539.

[20] Y. Liu, W. Zhang, Y. Chen, Y. Zhang, H. Bai, F. Feng, H. Cui, Y. Li, and W. Che. "Conversational Recommender System and Large Language Model Are Made for Each Other in E-commerce Pre-sales Dialogue". In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. 2023, pp. 9587–9605.

[21] Y. Feng, S. Liu, Z. Xue, Q. Cai, L. Hu, P. Jiang, K. Gai, and F. Sun. "A large language model enhanced conversational recommender system". In: *arXiv preprint arXiv:2308.06212* (2023).

[22] L. Friedman, S. Ahuja, D. Allen, Z. Tan, H. Sidahmed, C. Long, J. Xie, G. Schubiner, A. Patel, H. Lara, et al. "Leveraging large language models in conversational recommender systems". In: *arXiv preprint arXiv:2305.07961* (2023).

[23] C. Li, Y. Deng, H. Hu, M.-Y. Kan, and H. Li. "Incorporating external knowledge and goal guidance for llm-based conversational recommender systems". In: *arXiv preprint arXiv:2405.01868* (2024).

[24] C. K. Sah, L. Xiaoli, and M. M. Islam. "Unveiling bias in fairness evaluations of large language models: A critical literature review of music and movie recommendation systems". In: *arXiv preprint arXiv:2401.04057* (2024).

[25] C. Jiang, J. Wang, W. Ma, C. L. Clarke, S. Wang, C. Wu, and M. Zhang. "Beyond Utility: Evaluating LLM as Recommender". In: *Proceedings of the ACM on Web Conference 2025*. 2025, pp. 3850–3862.

[26] S. Meyer, S. Singh, B. Tam, C. Ton, and A. Ren. "A Comparison of LLM Finetuning Methods & Evaluation Metrics with Travel Chatbot Use Case". In: *arXiv preprint arXiv:2408.03562* (2024).

[27] S. Wang, E. Khramtsova, S. Zhuang, and G. Zuccon. "Feb4rag: Evaluating federated search in the context of retrieval augmented generation". In: *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2024, pp. 763–773.

[28] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, and I. Gurevych. "Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models". In: *arXiv preprint arXiv:2104.08663* (2021).

[29] H. Zhang, A. Korikov, P. Farinneya, M. M. Abdollah Pour, M. Bharadwaj, A. Pesaranghader, X. Y. Huang, Y. X. Lok, Z. Wang, N. Jones, and S. Sanner. "Recipe-MPR: A Test Collection for Evaluating Multi-aspect Preference-based Natural Language Retrieval". In: *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '23. New York, NY, USA: Association for Computing Machinery, July 2023, pp. 2744–2753. ISBN: 978-1-4503-9408-6. DOI: 10.1145/3539618.3591880. URL: https://dl.acm.org/doi/10.1145/3539618.3591880 (visited on 11/06/2024).

[30] S. Haussmann, O. Seneviratne, Y. Chen, Y. Ne'eman, J. Codella, C.-H. Chen, D. L. McGuinness, and M. J. Zaki. "FoodKG: a semantics-driven knowledge graph for food recommendation". In: *The Semantic Web–ISWC 2019: 18th International Semantic Web Conference, Auckland, New Zealand, October 26–30, 2019, Proceedings, Part II 18*. Springer. 2019, pp. 146–162.

[31] J. Marın, A. Biswas, F. Ofli, N. Hynes, A. Salvador, Y. Aytar, I. Weber, and A. Torralba. "Recipe1m+: A dataset for learning cross-modal embeddings for cooking recipes and food images". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.1 (2021), pp. 187–203.

[32] K. Chaudhari and A. Thakkar. "A comprehensive survey on travel recommender systems". In: *Archives of computational methods in engineering* 27 (2020), pp. 1545–1571.

[33] J. Lan, R. Shi, Y. Cao, and J. Lv. "Knowledge Graph-based Conversational Recommender System in Travel". In: *2022 International Joint Conference on Neural Networks (IJCNN)*. ISSN: 2161-4407. July 2022, pp. 1–8. DOI: 10.1109/IJCNN55064.2022.9892176. URL: https://ieeexplore.ieee.org/document/9892176/?arnumber=9892176 (visited on 10/27/2024).

[34] D. Jannach and L. Chen. "Conversational recommendation: A grand AI challenge". In: *AI Magazine* 43.2 (2022), pp. 151–163.

[35] P. Jandaghi, X. Sheng, X. Bai, J. Pujara, and H. Sidahmed. "Faithful persona-based conversational dataset generation with large language models". In: *arXiv preprint arXiv:2312.10007* (2023).

[36] L. Zhu, X. Huang, and J. Sang. "A LLM-based Controllable, Scalable, Human-Involved User Simulator Framework for Conversational Recommender Systems". In: *arXiv preprint arXiv:2405.08035* (2024).

[37] Z. Abbasiantaeb, Y. Yuan, E. Kanoulas, and M. Aliannejadi. "Let the llms talk: Simulating human-to-human conversational qa via zero-shot llm-to-llm interactions". In: *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*. 2024, pp. 8–17.

[38] X. Wang, H. A. Rahmani, J. Liu, and E. Yilmaz. "Improving Conversational Recommendation Systems via Bias Analysis and Language-Model-Enhanced Data Augmentation". In: *arXiv preprint arXiv:2310.16738* (2023).

[39] T. Ashby, A. Kulkarni, J. Qi, M. Liu, E. Cho, V. Kumar, and L. Huang. "Towards Effective Long Conversation Generation with Dynamic Topic Tracking and Recommendation". In: *Proceedings of the 17th International Natural Language Generation Conference*. 2024, pp. 540–556.

[40] Y. Lu, J. Bao, Z. Ma, X. Han, Y. Wu, S. Cui, and X. He. "AUGUST: an automatic generation understudy for synthesizing conversational recommendation datasets". In: *arXiv preprint arXiv:2306.09631* (2023).

[41] M. Leszczynski, S. Zhang, R. Ganti, K. Balog, F. Radlinski, F. Pereira, and A. T. Chaganty. "Talk the Walk: Synthetic Data Generation for Conversational Music Recommendation". In: *arXiv preprint arXiv:2301.11489* (2023).

[42] R. Tang, X. Han, X. Jiang, and X. Hu. "Does synthetic data generation of llms help clinical text mining?" In: *arXiv preprint arXiv:2303.04360* (2023).

[43] M. Khalil, F. Vadiee, R. Shakya, and Q. Liu. "Creating Artificial Students that Never Existed: Leveraging Large Language Models and CTGANs for Synthetic Data Generation". In: *arXiv preprint arXiv:2501.01793* (2025).

[44] A. Korikov, G. Saad, E. Baron, M. Khan, M. Shah, and S. Sanner. *Multi-Aspect Reviewed-Item Retrieval via LLM Query Decomposition and Aspect Fusion*. arXiv:2408.00878. Aug. 2024. DOI: 10.48550/arXiv.2408.00878. URL: http://arxiv.org/abs/2408.00878 (visited on 11/04/2024).

[45] M. M. Abdollah Pour, P. Farinneya, A. Toroghi, A. Korikov, A. Pesaranghader, T. Sajed, M. Bharadwaj, B. Mavrin, and S. Sanner. "Self-supervised Contrastive BERT Fine-tuning for Fusion-Based Reviewed-Item Retrieval". en. In: *Advances in Information Retrieval*. Ed. by J. Kamps, L. Goeuriot, F. Crestani, M. Maistro, H. Joho, B. Davis, C. Gurrin, U. Kruschwitz, and A. Caputo. Cham: Springer Nature Switzerland, 2023, pp. 3–17. ISBN: 978-3-031-28244-7. DOI: 10.1007/978-3-031-28244-7_1.

[46] M. Aliannejadi, H. Zamani, F. Crestani, and W. B. Croft. "Asking Clarifying Questions in Open-Domain Information-Seeking Conversations". en. In: *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Paris France: ACM, July 2019, pp. 475–484. ISBN: 978-1-4503-6172-9. DOI: 10.1145/3331184.3331265. URL: https://dl.acm.org/doi/10.1145/3331184.3331265 (visited on 10/17/2024).

[47] S. Kemper, J. Cui, K. Dicarlantonio, K. Lin, D. Tang, A. Korikov, and S. Sanner. "Retrieval-Augmented Conversational Recommendation with Prompt-based Semi-Structured Natural Language State Tracking". In: *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '24. New York, NY, USA: Association for Computing Machinery, July 2024, pp. 2786–2790. ISBN: 9798400704314. DOI: 10.1145/3626772.3657670. URL: https://dl.acm.org/doi/10.1145/3626772.3657670 (visited on 10/29/2024).

[48] P. Merinov. "Sustainability-oriented recommender systems". In: *Proceedings of the 31st ACM Conference on User Modeling, Adaptation and Personalization*. 2023, pp. 296–300.

[49] G. K. Patro, A. Chakraborty, A. Banerjee, and N. Ganguly. "Towards Safety and Sustainability: Designing Local Recommendations for Post-pandemic World". en. In: *Fourteenth ACM Conference on Recommender Systems*. Virtual Event Brazil: ACM, Sept. 2020, pp. 358–367. ISBN: 978-1-4503-7583-2. DOI: 10.1145/3383313.3412251. URL: https://dl.acm.org/doi/10.1145/3383313.3412251 (visited on 04/15/2024).

[50] A. Pachot, A. Albouy-Kissi, B. Albouy-Kissi, and F. Chausse. "Multiobjective recommendation for sustainable production systems". In: *MORS workshop held in conjunction with the 15th ACM Conference on Recommender Systems (RecSys), 2021*. 2021.

[51] D. Herzog, S. Sikander, and W. Wörndl. "Integrating route attractiveness attributes into tourist trip recommendations". In: *Companion Proceedings of The 2019 World Wide Web Conference*. 2019, pp. 96–101.

[52] F. J. Hoffmann, F. Braesemann, and T. Teubner. "Measuring sustainable tourism with online platform data". In: *EPJ Data Science* 11.1 (2022), p. 41.

[53] A. Banerjee, A. Satish, and W. Wörndl. "Enhancing tourism recommender systems for sustainable city trips using retrieval-augmented generation". In: *International Workshop on Recommender Systems for Sustainability and Social Good*. Springer. 2024, pp. 19–34.

[54] Y. Wang, W. Ma, M. Zhang, Y. Liu, and S. Ma. "A survey on the fairness of recommender systems". In: *ACM Transactions on Information Systems* 41.3 (2023), pp. 1–43.

[55] Y. Li, H. Chen, Z. Fu, Y. Ge, and Y. Zhang. "User-oriented fairness in recommendation". In: *Proceedings of the web conference 2021*. 2021, pp. 624–632.

[56] J. Zhang, K. Bao, Y. Zhang, W. Wang, F. Feng, and X. He. "Is chatgpt fair for recommendation? evaluating fairness in large language model recommendation". In: *Proceedings of the 17th ACM Conference on Recommender Systems*. 2023, pp. 993–999.

[57] Y. Deldjoo. "Understanding biases in ChatGPT-based recommender systems: Provider fairness, temporal stability, and recency". In: *ACM Transactions on Recommender Systems* (2024).

[58] M. Jiang, K. Bao, J. Zhang, W. Wang, Z. Yang, F. Feng, and X. He. "Item-side fairness of large language model-based recommendation system". In: *Proceedings of the ACM Web Conference 2024*. 2024, pp. 4717–4726.

[59] J. M. Lichtenberg, A. Buchholz, and P. Schwöbel. "Large language models as recommender systems: A study of popularity bias". In: *arXiv preprint arXiv:2406.01285* (2024).

[60] A. Klimashevskaia, D. Jannach, M. Elahi, and C. Trattner. "A survey on popularity bias in recommender systems". In: *User Modeling and User-Adapted Interaction* 34.5 (2024), pp. 1777–1834.

[61] F. M. Harper and J. A. Konstan. "The movielens datasets: History and context". In: *Acm transactions on interactive intelligent systems (tiis)* 5.4 (2015), pp. 1–19.

[62] J. Lu, Z. Pang, M. Xiao, Y. Zhu, R. Xia, and J. Zhang. "Merge, ensemble, and cooperate! a survey on collaborative strategies in the era of large language models". In: *arXiv preprint arXiv:2407.06089* (2024).

[63] T. Guo, X. Chen, Y. Wang, R. Chang, S. Pei, N. V. Chawla, O. Wiest, and X. Zhang. "Large language model based multi-agents: A survey of progress and challenges". In: *arXiv preprint arXiv:2402.01680* (2024).

[64] J. Li, Q. Zhang, Y. Yu, Q. Fu, and D. Ye. "More agents is all you need". In: *arXiv preprint arXiv:2402.05120* (2024).

[65] Y. Zhang, S. Qiao, J. Zhang, T.-H. Lin, C. Gao, and Y. Li. "A Survey of Large Language Model Empowered Agents for Recommendation and Search: Towards Next-Generation Information Retrieval". In: *arXiv preprint arXiv:2503.05659* (2025).

[66] D. Jiang, X. Ren, and B. Y. Lin. "LLM-Blender: Ensembling Large Language Models with Pairwise Ranking and Generative Fusion". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by A. Rogers, J. Boyd-Graber, and N. Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 14165–14178. DOI: 10.18653/v1/2023.acl-long.792. URL: https://aclanthology.org/2023.acl-long.792/.

[67] I. Sutskever, O. Vinyals, and Q. V. Le. "Sequence to sequence learning with neural networks". In: *Advances in neural information processing systems* 27 (2014).

[68] S. Longpre, L. Hou, T. Vu, A. Webson, H. W. Chung, Y. Tay, D. Zhou, Q. V. Le, B. Zoph, J. Wei, et al. "The flan collection: Designing data and methods for effective instruction tuning". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 22631–22648.

[69] B. Lv, C. Tang, Y. Zhang, X. Liu, P. Luo, and Y. Yu. "URG: A Unified Ranking and Generation Method for Ensembling Language Models". In: *Findings of the Association for Computational Linguistics: ACL 2024*. Ed. by L.-W. Ku, A. Martins, and V. Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 4421–4434. DOI: 10.18653/v1/2024.findings-acl.261. URL: https://aclanthology.org/2024.findings-acl.261/.

[70] Q. Wu, G. Bansal, J. Zhang, Y. Wu, B. Li, E. Zhu, L. Jiang, X. Zhang, S. Zhang, J. Liu, et al. "Autogen: Enabling next-gen llm applications via multi-agent conversation". In: *arXiv preprint arXiv:2308.08155* (2023).

[71] A. Yehudai, L. Eden, A. Li, G. Uziel, Y. Zhao, R. Bar-Haim, A. Cohan, and M. Shmueli-Scheuer. "Survey on Evaluation of LLM-based Agents". In: *arXiv preprint arXiv:2503.16416* (2025).

[72] K.-H. Huang, A. Prabhakar, S. Dhawan, Y. Mao, H. Wang, S. Savarese, C. Xiong, P. Laban, and C.-S. Wu. "CRMArena: Understanding the Capacity of LLM Agents to Perform Professional CRM Tasks in Realistic Environments". In: *arXiv preprint arXiv:2411.02305* (2024).

[73] Z. Wu, R. Peng, S. Zheng, Q. Liu, X. Han, B. I. Kwon, M. Onizuka, S. Tang, and C. Xiao. "Shall we team up: Exploring spontaneous cooperation of competing llm agents". In: *arXiv preprint arXiv:2402.12327* (2024).

[74] F. Bianchi, P. J. Chia, M. Yuksekgonul, J. Tagliabue, D. Jurafsky, and J. Zou. "How Well Can LLMs Negotiate? NegotiationArena Platform and Analysis". In: *International Conference on Machine Learning*. PMLR. 2024, pp. 3935–3951.

[75] Y. Du, S. Li, A. Torralba, J. B. Tenenbaum, and I. Mordatch. "Improving factuality and reasoning in language models through multiagent debate". In: *Forty-first International Conference on Machine Learning*. 2023.

[76] X. Chen, M. Mao, S. Li, and H. Shangguan. "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)*. 2025, pp. 462–470.

[77] J. Chen, S. Saha, and M. Bansal. "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)*. 2024, pp. 7066–7085.

[78] D. Wan, J. Chen, E. Stengel-Eskin, and M. Bansal. "MAMM-Refine: A Recipe for Improving Faithfulness in Generation with Multi-Agent Collaboration". In: *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. 2025, pp. 9882–9901.

[79] K. Shridhar, K. Sinha, A. Cohen, T. Wang, P. Yu, R. Pasunuru, M. Sachan, J. Weston, and A. Celikyilmaz. "The ART of LLM Refinement: Ask, Refine, and Trust". In: *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. 2024, pp. 5872–5883.

[80] Z. Wu and T. Ito. "The hidden strength of disagreement: Unraveling the consensus-diversity tradeoff in adaptive multi-agent systems". In: *arXiv preprint arXiv:2502.16565* (2025).

[81] Y. Wang, Z. Jiang, Z. Chen, F. Yang, Y. Zhou, E. Cho, X. Fan, X. Huang, Y. Lu, and Y. Yang. "Recmind: Large language model powered agent for recommendation". In: *arXiv preprint arXiv:2308.14296* (2023).

[82] J. Fang, S. Gao, P. Ren, X. Chen, S. Verberne, and Z. Ren. "A multi-agent conversational recommender system". In: *arXiv preprint arXiv:2402.01135* (2024).

[83] Z. Wang, Y. Yu, W. Zheng, W. Ma, and M. Zhang. "MACRec: A Multi-Agent Collaboration Framework for Recommendation". In: SIGIR '24. Washington DC, USA: Association for Computing Machinery, 2024, pp. 2760–2764. ISBN: 9798400704314. DOI: 10.1145/3626772.3657669. URL: https://doi.org/10.1145/3626772.3657669.

[84] G. Nie, R. Zhi, X. Yan, Y. Du, X. Zhang, J. Chen, M. Zhou, H. Chen, T. Li, Z. Cheng, S. Xu, and J. Hu. "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*. RecSys '24. Bari, Italy: Association for Computing Machinery, 2024, pp. 745–747. ISBN: 9798400705052. DOI: 10.1145/3640457.3688061. URL: https://doi.org/10.1145/3640457.3688061.

[85] Z. Hui, X. Wei, Y. Jiang, K. Gao, C. Wang, F. Ong, S.-e. Yoon, R. Pareek, and M. Gong. "MATCHA: Can Multi-Agent Collaboration Build a Trustworthy Conversational Recommender?" In: *arXiv preprint arXiv:2504.20094* (2025).

[86] G. Balakrishnan and W. Wörndl. "Multistakeholder Recommender Systems in Tourism". In: *Proc. Workshop on Recommenders in Tourism (RecTour 2021)* (2021).

[87] Z. Zhang, S. Liu, Z. Liu, R. Zhong, Q. Cai, X. Zhao, C. Zhang, Q. Liu, and P. Jiang. "Llm-powered user simulator for recommender system". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 39. 12. 2025, pp. 13339–13347.

[88] A. Banerjee, A. Satish, and W. Wörndl. "Enhancing Tourism Recommender Systems for Sustainable City Trips Using Retrieval-Augmented Generation". In: *arXiv preprint arXiv:2409.18003* (2024).

[89] M. Lewis. "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension". In: *arXiv preprint arXiv:1910.13461* (2019).

[90] A. Williams, N. Nangia, and S. Bowman. "A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, 2018, pp. 1112–1122. URL: http://aclweb.org/anthology/N18-1101.

[91] W. Yin, J. Hay, and D. Roth. "Benchmarking Zero-shot Text Classification: Datasets, Evaluation and Entailment Approach". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Ed. by K. Inui, J. Jiang, V. Ng, and X. Wan. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3914–3923. DOI: 10.18653/v1/D19-1404. URL: https://aclanthology.org/D19-1404/.

[92] L. Fröhling, G. Demartini, and D. Assenmacher. "Personas with Attitudes: Controlling LLMs for Diverse Data Annotation". In: *arXiv preprint arXiv:2410.11745* (2024).

[93] T. Ge, X. Chan, X. Wang, D. Yu, H. Mi, and D. Yu. "Scaling synthetic data creation with 1,000,000,000 personas". In: *arXiv preprint arXiv:2406.20094* (2024).

[94] M. Grootendorst. "BERTopic: Neural topic modeling with a class-based TF-IDF procedure". In: *arXiv preprint arXiv:2203.05794* (2022).

[95] P. Sanchez and L. W. Dietz. "Travelers vs. Locals: The Effect of Cluster Analysis in Point-of-Interest Recommendation". In: *Proceedings of the 30th ACM Conference on User Modeling, Adaptation and Personalization*. 2022, pp. 132–142.

[96] Y. Deldjoo. "Fairness of chatgpt and the role of explainable-guided prompts". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2023, pp. 13–22.

[97] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, et al. "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions". In: *ACM Transactions on Information Systems* (2024).

[98] G. Firebase. *Firebase Realtime Database Documentation*. 2025. URL: https://firebase.google.com/docs/database (visited on 02/13/2025).

[99] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing, et al. "Judging llm-as-a-judge with mt-bench and chatbot arena". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 46595–46623.

[100] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou. "Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers". In: *Advances in neural information processing systems* 33 (2020), pp. 5776–5788.

[101] L. Tang, P. Laban, and G. Durrett. "MiniCheck: Efficient Fact-Checking of LLMs on Grounding Documents". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. 2024, pp. 8818–8847.

[102] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma, et al. "Scaling instruction-finetuned language models". In: *Journal of Machine Learning Research* 25.70 (2024), pp. 1–53.

[103] Y. Zhu, S. Lu, L. Zheng, J. Guo, W. Zhang, J. Wang, and Y. Yu. "Texygen: A Benchmarking Platform for Text Generation Models". In: *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. SIGIR '18. Ann Arbor, MI, USA: Association for Computing Machinery, 2018, pp. 1097–1100. ISBN: 9781450356572. DOI: 10.1145/3209978.3210080. URL: https://doi.org/10.1145/3209978.3210080.

[104] Google. *Ground Gemini to your data*. URL: https://cloud.google.com/vertex-ai/generative-ai/docs/multimodal/ground-with-your-data#private-ground-gemini (visited on 02/13/2025).

[105] D. Machlab and R. Battle. "LLM In-Context Recall is Prompt Dependent". In: *arXiv preprint arXiv:2404.08865* (2024).

[106] Y. Gao, T. Sheng, Y. Xiang, Y. Xiong, H. Wang, and J. Zhang. *Chat-REC: Towards Interactive and Explainable LLMs-Augmented Recommender System*. 2023. arXiv: 2303.14524 [cs.IR].

[107] S. Lubos, T. N. T. Tran, A. Felfernig, S. Polat Erdeniz, and V.-M. Le. "Llm-generated explanations for recommender systems". In: *Adjunct Proceedings of the 32nd ACM Conference on User Modeling, Adaptation and Personalization*. 2024, pp. 276–285.

[108] Y. Deldjoo, N. Mehta, M. Sathiamoorthy, S. Zhang, P. Castells, and J. McAuley. "Toward Holistic Evaluation of Recommender Systems Powered by Generative Models". In: *SIGIR'25* (2025).

[109] R. Staab, M. Vero, M. Balunović, and M. Vechev. "Beyond memorization: Violating privacy via inference with large language models". In: *arXiv preprint arXiv:2310.07298* (2023).

[110] S. K. Sakib and A. B. Das. "Challenging fairness: A comprehensive exploration of bias in llm-based recommendations". In: *2024 IEEE International Conference on Big Data (BigData)*. IEEE. 2024, pp. 1585–1592.

[111] S. Erlich, N. Hazon, and S. Kraus. "Negotiation strategies for agents with ordinal preferences". In: *arXiv preprint arXiv:1805.00913* (2018).

[112] S. Patro. "Normalization: A preprocessing stage". In: *arXiv preprint arXiv:1503.06462* (2015).

[113] OpenAI. "OpenAI o3 and o4-mini System Card". In: (2025). URL: https://cdn.openai.com/pdf/2221c875-02dc-4789-800b-e7758f3722c1/o3-and-o4-mini-system-card.pdf.

[114] G. Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, K. Millican, et al. "Gemini: a family of highly capable multimodal models". In: *arXiv preprint arXiv:2312.11805* (2023).

[115] J. L. Gastwirth. “The estimation of the Lorenz curve and Gini index”. In: *The review of economics and statistics* (1972), pp. 306–316.

[116] L. Jost. “Entropy and diversity”. In: *Oikos* 113.2 (2006), pp. 363–375.

[117] N. Jegham, M. Abdelatti, L. Elmoubarki, and A. Hendawi. “How Hungry is AI? Benchmarking Energy, Water, and Carbon Footprint of LLM Inference”. In: *arXiv preprint arXiv:2505.09598* (2025).