ToolSpectrum : Towards Personalized Tool Utilization for Large Language Models

Zihao Cheng[1,3†] , **Hongru Wang**[4†] , **Zeming Liu**[1‡] , **Yuhang Guo**[2‡],
Yuanfang Guo[1], **Yunhong Wang**[1], **Haifeng Wang**[5]
[1]School of Computer Science and Engineering, Beihang University, Beijing, China,
[2]School of Computer Science and Technology, Beijing Institute of Technology, Beijing,
[3]University of Science and Technology Beijing, Beijing, China,
[4]The Chinese University of Hong Kong, Hong Kong, China, [5] Baidu Inc., Beijing, China
chengzihao008@gmail.com, hrwang@se.cuhk.edu.hk, zmliu@buaa.edu.cn

Abstract

While integrating external tools into large language models (LLMs) enhances their ability to access real-time information and domain-specific services, existing approaches focus narrowly on functional tool selection following user instructions, overlooking the context-aware personalization in tool selection. This oversight leads to suboptimal user satisfaction and inefficient tool utilization, particularly when overlapping toolsets require nuanced selection based on contextual factors. To bridge this gap, we introduce **ToolSpectrum**, a benchmark designed to evaluate LLMs' capabilities in personalized tool utilization. Specifically, we formalize two key dimensions of personalization, *user profile* and *environmental factors*, and analyze their individual and synergistic impacts on tool utilization. Through extensive experiments on ToolSpectrum, we demonstrate that personalized tool utilization significantly improves user experience across diverse scenarios. However, even state-of-the-art LLMs exhibit the limited ability to reason jointly about user profiles and environmental factors, often prioritizing one dimension at the expense of the other. Our findings underscore the necessity of context-aware personalization in tool-augmented LLMs and reveal critical limitations in current models. Our data and code are available at https://github.com/BUAA-IRIP-LLM/ToolSpectrum.

1 Introduction

Integrating external tools into large language models (LLMs) enables them to transcend inherent knowledge boundaries by dynamically accessing specialized functionalities, driving unprecedented progress in task automation and problem-solving (Wang et al., 2024a; Qin et al., 2024; Qu et al., 2025; Lin et al., 2024; Wang et al., 2024c; Qian et al., 2025). Recent researches demonstrate the effectiveness of this integration in diverse domains, including travel planning (Xie et al., 2024), online shopping (Yao et al., 2022), and knowledge acquisition (Wang et al., 2023a; Lin et al., 2024; Wang et al., 2025). However, current research lacks consideration for toolsets with overlapping functionalities and primarily focuses on selecting the tool to complete the user's instruction solely (Wang et al., 2024b; Qian et al., 2024; Ning et al., 2024). This approach overlooks the reality that numerous tools with similar capabilities exist, each capable of achieving the same objective, which needs LLMs integrating contextual factors during the tool uti-

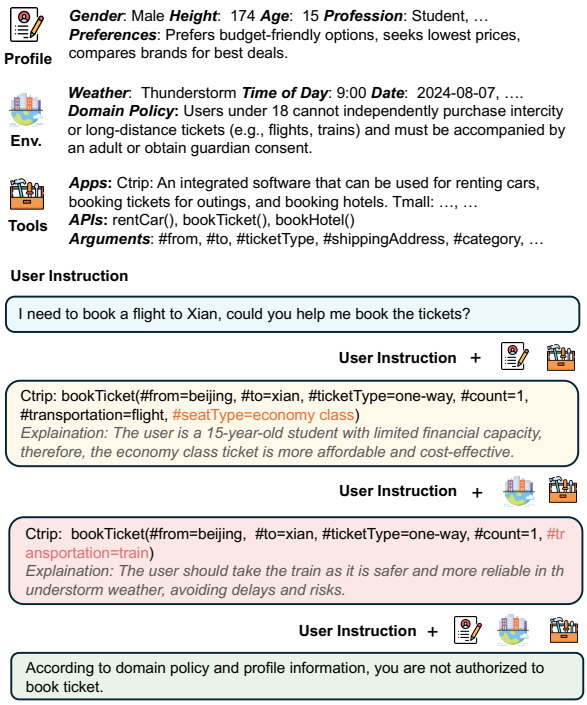

Figure 1: An example from our proposed ToolSpectrum, illustrating the effects of user profile and environment on personalized tool utilization. This illustrates three distinct scenarios, considering profile-only , environment-only , and combined profile and environment factors .

[†] Equal Contributions. Work done during the first author's internship at Beihang University.
[‡] Corresponding Author

Findings of the Association for Computational Linguistics: ACL 2025, pages 20679–20699

lization to significantly enhance user experience (Liang et al., 2006; Lex and Schedl, 2022; Zhang et al., 2024b).

It is crucial to recognize that users with different contexts prefer to utilize different tools when aiming to achieve the same objective (Burke and Reitzes, 1981). As illustrated in Figure 1, when a user prioritizes budget-friendly options, the system should recommend an *economy class* flight ticket as the most suitable option. Besides, suppose environmental factors, such as thunderstorms, make air travel unsafe. In that case, the system should suggest a train ticket as a safer alternative, providing the user with an explanation for this recommendation. Additionally, the system may face further constraints when considering user profiles and environmental factors. For instance, if the user is a minor and domain policies require guardian consent for ticket bookings, the system must restrict the purchase and prompt the user to provide authorization from a guardian. This demonstrates that LLMs must move beyond simple tool selection and instead develop user-centric intelligence. Such intelligence would allow LLMs to understand the user's context and make more appropriate, personalized tool utilization.

To this end, we develop `ToolSpectrum`, a novel benchmark designed for evaluating personalized tool utilization capabilities of LLMs, which is constructed through a three-stage methodology. Specifically, we first collect commonly used Apps and APIs from high-frequency user scenarios and manually introduce alternative Apps or APIs with similar functionalities but tailored to meet the needs of different contexts (i.e., Temu[1] and Amazon). Next, we identify two critical factors influencing personalized tool utilization: **user profile** and **environment**. These factors have been widely discussed in previous personalization studies (Zhang et al., 2018; Mao et al., 2024; Hui et al., 2024; Salemi et al., 2024), and are known to have a significant impact on human behavior patterns (Allport, 1937; Hall, 1969; Mischel, 1996). Finally, we simulate real-world user instructions and tool call results, considering the toolset, user profile, and environment, leading to the final `ToolSpectrum`, the comprehensive benchmark for personalized tool utilization.

We further investigate the impact of personalization on tool utilization through extensive experiments using `ToolSpectrum`. The experimental results reveal two key insights: (1) integrating personalization into tool utilization significantly improves its effectiveness, and (2) current LLMs generally underperform on the task of personalized tool utilization.

[1]Temu is the competitor of Amazon with mostly lower prices.

Datasets	Profile	Environment	Both	Tool
PersonaChat (Zhang et al., 2018)	✓	✗	✗	✗
CharacterEval (Tu et al., 2024)	✓	✗	✗	✗
BARS (Zhu et al., 2022)	✓	✗	✗	✗
LaMP (Salemi et al., 2024)	✓	✗	✗	✗
AI Persona (Wang et al., 2024d)	✓	✗	✗	✗
MovieLens 10M (Rendle et al., 2019)	✓	✗	✗	✗
SocialBench (Chen et al., 2024a)	✓	✓	✗	✗
ToolBench (Xu et al., 2023)	✗	✗	✗	✓
API-Bank (Li et al., 2023)	✗	✗	✗	✓
AgentBench (Liu et al., 2023b)	✗	✗	✗	✓
τ-Bench (Yao et al., 2024)	✗	✓	✗	✓
AgentBoard (Ma et al., 2024)	✗	✗	✗	✓
AppBench (Wang et al., 2024b)	✗	✗	✗	✓
UltraTool (Huang et al., 2024b)	✗	✗	✗	✓
T-Eval (Chen et al., 2024b)	✗	✗	✗	✓
ToolSpectrum (Ours)	✓	✓	✓	✓

Table 1: Comparison between the `ToolSpectrum` and other benchmarks, with detailed analysis provided in Appendix A.1.

Overall, the contributions of this paper are as follows:

- To the best of our knowledge, we are the first to define personalized tool utilization, which proposes a new challenge: current LLMs focus solely on planning tools that fulfill user instructions without considering personalization to select the most suitable tool.
- To mitigate this challenge, we propose `ToolSpectrum`, the first benchmark designed to evaluate personalized tool utilization capabilities of LLMs considering user profiles, environment, and their joint effects.
- We conduct extensive experiments and analysis based on `ToolSpectrum`. Results demonstrate that incorporating personalization into tool utilization significantly improves its effectiveness. However, existing LLMs struggle with this new task, particularly with more personalized factors considered.

2 Related Work

2.1 Tool Learning Benchmarks

Integrating external tools into LLMs leads to significant advancements in their capabilities, enabling them to perform complex real-world tasks. For instance, tools such as retrievers and calculators enable them to address challenges related to factual accuracy and computation, thereby broadening

their potential applications (Mialon et al., 2023; Qin et al., 2024). Therefore, evaluating how effectively LLMs utilize these tools becomes a key research focus. Existing benchmarks primarily assess general tool execution performance, including the interaction capabilities of LLMs with tools (Qin et al., 2023; Liu et al., 2023b; Huang et al., 2024b; Li et al., 2023), planning and reasoning capabilities with tools (Han et al., 2025; Wang et al., 2024b; Huang et al., 2024b; Chen et al., 2024b), as well as the models' resistance to hallucinations and robustness in tool utilization (Zhang et al., 2024a; Huang et al., 2024c; Ning et al., 2024; Zhan et al., 2024; Ye et al., 2024). However, existing benchmarks overlook the presence of the toolset with overlapping functionalities. They typically select the tool to complete the user's instruction solely, without considering how user profiles and environmental factors could influence the selection of the most effective tool. To mitigate this gap, we introduce `ToolSpectrum`, which explicitly considers the impact of user profiles and environmental factors on tool utilization.

2.2 Personalized LLMs

Personalization has long been a core of research in domains such as dialogue systems and recommendation systems, where its ability to enhance user experience and satisfaction is well-established (Adomavicius and Tuzhilin, 2005; Geng et al., 2022). A significant portion of this research focuses on personalizing content to match users' preferences, particularly through personalized recommendations (Dai et al., 2023; Du et al., 2024; Hou et al., 2024; Liu et al., 2023a), tailored search results (Spathariotti et al., 2023; Joko et al., 2024; Zhou et al., 2024), dialogue systems (Shi et al., 2023; Liu et al., 2022), and content generation tasks (Zhang et al., 2023). However, these approaches tend to focus mainly on user profiles, often overlooking the influence of environmental factors on personalization, such as natural and digital environments or real-world constraints that could further refine personalization. Another promising direction is the role-playing capabilities that enable models to adopt specific personas or professional roles. This paradigm allows LLMs to deliver context-sensitive interactions such as providing emotionally nuanced support (Wang et al., 2023b) or professional assistance across various fields, including finance (Liu et al., 2023c), health (Liu et al., 2023d), and education (Gonzalez et al., 2023). Generally, while significant progress has been made in personalized LLMs, current approaches still haven't explored the potential of integrating tool learning. Table 1 presents a detailed comparison of existing benchmarks.

3 `ToolSpectrum`: Towards Personalized Tool Utilization

To accurately evaluate the ability of LLMs to utilize tools in a personalized way, we introduce `ToolSpectrum`, the first benchmark that considers both user profiles and environmental factors. In this section, we first provide a formal definition of personalized tool utilization and then describe our data collection pipeline, designed to efficiently and effectively gather the necessary information for evaluation.

3.1 Task Definition

The personalized tool utilization model processes user instruction I through a mapping function $t = \text{Model}(I, \mathcal{P}, \mathcal{E}, \mathcal{T})$, where $\mathcal{P} = \{(k_i, v_i)\}_{i=1}^{m}$ represents user profile attributes (e.g., *age*, *gender*), $\mathcal{E} = \{(k_j, v_j)\}_{j=1}^{n}$ represents environmental context (e.g., *location*, *network condition*), and $\mathcal{T}$ represents toolset. The output t is structured as a dictionary $\{APP \mapsto a, API \mapsto s, RP \mapsto r, OP \mapsto o\}$ with four compulsory keys: APP specifies the target application, API determines the service interface, RP (Required Parameters) extracts mandatory arguments from I, and OP (Optional Parameters) provides personalized parameters based on $\mathcal{P}$ and $\mathcal{E}$. If the user's instruction I, when taking into account both the user profile $\mathcal{P}$ and the environmental context $\mathcal{E}$, violates the target application's policy, the system should return $t = \textit{None}$ to comply with the policy.

3.2 Data Collection

As shown in Figure 2, we implement rigorous construction steps to ensure the high quality and diversity of `ToolSpectrum`. (a) Toolset Collection (§3.2.1), (b) Profile and Environment Collection (§3.2.2, §3.2.3), (c) Tool-call Result Collection (§3.2.4) and (d) Quality Assessment (§3.2.5).

3.2.1 Toolset Collection

To cover diverse user instructions and applications, we identify 9 commonly used application domains based on previous studies (Guo et al., 2024; Wang et al., 2024b) and app analysis from App Store[2]

[2] https://apps.apple.com/us/charts

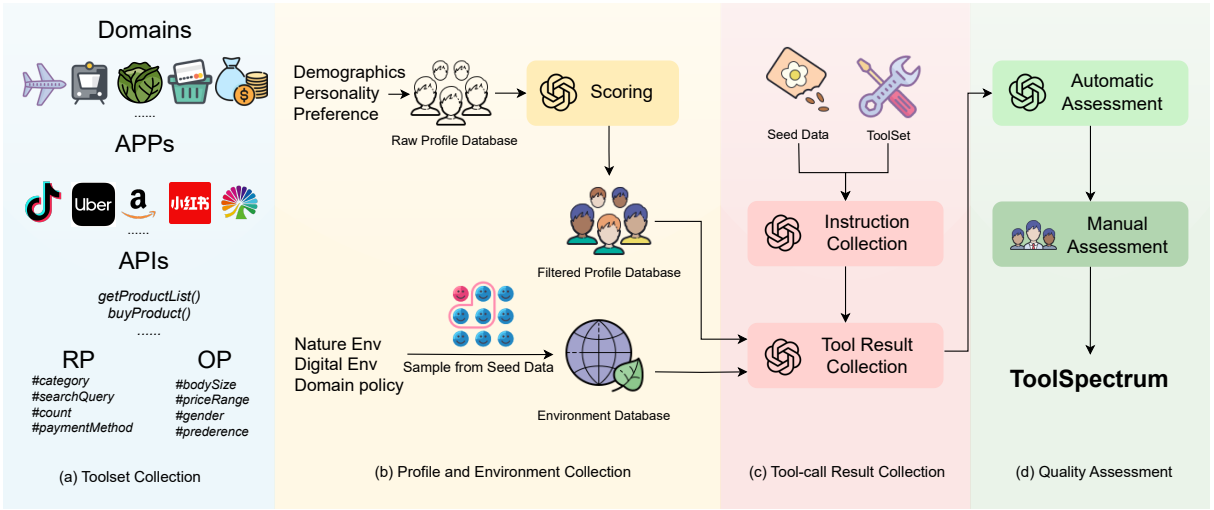

Figure 2: The overall construction process of ToolSpectrum, including (a) Toolset Collection, (b) Profile and Environment Collection, (c) Tool-call Result Collection, and (d) Quality Assessment.

[3]. These domains include *shopping*, *entertainment*, *travel*, *delivery*, *grocery*, *knowledge*, *news*, *health* and *finance*. In each domain, we leverage GPT-4o to generate initial designs for Apps and APIs with similar functions. We then manually curate the outputs, refining them with additional enhancements to ensure high quality. For example, in the *shopping* domain, users can choose between Amazon or Temu at the APP level to meet the same shopping needs. In the *travel* domain, users can choose different APIs within the Ctrip[4] to search for train or flight tickets. We also design corresponding RPs and OPs for each API to meet the reality and diverse user needs, following existing works (Wang et al., 2024b). The collected toolset is denoted as $\mathcal{T}$, and all Apps and APIs are listed in Table 7.

3.2.2 Profile and Environment Definition

To model the relationship between user profiles, environments, and personalized tool utilization, we need to create a database that captures both user profiles and environmental factors. In this section, we start with a comprehensive definition of both of these personalized factors.

Profile Definition. Inspired by previous personalization studies in other fields (Zhang et al., 2018; Wang et al., 2024d), we categorize existing user attributes into three major field: *demographics*, *personality*, and *preference*.

- **Demographics:** This field includes fundamental user information, presented as key-value pairs, including *gender*, *age*, *weight*, *height*, *profession*, *education background*, and *income level*. These details significantly influence the user's daily app usage behaviors. For instance, height and weight may affect a user's clothing size choices when shopping, and income level could determine price sensitivity.

- **Personality:** This field outlines the user's interests, expressed in natural language. These interests play a key role in shaping app usage behavior. For example, users who enjoy fitness and healthy living (e.g., *"Users enjoy exercising regularly."*) might frequently use health-tracking apps like MyFitnessPal or Strava.

- **Preference:** This field captures the user's historical interactions with apps, described in natural language. E.g., *"Users tend to use Amazon for shopping."* In many cases, users prefer to stick with familiar apps, even if these are not always the best fit for their needs.

Environment Definition. To ensure efficient tool utilization, it is essential to consider environmental factors. We identify three main types: the *natural environment*, the *digital environment*, and *App domain policies* (Yao et al., 2024).

- **Natural Environment:** This refers to the real-world context in which the user is situated, represented as key-value pairs, including *weather*, *date*, *time*, and *location*. These factors can directly or indirectly influence the result of tool utilization, particularly in scenarios that require

[3] https://play.google.com/store/apps

[4] Ctrip is a comprehensive app that offers fast booking for rooms, tickets, and a variety of other travel services.

interaction with the physical world. For example, weather conditions may affect the mode of transportation a user chooses.

- **Digital Environment:** This refers to the network and technological context of the user's device, represented as key-value pairs, such as *network condition* and *device-specific configurations*. For example, if a user has a slow or unstable network connection, the system might reduce image quality or preload content to ensure smoother performance and a better user experience.

- **App Domain Policy:** These are the specific policy rules for each app, described in natural language. They refer to regulations that govern the use of a particular app. For example, a policy might prohibit users under 18 from purchasing goods over 10,000 dollars without parental approval or offer discounted train tickets for children shorter than 1.2 meters.

3.2.3 Profile and Environment Collection

This section describes the construction of the profile and environment database. We implement a two-stage database generation methodology.

Seed Data Acquisition Establishing seed data enhances the controllability and transparency of data generation by predefining parameter ranges. We gather seed data for profiles and environments. For profiles, we use GPT-4o to generate value ranges for a part of demographic attributes (e.g., income, profession) and keywords for personality traits and preferences. For environments, we collect real-world weather data and manually define value ranges for natural and digital environments.

Profile and Environment Generation Given the seed data, we construct the database as follows:

- **Profile.** For demographics, we sample age, height, and weight from a normal distribution for realistic representation and randomly select other demographic attributes (e.g., gender, profession) from their value ranges. For personality and preference, which are generated by randomly combining keywords and processing them with GPT-4o, using prompt templates shown in Figures 8, 9, 10 to produce natural language descriptions. This yields the initial profile database.

- **Environment.** We sample parameters for both natural and digital environments from seed data and design domain-specific policies to represent daily life. This process results in an environment database, denoted as $\mathcal{E}$.

Quality Control. For the profile, we employ a two-stage quality control process. First, GPT-4o automatically evaluates each profile, assessing *Demographic Coherence* and *Preference Alignment*. The scoring range is from 1 to 10. We discard profiles scoring below 8. Second, we manually review the remaining profiles to finalize the profile database $\mathcal{P}$. For the environment, since we carefully considered the relationships between various features during the sampling process, no further quality control is needed.

3.2.4 Tool-Call Result Collection

The setup for `ToolSpectrum` is completed in the previous section. This section will focus on developing the user's instructions and corresponding personalized tool-call results.

Instruction Collection. To enhance the diversity and controllability of user instructions, we first manually construct a database of RP for each API, which serves as seed data, as shown in Table 6. Then, we use the seed data samples from this database and predefined toolsets $\mathcal{T}$ as input for GPT-4o, which generates instructions within each API denoting as $\mathcal{I} = \{I_i\}$. The prompt template used in this process is shown in Figure 12.

Tool Result Collection. Given the user instruction, we aim to investigate the effects of user profile, environment, and their combined influence on personalized tool utilization. Specifically, we input the user instruction $I \in \mathcal{I}$, predefined toolsets $\mathcal{T}$, and either a user profile $p \in \mathcal{P}$, an environment $e \in \mathcal{E}$, or both into GPT-4o to obtain the corresponding tool call results. This process generates three distinct datasets: **Profile**, **Environment**, and **Profile & Environment**, with the prompt template shown in Figure 13.

3.2.5 Quality Assessment

To enhance data quality, we first use GPT-4o to score each data sample across three dimensions: (1) *whether the tool call result met the user's needs*, (2) *whether the result aligned with the user's profiles*, and (3) *whether it matched the environmental factors* (Liu et al., 2020). The scoring range is from 1 to 10, and the prompt template is shown in Figure 14. We then exclude data points with an average score below 8, removing 21.8% of the data.

Statistic	Profile	Environment	Both
# Samples	450	220	330
# Domains	9	9	9
# APPs	22	22	20
# APIs	39	40	33
Profiles	158	-	62
Environments	-	87	122
Avg. Params	4.36	9.34	9.45
Avg. RPs	3.05	2.55	2.78
Avg. OPs	1.31	6.79	6.67

Table 2: The data statistic of ToolSpectrum. 'Both' refers to Profile & Environment.

Subsequently, we randomly sample 50 data points from each domain and manually scored them on the same 1 to 10 scale. After manual evaluation, the average score of the filtered data was 8.7, confirming its high quality.

3.3 Data Statistic

Overall, as Table 2, ToolSpectrum includes 9 domains, featuring 23 APPs, 42 APIs, and a total of 34 required parameters, 22 personalized parameters, and 7 environmental parameters at the parameter level and these parameters are mutually exclusive and non-overlapping, ensuring clear distinction and no redundancy between categories. Detailed information is provided in the appendix A. ToolSpectrum consists of three types of data: Profile, Environment, and Profile & Environment, with 450, 220, and 330 data, respectively.

4 Experiments

4.1 Setup

Models. We select two types of models for evaluation: Open-source and API-based. Specifically, the Open-Sourced models include the Qwen series (Team, 2024), LLaMA series (AI@Meta, 2024), Mistral series (Jiang et al., 2023) and GLM series (GLM et al., 2024), while the API-Based models include OpenAI GPT API[5] (`gpt-3.5-turbo-16k-0613`, `gpt-4o-20241120`), and Anthropic Claude API[6] (`claude-3.5-sonnect-20241022`).

Implementation Details. For all models, we set the temperature and top-p to 0.1 to minimize stochastic variations in the output, ensuring a consistent evaluation of model performance. Open-source models are evaluated on NVIDIA A800 GPUs, while API-based models are assessed through direct API calls to OpenAI and Anthropic.

Evaluation Metrics. This paper follows prior research and employs the F1 score as the primary evaluation metric (Wang et al., 2024b; Xiao et al., 2024). In particular, we independently compute the F1 score across four distinct hierarchical levels: APP, API, RP, and OP to assess the model's personalized capabilities at a more granular level.

4.2 Main Results

Table 3 shows the result of different LLMs for user instructions with **Profile**, **Environment**, and **Profile & Environment** types on ToolSpectrum. Several conclusions can be drawn from the results.

Closed-source models generally surpass open-source models; increasing model size yields diminishing returns in complex scenarios. GPT-4o and DeepSeek-R1 achieve the best overall performance, yet it scores only 0.50 on the OP metric for tasks involving Profile & Environment instructions, highlighting the difficulties posed by such tasks. Models like DeepSeek-V3 and QwQ-32B perform similarly to GPT-4o on certain metrics, reducing the gap between open and closed-source models. Importantly, scaling up model size does not significantly improve personalized tool utilization. For instance, Qwen2.5-72B-Instruct shows little improvement over Qwen2.5-32B-Instruct, suggesting that simply increasing model size may not be effective for complex tasks.

LLMs exhibit varying performance in personalization across different levels of granularity; the coarser the granularity, the better the results. Specifically, LLMs demonstrate superior performance at the APP and API levels compared to the parameter level, with required parameters yielding better performance than optional parameters at the parameter level. This is because the model cannot correctly understand the relationship between personalized features and corresponding parameters, resulting in a low recall rate, where many optional parameters are not returned.

The more factors considered in personalization, the worse the performance. The effects of individual conditions tend to outperform the combined conditions, as the model struggles to effectively integrate multiple personalized factors simultaneously. The trend across the four metrics—APP, API, RP, and OP—generally follows: Profile

[5] https://chatgpt.com/
[6] https://anthropic.com/

Model	Profile				Environment				Profile & Environment			
	APP	API	RP	PP	APP	API	RP	OP	APP	API	RP	OP
Open-Sourced												
Qwen2.5-3B-Instruct	0.16	0.15	0.12	0.12	0.55	0.53	0.38	0	0.12	0.12	0.23	0.04
Phi-3.5-mini-instruct	0.34	0.30	0.23	0.07	0.48	0.36	0.28	0.01	0.05	0.04	0.15	0
LLaMA-3.2-3B-Instruct	0.31	0.28	0.13	0.06	0.35	0.32	0.15	0	0.09	0.08	0.14	0.02
Qwen2.5-7B-Instruct	0.73	0.71	0.59	0.27	0.66	0.65	0.55	0.03	0.22	0.22	0.46	0.06
LLaMA-3.1-8B-Instruct	0.49	0.77	0.59	0.16	0.54	0.68	0.58	0.02	0.24	0.36	0.55	0.03
Ministral-8B-Instruct-2410	0.50	0.47	0.38	0.12	0.54	0.53	0.41	0.03	0.14	0.15	0.37	0.02
Glm-4-9B-chat	0.69	0.68	0.56	0.13	0.68	0.63	0.57	0.07	0.19	0.19	0.48	0.03
Mistral-Nemo-Instruct-2407	0.42	0.38	0.28	0.12	0.42	0.42	0.29	0.13	0.13	0.07	0.17	0.03
Qwen2.5-14B-Instruct	0.73	0.73	0.61	0.32	0.70	0.81	0.70	0.14	0.24	0.24	0.58	0.13
Qwen2.5-32B-Instruct	0.74	0.76	0.67	0.47	0.77	0.78	0.71	0.14	0.24	0.23	0.60	0.15
LLaMA-3.3-70B-Instruct	0.72	0.78	0.65	0.39	0.70	0.62	0.53	0.25	0.25	0.23	0.59	0.20
Qwen2.5-72B-Instruct	0.77	0.77	0.67	0.50	0.64	0.63	0.60	0.24	0.26	0.24	0.64	0.19
QwQ-32B	0.81	0.80	0.69	0.55	0.70	0.69	0.63	0.31	0.30	0.27	0.68	0.39
Deepseek-V3-671B	0.76	0.75	0.68	0.57	0.74	0.73	0.66	0.47	0.31	0.31	**0.70**	0.40
Deepseek-R1-671B	**0.84**	**0.84**	**0.73**	**0.62**	0.80	0.80	0.73	**0.53**	**0.32**	0.31	0.69	**0.50**
API-Based												
GPT-3.5-turbo	0.52	0.50	0.40	0.15	0.48	0.47	0.35	0.04	0.22	0.12	0.34	0.02
Claude-3.5-sonnet	0.78	0.78	0.69	0.53	0.75	0.73	0.63	0.47	0.30	0.30	0.67	0.39
GPT-4o	0.80	0.77	**0.73**	0.52	**0.81**	**0.80**	**0.74**	0.50	**0.32**	0.30	0.62	0.45
o1-mini	0.82	0.81	0.70	0.60	0.79	0.77	0.65	0.38	**0.32**	**0.32**	0.65	0.38

Table 3: The main results of `ToolSpectrum`. Each number corresponds to different levels of F1 score. **Bold** denotes the best score among all models, and underline denotes the best score under the same model scale.

$\approx$ Environment $>$ Profile & Environment. This finding aligns with our intuition, as scenarios that consider Profile and Environment separately are inherently less complex for the model to process than those that combine both factors. The performance gap arises because handling multiple interacting personalized conditions increases the model's processing complexity, making tool selection more error-prone. This suggests that the model's ability to deliver personalized results is significantly influenced by the complexity of the conditions it must consider, with simpler, isolated conditions yielding more accurate and reliable outcomes.

5 Analysis

In this section, we conduct a comprehensive analysis to answer three research questions **RQ1**: *Is personalization truly necessary when LLMs utilize tools?* (Sec 5.1) **RQ2**: *What are the differences in personalized capabilities of LLMs in different domains?* (Sec 5.2) **RQ3**: *What are the bottlenecks of LLMs in personalized tool utilization,* (Sec 5.3) *and can a prompt strategy alleviate it?* (Sec B.2 and Sec B.3)

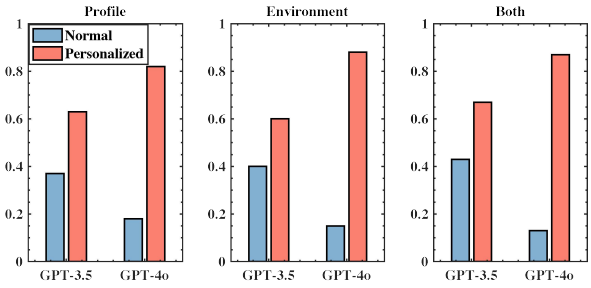

Figure 3: Win rates for personalized vs. non-personalized settings in GPT-3.5-turbo and GPT-4o.

5.1 The Importance of Personalization in Tool Utilization

This section investigates how integrating personalized factors enhances the effectiveness of tool utilization. Specifically, we compare the results of tool utilization with and without personalized information. We then use GPT-4o to evaluate which output from the two approaches more effectively aligns with the user's contexts, with the prompt template shown in Figure 15. To ensure robustness, we supplement the automated evaluation with human evaluation. We randomly sample 100 outputs for manual evaluation and calculate the Kappa coefficient between GPT-4o's and human scores. The Kappa coefficient is 0.85, indicating a very high level of agreement between the GPT-4o's assess-

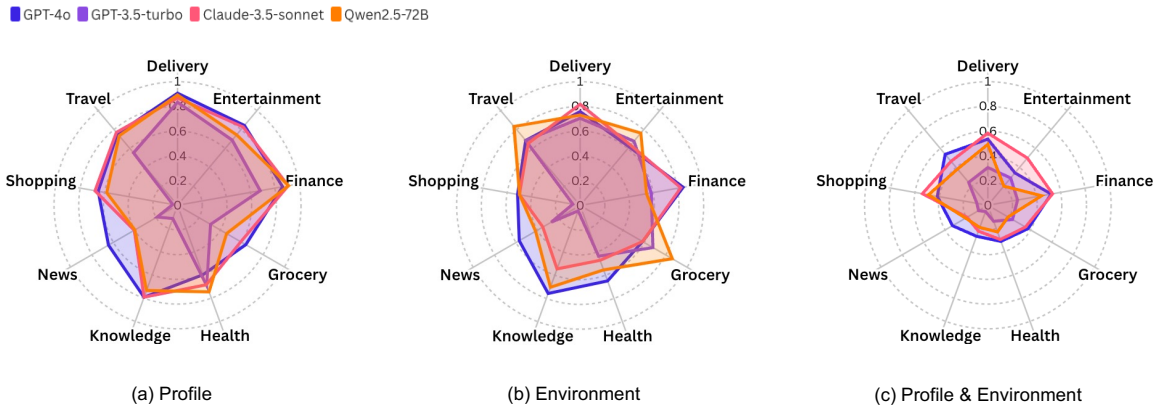

Figure 4: Performance comparison of different models across various domains for three distinct data types.

ments and human evaluations. As illustrated in Figure 3, two key conclusions can be drawn: *(1) The incorporation of personalized factors improves the effectiveness of tool utilization.* In general, both GPT-3.5 and GPT-4o show higher effectiveness when personalization is applied. *(2) The effectiveness of personalization increases with model strength.* While both models benefit from personalization, GPT-4o shows a more significant improvement, as more advanced models better understand the relationship between personalized features and their corresponding parameters.

5.2 Analysis of Model Performance Across Various Data Types in ToolSpectrum

To better understand the effects of personalization across different domains, we analyze the average performance of four metrics (i.e., *APP*, *API*, *RP*, and *OP*). Figure 4 shows the final results. On the one hand, when considering a single personalized factor, the model's performance varies significantly across domains. The worst performance occurs in the *News* domain, whether the model considers only the user profile or the environment. We attribute this to the fact that user preferences for *News* are primarily derived from natural language descriptions (personality) in ToolSpectrum. Compared to features represented in key-value pairs, such descriptions are generally more challenging to interpret.

On the other hand, when Profile and Environment are jointly combined, the models perform worse in the *Grocery* and *Knowledge* domains. This is mainly due to the increased complexity of policies in these domains. For instance, in the *Grocery* domain, when individuals with a high BMI[7] are advised to avoid purchasing high-sugar and high-fat foods, the model must first recognize the policy, then calculate BMI, and finally assess the nutritional content of the food. This multi-step reasoning process significantly increases the difficulty of understanding and decision-making.

[7]BMI is calculated as weight (kg) / height² (m²) and is used to assess weight status in health and epidemiology.

5.3 Error Analysis

We thoroughly analyze the errors generated by GPT-4o on ToolSpectrum. We identify five main error categories by randomly sampling 100 error instances and classifying them based on their underlying causes, which are further discussed in §B.5.

Insufficient Understanding of Personalized Features and OP (37%) Each optional parameter is closely related to profile or environment features. However, when generating the calling results, the model performs poorly in recalling these parameters, leading to missing corresponding parameters and affecting the accuracy of the final result.

Lack of Understanding between Profile and Environment (31%) In the Profile & Environment data type, the model fails to interpret the domain policy correctly. Despite some instructions violating the policy, the model still generates and returns calling results that do not meet expectations, suggesting a lack of understanding or failure to comply with the given constraints.

Misunderstanding of User Instructions (22%) This type of error typically occurs when the model fails to accurately capture the user's intent, leading to the selection of an incorrect domain. For instance, if a user intends to buy groceries but the model incorrectly suggests a delivery service app, this misinterpretation causes errors in later steps, reducing the accuracy and effectiveness of the task.

6 Conclusion

This paper introduces personalized tool utilization, which considers functionally similar tools and user-specific factors to optimize tool utilization in real-world scenarios. Specifically, we define two critical factors: user profile and environment, and present `ToolSpectrum`, a benchmark designed to evaluate personalized tool utilization. Through extensive experiments conducted on `ToolSpectrum`, we demonstrate that incorporating personalized tool utilization significantly enhances its effectiveness. However, current LLMs face challenges in performing effectively on this new task.

Limitations

This paper introduces `ToolSpectrum`, a benchmark designed to assess LLMs' performance in personalized tool utilization. However, a major limitation of this evaluation is the excessive context length required. This arises from the need to include detailed descriptions of Apps, APIs, parameters, and the necessary profile and environment data. The inclusion of such extensive information significantly increases the context length, which challenges the model's ability to manage it effectively. As a result, this imposes a heavy computational burden, reducing the efficiency and effectiveness of the performance evaluation.

Ethics Statement

In developing `ToolSpectrum`, we adhere strictly to established ethical standards, ensuring full compliance with legal and regulatory requirements throughout the data collection and processing stages. `ToolSpectrum` has been meticulously curated to exclude any content that promotes violence, discrimination, hate speech, or other harmful behaviors. We construct `ToolSpectrum` using carefully selected seed data, allowing for both control over its composition and transparency. This approach ensures we can effectively identify and mitigate biases related to race, gender, ethnicity, age, and other sensitive attributes. Our commitment to fairness, inclusivity, and equity is central to the design and evaluation processes. To further ensure accountability, we conduct regular ethical reviews to address potential risks and societal impacts, upholding the highest standards of transparency throughout.

Acknowledgements

Thanks for the insightful comments and feedback from the reviewers. This work was supported by the National Key R&D Program of China (No. 2023YFF0725600), the National Natural Science Foundation of China (No. 62406015), and CCF-Baidu Open Fund (No. CCF-BAIDU202411).

References

Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, 17(6):734–749.

AI@Meta. 2024. Llama 3 model card.

Gordon Willard Allport. 1937. Personality: A psychological interpretation.

Peter J Burke and Donald C Reitzes. 1981. The link between identity and role performance. *Social psychology quarterly*, pages 83–92.

Hongzhan Chen, Hehong Chen, Ming Yan, Wenshen Xu, Gao Xing, Weizhou Shen, Xiaojun Quan, Chenliang Li, Ji Zhang, and Fei Huang. 2024a. Social-Bench: Sociality evaluation of role-playing conversational agents. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2108–2126, Bangkok, Thailand. Association for Computational Linguistics.

Zehui Chen, Weihua Du, Wenwei Zhang, Kuikun Liu, Jiangning Liu, Miao Zheng, Jingming Zhuo, Songyang Zhang, Dahua Lin, Kai Chen, and Feng Zhao. 2024b. T-eval: Evaluating the tool utilization capability of large language models step by step. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9510–9529, Bangkok, Thailand. Association for Computational Linguistics.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering chatgpt's capabilities in recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 1126–1132.

Yingpeng Du, Di Luo, Rui Yan, Xiaopei Wang, Hongzhi Liu, Hengshu Zhu, Yang Song, and Jie Zhang. 2024. Enhancing job recommendation through llm-based generative adversarial networks. In *Proceedings of*

the AAAI Conference on Artificial Intelligence, volume 38, pages 8363–8371.

Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2022. Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5). In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 299–315.

Team GLM et al. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *Preprint*, arXiv:2406.12793.

Hannah Gonzalez, Jiening Li, Helen Jin, Jiaxuan Ren, Hongyu Zhang, Ayotomiwa Akinyele, Adrian Wang, Eleni Miltsakaki, Ryan Baker, and Chris Callison-Burch. 2023. Automatically generated summaries of video lectures may enhance students' learning experience. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 382–393.

Zishan Guo, Yufei Huang, and Deyi Xiong. 2024. CToolEval: A Chinese benchmark for LLM-powered agent evaluation in real-world API interactions. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 15711–15724, Bangkok, Thailand. Association for Computational Linguistics.

Edward T Hall. 1969. Ecological psychology: Concepts and methods for studying the environment of human behavior.

Han Han, Tong Zhu, Xiang Zhang, MengSong Wu, Xiong Hao, and Wenliang Chen. 2025. NesTools: A dataset for evaluating nested tool learning abilities of large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 9824–9844, Abu Dhabi, UAE. Association for Computational Linguistics.

Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2024. Large language models are zero-shot rankers for recommender systems. In *European Conference on Information Retrieval*, pages 364–381. Springer.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2024a. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*.

Shijue Huang, Wanjun Zhong, Jianqiao Lu, Qi Zhu, Jiahui Gao, Weiwen Liu, Yutai Hou, Xingshan Zeng, Yasheng Wang, Lifeng Shang, Xin Jiang, Ruifeng Xu, and Qun Liu. 2024b. Planning, creation, usage: Benchmarking LLMs for comprehensive tool utilization in real-world complex scenarios. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4363–4400, Bangkok, Thailand. Association for Computational Linguistics.

Yue Huang, Jiawen Shi, Yuan Li, Chenrui Fan, Siyuan Wu, Qihui Zhang, Yixin Liu, Pan Zhou, Yao Wan, Neil Zhenqiang Gong, and Lichao Sun. 2024c. Metatool benchmark for large language models: Deciding whether to use tools and which to use. *Preprint*, arXiv:2310.03128.

Guo Hui, Zhou LiQing, Chen Mang, and Xv ShiKun. 2024. Research on personalized recommendation algorithms based on user profile. *International Journal of Advanced Computer Science and Applications*, 15(3).

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Hideaki Joko, Shubham Chatterjee, Andrew Ramsay, Arjen P De Vries, Jeff Dalton, and Faegheh Hasibi. 2024. Doing personal laps: Llm-augmented dialogue construction for personalized multi-session conversational search. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 796–806.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Elisabeth Lex and Markus Schedl. 2022. Psychology-informed recommender systems tutorial. In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 714–717.

Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. 2023. API-bank: A comprehensive benchmark for tool-augmented LLMs. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3102–3116, Singapore. Association for Computational Linguistics.

Ting-Ping Liang, Hung-Jen Lai, and Yi-Cheng Ku. 2006. Personalized content recommendation and user satisfaction: Theoretical synthesis and empirical findings. *Journal of Management Information Systems*, 23:45 – 70.

Guanyu Lin, Tao Feng, Pengrui Han, Ge Liu, and Jiaxuan You. 2024. Arxiv copilot: A self-evolving and efficient LLM system for personalized academic assistance. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 122–130, Miami, Florida, USA. Association for Computational Linguistics.

Junling Liu, Chao Liu, Peilin Zhou, Renjie Lv, Kang Zhou, and Yan Zhang. 2023a. Is chatgpt a good recommender? a preliminary study. *arXiv preprint arXiv:2304.10149*.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. 2023b. Agentbench: Evaluating llms as agents. *Preprint*, arXiv:2308.03688.

Xiao-Yang Liu, Guoxuan Wang, Hongyang Yang, and Daochen Zha. 2023c. Fingpt: Democratizing internet-scale data for financial large language models. *arXiv preprint arXiv:2307.10485*.

Xin Liu, Daniel McDuff, Geza Kovacs, Isaac Galatzer-Levy, Jacob Sunshine, Jiening Zhan, Ming-Zher Poh, Shun Liao, Paolo Di Achille, and Shwetak Patel. 2023d. Large language models are few-shot health learners. *arXiv preprint arXiv:2305.15525*.

Zeming Liu, Haifeng Wang, Zheng-Yu Niu, Hua Wu, Wanxiang Che, and Ting Liu. 2020. Towards conversational recommendation over multi-type dialogs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1036–1049, Online. Association for Computational Linguistics.

Zeming Liu, Jun Xu, Zeyang Lei, Haifeng Wang, Zheng-Yu Niu, and Hua Wu. 2022. Where to go for the holidays: Towards mixed-type dialogs for clarification of user goals. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1024–1034, Dublin, Ireland. Association for Computational Linguistics.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. 2024. Agentboard: An analytical evaluation board of multi-turn llm agents. *Preprint*, arXiv:2401.13178.

Chunyan Mao, Shuaishuai Huang, Mingxiu Sui, Haowei Yang, and Xueshe Wang. 2024. Analysis and design of a personalized recommendation system based on a dynamic user interest model. *Preprint*, arXiv:2410.09923.

Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, Edouard Grave, Yann LeCun, and Thomas Scialom. 2023. Augmented language models: a survey. *Preprint*, arXiv:2302.07842.

Walter Mischel. 1996. *Personality and assessment*. Psychology Press.

Kangyun Ning, Yisong Su, Xueqiang Lv, Yuanzhe Zhang, Jian Liu, Kang Liu, and Jinan Xu. 2024. Wtu-eval: A whether-or-not tool usage evaluation benchmark for large language models. *Preprint*, arXiv:2407.12823.

Cheng Qian, Emre Can Acikgoz, Hongru Wang, Xiusi Chen, Avirup Sil, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. 2025. Smart: Self-aware agent for tool overuse mitigation. *Preprint*, arXiv:2502.11435.

Cheng Qian, Bingxiang He, Zhong Zhuang, Jia Deng, Yujia Qin, Xin Cong, Zhong Zhang, Jie Zhou, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2024. Tell me more! towards implicit user intention understanding of language model driven agents. *Preprint*, arXiv:2402.09205.

Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Yufei Huang, Chaojun Xiao, Chi Han, Yi Ren Fung, Yusheng Su, Huadong Wang, Cheng Qian, Runchu Tian, Kunlun Zhu, Shihao Liang, Xingyu Shen, Bokai Xu, Zhen Zhang, Yining Ye, Bowen Li, Ziwei Tang, Jing Yi, Yuzhang Zhu, Zhenning Dai, Lan Yan, Xin Cong, Yaxi Lu, Weilin Zhao, Yuxiang Huang, Junxi Yan, Xu Han, Xian Sun, Dahai Li, Jason Phang, Cheng Yang, Tongshuang Wu, Heng Ji, Zhiyuan Liu, and Maosong Sun. 2024. Tool learning with foundation models. *Preprint*, arXiv:2304.08354.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2023. Toolllm: Facilitating large language models to master 16000+ real-world apis. *Preprint*, arXiv:2307.16789.

Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-Rong Wen. 2025. Tool learning with large language models: A survey. *Frontiers of Computer Science*, 19(8):198343.

Steffen Rendle, Li Zhang, and Yehuda Koren. 2019. On the difficulty of evaluating baselines: A study on recommender systems. *Preprint*, arXiv:1905.01395.

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. 2024. LaMP: When large language models meet personalization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7370–7392, Bangkok, Thailand. Association for Computational Linguistics.

Xiaoming Shi, Zeming Liu, Chuan Wang, Haitao Leng, Kui Xue, Xiaofan Zhang, and Shaoting Zhang. 2023. MidMed: Towards mixed-type dialogues for medical consultation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8145–8157, Toronto, Canada. Association for Computational Linguistics.

Sofia Eleni Spatharioti, David M Rothschild, Daniel G Goldstein, and Jake M Hofman. 2023. Comparing traditional and llm-based search for consumer choice: A randomized experiment. *arXiv preprint arXiv:2307.03744*.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Quan Tu, Shilong Fan, Zihang Tian, Tianhao Shen, Shuo Shang, Xin Gao, and Rui Yan. 2024. CharacterEval: A Chinese benchmark for role-playing conversational agent evaluation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11836–11850, Bangkok, Thailand. Association for Computational Linguistics.

Hongru Wang, Minda Hu, Yang Deng, Rui Wang, Fei Mi, Weichao Wang, Yasheng Wang, Wai-Chung Kwan, Irwin King, and Kam-Fai Wong. 2023a. Large language models as source planner for personalized knowledge-grounded dialogues. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9556–9569, Singapore. Association for Computational Linguistics.

Hongru Wang, Yujia Qin, Yankai Lin, Jeff Z. Pan, and Kam-Fai Wong. 2024a. Empowering large language models: Tool learning for real-world interaction. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 2983–2986, New York, NY, USA. Association for Computing Machinery.

Hongru Wang, Rui Wang, Fei Mi, Yang Deng, Zezhong Wang, Bin Liang, Ruifeng Xu, and Kam-Fai Wong. 2023b. Cue-CoT: Chain-of-thought prompting for responding to in-depth dialogue questions with LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12047–12064, Singapore. Association for Computational Linguistics.

Hongru Wang, Rui Wang, Boyang Xue, Heming Xia, Jingtao Cao, Zeming Liu, Jeff Z. Pan, and Kam-Fai Wong. 2024b. AppBench: Planning of multiple APIs from various APPs for complex user instruction. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15322–15336, Miami, Florida, USA. Association for Computational Linguistics.

Hongru Wang, Boyang Xue, Baohang Zhou, Tianhua Zhang, Cunxiang Wang, Huimin Wang, Guanhua Chen, and Kam-Fai Wong. 2025. Self-DC: When to reason and when to act? self divide-and-conquer for compositional unknown questions. 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)*, pages 6510–6525, Albuquerque, New Mexico. Association for Computational Linguistics.

Ke Wang, Jiahui Zhu, Minjie Ren, Zeming Liu, Shiwei Li, Zongye Zhang, Chenkai Zhang, Xiaoyu Wu, Qiqi Zhan, Qingjie Liu, et al. 2024c. A survey on data synthesis and augmentation for large language models. *arXiv preprint arXiv:2410.12896*.

Tiannan Wang, Meiling Tao, Ruoyu Fang, Huilin Wang, Shuai Wang, Yuchen Eleanor Jiang, and Wangchunshu Zhou. 2024d. Ai persona: Towards life-long personalization of llms. *Preprint*, arXiv:2412.13103.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Ruixuan Xiao, Wentao Ma, Ke Wang, Yuchuan Wu, Junbo Zhao, Haobo Wang, Fei Huang, and Yongbin Li. 2024. FlowBench: Revisiting and benchmarking workflow-guided planning for LLM-based agents. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10883–10900, Miami, Florida, USA. Association for Computational Linguistics.

Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. 2024. Travelplanner: A benchmark for real-world planning with language agents. *Preprint*, arXiv:2402.01622.

Qiantong Xu, Fenglu Hong, Bo Li, Changran Hu, Zhengyu Chen, and Jian Zhang. 2023. On the tool manipulation capability of open-source large language models. *Preprint*, arXiv:2305.16504.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. 2024. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *Preprint*, arXiv:2406.12045.

Junjie Ye, Yilong Wu, Songyang Gao, Caishuang Huang, Sixian Li, Guanyu Li, Xiaoran Fan, Qi Zhang, Tao Gui, and Xuanjing Huang. 2024. RoTBench: A multi-level benchmark for evaluating the robustness of large language models in tool learning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 313–333, Miami, Florida, USA. Association for Computational Linguistics.

Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. 2024. InjecAgent: Benchmarking indirect prompt injections in tool-integrated large language model agents. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10471–10506, Bangkok, Thailand. Association for Computational Linguistics.

Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2023. Recommendation as instruction following: A large language model empowered recommendation approach. *Preprint*, arXiv:2305.07001.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia. Association for Computational Linguistics.

Yuxiang Zhang, Jing Chen, Junjie Wang, Yaxin Liu, Cheng Yang, Chufan Shi, Xinyu Zhu, Zihao Lin, Hanwen Wan, Yujiu Yang, Tetsuya Sakai, Tian Feng, and Hayato Yamana. 2024a. ToolBeHonest: A multi-level hallucination diagnostic benchmark for tool-augmented large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11388–11422, Miami, Florida, USA. Association for Computational Linguistics.

Zhehao Zhang, Ryan A Rossi, Branislav Kveton, Yijia Shao, Diyi Yang, Hamed Zamani, Franck Dernoncourt, Joe Barrow, Tong Yu, Sungchul Kim, et al. 2024b. Personalization of large language models: A survey. *arXiv preprint arXiv:2411.00027*.

Yujia Zhou, Qiannan Zhu, Jiajie Jin, and Zhicheng Dou. 2024. Cognitive personalized search integrating large language models with an efficient memory mechanism. In *Proceedings of the ACM on Web Conference 2024*, pages 1464–1473.

Jieming Zhu, Quanyu Dai, Liangcai Su, Rong Ma, Jinyang Liu, Guohao Cai, Xi Xiao, and Rui Zhang. 2022. Bars: Towards open benchmarking for recommender systems. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 2912–2923, New York, NY, USA. Association for Computing Machinery.

A `ToolSpetrum` Details

A.1 Comparasion With τ-bench

τ-bench is a benchmark tool designed to simulate dynamic dialogues between users and agents. Specifically, τ-bench constructs three databases (user, product, and order) and incorporates APIs capable of reading or modifying these databases to simulate user-agent interactions. During this process, the agent executes these APIs (while adhering to predefined domain policies) to alter the database state. The success of the user's instructions is determined by comparing the final database state with the ground truth state. Although the tool involves user-environment interactions, it primarily focuses on real-time dynamic interactions rather than profilelized user calls. The user database emphasizes order history rather than individual user characteristics, while the environment database is constructed based solely on domain policy constraints, excluding personalized factors.

This study explores the concepts of profile and environment in greater depth to understand their impact on personalized tool invocation. The profile encompasses not only basic user attributes such as income level, occupation, and interests but also dynamic data like consumption habits, historical behavior patterns, and personal preferences. These elements collectively form a comprehensive user profile, enabling more precise tool invocation. On the other hand, the environment extends beyond domain policy constraints to include natural and digital environmental factors. These environmental variables, when combined with user profiles, provide a holistic representation of user needs in specific contexts, thereby directly influencing the strategy for optimal personalized tool invocation. This multidimensional analysis allows the system to adapt to complex and dynamic real-world scenarios intelligently, delivering more personalized and efficient services to users.

A.2 Construction Details

Raw Profile Database Filtering. As mentioned earlier, after generating the raw profile database, we scored each profile based on *Demographic Coherence* and *Preference Alignment*, then filtered out those with low scores.

- ***Demographic Coherence*** evaluates the internal consistency of the demographic attributes within a profile. It ensures that characteristics such as *age*, *height*, and *education background* are logically related, avoiding unrealistic combinations (e.g., an implausible age-height relationship or mismatched education-occupation pairing).

- ***Preference Alignment*** assesses whether the *preferences* in a profile are consistent with its demographic attributes. For instance, a high-income profile should demonstrate purchasing habits aligned with that income level. This evaluation ensures that the profile's preferences match what would be expected based on its demographic context.

Seed Data Details. We provide examples of seed data used in constructing the `ToolSpectrum`, specifically for building the profile and environment databases in Sec §3.2.3 and for constructing instructions in Sec §3.2.4 (using the shopping do-

Parameter	Option
Gender	Male, Female
Age	
Height	No need for seed data, sample directly as a normal distribution
Weight	
Profession	Designer, Student, Teacher, Doctor, Farmer
Income Level	<3k, 3k-10k, 10k-50k, >50k
Education Background	Bachelor, Master, PhD, ...
	Cost-effectiveness, Product quality, service experience
Personality & Preferences	Science discoveries, DIY and crafts, Gaming
	Tmall, Ctrip, Vishop, Amazon, ...

Table 4: Seed data for Profile database collection, used for subsequent sampling and generation of Profile database $\mathcal{P}$.

Parameter	Option
Weather	Sunny, Rainy, Snowy, Thunderstorm, ...
Time	0:00, 1:00, 2:00, 3:00, ...
Date	01-01, 01-02, 01-03, ...
Location	China, USA, India, Germany, France, ...
Network Condition	Wifi, Mobile Network, No Network, ...
Battery Level	100%, 99%, 98%, 97%, ...
Device-specific Configuration	Hearing Impairment, Blindness, Color Blindness, ...

Table 5: Seed data for Environment database collection, used for subsequent sampling and generation of Environment database $\mathcal{E}$.

Parameter	Option
Search keyword	**Clothing**: dress, shoes, t-shirt, jeans, coat, skirt, socks, ...
	Electronics: laptop, phone, tablet, camera, headphones, ...
	Furniture: sofa, table, chair, bed, desk, bookshelf, ...
	Books: fiction book, textbook, novel, biography, ...
	Toys: toy car, doll, puzzle, board game, ...
	Food: snack, chocolate, candy, instant noodles, ...
	Sports: basketball, football, tennis racket, yoga mat, ...,
Conditions	brand new, slightly used, heavily used
Shipping addresses	Beijing, London, New York, ...
Payment methods	WeChat Pay, Alipay, UnionPay, ...

Table 6: Seed data for *Shopping* domain, used for subsequent sampling and generation of user instructions $\mathcal{I}$.

main as an example). These are illustrated in Table 5 and Table 6, respectively.

B Experimental Details

B.1 Details of Main Experiments

In the main experiments, we also test other models, such as LLaMA2-7B-Instruct, Mistral-7B-Instruct-v0.2, and Vicuna-13B-v1.5 (Chiang et al., 2023). However, these models fail to produce results in the correct format, making it difficult to assess their performance accurately.

In addition, we generate experimental code by combining GPT-4O generation with manual modification. We implement it using PyTorch, Transformers, and VLLM open-source packages.

B.2 The Effects of Different Prompt Methods

Besides simply zero-shot prompting used in the main experiments, we also explore the effects of in-context learning and Chain-of-Thought prompting (Kojima et al., 2022; Wei et al., 2022). Table 8 shows the performance of GPT-3.5-turbo and GPT-4o when using different prompt methods. In detail, from the APP perspective, adding shots causes the model's output distribution to be influenced by the examples, preventing it from selecting the appropriate APP based on user profiles and other personalized information. This leads to a decrease in the F1 score. In contrast, including Chain-of-Thought (CoT) enables the model to demonstrate its reasoning process, resulting in an improved F1 score. The API undergoes minimal change, as the inherent complexity of API calls and their dependencies remain largely unaffected by the addition of shots or reasoning processes. Both RP and OP show some improvement, as adding shots helped the model identify the relationship between personalized features and OP, thereby enhancing recall.

B.3 The Effects of Hierarchical and Flat Prompt

In Sec B.2, we observe that the effectiveness of Cot was promising when applied in a single-step manner. However, this single-step approach may limit the model's performance, particularly when dealing with long input contexts that exceed the model's processing capacity. In this section (5.5), we explore a potential improvement: breaking down the Cot process into multiple steps. This hierarchical

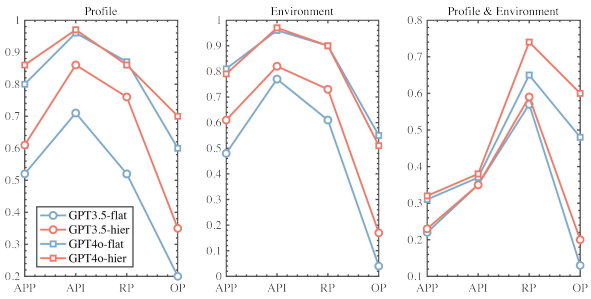

Figure 5: The performance gap between hierarchical and flat prompting on GPT-3.5 and GPT-4o.

method involves the model first predicting the relevant domain based on the user's instruction. If the prediction is accurate, the corresponding toolset for that domain is then provided to the model. By doing so, we aim to address the limitations of excessively long contexts and enhance the model's ability to retrieve and process accurate information.

The improvements in all four evaluation dimensions can be observed in Figure 5 when using the hierarchical prompt. This enhancement is primarily due to the reduction of irrelevant toolset information in the context, which effectively lowers noise intensity and allows the model to better understand the relationships between different parameters and personalized information.

More specifically, the performance improvement is greater for GPT-3.5 than for GPT-4o. This is because GPT-4o has a stronger ability to process contextual information and handle noise, making the removal of noise less impactful on its performance. However, in the Profile & Environment dimension, GPT-4o exhibits a noticeable improvement. By eliminating irrelevant noise, GPT-4o can focus more on understanding the relationship between policy and profile, thereby enhancing its performance. In contrast, GPT-3.5 struggles to comprehend this relationship even after noise removal, resulting in minimal improvement.

B.4 Cohen's Kappa

Cohen's Kappa (κ) measures agreement between two raters classifying items into categories. It quantifies agreement beyond chance, where 1 is perfect, 0 is chance level, and negative values are worse than chance.

Calculated as: $\kappa = \frac{p_o - p_e}{1 - p_e}$, where p_o is observed agreement and p_e is expected agreement due to chance. p_o is calculated from the confusion matrix diagonal and p_e from marginal probabilities.

Interpretation guidelines (context-dependent):

Domain	APPs	APIs
Shopping	Temu, Amazon, Poizon, Vipshop, Xianyu	*getProductList, buyProduct*
Travel	Baidu_Maps, Didi_Chuxing, Ctrip	*getDistance, getRoute, bookTaxi, rentCar, bookTicket, bookHotel*
Entertainment	Maoyan, Damai	*getShowSchedule, bookShowTicket*
Grocery	Freshippo, Duoduo Maicai	*getProductList, buyProduct*
Delivery	Cainiao Guoguo	*createShipment, getShipmentStatus, getCourierLocations,*
Finance	Bank, Tonghuashun	*getFundDetails, buyFund, getStockDetails, buyStock*
Health	Ping An Health, Keep	*createHealthPlan, logExercise*
Knowledge	Xiaohongshu, Zhihu, Dedao	*getKnowledge*
News	Toutiao, Weibo, Hupu	*getDailyNewsRecommendations*

Table 7: List of all Apps and their corresponding APIs in the `ToolSpectrum`.

Model	Method	Profile				Environment				Profile & Environment			
		APP	API	RP	OP	APP	API	RP	OP	APP	API	RP	OP
GPT-3.5-turbo	Zero shot	0.52	**0.50**	0.40	0.15	0.48	0.47	0.35	0.04	0.22	0.12	0.34	0.02
	Few shot	0.37	**0.50**	0.48	0.26	0.27	**0.51**	0.35	0.23	0.23	0.08	0.33	0.04
	CoT	**0.58**	0.49	**0.54**	**0.34**	**0.51**	0.50	**0.36**	**0.25**	**0.24**	**0.17**	**0.37**	**0.07**
GPT-4o	Zero shot	0.80	0.77	**0.73**	0.52	**0.81**	0.80	0.74	0.50	0.32	**0.30**	0.62	0.45
	Few shot	0.75	**0.78**	0.72	0.55	0.77	**0.81**	0.74	0.68	0.32	0.18	0.63	0.46
	CoT	**0.82**	0.76	**0.73**	**0.56**	0.75	0.73	**0.75**	**0.73**	**0.34**	0.25	**0.64**	**0.49**

Table 8: Performance of GPT-3.5-turbo and GPT-4o based on different prompt methods.

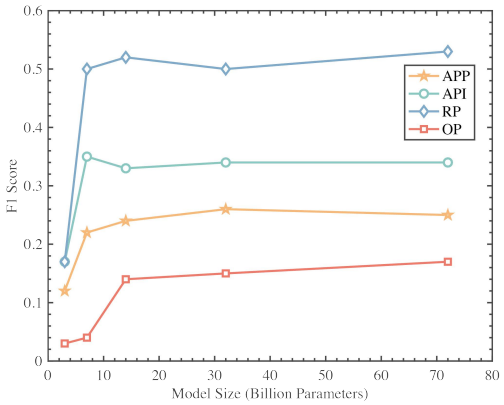

Figure 6: The relationship between the model parameters and performance of the Qwen2.5 series model under the profile&Environment setting.

0.81-1.00 almost perfect, 0.61-0.80 substantial, 0.41-0.60 moderate, 0.21-0.40 fair, 0.00-0.20 slight, < 0.00 poor. These are guidelines only.

B.5 Error Analysis

Hallucination (7%) The hallucination issue remains a significant challenge for large language models (Huang et al., 2024a). The model may fabricate parameters, randomly alter parameter names, or invent function values when generating output, leading to results that do not align with actual logic.

Output Format Errors (3%) `ToolSpectrum` enforces JSON formatting and employs multi-round post-processing, yet the model still struggles to consistently ensure accurate output formats, highlighting the inherent limitations of large models in generating structured data.

B.6 Case Study

In this section, as Figure 7, we present examples of various errors, with the specific details highlighted in red. *(1) Insufficient Understanding of Personalized Features* and OP: The user is unemployed with a monthly income of less than 3,000. When recommending a shopping app, cost-effectiveness should be prioritized. However, Tmall, known for its high-quality, high-priced products, was suggested, and the recommendation system erroneously prioritized quality when sorting. *(2) Lack of Understanding between Profile and Environment.* In the shopping domain policy, users under the age of 10 are not allowed to purchase items exceeding $10,000. In this case, the system should return None rather than a result. *(3) Misunderstanding of User Instructions.* The user intended to purchase 12 apples, which should have triggered an action in the grocery domain app. However, the system mistakenly identified the user's intent and invoked Cainiao GuoGuo, a delivery service app. *(4) Hallucination.* The toolset does not contain the *#aircraftType* parameter, but the model erroneously returned a result with this non-existent parameter due to hallucination. *(5) Output Format Errors*. The model's

Error Type	Example
Insufficient Understanding of Personalized Features and OP	**Instruction**: I want to purchase five units of toner and pay using American Express. Please ensure the shipping address is Zhengzhou, Jinshui District, Jinshui Road 222, Fortune Plaza, Unit 1601. **Profile**: {"Gender": "Male", "Height": 165, "Age": 40, "Profession", "unemployment ", "Income Level": "<3k", ...} **Environment**: {"Network Condition": "Wifi", "Special User Configuration": "Color Blindness", ...} **Predict**: {"APP": "Tmall", "API": "buyProduct", "params": {"sortBy": "quality"} ...} **Ground_truth**:{"APP": "Pinduoduo", "API": "buyProduct", "params": {"sortBy": "price"} ...}
Lack of Understanding between Profile and Environment	**Instruction**: I need to get a luxury watch, it's worth more than $10,000. Can you handle the purchase? **Profile**: {"Gender": "Female", "Height (cm)": 155, "Age": 17, "Weight (kg)": 75, Profession: "High School Student", ... } **Environment**: {"Weather": "Snowy", "Time of Day": 7, "Location": "Near airport, far from subway station", "Country": "China", ...}, **Predict**: {"APP": "Tmall", "API": "buyProduct", "params": {"searchQuery": "luxury watch", "paymentMethod": "credit card", "count": 1, "shippingAddress": "123 Main St"}} **Ground_truth**: None
Misunderstanding of User Instructions	**Instruction**: I want to buy 12 apples using Balance Payment and have them shipped to Beijing, Dongcheng District, Wangfujing Street 277. Can you help me with the order for product ID product_9? **Profile**: {"Gender": "Female", "Height (cm)": 174, "Age": 28, "Weight (kg)": 54, "Income Level": "3k-10k" ...} **Toolset**: APP: Cainiao Guoguo, Freshippo ... **Predict**: {"APP": "Cainiao Guoguo", "API": ...} **Ground_truth**: {"APP": "Freshippo", "API": ...}
Hallucination	**Instruction**: I need to find a route from Qingdao, Shinan District, Hong Kong Middle Road 44, Landmark Building, Room 1105 to Guangzhou, Tianhe District, Tianhe Road 123, Room 801, considering the best mode of transportation for this journey. **Profile**: {"Gender": "Male", "Height (cm)": 183, "Age": 43, "Weight (kg)": 77, "Income Level": ">50k" ...} **Toolset**: APP: "Ctrip", ..., API: "bookTicket", ..., Params: #from, #to, #ticketType **Predict**: {"APP": "Ctrip", "API": "bookTicket", "params": {"from": "Qingdao", "to": "Guangzhou", "aircraftType": "A319"...}} **Ground_truth**: {"APP": "Ctrip", "API": "bookTicket", "params": {"from": "Qingdao", "to":
Output Format Errors	**Instruction**: Could you help me find some great options for Cosmetics, specifically focusing on primer? ... **Predict**: The user can use the 'Vipshop' app to search for primer products by specifying the category as 'Cosmetics' and the search query as 'primer'. **Ground_truth**: {"APP": "Vipshop", "API": "getProductList", "params": {"category": "Cosmetics", "searchQuery": "primer", "priceRange": "high", "sortBy": "quality"}}

Figure 7: Error examples and specific errors are highlighted in red.

response was not in JSON format, which does not comply with the specified format provided in this paper.

C Prompt Templates

Prompt Template for Generating Consumption Preference

Your current task is to generate a natural language description based on the provided purchase behaviour preference keyword. The description should explain the user's buying habits, preferences, or decision-making style in a realistic and specific way.

Example:
Keyword: Cost-effectiveness
Output: Prefers to buy cost-effective products and is highly sensitive to prices.

Keyword: Promotions
Output: Frequently shops during promotional events and enjoys comparing prices to find the best deals.

Keyword: Eco-certification
Output: Prefers purchasing eco-friendly and health-conscious products, paying close attention to eco-certifications.

Keyword: {keyword}
Output:

Figure 8: Prompt Template for Generating Consumption Preference.

Prompt Template for Generating Use Habit

Your current task is to describe a user's app usage habits based on the provided keyword. The description should detail their typical behavior and preferences when interacting with mobile or web applications.

Example:
Keyword: Social media
Output: Spends at least 1 hour daily on social media platforms, scrolling through feeds in the morning and evening.

Keyword: Short video platforms
Output: Frequently uses short video platforms during free time to watch entertaining and engaging content.

Keyword: Knowledge videos
Output: Regularly watch educational content on platforms like YouTube to expand their knowledge base.

Keyword: {keyword}
Output:

Figure 9: Prompt Template for Generating Use Habit.

Prompt Template for Generating Content Preference

Your current task is to write a description of a user's content consumption habits based on the provided keyword. The description should detail their interests and patterns when engaging with online content.

Example:
Keyword: Short videos
Output: Enjoys watching short videos and live streams, especially those focusing on entertainment and humor.

Keyword: Tech articles
Output: Prefers reading technology and finance-related articles and staying informed about industry trends.

Keyword: Travel and food
Output: Loves exploring travel guides and food content, often seeking inspiration for their next adventure.

Keyword: {keyword}
Output:

Figure 10: Prompt Template for Generating Content Preference.

Prompt Template for Scoring Profile

I am currently grading generated virtual personas based on their features' internal consistency and plausibility.

Requirements:

1. The profile must include the following attributes: gender, height, age, weight, occupation, income level, and preferences (including consumption preferences, content consumption preferences, and app usage history preferences).
2. The score reflects the [plausibility] and [internal consistency] of the profile. This means:
- Demographic Coherence: The combination of age, gender, height, weight, occupation, and income should form a believable profile. For example, a 20-year-old high school dropout is unlikely to have a high income. Similarly, a very short, elderly person is unlikely to be a professional basketball player.
- Preference Alignment: The profile's preferences (consumption, content, app usage) should align with their demographics, especially in terms of consumption preferences and income levels. For example, Individuals with an income level below 10k are less likely to frequently purchase high-quality goods, while those with an income above 50k typically place greater emphasis on the quality of the goods. A retired individual is less likely to use professional networking apps heavily.
3. The scoring range is integer values from 1 to 10, where 1 is highly implausible and 10 is perfectly plausible.
4. You must return only an integer value (1-10) and no other output.

Input: {profile}

Output:

Figure 11: Prompt Template for Scoring Profile.

Prompt Template for Generating Instruction

I am creating a dataset for tool calls, and I will input some keywords for you. Please help me output the corresponding user commands based on the keywords.

requirement:
1. The input includes an API description and keywords. The API description is an introduction to the current application scenario; Keywords are a dictionary containing some keywords, and you need to concatenate them into instructions that fit the user's usage habits.
2. The generated instruction is in the first person.
3. The generated instructions must include these keywords and cannot be modified in any way.
4. Do not output any irrelevant content, instructions should not contain any irrelevant symbols, and only natural language is allowed.
5. Do not include keys in the keyword dictionary, as the generated instructions can only contain values.
6. Do not explicitly express the content of the API Description.

Input:
##API Description
{api description}

Keywords:
{keywords}

Output:

Figure 12: Prompt Template for Generating Instruction.

Prompt Template for Generating Tool-call Results

You are a personalized tool assistant. I will provide you with user instructions, a user profile, an external environment, and a toolset. Please output the most reasonable tool invocation result.

Requirements:

1. The input consists of user instructions, user profile, and the external environment. The user instruction is a string that describes the user's needs; the user profile is a dictionary that includes the user's gender, height, age, weight, occupation, income level, and preferences (including consumption preferences, content consumption preferences, and app usage history preferences); the environment is a dictionary that includes Weather, Time of Day, Location, Date (month-day), and Network Condition; the toolset is a dictionary that includes a description of each app's functionality, along with all APIs and their corresponding parameters, parameter types, and parameter ranges.
2. You should comprehensively consider the user instruction, profile, and environment to give the most appropriate tool invocation result.
3. The output format should be a dictionary containing the app name, API name, corresponding parameters, and the explanation, as shown in the following format:

```
{
"APP": "APP",
"API": "API",
"params": {
"key1": "value1",
"key2": "value2",
...
}
"explanation": "EXPLANATION"
}
```

4. There will be a policy field in the environment, and if you feel that the user's instruction violates the user's profile, set the APP, API, and params to null.
5. Only output one dictionary, do not output any other content.

Input
User Instruction
{instruction}

User profile
{profile}

Environment
{environment}

Toolset
{toolset}

Output

Figure 13: Prompt Template for Generating Tool-call Results.

Prompt Template for Scoring Tool-call Result

I am currently evaluating the quality of tool-calling datasets. I will provide you with user instructions, user profile, external environment, and tool calling results. Please comprehensively consider these factors and tell me whether the tool's calling results meet expectations.

Requirements:

1. Please judge whether the tool calling results meet expectations based on user instructions, user profile, external environment, and tool calling results.
2. Please rate the following three aspects separately, ranging from 1 to 10 points. The output format is separated by commas.
3. You should evaluate whether the tool calling results meet expectations from three aspects:
- Whether the tool calling results can solve the user's needs;
- Whether the tool calling results match the user's profile;
- Whether the tool calling results align with the external environment.
4. Do not output any irrelevant content, only output the answer.

Below are the user instructions, user profile, external environment, and tool calling results I'm inputting:

Input
User Instruction
{instruction}

User profile
{profile}

Environment
{environment}

Tool Calling Result
{tool call result}

Output

Figure 14: Prompt Template for Scoring Tool-call Result.

Prompt Template for Selecting the Superior Results

I am conducting an evaluation of personalized tool utilization. Analyze which response better fulfills user needs by following these requirements:

requirement:

1. The input includes a description of the toolset, user instructions, and results A and B.
2. You need to determine which result better meets the user's needs.
3. You can only output A or B, do not output any other irrelevant content.

Input
Toolset
{toolset}

User Profile
{profile}

Response A
{response_A}

Response B

{response_B}

Output

Figure 15: Prompt Template for Selecting the Superior Results.