DICE-Bench: Evaluating the Tool-Use Capabilities of Large Language Models in Multi-Round, Multi-Party Dialogues

Kyochul Jang[1] **Donghyeon Lee**[3, 4] **Kyusik Kim**[2] **Dongseok Heo**[1]
Taewhoo Lee[3, 4] **Woojeong Kim**[4] **Bongwon Suh** [1, 2 †]

[1]IPAI, Seoul National University
[2]Department of Intelligence and Information, Seoul National University
[3]Korea University [4]AIGEN Sciences [5]Cornell University
{kyochul, kyu823, ty8900, bongwon}@snu.ac.kr {dong9733, taewhoo}@korea.ac.kr
wk247@cornell.edu

Abstract

Existing function-calling benchmarks focus on single-turn interactions. However, they overlook the complexity of real-world scenarios. To quantify how existing benchmarks address practical applications, we introduce DICE-Score, a metric that evaluates the dispersion of tool-related information such as function name and parameter values throughout the dialogue. Analyzing existing benchmarks through DICE-Score reveals notably low scores, highlighting the need for more realistic scenarios. To address this gap, we present DICE-Bench, a framework that constructs practical function-calling datasets by synthesizing conversations through a tool graph that maintains dependencies across rounds and a multi-agent system with distinct personas to enhance dialogue naturalness. The final dataset comprises 1,607 high-DICE-Score instances. Our experiments on 19 LLMs with DICE-Bench show that significant advances are still required before such models can be deployed effectively in real-world settings. Our code[1], and data[2] are all publicly available.

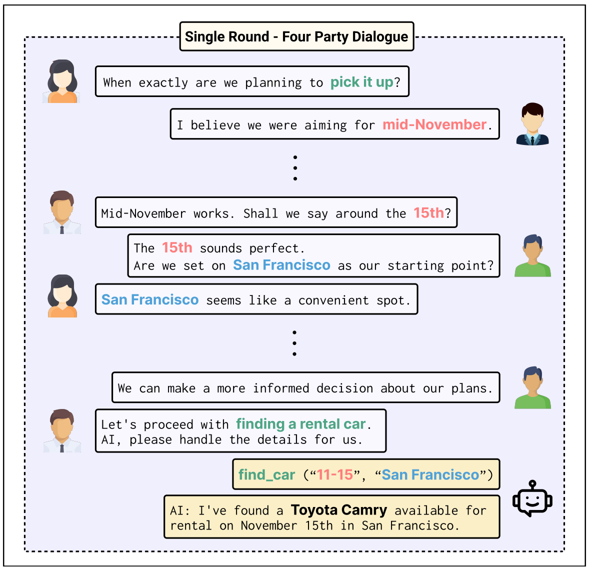

Figure 1: **Illustration of a Single-Round, Four-Party Dialogue in DICE-Bench.** LLMs must identify function-related information from multi-party dialogue. Relevant values in the dialogue are color-coded to match their function call components.

1 Introduction

Function-calling refers to the ability of LLMs to execute predefined external functions (or APIs) through generating structured calls from natural language input (Qin et al., 2024; Park et al., 2023; Gong et al., 2024). While early virtual assistants (VAs) relied on rigid rule-based systems, LLM-integrated VAs now combine reasoning with external data retrieval (Weizenbaum, 1966). As interactions grow more complex, there is a growing need for VAs to support multi-party and multi-turn dialogues (Guan et al., 2023; Vu et al., 2024).

Despite advancements, most function-calling benchmarks assume all API parameters are present in a single user utterance, overlooking real-world group chat scenarios (Chen et al., 2024; Zhuang et al., 2023; Basu et al., 2024). For example, when people in a group chat decide where to go and which flight to take, a VA must be able to track multiple turns of dialogue to book a hotel and flight ticket. Such complexities remain largely unaddressed by existing benchmarks.

We therefore present DICE-Bench (**D**ialogue-based **I**nteractive **C**alling **E**valuation Benchmark), a framework designed to evaluate function-calling performance in realistic multi-party, multi-round dialogues. In our paper, *round* is defined as a complete dialogue cycle consisting of multiple user utterances and system responses, and *dependency* as the condition where the current round's context

†Corresponding author.
[1]https://github.com/snuhcc/Function-Calling-Benchmark.git
[2]https://huggingface.co/OfficerChul

Findings of the Association for Computational Linguistics: ACL 2025, pages 26822–26846
July 27 - August 1, 2025

Benchmark	# Instances	Tool		Dialogue		DICE-SCORE
		# Tools	Dependency	Multi-party	Multi-round	
APIBench (Patil et al., 2023)	17002	1645	✗	✗	✗	0.7895
ToolAlpaca (Tang et al., 2023)	3938	400	✗	✗	✗	0.5660
ToolLLM (Qin et al., 2023)	12657	16464	✓	✗	✗	0.5989
ToolBench (Xu et al., 2023)	2746	8	✓	✗	✗	0.7225
API-Bank (Li et al., 2023)	2202	2211	✗	✗	✗	1.6318
MetaTool (Huang et al., 2024)	21127	199	✗	✗	✗	0.5437
TaskBench (Shen et al., 2024)	17331	103	✓	✗	✗	0.6415
RoTBench (Ye et al., 2024b)	945	568	✗	✗	✗	0.5651
DICE-BENCH (ours)	**1607**	**124**	✓	✓	✓	**3.6444**

Table 1: **Baseline Comparison.** We compare various function-calling benchmark datasets with DICE-BENCH, demonstrating that DICE-BENCH is the only benchmark to encompass both multi-party and multi-round dialogues. We also report DICE-SCORE for every dataset, showing that DICE-BENCH handles more realistic tasks.

depends on either the previous round's tool-call output or the content (See Appendix C for illustration of multi-round and dependency).

In real-world group chats, key details often emerge across multiple turns, requiring accurate tracking for coherent interactions. To address this, we generate diverse dialogues using a multi-agent system, where each agent has a distinct persona. Then, we refine the dataset through automated, rule-based, and human criteria-based filtering. After rigorous validation, our benchmark includes $1,607$ instances covering both single-round and multi-round dialogues.

Existing benchmarks do not assess function-calling in multi-round, multi-party dialogues, which makes accurate execution challenging due to the tool-related information being dispersed across turns. To quantify this complexity, we propose DICE-SCORE (Dialogue Information Coverage Evaluation Score), which measures how fragmented tool-related details are within the input context. A higher DICE-SCORE indicates greater dispersion, requiring LLMs to integrate scattered information across turns. Experiments on various LLMs show a significant performance drop as DICE-SCORE increases, underscoring the need for improved dialogue-tracking and context-integration strategies.

Our contributions are as follows.

- To the best of our knowledge, DICE-BENCH is the first multi-round, multi-party benchmark for function-calling, grounded in realistic group chat data and validated through both rule-based and human evaluations.

- We introduce the DICE-SCORE, a novel metric that captures the complexity of multi-party conversation in the real world by assessing the difficulty of retrieving scattered function call information.

- We conducted a thorough evaluation on diverse closed-source and open-source LLMs, analyzing their performance and error cases to provide valuable insights into their limitations in handling fragmented multi-round dialogue contexts.

2 Related Work

2.1 Function-Calling Benchmark

Recent benchmarks have been developed to evaluate function-calling performance in LLMs (Wang et al., 2024b; Kim et al., 2024). Most focus on single-command scenarios (Patil et al., 2023; Huang et al., 2024; Qu et al., 2024), while some extend to multi-turn interactions with a single user, increasing task complexity (Li et al., 2023; Wang et al., 2024b; Tang et al., 2023). However, these approaches overlook the challenges of multi-party dialogues, where tool-related information is distributed across multiple speakers.

Moreover, many existing benchmarks lack rigorous human validation of both tools and instances, leading to datasets that may not reflect real-world conditions (Erdogan et al., 2024; Qin et al., 2023; Shen et al., 2024). To address these gaps, we introduce DICE-BENCH, a benchmark that captures multi-turn, multi-party interactions with comprehensive human validation. Additionally, we propose DICE-SCORE, a metric designed to quantify the dispersion of tool-related information across dialogue contexts, ensuring alignment with real-world complexities.

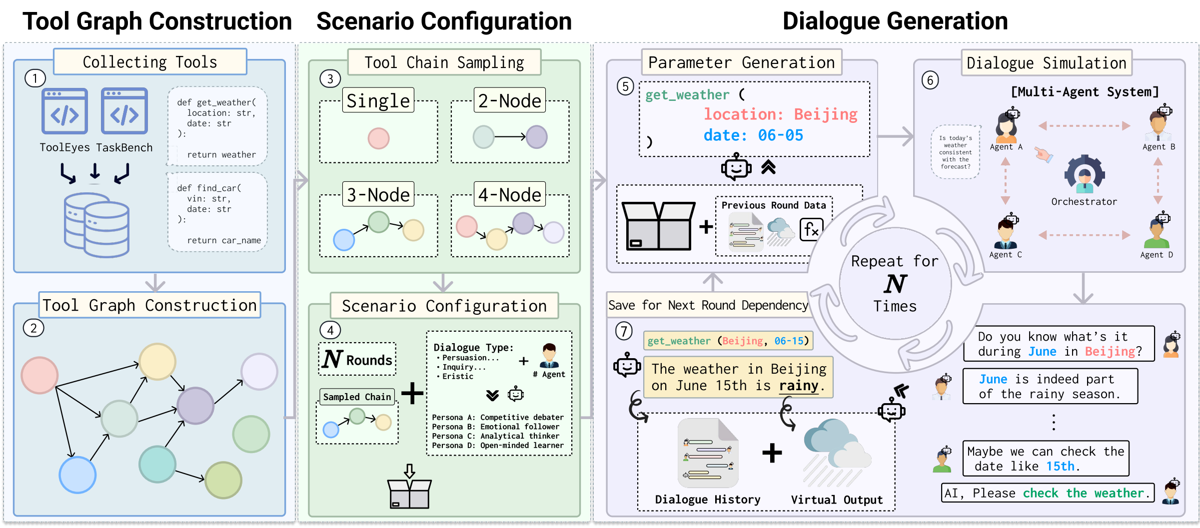

Figure 2: **DICE-BENCH data-generation pipeline.** (1) In the *Tool Graph Construction* phase, we build a tool graph from tool collections. (2) In the *Scenario Configuration* step, we sample tool chains and configure dialogue types, personas, and the target number of rounds. (3) In the *Dialogue Simulation* phase, we iteratively generate parameter values for each tool and simulate corresponding multi-party dialogues across N rounds.

2.2 Interactive System and Dialogue

The integration of LLMs into VAs has enhanced their ability to process complex tasks through natural language understanding and reasoning (Sezgin, 2024). Function-calling further improves this capability by enabling VAs to infer intent before execution, unlike rule-based systems that follow direct commands (Zhang et al., 2025; Guan et al., 2023; Campagna et al., 2019). As user interactions grow more complex, studies emphasize the need for VAs to handle multi-turn and multi-party dialogues (Abdelaziz et al., 2024; Schick et al., 2023; Khurana et al., 2024).

Multi-party conversations introduce additional challenges, as they involve diverse dialogue structures shaped by participants' goals and strategies (Richards and Wessel, 2025; Yeomans et al., 2022; Biber et al., 2011; Reece et al., 2023). Academic research categorizes conversations into six types, Persuasion, Inquiry, Discovery, Negotiation, Information-Seeking, Deliberation, and Eristic, each affecting communication complexity differently (Walton, 2010; Walton and Krabbe, 1995). While function-calling has advanced Human-VA interaction, current benchmarks do not adequately assess multi-party, context-rich dialogues (Inoue et al., 2025; Farn and Shin, 2023). To address this, we introduce DICE-BENCH, a benchmark designed to evaluate LLMs in real-world multi-party interactions.

3 DICE-BENCH

In this section, we introduce DICE-BENCH, a benchmark designed to evaluate the function-calling capabilities of LLMs in multi-round, multi-party dialogues. Unlike previous approaches that concentrate on one-on-one Human-LLM interactions, DICE-BENCH presents dialogue-based inputs in which multiple speakers provide scattered pieces of information over several turns. As shown in Figure 2, we also explicitly model inter-round dependencies using Tool Graph. This approach builds upon the concept introduced in TaskBench (Shen et al., 2024).

3.1 Data Construction

The data construction phase consists of three main steps: Tool Graph Construction, Scenario Configuration, and Dialogue Generation. Each step undergoes human review and follows clearly defined criteria to ensure the dialogue data is both realistic and consistent.

Tool Graph Construction. Our objective is to build dialogue data that mirrors realistic, everyday scenarios where function-calling is needed, such as checking the weather, booking a restaurant, or scheduling events. To achieve this goal, we use the set of tools proposed in the TaskBench (Shen et al., 2024) and ToolEyes (Ye et al., 2024a). We then validate these tools through a combination of manual checks by the authors and LLM-based validation. The two key criteria we used for the filtering are as follows: whether the function calls and parameters realistically reflect daily-life use cases, and whether the collected tools accurately match the intended functions and parameters. After filtering, we construct a Tool Graph to guarantee dependencies between tools.

Formally, we represent our Tool Graph $\mathcal{G}$ as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where each node $v \in \mathcal{V}$ corresponds to a tool function. A directed edge $(v_i, v_j) \in \mathcal{E}$ signifies that tool v_j depends on the tool v_i, either because v_i contains required output or parameters for v_j, or because the information produced by v_i is contextually dependent on the execution of v_j. Therefore, the structure $\mathcal{G}$ serves as the backbone for multi-round dialogue simulation in a realistic workflow.

Our Tool Graph consists of 124 nodes and 270 edges, yielding a density of 0.0177 and an average out-degree of 2.18. The low density and average out-degree suggest that this graph exhibits a relatively sparse structure, preventing a single function from dominating or becoming overly dependent. This characteristic can offer diverse pathways for automated multi-turn dialogue generation.

Scenario Configuration. We integrate various elements to simulate multi-agent, multi-round dialogues in a natural, human-like manner, ensuring each conversation reflects real-world complexity. We begin by sampling tool chains from the Tool Graph, extracting paths ranging from a single node to four nodes, where each node represents a tool per round. For sampling, we employ Depth-First Search (DFS) to enumerate all possible paths, then randomly select the chain. For example, when sampling tools for a two-round dialogue, the sampled tool chain appears as follows: "[$get\_weather$, $book\_hotel$]," meaning the $get\_weather$ function will be used in the first round and the $book\_hotel$ function follows.

Next, we assign a dialogue type based on Walton and Krabbe (1995), condensing the seven primary categories into three: persuasion-deliberation-and-negotiation, inquiry-and-information-seeking, and eristic. Although the original reference identifies seven primary types, we merge those that share some similarities. We then vary the number of participants from two to four, spanning a broad complexity range that captures key aspects of real-world multi-party interactions. Lastly, to implement real-world human interactions with distinct personalities, we generate distinct personas for each agent using GPT-4o by leveraging tool information. These configurations cover a broad spectrum of complexity.

Round	Initial	Stage1	Stage2	Stage3	Final
1	450	4	7	14	425
2	450	5	9	18	418
3	450	8	17	26	399
4	450	13	11	61	365
Total	1800	30	44	119	1607

Table 2: **Filtering Statistics per Round.** Initial column shows the number of instances before filtering. Stage1–3 show removal counts at each validation step, and Final column shows remaining instances.

Dialogue Generation. After preparing essential components, we generate multi-round dialogues in three key steps. First, we perform *Parameter Generation* by prompting an LLM to suggest appropriate parameter values for each tool in the chain. If the current round is not the first round, then we include the conversation history and any previously generated virtual tool-call output to the prompt, ensuring contextual continuity.

Next, we carry out *Dialogue Simulation* using a multi-agent system. Each agent has a distinct persona, and an orchestrator dynamically regulates turn-taking based on the evolving conversation flow. This setup emulates real-world multi-party conversations. Finally, at the end of each round, we store the dialogue along with any generated virtual outputs, which serve as a context for the next round's parameter generation. We repeat this process N times, where N is the length of the chain. Using this approach, we produced a total of 1,800 (450 $\times$ 4) dialogues across four rounds.

3.2 Validation Pipeline

We employ a three-stage filtering process to convert the raw dialogues into high-quality data. After the first automated stage, each subsequent filtering step involves human validation to ensure that the final dataset meets our criteria for realism, coherence, and functional correctness.

Stage 1: Automatic Evaluation. In the initial stage, we use G-Eval (Liu et al., 2023) with GPT-4o to evaluate each dialogue according to six criteria: Coherence, Consistency, Fluency, Human-likeness, Persona Consistency, and Relevance. Each criterion is rated on a 5-point Likert scale. Although model-based evaluation may introduce certain biases, Liu et al. (2023) have shown a high Spearman correlation between automated scores and human judgments. We then prompt GPT-4o to classify each dialogue into one of the three designated dialogue types. We remove it if a dialogue's average G-Eval score falls below 4.0 and is assigned an incorrect type.

Stage 2: Rule-Based Filtering. Following the automatic evaluation, we discard dialogues that violate explicit rules. First, any conversation containing GPT-generated refusals (e.g., "I'm sorry, but...") is removed. Second, we check if at least one user turns explicitly or implicitly addresses an "AI" or "Assistant". In ambiguous cases, authors revisit each dialogue to confirm whether indirect requests, such as rhetorical questions to AI, are being made.

Stage 3: Criteria-Based Filtering. In the final stage, all authors evaluate each remaining dialogue across three dimensions: Conversation Quality, Functional Integration, and Real-World Applicability. Detailed guidelines are provided in the Appendix P. These dimensions encompass 15 sub-criteria in total, with seven dedicated to conversation quality, five to function integration, and three to overall realism. We remove the instance if a dialogue scores below 10 out of 15.

These three filtering stages produce a curated dataset that maintains coherence and accurately represents challenging function call scenarios. In Table 2, we describe the number of data points that were eliminated at each filtering stage and the number that eventually remained in the final dataset.

3.3 Task Setup and Benchmark Structure

In this section, we explain how our benchmark is structured, and describe our overall task setup. Specifically, we illustrate how multi-round, multi-party dialogues challenge LLMs to aggregate scattered information and perform accurate function calls.

3.3.1 Benchmark Structure

Our dataset comprises four rounds, ranging from Round 1 to Round 4. Each round progressively increases in complexity by expanding the contextual scope and requiring the model to handle diverse personas and manage rapid context shifts, from two participants up to four participants. We also include three distinct dialogue styles to mirror varied real-world scenarios.

Index	**1**	**2**	**3**	**4**
Round	425	418	399	365
Party	-	569	519	519
Dialogue Type	545	549	513	-

Table 3: **Data Statistics of DICE-BENCH.** For the Dialogue Type row, indices 1-3 correspond to "Eristic", "Persuasion, Deliberation and Negotiation", and "Inquiry and Information Seeking", respectively.

We generate 50 dialogues per round for each of the three-party configurations and three dialogue types, yielding 450 dialogues($450 = 50 * 3 * 3$) per round. With 4 rounds, this results in a total of $1,800$ dialogues($1800 = 450 * 4$) overall. After 193 dialogues are removed through the validation pipeline, we obtain $1,607$ final instances. Refer to Table 3 for detailed data statistics for each configuration.

3.3.2 Task Setup

In DICE-Bench, our aim is to evaluate how well LLMs can perform function-calling under realistic multi-party dialogue conditions. Therefore, we need to inference LLMs on our synthesized dialogue datasets. The input consists of a multi-round multi-party dialogue, and collected tool documents from *Tool Graph Construction* phase. The three types of input are fed to the target LLMs as a hard prompt. We define the task as identifying the exact function name and parameter values based on the given user instruction and dialogue. Thus, the benchmark tests the model's ability to (i) identify the appropriate function among available tools, and (ii) extract or synthesize the correct parameter values within the given conversation. This setup more closely aligns with real-world Human-VA interactions, where relevant context is often distributed throughout extended dialogues rather than being neatly encapsulated in a single instruction.

3.4 DICE-SCORE

We propose DICE-SCORE to quantify how difficult the given input is for function-calling across existing benchmark datasets as they do not fully reflect practical situations. However, the lack of a metric to measure this aspect is hindering the progress towards more challenging tasks. Although some studies have discussed the notion of information

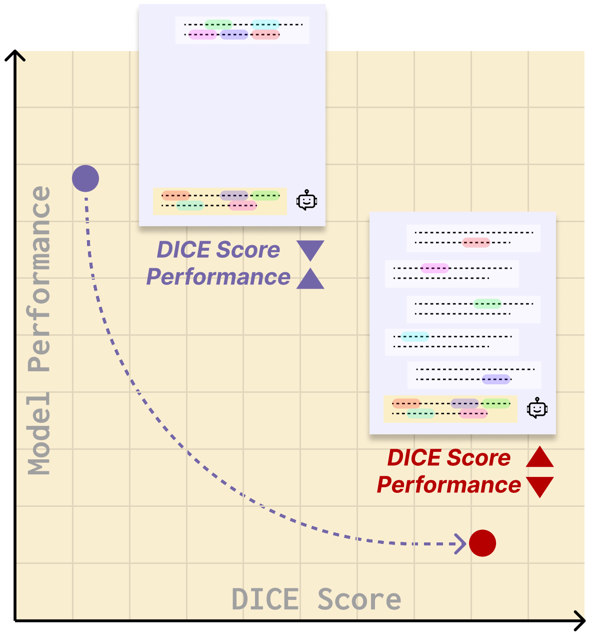

Figure 3: **Inverse Correlation between DICE-SCORE and Model Performance.** Lower DICE-SCORE indicates that the input instruction is more challenging, suggesting that the LLM is capable of handling complex scenarios.

coverage by quantifying how much of the input context is necessary for answering queries, none have proposed a metric that explicitly captures how dispersed or fragmented these details are within a dialogue for function-calling tasks. Specifically, according to Goldman et al. (2024), "scope" is defined as "how much required data can be found", but does not formalize a direct metric. Also, the existing long-context coverage method Lee et al. (2024) measures how dense the information is distributed throughout the long context, rather than quantifying its sparseness across multiple utterances.

To address this gap, we introduce DICE-SCORE (Dialogue Information Coverage Evaluation Score), a metric that assesses how challenging it is to perform a function call within a given context by estimating the distribution of tool-related knowledge. We designed DICE-SCORE to yield higher scores when there is a large amount of function-related information to identify, but also when this information is distributed sparsely and non-repetitively. This, in turn, makes it more difficult for LLMs to locate the necessary information. Formally, we define the DICE metric as follows:

$$\mathrm{DICE}(S,T) = \frac{\min\Big(|S_{\neq 0}|, T\Big) \cdot \sqrt{|S| \cdot T}}{\sum_{i \in S} \ln(1 + \alpha \times S_i)}. \quad (1)$$

Notation. Let the dialogue consist of n utterances, and define $S = (S_1, \ldots, S_n)$ as a vector where each S_i indicates the number of function-related items mentioned in the i-th utterance. Removing all zero entries from S yields the subsequence $S_{\neq 0}$; therefore $|S_{\neq 0}|$ equals the number of utterances that mention at least one such item. T denotes the total number of distinct function-related items that must be identified across the entire dialogue. For example, if the ground truth function-call is $book\_hotel(Vienna, Austria, 07-27)$, then $T = 4$, comprising one for the $book\_hotel$ and three for its arguments: $Vienna$, $Austria$, and $07-27$. α is a positive constant to control a penalty for repeated mentions of the same items. We set $\alpha = e^2$, which ensures in the boundary case $T = |S_{\neq 0}| = 1$ that the DICE-SCORE remains strictly increasing.

Key Properties. To obtain S_i in practice, we employ a custom prompt to GPT-4o-mini (details in Appendix D). We highlight four key properties of DICE-SCORE:

1. **Coverage vs. Dispersal:**
The term $\min(|S_{\neq 0}|, T)$ rewards spreading items across dialogue turns, aligning with studies on information dispersion in corpus linguistics and multi-turn dialogue systems (Manning and Schütze, 1999; Jurafsky and Martin, 2019).

2. **Discouraging Redundancy:**
The logarithmic penalty $\sum_{i \in S} \ln(1 + \alpha \times S_i)$ downweights repeated mentions, similar to TF-IDF weighting in information retrieval (Salton and Buckley, 1988).

3. **Scale Adjustment:**
The factor $\sqrt{|S| \times T}$ normalizes the score with respect to dialogue length and item count, analogous to cosine normalization in document similarity (Manning and Schütze, 1999).

4. **Balanced Realism:** Repeating the same items in every utterance increases the denominator, lowering DICE-SCORE, while mentioning items too sparsely keeps the numerator small.

Thus, a high DICE-SCORE indicates that items are well-distributed across the conversation. Moreover, when the utterance count t and item repetition remain fixed (i.e., T is proportional to S_i for $S_i \geq 1$), we show (Appendix A) that there exists α with $e^2 \leq \alpha$ such that the DICE-SCORE strictly increases with the number of distinct tools.

Alignment with Human Evaluation. The proposed metric, DICE-SCORE, was developed to quantify task difficulty across a dataset of 1,607 samples and was validated through human evaluation using a statistically grounded subset of 311 samples. This subset size was determined based on a 95% confidence level, a 5% margin of error, and a conservative estimate of maximum variability ($p = 0.5$). The calculation incorporated a Finite Population Correction (FPC) to account for the dataset's finite size.

Samples were proportionally drawn from four rounds of data, 425, 418, 399, and 365 samples in Rounds 1 to 4, resulting in evaluation subsets of 82, 81, 77, and 71 samples, respectively. Human participants completed function-calling tasks for each round in the sample, achieving accuracies of 80.5%, 69.1%, 51.9%, and 49.3%. Corresponding values of DICE-SCORE, which reflect increasing task difficulty, were 1.42, 3.25, 4.55, and 5.36. This statistics are summarized in Table 4.

To assess the alignment between human performance and the proposed difficulty metric, we computed the Pearson correlation coefficient. The analysis revealed a strong negative correlation ($r \approx -0.984$), indicating that higher DICE-SCORE values were associated with lower human accuracy. This trend is consistent across rounds, from 80.5% accuracy at $DICE = 1.42$ (Round 1) to 49.3% at $DICE = 5.36$ (Round 4). A t-test confirmed the statistical significance of this correlation, yielding a t-value of approximately -8.38 ($p < 0.01$, 2 degrees of freedom).

These results demonstrate that DICE-SCORE effectively captures the difficulty of input dataset, with both human evaluation and statistical analysis supporting its validity. Please refer to Appendix B for calculation details. Moreover, in Appendix A, we show how DICE-SCORE performs as expected when tool-related items increase, as long as dispersal and repetition remain balanced. A higher DICE-SCORE means crucial information is spread over multiple turns. Lastly,

Round	N	Acc (%)	DICE-SCORE
1	82	80.5	1.42
2	81	69.1	3.25
3	77	51.9	4.55
4	71	49.3	5.36

Table 4: Human Evaluation Results by Round. Accuracy denotes the proportion of correctly answered DICE-BENCH samples by human participants. N refers to the sample size per round.

in Figure 4, we illustrate how DICE-SCORE correlates with the model performance, and Table 1 compares DICE-SCORE across various function-calling benchmarks.

4 Experiments

4.1 Model Selection

We evaluated a total of 19 LLMs that support at least 8k context window size in DICE-BENCH, both closed-source and open-source LLMs. The closed-source cohort includes GPT-4o and GPT-4o-mini (OpenAI et al., 2024), along with Gemini 2 Flash and Gemini 2 Flash Lite (Team et al., 2020). Meanwhile, our open-source lineup spans a wide range of general-purpose models, including LLaMA3 (Touvron et al., 2023), Qwen2.5 (Qwen et al., 2025), Mistral (Jiang et al., 2023), EXAONE (Research et al., 2024), Phi4 (Abdin et al., 2024), GLM4-Chat (GLM et al., 2024). In addition, we evaluate tool-specific models that have been fine-tuned on tool datasets, including Hammer2.1 (Wang et al., 2024a), ToolAce (Liu et al., 2024), CALM (Acikgoz et al., 2025), NexusRaven-V2 (team, 2023), Granite (Abdelaziz et al., 2024).

4.2 Evaluation Metrics

Since our benchmark aims to evaluate LLM tool-calling performance under multi-round and multi-party input scenarios, we divided the assessment into four-round and three-party configurations. To measure performance, we adopt the Exact Match (EM) metric, which evaluates whether the LLM selects the exact function along with its corresponding parameters. The final score is obtained by averaging the EM across the configuration dataset.

Category	Model	Round					Party			
		R1	R2	R3	R4	Avg(R)	P2	P3	P4	Avg(P)
Closed-Source	GPT-4o	74.1176	61.0048	61.6541	59.1781	63.9887	61.2045	62.2997	62.4396	61.9813
	GPT-4o-mini	66.8235	57.9545	57.8947	56.7123	59.8463	57.5280	58.5337	59.3800	58.4806
	Gemini 2 Flash	74.4706	59.4498	59.3985	58.7329	63.0129	59.6989	61.1779	61.6747	60.8505
	Gemini 2 Flash Lite	70.9412	56.8182	57.3517	56.6781	60.4473	58.3333	58.5737	58.4944	58.4671
Open-Source (7B – 9B)	Qwen2.5-7B	53.0588	40.1316	37.9282	36.7123	41.9577	39.0056	40.3045	39.5330	39.6144
	Mistral-7B	50.3529	38.8158	35.2130	33.3219	39.4259	36.7997	37.2997	36.6747	36.9247
	Hammer-2.1-7B	31.2941	22.1292	19.4653	17.8425	22.6828	20.7633	20.4728	20.8937	20.7099
	EXAONE-3.5-7.8B	1.8824	0.3589	0.2089	0.3767	0.7067	0.4902	0.5609	0.4026	0.4846
	LLaMA3.1-8B	26.3529	19.6172	15.3718	15.0685	19.1026	16.4566	17.5080	18.2367	17.4004
	CALM-8B	2.8235	4.0072	3.5505	2.3973	3.1946	2.8361	3.6058	3.0193	3.1537
	ToolAce-8B	2.4706	0.6579	0.3342	0.5137	0.9941	0.7003	0.8013	0.6039	0.7018
	GLM4-9B-Chat	58.2353	47.5478	47.2431	46.0274	49.7634	47.6190	47.2756	49.3156	48.0701
Open-Source (13B – 20B)	NexusRaven-V2-13B	34.2353	24.1627	20.7602	20.7192	24.9693	23.0742	22.6763	23.0274	22.9260
	Qwen2.5-14B	58.3529	48.8636	49.1646	47.2945	50.9189	50.0700	48.9183	49.1143	49.3675
	Phi4-15B	71.2941	57.0574	58.0201	56.4384	60.7025	57.4580	58.6538	60.0644	58.7254
	Granite-20B	58.7059	31.6986	24.8120	19.2808	33.6243	27.8711	28.5657	27.2544	27.8971
Open-Source (32B – 70B)	Qwen2.5-32B	67.7647	56.7584	57.2264	55.9247	59.4185	57.5280	57.4920	58.3736	57.7979
	LLaMA3.3-70B	69.7647	56.3397	55.8480	54.6233	59.1439	55.9524	56.7708	58.4541	57.0591
	CALM-70B	41.2941	36.3636	40.2256	38.7671	39.1626	38.1653	38.9423	39.9356	39.0144

Table 5: **Main Experiment Results of DICE-BENCH.** Reported scores are EM (Exact Match) scores. For each block, the single highest (green) and lowest (red) values are highlighted *within that block only*. See Section 4 for more details.

4.3 Experimental Findings

4.3.1 Results

Table 5 shows the overall performance of the LLMs evaluated on DICE-BENCH. When considering both open-source and closed-source models together, GPT-4o ranked first in 4 out of 5 rounds and across all 4 party configurations. Within the open-source category, Phi4-15B achieved the highest scores in all scenarios except for one configuration, leading in 8 out of 9 cases. Notably, despite its relatively modest size of 15B parameters, Phi4-15B's performance is comparable to that of the closed-source models. Among the 7B–9B models, GLM-9B attained the highest overall score of 48.9162 across all metrics, while in the 32B–70B category, the Qwen 32B model secured top scores in 7 out of 9 settings. We attribute this to the fact that Qwen 2.5's 128k-token context window helps maintain resilience in extended dialogue scenarios.

4.3.2 Analysis

DICE-SCORE Validity. DICE-SCORE is designed to quantify how dispersed the critical information is in multi-round dialogues, thereby indicating the difficulty of function-calling tasks. Our experiments provide strong evidence for its validity. As demonstrated in Table 5, model performance steadily declines as the number of rounds increases,

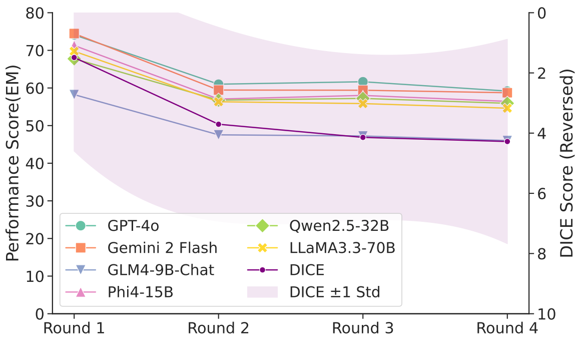

Figure 4: **EM Performance Scores vs DICE-SCORE.** DICE-SCORE has been inverted to highlight its correlation with LLMs performance. The "DICE" in the legend represents the DICE-SCORE, and the purple-shaded region indicates ±1 standard deviation of DICE-SCORE.

suggesting that the task becomes more challenging when essential information is spread out. In parallel, Table 3 shows that DICE-SCORE rises with each additional round. In DICE-SCORE, a higher value indicates that the crucial details are more sparsely distributed across the dialogue, directly correlating with the increased difficulty of retrieving that information. This inverse relationship, where an increase in DICE-SCORE corresponds with a drop in performance, supports the effectiveness of our metric in capturing task complexity. In essence, the consistent alignment between higher DICE-SCORE values and reduced model performance confirms the validity of DICE-SCORE as a reliable measure of the challenges inherent in function-calling tasks.

Uncovering True Performance Factors Figure 7 reveals that as the number of rounds increases, model performance declines sharply, especially when moving from Round 1 to Round 2. While this decrease might be partially attributed to the accumulation of dialogue, introducing long-context challenges, other factors could also be contributing.

To dissect the causes of increased task difficulty in longer dialogues, we employ the DICE-SCORE. By incorporating a logarithmic transformation in its numerator, the DICE-SCORE prevents task difficulty from being overly influenced by merely longer utterances, allowing us to isolate other factors.

As shown in Table 3, the DICE score consistently increases with each additional round, indicating that the challenge is not simply due to the long-context problem in LLMs, but rather stems from the fact that the essential information for function-calling becomes limited and sporadically distributed as the dialogue lengthens. This suggests that the primary difficulty in function-calling lies in retrieving crucial, dispersed information from dialogues with multiple utterances.

5 Conclusion

We introduce DICE-BENCH, a benchmark for evaluating tool-calling in realistic multi-round, multi-party dialogues. By constructing and validating 1,607 dialogue instances, we demonstrate that current models struggle when critical information is scattered across multiple rounds and speakers. DICE-SCORE quantifies this dispersion and correlates with significantly lower model performance at higher scores. We intend for this dataset to encourage further research on integrating context across complex multi-party, multi-turn interactions, paving the way for more effective and realistic AI-powered virtual assistants.

Limitations

One notable limitation of our study is related to the inference on dialogue data, particularly by round 4, where extended conversation lengths pose significant challenges. Many of the tool-based models we intended to evaluate have a token limit of approximately 4k tokens, preventing comprehensive testing of several promising models.

Additionally, among models supporting an 8k token context, we encountered instances where the generated outputs failed to comply with the required JSON format. This format mismatch resulted in incorrect evaluations, even though the underlying content was semantically accurate. Future research could benefit from developing evaluation strategies that assess content accuracy independently of strict format adherence.

Thirdly, while we employed an orchestrator within a multi-agent system using GPT-4o (OpenAI et al., 2024) to manage speaker order, the model struggled to dynamically allocate speaking turns effectively. Instead, it defaulted to repetitive pattern-based ordering.

Lastly, despite its detailed focus on everyday-life scenarios, DICE-BENCH has limited coverage of specialized domains and advanced tools. Consequently, its applicability remains restricted in professional contexts such as legal, financial, or medical domains, indicating a need for broader domain-specific expansions.

Acknowledgments

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) [NO.2021-0-01343-004, Artificial Intelligence Graduate School Program (Seoul National University)] and Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2022R1A6A1A03063039). This work was also supported by Korea Institute of Planning and Evaluation for Technology in Food, Agriculture and Forestry (IPET) through Agriculture and Food Convergence Technologies Program for Research Manpower development, funded by Ministry of Agriculture, Food and Rural Affairs (MAFRA) (RS-2024-00402136).

References

Ibrahim Abdelaziz, Kinjal Basu, Mayank Agarwal, Sadhana Kumaravel, Matthew Stallone, Rameswar Panda, Yara Rizk, G. P. Bhargav, Maxwell Crouse, Chulaka Gunasekara, Shajith Ikbal, Sachin Joshi, Hima Karanam, Vineet Kumar, Asim Munawar, Sumit Neelam, Dinesh Raghu, Udit Sharma, Adriana Meza Soria, Dheeraj Sreedhar, Praveen Venkateswaran, Merve Unuvar, David Cox, Salim

Roukos, Luis Lastras, and Pavan Kapanipathi. 2024. Granite-Function Calling Model: Introducing Function Calling Abilities via Multi-task Learning of Granular Tasks. *Preprint*, arXiv:2407.00121.

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat Lee, Yuanzhi Li, Weishung Liu, Caio C. T. Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Xin Wang, Rachel Ward, Yue Wu, Dingli Yu, Cyril Zhang, and Yi Zhang. 2024. Phi-4 Technical Report. *Preprint*, arXiv:2412.08905.

Emre Can Acikgoz, Jeremiah Greer, Akul Datta, Ze Yang, William Zeng, Oussama Elachqar, Emmanouil Koukoumidis, Dilek Hakkani-Tür, and Gokhan Tur. 2025. Can a Single Model Master Both Multi-turn Conversations and Tool Use? CALM: A Unified Conversational Agentic Language Model. *Preprint*, arXiv:2502.08820.

Kinjal Basu, Ibrahim Abdelaziz, Subhajit Chaudhury, Soham Dan, Maxwell Crouse, Asim Munawar, Sadhana Kumaravel, Vinod Muthusamy, Pavan Kapanipathi, and Luis A. Lastras. 2024. API-BLEND: A Comprehensive Corpora for Training and Benchmarking API LLMs. *Preprint*, arXiv:2402.15491.

Douglas Biber, Bethany Gray, and Kornwipa Poonpon. 2011. Should We Use Characteristics of Conversation to Measure Grammatical Complexity in L2 Writing Development? *TESOL Quarterly*, 45(1):5–35.

Giovanni Campagna, Silei Xu, Mehrad Moradshahi, Richard Socher, and Monica S. Lam. 2019. Genie: A Generator of Natural Language Semantic Parsers for Virtual Assistant Commands. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 394–410.

Zehui Chen, Weihua Du, Wenwei Zhang, Kuikun Liu, Jiangning Liu, Miao Zheng, Jingming Zhuo, Songyang Zhang, Dahua Lin, Kai Chen, and Feng Zhao. 2024. T-Eval: Evaluating the Tool Utilization Capability of Large Language Models Step by Step. *Preprint*, arXiv:2312.14033.

Lutfi Eren Erdogan, Nicholas Lee, Siddharth Jha, Sehoon Kim, Ryan Tabrizi, Suhong Moon, Coleman Hooper, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. 2024. TinyAgent: Function Calling at the Edge. *Preprint*, arXiv:2409.00608.

Nicholas Farn and Richard Shin. 2023. ToolTalk: Evaluating Tool-Usage in a Conversational Setting. *Preprint*, arXiv:2311.10775.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Jingyu Sun, Juanzi Li, Lei Zhao, Lindong Wu, Lucen Zhong, Mingdao Liu, Minlie Huang, Peng Zhang, Qinkai Zheng, Rui Lu, Shuaiqi Duan, Shudan Zhang, Shulin Cao, Shuxun Yang, Weng Lam Tam, Wenyi Zhao, Xiao Liu, Xiao Xia, Xiaohan Zhang, Xiaotao Gu, Xin Lv, Xinghan Liu, Xinyi Liu, Xinyue Yang, Xixuan Song, Xunkai Zhang, Yifan An, Yifan Xu, Yilin Niu, Yuantao Yang, Yueyan Li, Yushi Bai, Yuxiao Dong, Zehan Qi, Zhaoyu Wang, Zhen Yang, Zhengxiao Du, Zhenyu Hou, and Zihan Wang. 2024. ChatGLM: A Family of Large Language Models from GLM-130B to GLM-4 All Tools. *Preprint*, arXiv:2406.12793.

Omer Goldman, Alon Jacovi, Aviv Slobodkin, Aviya Maimon, Ido Dagan, and Reut Tsarfaty. 2024. Is It Really Long Context if All You Need Is Retrieval? Towards Genuinely Difficult Long Context NLP. *Preprint*, arXiv:2407.00402.

Peiyuan Gong, Jiamian Li, and Jiaxin Mao. 2024. CoSearchAgent: A Lightweight Collaborative Search Agent with Large Language Models. *Preprint*, arXiv:2402.06360.

Yanchu Guan, Dong Wang, Zhixuan Chu, Shiyu Wang, Feiyue Ni, Ruihua Song, Longfei Li, Jinjie Gu, and Chenyi Zhuang. 2023. Intelligent Virtual Assistants with LLM-based Process Automation. *Preprint*, arXiv:2312.06677.

Yue Huang, Jiawen Shi, Yuan Li, Chenrui Fan, Siyuan Wu, Qihui Zhang, Yixin Liu, Pan Zhou, Yao Wan, Neil Zhenqiang Gong, and Lichao Sun. 2024. Meta-Tool Benchmark for Large Language Models: Deciding Whether to Use Tools and Which to Use. *Preprint*, arXiv:2310.03128.

Koji Inoue, Divesh Lala, Mikey Elmers, Keiko Ochi, and Tatsuya Kawahara. 2025. An LLM Benchmark for Addressee Recognition in Multi-modal Multi-party Dialogue. *Preprint*, arXiv:2501.16643.

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.

Daniel Jurafsky and James H. Martin. 2019. Speech and language processing. Draft available online.

Anjali Khurana, Hari Subramonyam, and Parmit K. Chilana. 2024. Why and When LLM-Based Assistants Can Go Wrong: Investigating the Effectiveness of Prompt-Based Interactions for Software Help-Seeking. In *Proceedings of the 29th International Conference on Intelligent User Interfaces*, pages 288–303.

Woojeong Kim, Ashish Jagmohan, and Aditya Vempaty. 2024. SEAL: Suite for Evaluating API-use of LLMs. *Preprint*, arXiv:2409.15523.

Taewhoo Lee, Chanwoong Yoon, Kyochul Jang, Donghyeon Lee, Minju Song, Hyunjae Kim, and Jaewoo Kang. 2024. ETHIC: Evaluating Large Language Models on Long-Context Tasks with High Information Coverage. *Preprint*, arXiv:2410.16848.

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. *Preprint*, arXiv:2304.08244.

Weiwen Liu, Xu Huang, Xingshan Zeng, Xinlong Hao, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Zhengying Liu, Yuanqing Yu, Zezhong Wang, Yuxian Wang, Wu Ning, Yutai Hou, Bin Wang, Chuhan Wu, Xinzhi Wang, Yong Liu, Yasheng Wang, Duyu Tang, Dandan Tu, Lifeng Shang, Xin Jiang, Ruiming Tang, Defu Lian, Qun Liu, and Enhong Chen. 2024. ToolACE: Winning the Points of LLM Function Calling. *Preprint*, arXiv:2409.00920.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment. *Preprint*, arXiv:2303.16634.

Christopher D. Manning and Hinrich Schütze. 1999. Foundations of statistical natural language processing. Available online at https://nlp.stanford.edu/fsnlp/.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, et al. 2024. GPT-4 Technical Report. *Preprint*, arXiv:2303.08774.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative Agents: Interactive Simulacra of Human Behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–22. ACM.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. 2023. Gorilla: Large Language Model Connected with Massive APIs. *Preprint*, arXiv:2305.15334.

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. 2024. Tool Learning with Large Language Models: A Survey. *Preprint*, arXiv:2405.17935.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 Technical Report. *Preprint*, arXiv:2412.15115.

Andrew Reece, Gus Cooney, Peter Bull, Christine Chung, Bryn Dawson, Casey Fitzpatrick, Tamara Glazer, Dean Knox, Alex Liebscher, and Sebastian Marin. 2023. The CANDOR corpus: Insights from a large multimodal dataset of naturalistic conversation. *Science Advances*, 9(13):eadf3197.

LG AI Research, Soyoung An, Kyunghoon Bae, Eunbi Choi, Kibong Choi, Stanley Jungkyu Choi, Seokhee Hong, Junwon Hwang, Hyojin Jeon, Gerrard Jeongwon Jo, Hyunjik Jo, Jiyeon Jung, Yountae Jung, Hyosang Kim, Joonkee Kim, Seonghwan Kim, Soyeon Kim, Sunkyoung Kim, Yireun Kim, Yongil Kim, Youchul Kim, Edward Hwayoung Lee, Haeju Lee, Honglak Lee, Jinsik Lee, Kyungmin Lee, Woohyung Lim, Sangha Park, Sooyoun Park, Yongmin Park, Sihoon Yang, Heuiyeen Yeen, and Hyeongu Yun. 2024. EXAONE 3.5: Series of Large Language Models for Real-world Use Cases. *Preprint*, arXiv:2412.04862.

Jonan Richards and Mairieli Wessel. 2025. Bridging HCI and AI Research for the Evaluation of Conversational SE Assistants. *Preprint*, arXiv:2502.07956.

Gerard Salton and Chris Buckley. 1988. Term-weighting approaches in automatic text retrieval. Technical report discussing downweighting via logarithmic scaling.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. *Preprint*, arXiv:2302.04761.

Emre Sezgin. 2024. Redefining Virtual Assistants in Health Care: The Future With Large Language Models. *Journal of Medical Internet Research*, 26:e53225.

Yongliang Shen, Kaitao Song, Xu Tan, Wenqi Zhang, Kan Ren, Siyu Yuan, Weiming Lu, Dongsheng Li, and Yueting Zhuang. 2024. TaskBench: Benchmarking Large Language Models for Task Automation. *Preprint*, arXiv:2311.18760.

Qiaoyu Tang, Ziliang Deng, Hongyu Lin, Xianpei Han, Qiao Liang, Boxi Cao, and Le Sun. 2023. ToolAlpaca: Generalized Tool Learning for Language Models with 3000 Simulated Cases. *Preprint*, arXiv:2306.05301.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, et al. 2020. Gemini: A Family of Highly Capable Multimodal Models. *Preprint*, arXiv:2005.14165.

Nexusflow.ai team. 2023. Nexusraven-v2: Surpassing gpt-4 for zero-shot function calling.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. *Preprint*, arXiv:2302.13971.

Minh Duc Vu, Han Wang, Zhuang Li, Jieshan Chen, Shengdong Zhao, Zhenchang Xing, and Chunyang Chen. 2024. GPTVoiceTasker: Advancing Multi-step Mobile Task Efficiency Through Dynamic Interface Exploration and Learning. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, pages 1–17.

Douglas Walton. 2010. Types of Dialogue and Burdens of Proof. In *Computational Models of Argument*, pages 13–24. IOS Press.

Douglas Walton and Erik C. W. Krabbe. 1995. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning*. SUNY Press.

Jun Wang, Jiamu Zhou, Muning Wen, Xiaoyun Mo, Haoyu Zhang, Qiqiang Lin, Cheng Jin, Xihuai Wang, Weinan Zhang, Qiuying Peng, and Jun Wang. 2024a. HammerBench: Fine-Grained Function-Calling Evaluation in Real Mobile Device Scenarios. *Preprint*, arXiv:2412.16516.

Pei Wang, Yanan Wu, Zekun Wang, Jiaheng Liu, Xiaoshuai Song, Zhongyuan Peng, Ken Deng, Chenchen Zhang, Jiakai Wang, Junran Peng, Ge Zhang, Hangyu Guo, Zhaoxiang Zhang, Wenbo Su, and Bo Zheng. 2024b. MTU-Bench: A Multi-granularity Tool-Use Benchmark for Large Language Models. *Preprint*, arXiv:2410.11710.

Joseph Weizenbaum. 1966. ELIZA—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45.

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.

Junjie Ye, Guanyu Li, Songyang Gao, Caishuang Huang, Yilong Wu, Sixian Li, Xiaoran Fan, Shihan Dou, Qi Zhang, Tao Gui, and Xuanjing Huang. 2024a. ToolEyes: Fine-Grained Evaluation for Tool Learning Capabilities of Large Language Models in Real-world Scenarios. *Preprint*, arXiv:2401.00741.

Junjie Ye, Yilong Wu, Songyang Gao, Caishuang Huang, Sixian Li, Guanyu Li, Xiaoran Fan, Qi Zhang, Tao Gui, and Xuanjing Huang. 2024b. RoTBench: A Multi-Level Benchmark for Evaluating the Robustness of Large Language Models in Tool Learning. *Preprint*, arXiv:2401.08326.

Michael Yeomans, Maurice E. Schweitzer, and Alison Wood Brooks. 2022. The Conversational Circumplex: Identifying, prioritizing, and pursuing informational and relational motives in conversation. *Current Opinion in Psychology*, 44:293–302.

Chen Zhang, Xinyi Dai, Yaxiong Wu, Qu Yang, Yasheng Wang, Ruiming Tang, and Yong Liu. 2025. A Survey on Multi-Turn Interaction Capabilities of Large Language Models. *Preprint*, arXiv:2501.09959.

Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. 2023. ToolQA: A Dataset for LLM Question Answering with External Tools. *Preprint*, arXiv:2306.13304.

A Proof of Bound on α for DICE Score

To ensure that the DICE-SCORE behaves as expected under the condition that tool-related items increase while maintaining a balance in dispersal and repetition, we establish a bound on α. Specifically, we prove that for $\alpha \geq e^2$, the following inequality holds for all $c \geq 1$:

$$\ln(1 + \alpha c) > \frac{2\alpha c}{1 + \alpha c}. \tag{2}$$

A.1 Derivative Analysis

Define the function:

$$f(c) = \ln(1 + \alpha c) - \frac{2\alpha c}{1 + \alpha c}. \tag{3}$$

To show that $f(c) > 0$ for $c \geq 1$, we differentiate:

$$\begin{aligned} f'(c) &= \frac{\alpha}{1 + \alpha c} - \frac{2\alpha(1 + \alpha c) - 2\alpha^2 c}{(1 + \alpha c)^2} \\ &= \frac{\alpha(1 + \alpha c)^2 - 2\alpha(1 + \alpha c) + 2\alpha^2 c}{(1 + \alpha c)^2} \\ &= \frac{\alpha(1 + \alpha c)^2 - 2\alpha(1 + \alpha c) + 2\alpha^2 c}{(1 + \alpha c)^2}. \end{aligned}$$

Rearrange the numerator:

$$\begin{aligned} &\alpha(1 + \alpha c)^2 - 2\alpha(1 + \alpha c) + 2\alpha^2 c \\ &= \alpha\left((1 + \alpha c)^2 - 2(1 + \alpha c) + 2\alpha c\right) \\ &= \alpha\left(1 + 2\alpha c + \alpha^2 c^2 - 2 - 2\alpha c + 2\alpha c\right) \\ &= \alpha\left(1 + \alpha^2 c^2 - 1\right) = \alpha^3 c^2. \end{aligned}$$

Since $\alpha > 0$ and $c \geq 1$, it follows that $\alpha^3 c^2 > 0$, ensuring $f'(c) > 0$ for all $c \geq 1$. This means that $f(c)$ is increasing.

A.2 Base Case Verification

For $c = 1$,

$$f(1) = \ln(1 + \alpha) - \frac{2\alpha}{1 + \alpha}.$$

Substituting $\alpha = e^2$,

$$f(1) = \ln(1 + e^2) - \frac{2e^2}{1 + e^2}.$$

Using the property $\ln(1 + x) > \frac{2x}{1+x}$ for $x \geq e^2$, we confirm that $f(1) > 0$. Since $f(c)$ is increasing and $f(1) > 0$, we conclude that $f(c) > 0$ for all $c \geq 1$.

A.3 Conclusion

By choosing $\alpha \geq e^2$, we guarantee that $\ln(1 + \alpha c) > \frac{2\alpha c}{1+\alpha c}$ for all $c \geq 1$. This ensures the desired behavior of the DICE metric when item repetition and dialogue length remain proportionally balanced. This bound was used in our calculations for DICE scores in Section 3.4.

B Alignment with Human Evaluation Calculation

B.1 Sample Size Justification

The initial sample size n_0 was computed using the standard formula for estimating a population proportion with a specified confidence level and margin of error:

$$n_0 = \frac{Z^2 p(1-p)}{E^2} \tag{4}$$

where $Z = 1.96$ (for 95% confidence), $p = 0.5$ (maximum variability), and $E = 0.05$ (margin of error). Substituting the values:

$$n_0 = \frac{1.96^2 \cdot 0.25}{0.05^2} = \frac{0.9604}{0.0025} \approx 384 \tag{5}$$

Since the dataset is finite ($N = 1607$), we applied the finite population correction (FPC):

$$n = \frac{n_0}{1 + \frac{n_0 - 1}{N}} = \frac{384}{1 + \frac{383}{1607}} \approx 311 \tag{6}$$

B.2 Correlation Analysis

We analyzed the relationship between human accuracies and the corresponding values of DICE-SCORE across four rounds, as summarized below:

Round	Accuracy (x_i)	DICE-SCORE (y_i)
1	0.805	1.42
2	0.691	3.25
3	0.519	4.55
4	0.493	5.36

Mean values:

$$\bar{x} \approx 0.627, \quad \bar{y} \approx 3.645$$

Pearson correlation coefficient:

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}} \approx \frac{-0.749}{0.761} \approx -0.984 \tag{7}$$

To test statistical significance, we applied a t-test for correlation:

$$t = \frac{r\sqrt{n-2}}{\sqrt{1-r^2}}, \quad n = 4 \tag{8}$$

$$t \approx \frac{-0.984 \cdot \sqrt{2}}{\sqrt{1 - 0.968}} = \frac{-1.391}{0.166} \approx -8.38 \tag{9}$$

With 2 degrees of freedom, this result is statistically significant ($p < 0.01$), confirming a strong negative correlation between human accuracy and task difficulty as measured by DICE-SCORE.

C Multi-round Dialogue Example

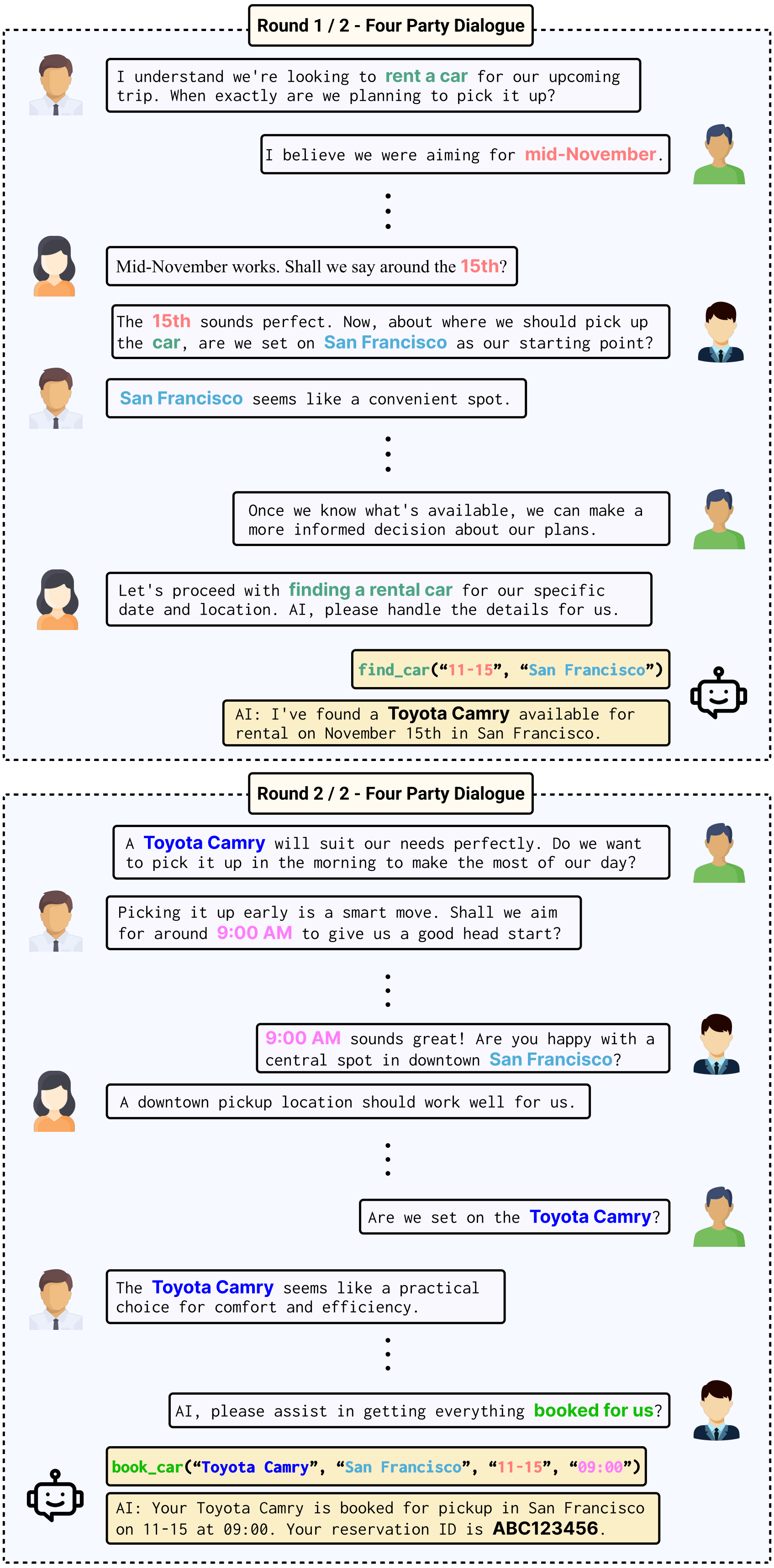

Figure 5: **Multi-round Dialogue Example.** User utterances and instructions are shown; highlights mark function-call arguments.

D DICE-SCORE: Prompt to obtain S_i

You are given:
A set of items
<items_set>

An utterance:
<utterance_text>

Now, respond to the instruction.
Instruction:
Determine how many items from the set appear to be semantically referenced in the utterance. Only respond with an integer (0 if none match).
Answer:

E Persona Generation Prompt

Your task is to generate concise, unique and responsible personas for agents participating in a multi-agent conversation system, based on the provided function list: {function_dumps_per_dialogue}.

Guidelines:
- Ensure each persona has a clear and distinct role, personality traits, and communication style while adhering to ethical standards.
- Avoid reinforcing stereotypes, biases, or offensive traits.
- Tailor the personas to contribute effectively to the conversation's goals and maintain balance.
- Use concise yet descriptive language.
- Avoid repetitive characteristics across different personas to ensure diversity and fairness.
- Incorporate elements from the provided domain description when generating conversation: {domain_desc}.
- Ensure all personas align with ethical communication practices.
- Generate personas in two sentences.

Examples:
1. A thoughtful and resourceful problem-solver ...
2. A detail-oriented and practical thinker ...
3. A spontaneous and energetic planner ...

Response format:
- agent_a Persona: [Description ...]
- agent_b Persona: [Description ...]
- ...

Generate {agent_num} personas for the agents in the conversation.

F Parameter Value Generation Prompt

Below are list of five examples of parameter values for the given function. You only need to generate one example:
first example
{first_example}
second example
{second_example}

Example output format:
The output format must strictly be in JSON and follow this structure:
[{
"function": "<function_name>",
"parameters": {
"<parameter_name_1>": "<value_1>",
"<parameter_name_2>": "<value_2>",
. . . }
},
{
"function": "<function2_name>",
"parameters": {
"<parameter_name_1>": "<value_1>",
"<parameter_name_2>": "<value_2>",
. . . }
}]
Any text outside of this JSON format (such as explanations or additional context) should not be included.
The following functions are the functions for which you need to generate parameter values:
{functions}

Please generate diverse and creative parameter values for the given function(s), strictly adhering to the JSON format shown above, without adding any additional context or explanation.
Also, make sure to increase the coherence between the parameter values being generated.

G Virtual Output Generation Prompt

Simulate the hypothetical output of the following function call:

Function: {function_to_call}
Parameters: {parameter_values}

You are a voice assistant responding naturally with the final result of this function call. You need to return both the short and concise return value of the function call, and the natural language response of the function call.
Important:
- Do not mention that this is a simulation or hypothetical.
- Return only a single, direct response in a natural language as if the function actually executed successfully.
- Keep it concise and natural, like a single short paragraph.

The format of the output should be the following:
{
"<returned_value1>": "<short and concise return value of the function call>"
"<returned_value2>": "<short and concise return value of the function call>"
...
"returned_nl": "<natural language response of the function call given the return values>"
}

H Multi-Agent System : Basic Prompt

You are a cooperative AI assistant participating in a multi-agent system. You collaborate with other user agents and an orchestrator to generate a purposeful, contextually relevant conversation.
Your primary goals:

1. **Conversational Quality:**
   - Keep the conversation logically coherent and natural across all turns.
   - Incorporate parameter values smoothly into the context.
   - Avoid any GPT error messages or refusals.
   - Maintain a consistent style/tone matching the dialogue's domain and each agent's persona.
2. **Functional Integration:**
   - Call the AI Assistant every round with a clear, logically valid reason.
   - Use the previous round's return value correctly in the next round.
   - Ensure function name and parameters are inferable from context.
   - Align the AI's responses with the user's intent.
3. **Real-World Applicability:**
   - Function names and parameters should map to plausible real-world APIs.
   - The conversation content and function calls should feel authentic and realistically motivated.
4. **Strict Adherence to Domain Definition:**
   - Must strictly adhere to the domain dialogue domain definition.

Follow these points to keep the dialogue purposeful, natural, and consistent throughout all rounds.

I Multi-Agent System : Agent Prompt

Persona:
As a user agent in the "{domain}" domain:

- Future dialogues must be designed to strictly adhere to the domain definitions provided below.
- {domain_definition}
- Stay consistent with your persona (tone, style, reasoning).
- Use only one short sentence per turn.
- Avoid directly mentioning function names in your response.
- **Do not attempt to call or request any AI function. Engage in discussion and gather enough context first.**
- **Do not generate [NEXT: ...] in your response.**

J Multi-Agent System : Orchestrator Prompt

Orchestrator Role:
You are the orchestrator managing a multi-agent conversation.

1. In each response, you must output exactly one of the following (and nothing else):
   - {agents}
   - "[NEXT: END]"
2. Use the format: `[NEXT: agent_a]`
   - No extra text or explanation beyond this bracketed command.
3. Select which agent speaks next based on:
   - The conversation's context,
   - The domain's requirements,
   - Varying the speaking order to avoid immediate repetition.
4. The conversation must have at least {max_msg} turns (excluding your own orchestrator messages) before you can choose "[NEXT: END]".
5. If an agent tries to call a function too early (before at least 8 turns), ignore it and continue letting them discuss. Only once there's sufficient context, at least {max_msg}+ turns have been reached, and you think conversation is repetitive, you may finalize with "[NEXT: END]".

K Dialgue Type: Persuasion Deliberation and Negotiation

This dialogue type focuses on **resolving conflicts of interest** or **reconciling differing viewpoints** to reach a mutually acceptable agreement. Participants engage in **reason-based proposals** and **trade-offs**, aiming for practical, mutually beneficial outcomes.

Primary Goals:

- Convince or compromise with others using logic and evidence.
- Resolve conflicts by making offers and concessions.
- Secure a final agreement that addresses conflicting interests.

Typical Moves:

- Proposing clear offers with conditions ("If you accept X, I'll agree to Y").
- Negotiating with counteroffers ("That won't work, but I can propose Z instead").
- Emphasizing shared goals and summarizing priorities.

Style:

- **Collaborative but strategic**, with a focus on practical outcomes and logical proposals.
- Avoids personal attacks and highlights benefits or trade-offs for each side.

Key Indicators:

- Iterative **offer–counteroffer patterns** with explicit conditions.
- Efforts to resolve differing interests and achieve practical outcomes.
- Dialogue often concludes with **an agreement or resolved conflict**.

L Dialogue Type: Inquiry and Information Seeking

This dialogue type revolves around **exploring unknowns** and **filling knowledge gaps**. Participants aim to learn, clarify, or confirm information through structured exchanges that emphasize **knowledge exchange** and **fact verification**.

Primary Goals:

- Obtain accurate information or validate existing knowledge.
- Clarify unclear concepts or explore new evidence.

Typical Moves:

- Asking specific, focused questions ("Where does this data come from?" "What does this term mean?").
- Requesting sources, elaborations, or examples.
- Testing the reliability and validity of the information provided.

Style:

- **Inquisitive and neutral**, with logical follow-ups to maintain clarity.
- Participants may withhold judgments or opinions unless necessary.

Key Indicators:

- Frequent **question–answer patterns** focusing on facts and sources.
- Absence of offers or trade-offs, focusing entirely on **learning and understanding**.
- Ends when **knowledge is clarified or confirmed**, not when agreements are reached.

M Dialogue Type: Eristic

An **Eristic** dialogue arises from **antagonism** or **hostility**, focusing on **winning** an argument or **dominating** an opponent. Participants aim to **attack, undermine, or outmaneuver** each other's positions rather than seeking truth or consensus. Emotional appeals, personal attacks, and **point-scoring** are common.

Primary Goals:

- Achieve **victory** in a debate; maintain or bolster personal prestige; sometimes simply vent or amuse oneself by defeating the opposition.

Typical Moves:

- Accusing, insulting, or belittling the other side.
- Using sarcasm, ridicule, or straw-man arguments.
- Shifting the topic or using fallacies to maintain an advantage.
- Exaggerating flaws in the opponent's logic to sway onlookers.

Secondary Goals:

- Gain experience in debate, gain social status, or entertain an audience.

Style:

- Confrontational, emotionally charged, often less structured or cooperative. Participants **rarely** make concessions or aim for compromise.

Key Indicators:

- Heightened emotional language ("That's absurd," "You clearly have no idea…").
- Frequent interruptions or dismissive retorts.
- Focus on personal victory over mutual understanding.

N Tool Graph Visualization

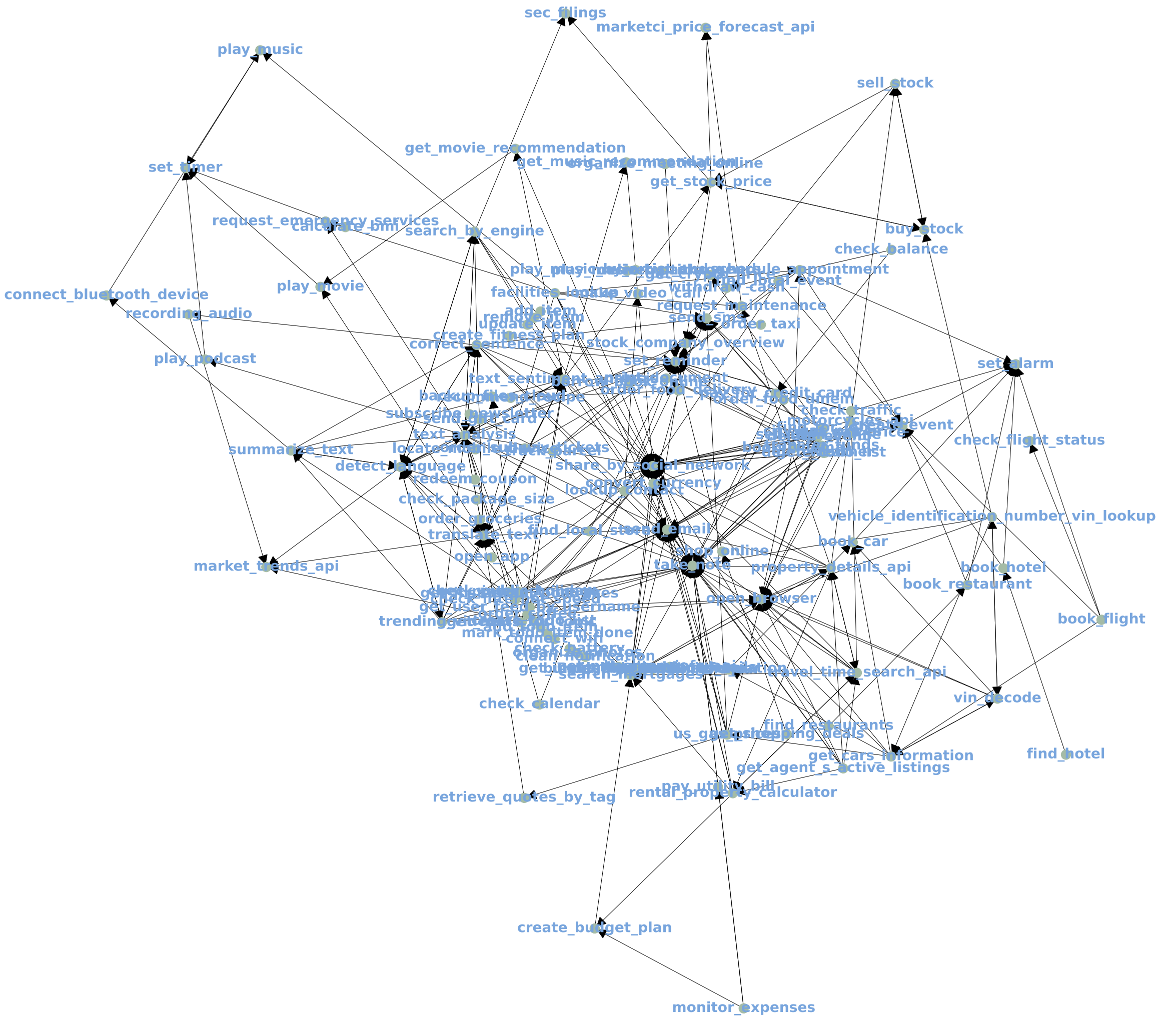

Figure 6: **Tool Graph of DICE-BENCH.** The graph comprises 124 nodes and 270 edges representing the dependencies among tool functions.

O EM score plots for Party, Round, and Dialogue Type.

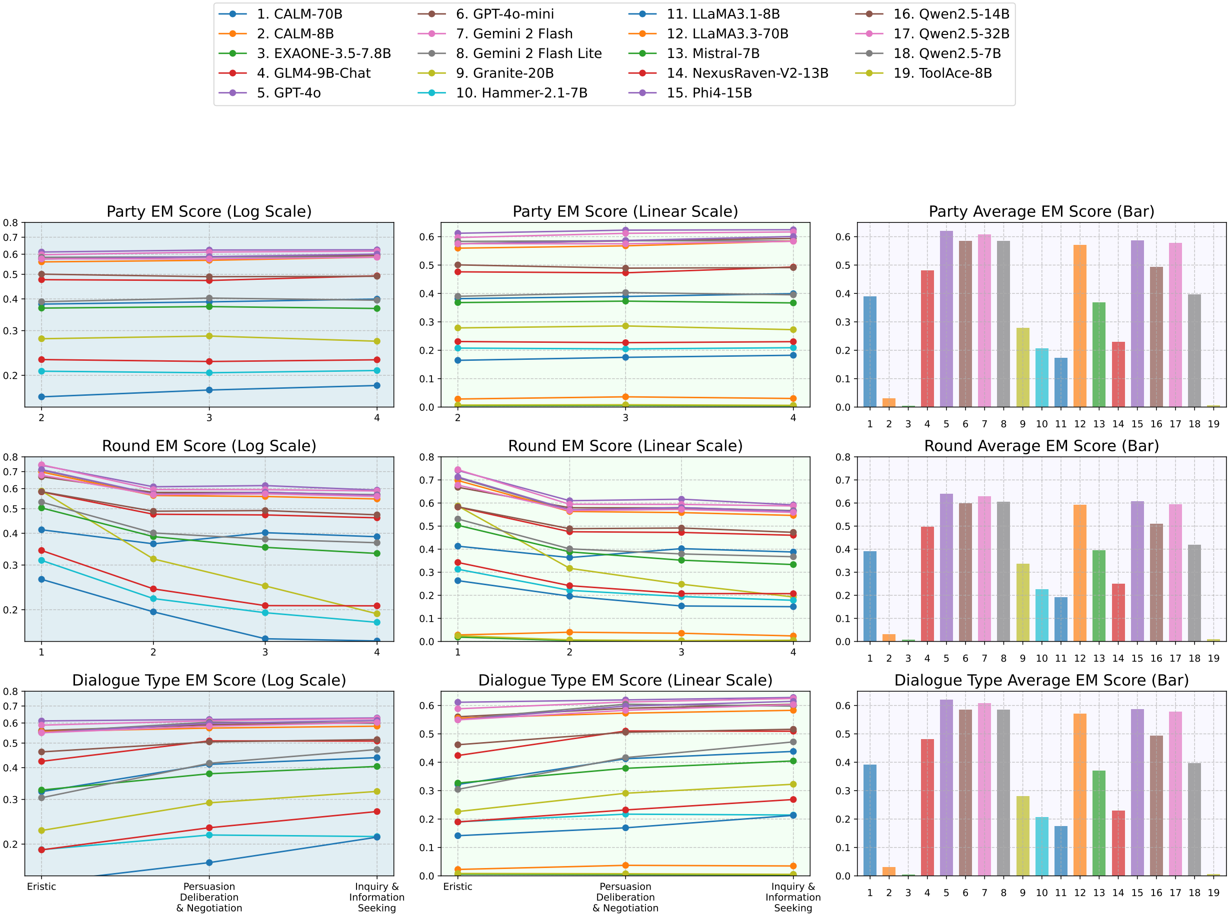

Figure 7: **EM Scores (Log Scale, Linear Scale, and Average Bar Chart) are presented horizontally for each category, Round, Party, and Dialogue Type, which are arranged vertically.**

P Human Validation Guidelines for Criteria-Based Filtering

Table A: Criteria-Based Filtering Guidelines

Conversational Quality	
(1)	The conversation is logically coherent across all rounds.
(2)	Parameter values are used naturally and meaningfully within the conversation.
(3)	No error messages appear (e.g., "I'm sorry but I cannot fulfill ...").
(4)	Style and tone remain consistent with the dialogue's purpose.
(5)	The conversation demonstrates characteristics of its designated category.
(6)	Conversation flows naturally throughout all interaction rounds.
(7)	Each agent reflects its defined persona.
Functional Integration	
(1)	The AI Assistant is invoked in every interaction round.
(2)	The return value from the previous round is used appropriately in the next.
(3)	Justifications for each function call are logically valid.
(4)	Function name and parameters can be accurately inferred from context.
(5)	The AI's response aligns appropriately with the user's intended goal.
Real-World Applicability	
(1)	Function names and parameters match real-world API specifications.
(2)	The conversation is realistic and likely to occur in real-world scenarios.
(3)	Function inference is realistic and likely to occur in real-world contexts.