Talk2Code: A Multi-Turn Interaction Benchmark with Dual-Track Evaluation for Code Generation

Weibin Yang[1*], Liangru Xie[2*], Jieyun Cai[2], Yuxiang Yan[2], Hong-Ning Dai[3], Hao Wang[2†]

[1]Guangzhou Institute of Technology, Xidian University
[2]School of Cyber Engineering, Xidian University
[3]Department of Computer Science, Hong Kong Baptist University
{weibin.yang, cjy, yanyuxiang}@stu.xidian.edu.cn, {xielr, haow}@ieee.org, henrydai@comp.hkbu.edu.hk

Abstract

While large language models (LLMs) have demonstrated strong capabilities in code generation, current benchmarks primarily focus on single-turn scenarios, neglecting the complexity of multi-turn interactions and user diversity. To address this gap, we introduce Talk2Code, the first benchmark for user-stratified multi-turn dialogue code generation evaluation across algorithmic problem-solving and backend programming tasks. A distinctive feature of our benchmark is its user-stratified interaction modeling. For identical coding tasks, we construct dialogue trajectories tailored for novice, intermediate, and expert users, capturing their distinct expectations and communication patterns. To facilitate comprehensive evaluation, we propose a multi-dimensional evaluation framework assessing both code quality and interaction experience through a novel Dual-track Evaluation Method. In the Direct Generation Track, the benchmark provides golden dialogue context (excluding the final code) directly to the LLM for code generation. In contrast, the Interactive Dialogue Track simulates realistic multi-turn interactions, prompting the model to proactively clarify instructions and gather requirements before generating solutions. Code quality is evaluated in both tracks by Test Pass Rate and Success Rate, while interaction experience is assessed exclusively within the Interactive Dialogue Track through subjective and alignment indicators. Our benchmark and multi-dimensional indicator system collectively establish a new paradigm for evaluating adaptive, user-aware AI coding assistants.

Code — https://github.com/logpum/Talk2Code

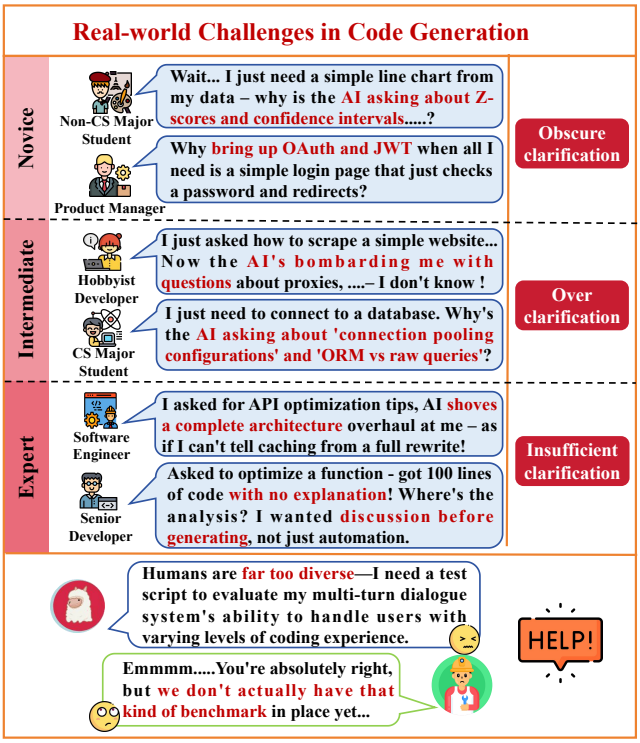


Figure 1: Users with different programming expertise experience distinct challenges—obscure, excessive, or insufficient clarification—in multi-turn code generation dialogues.

Introduction

Large language models (LLMs) have recently demonstrated significant breakthroughs in code generation, increasingly integrating into real-world software development. Existing benchmarks for LLM-based code generation evaluation, e.g., HumanEval (Chen et al. 2021), HumanEval-XL (Peng, Chai, and Li 2024), MBPP (Austin et al. 2021), and APPS (Hendrycks et al. 2021), primarily rely on a single-turn paradigm, prompting models with function signatures and comments to produce code implementations. Recent multi-turn benchmarks—CodeFlowBench (Wang et al. 2025b) focuses on function dependency and code reuse, while CodeIF-Bench (Wang et al. 2025a) emphasizes instruction-following verification-assume consistent user interaction patterns. However, real-world programming often involves more complex interactions where users express unclear or incomplete requirements that cannot be resolved in a single interaction. Thus, LLMs must engage in multi-turn dialogues, proactively clarifying intentions and iteratively refining requirements (Shi, Feng, and Lipani 2022; Zou et al. 2023; Khalid, Alikhani, and Stone 2020) while adapting to diverse user cognitive profiles to generate high-quality code.

Users' programming expertise significantly affects their

*These authors contributed equally.
†Corresponding authors.

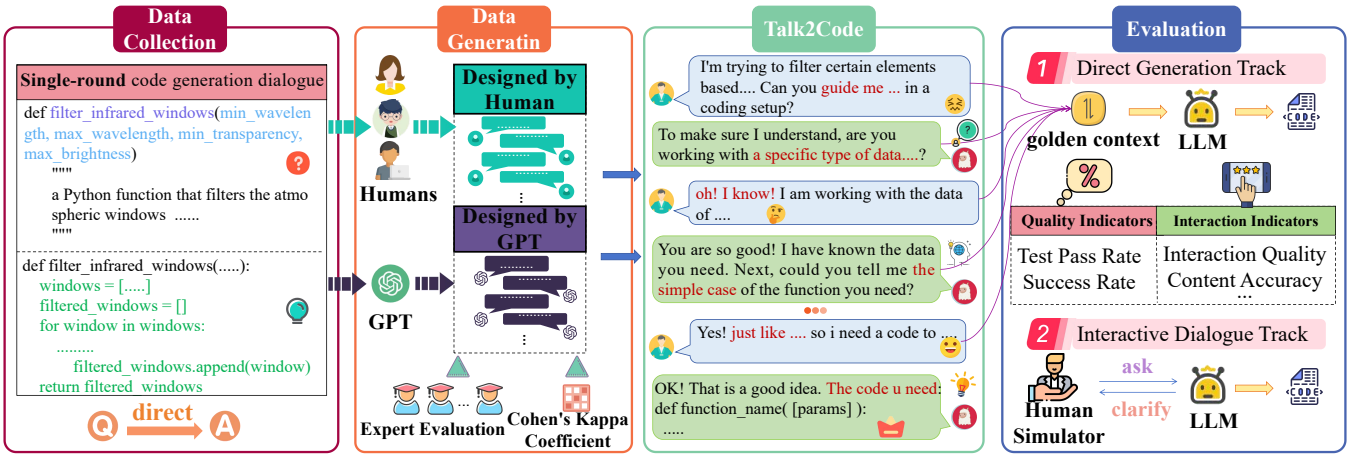


Figure 2: Overview of the Talk2Code Benchmark Construction and Evaluation Framework.

ability to articulate requirements and understand model clarifications (Naszadi, Oliehoek, and Monz 2024; Palta et al. 2025). Take Figure 1 as an example. We observe that 1) novice users find the model's clarification responses overly technical and difficult to grasp; 2) intermediate users perceive model clarifications as excessively detailed and burdensome; 3) experienced programmers often observe model clarifications lacking sufficient depth and precision, thereby resulting in generated code that inadequately satisfies their professional requirements. Such variations significantly influence interaction quality and code outputs, revealing a critical gap: identical dialogue complexity affects different user types in qualitatively different ways—a phenomenon invisible to existing uniform evaluation approaches.

We observe these misalignments manifest concretely in practice. Taking a musical note parsing task as example—converting ASCII symbols ('o', 'o—', '.—') to beat durations: novice users asking 'how does the dictionary work' receive complex error handling discussions instead of simple explanations, leading to verbose manual parsing code rather than the elegant solution `[note_map[x] for x in music_string.split()]`. Intermediate users face cognitive overload when models persistently ask "how should we handle invalid symbols?" despite users already demonstrating task comprehension. Expert users experience inefficient interactions where models ignore explicit specifications like "return a list of integers" and generate inefficient character-by-character parsing instead of leveraging space-separated input structure. These systematic misalignments indicate an urgent need for user-stratified multi-turn dialogue code generation evaluation frameworks.

Aiming to bridge this gap, we introduce Talk2Code, the first comprehensive benchmark explicitly designed for user-stratified multi-turn dialogue-based code generation evaluation, as illustrated in Figure 2. Talk2Code builds upon Multilingual HumanEval (Athiwaratkun et al. 2023) and CoLBench (Zhou et al. 2025) datasets, encompassing algorithmic and backend programming tasks. Grounded in cognitive psychology studies (Howarth and Anderson 2007; Lopes, Lohan, and Hastie 2018; Kurt 2011), we defined six distinct user archetypes and developed structured guidelines for constructing multi-turn dialogues. Data generation combines trained dialogue volunteers and a GPT-4o-based automated pipeline (Zheng et al. 2023), validated through expert evaluation with Cohen's Kappa of 0.81. Talk2Code contains 2,664 dialogues across six user archetypes, comprising 14,396 interactive turns.

As a key innovation of this work, we design a *dual-track*, *multi-dimensional* evaluation framework that systematically integrates code quality and user interaction experience. First, the Direct Generation Track evaluates the model's ability to infer user intent from golden dialogue context (excluding final code) and directly generate code. Second, the Interactive Dialogue Track introduces a human simulator that engages the model in real-time multi-turn clarification, simulating realistic development scenarios. Code quality is measured consistently across both tracks using test-case pass rate and code success rate. For interaction experience, we propose a two-dimensional evaluation method: subjective experience evaluation adopts GPT-4 as an automatic evaluator to score six criteria including question design, cognitive load, interaction fluency, adaptability, communication efficiency, and learning support. Moreover, objective semantic evaluation introduces novel metrics such as cumulative entity consistency and semantic satisfaction, which dynamically trace how well the model understands and refines user requirements throughout the dialogue.

We applied Talk2Code to evaluate 21 LLMs (including 3 closed-source and 18 open-source LLMs), and results reveal significant limitations in both code generation quality and interaction effectiveness under multi-turn conditions.

Our main contributions include:

1. Introducing Talk2Code, the first benchmark for multi-turn dialogue code generation with systematic user-stratified modeling across different programming expertise levels.
2. Developing a comprehensive dual-track evaluation methodology that integrates code quality and interaction experience assessment, including novel objective metrics

for semantic alignment and satisfaction tracking.

3. Demonstrating significant performance gaps in user-adaptive multi-turn scenarios across 21 LLMs, revealing challenges beyond existing dependency and instruction-following perspectives.

We believe our work significantly advances the development and alignment of LLMs within realistic code-generation applications, ultimately improving user satisfaction and broadening the model's practical impact.

Related Work

Code Generation Benchmarks

Code generation benchmarks assess LLM code generation capabilities across varying complexity levels. HumanEval (Chen et al. 2021) established the function-level completion paradigm using docstrings and signatures. MBPP (Austin et al. 2021) extended this to basic programming problems, while APPS (Hendrycks et al. 2021) targeted complex algorithmic challenges. Recent benchmarks expanded evaluation scope and complexity. HumanEval-XL (Peng, Chai, and Li 2024) and mHumanEval (Raihan, Anastasopoulos, and Zampieri 2025) addressed multilingual programming across Python, Java, and other languages. CodeContests (Li et al. 2022) introduced competition-level tasks, while LiveCodeBench (Jain et al. 2024) focused on real-time evaluation. Despite these advances, existing benchmarks remain limited to single-turn evaluation with complete specifications, neglecting the multi-turn interactive nature of real-world programming assistance.

Recent Multi-turn Code Generation Benchmarks

Recent studies explore multi-turn code generation scenarios. CodeFlowBench (Wang et al. 2025b) emphasizes function dependencies and code reuse, while CodeIF-Bench (Wang et al. 2025a) focuses on instruction-following in interactive code generation. MINT (Wang et al. 2024) evaluates tool-augmented task-solving with natural language feedback in multi-turn settings. However, existing benchmarks overlook how user expertise impacts multi-turn interactions. Talk2Code uniquely focuses on user-stratified interaction modeling—evaluating how LLMs adjust dialogue complexity for different programming expertise levels. We hypothesize that code generation effectiveness depends on aligning dialogue complexity with user cognitive capacity—an aspect unexplored in current dependency-focused, instruction-following, and tool-augmented evaluation methods.

Talk2Code Benchmark

Data Construction

Programming User Taxonomy In real-world scenarios, LLM users exhibit substantial variation in programming expertise, interaction preferences, and requirement articulation abilities (Vaithilingam, Zhang, and Glassman 2022). To systematically capture this diversity, we define three proficiency levels: Novice, Intermediate and Expert, with two representative archetypes each, yielding six user profiles: non-CS Major Student, Product Manager, Hobbyist Developer, CS Major Student, Software Engineer, and Senior Developer. For each archetype, we construct persona profiles encompassing programming background, communication style, request types, cognitive tendencies, and interaction expectations. These profiles serve as behavioral templates for generating role-consistent multi-turn dialogues.

Manual Dialogue Curation To create high-quality multi-turn code generation dialogues, we develop a structured design guideline grounded in cognitive psychology and human-computer interaction research. The guideline emphasizes cognitive load theory, problem formulation, and vocabulary preferences across expertise levels. It captures user-specific traits such as domain knowledge, technical fluency, expression clarity, and preferred interaction complexity. For novice users, it recommends simplified, goal-oriented language; for intermediate users, step-by-step clarification; and for experts, concise technical communication focused on performance and architectural precision. We use Multilingual HumanEval (Athiwaratkun et al. 2023) and ColBench (Zhou et al. 2025) as base datasets, covering diverse algorithmic and backend programming tasks. Human dialogue authors with programming expertise transformed single-turn prompts into multi-turn interactions through incremental information disclosure: starting with vague or partial requests and progressively refining constraints and implementation details. Following the self-dialogue strategy in (Byrne et al. 2019), each author acted as both user and assistant to ensure coherence and reduce interpretation drift.

Evaluation Framework

We evaluate LLMs in multi-turn code generation from two perspectives: code quality and interaction experience. An effective model should generate syntactically and functionally correct code, understand user intent, clarify ambiguities, and support user expression throughout interaction. To achieve this, we design a dual-track evaluation method that assesses model performance across diverse real-world scenarios.

Dual-Track Evaluation Method. We adopt a dual-track method to evaluate LLMs in multi-turn code generation, as shown in Figure 3. The *Direct Generation Track* provides full dialogue context (excluding the reference code) to test the model's ability to infer intent and generate correct code under ideal conditions. The *Interactive Dialogue Track* simulates real-world workflows by allowing a human simulator to interact with the model, assessing its ability to clarify intent, adapt to evolving inputs, and produce usable code.

Both tracks share two core metrics: test-case pass rate, measuring functional correctness, and code success rate, reflecting quality judgments. While the Direct Track focuses on static comprehension, the Interactive Track evaluates the interaction process through subjective (e.g., clarity, cognitive load) and objective measures of user intent alignment.

Evaluation Metrics Design. Under the dual-track evaluation method, we define metrics to assess model performance in code quality and interaction. In the Direct Generation Track, we use test-case pass rate and code success rate.

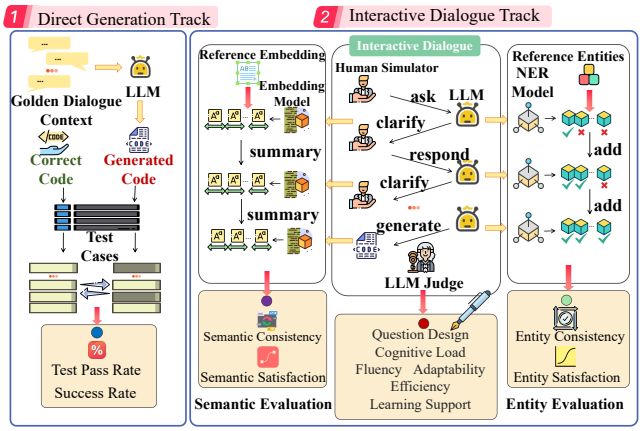


Figure 3: Overview of the Dual-Track Evaluation Method.

Test-case Pass Rate (TP). This metric measures functional correctness by computing the proportion of passed test cases:

$$\text{PassRate} = \frac{N_{\text{passed}}}{N_{\text{total}}}, \tag{1}$$

where N_{passed} is the number of passed test cases, and N_{total} is the total number of test cases.

Success Rate (SR). This metric measures the proportion of generated programs that pass all test cases. The success rate is defined as:

$$\text{SuccessRate} = \frac{1}{M}\sum_{i=1}^{M} s_i, \tag{2}$$

where $s_i \in \{0, 1\}$ is the label for the i-th sample, and M is the total number of evaluated samples.

For the Interactive Dialogue Track, we introduce subjective and objective criteria to evaluate the model's ability to understand and respond to user intent during interaction.

Subjective Evaluation. We follow recent automatic evaluation practices (Dubois et al. 2023; Duan et al. 2024) and use GPT-4o to rate interaction quality across six Likert-scale dimensions: *question design* (QD), *cognitive load* (CL), *interaction fluency* (IF), *user adaptability* (UA), *communication efficiency* (CE), and *learning support* (LS).

To ensure the reliability of our GPT-4o-based subjective scoring, we conducted validation studies: (1) Human-AI agreement: Three expert evaluators independently scored 200 randomly sampled dialogues using our six-dimension framework, achieving Cohen's k = 0.78 with GPT-4o scores; (2) Inter-rater reliability: The same human evaluators achieved k = 0.82 among themselves.

Objective Evaluation. To evaluate how well the model captures user intent during interaction, we define two core similarity functions: **Entity Match Density** ρ_k and **Semantic Similarity** ℓ_k at dialogue turn k .

$$\rho_k = \frac{\left|\bigcup_{i=1}^{k}\left(K_{\text{true}} \cap K_{A_i}\right)\right|}{\left|\left(\bigcup_{i=1}^{k}\left(K_{\text{true}} \cap K_{A_i}\right)\right) \cup \left(K_{A_k} \setminus K_{\text{true}}\right)\right|}, \tag{3}$$

where K_{true} denotes the entity set from the reference answer and K_{A_k} represents the entity set extracted from the model's response at turn k.

The score ℓ_k is derived from cross-entropy under a binary semantic similarity assumption. We have $\ell_k = \log P_k$. Since the target label is set to 1, the expression simplifies to the log-probability of the predicted semantic alignment P_k at turn k.

Building on ρ_k and ℓ_k, we define two global metrics over n dialogue turns to measure overall consistency in the model's understanding: **Cumulative Entity Consistency** (EC) C and **Cumulative Semantic Consistency** (SC) H. C is computed as the CDF of a Beta distribution with fixed parameters $a = b = 2$:

$$\begin{aligned} C &= \int_0^{\rho_{n-1}} \text{pdf}(t; a, b)\, dt \\ &= \frac{1}{B(a,b)} \int_0^{\rho_{n-1}} t^{a-1}(1-t)^{b-1}\, dt \end{aligned}, \tag{4}$$

H is computed as the geometric mean of semantic similarity probabilities up to turn $n-1$:

$$H = e^{\frac{1}{n-1}\sum_{i=1}^{n-1}\ell_k}, \tag{5}$$

These metrics capture how well the model accumulates understanding of user intent across the interaction. To assess turn-level dynamics, we define **Entity Satisfaction** (ES) and **Semantic Satisfaction** (SS) as local indicators. To quantify entity-level progression, **Entity Satisfaction** is defined using incremental metrics derived from changes in ρ_k across turns.

$$\begin{aligned} \Delta_k &= \rho_{k+1} - \rho_k \\ \bar{\Delta} &= \frac{1}{n-1}\sum_{i=1}^{n-2}\Delta_i = \frac{\rho_{n-1} - \rho_1}{n-1} \\ I &= \frac{1}{n-1}\sum_{i=1}^{n-1}\mathbf{1}(\Delta_i > 0) \\ J &= \bar{\Delta} \cdot I. \end{aligned}, \tag{6}$$

In parallel, **Semantic Satisfaction** applies the same formulation to semantic similarity scores ℓ_k. Both metrics capture the 'learning trajectory' in multi-turn interactions—whether the model progressively identifies more correct entities and improves overall problem understanding, rather than treating each turn independently. This is particularly crucial for novice users who struggle to articulate precise requirements initially.

This objective evaluation offers a structured, turn-aware view of how the model incrementally processes and responds to user input, supporting comprehensive assessment at both local and global levels. A representative example of such interaction-level evaluation is illustrated in Figure 4.

Experiments

Experimental Setup

Settings In the Direct Generation Track, experiments use golden dialogue contexts as input, requiring LLMs to

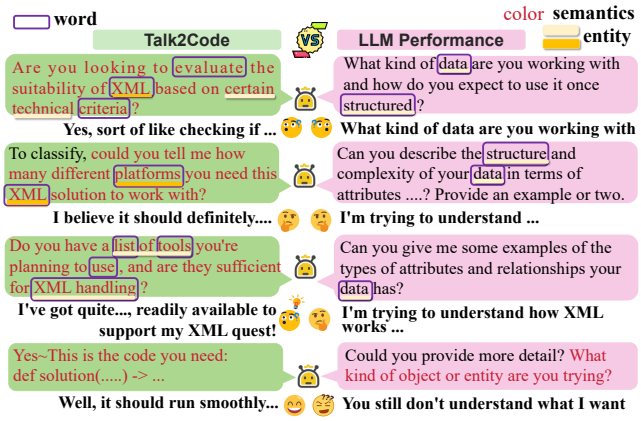


Figure 4: Interaction-Level Evaluation in Talk2Code under the Objective Evaluation.

synthesize multi-turn information and generate code. In the Interactive Dialogue Track, we employ Llama3.3-70B-Instruct (Dubey et al. 2024) as a human simulator for real-time interactions with evaluated LLMs, following prior multi-turn evaluation setups (Zhou et al. 2025) to ensure scalable and reliable benchmarking. To ensure high-fidelity interactions, the simulator operates within a persona-driven role-playing framework, being assigned one of six predefined user archetypes, which dictate its communication style and technical fluency. It initiates the first Talk2Code turn as inquiry and retains access to problem descriptions and reference answers for responding to clarification requests. Each LLM must complete the interaction within 10 turns. We adopt model-specific dialogue formats and system prompts, with temperature and sampling parameters set to defaults. For entity- and semantic-level evaluations, we use Qwen2.5-14B (Qwen et al. 2025) for entity extraction and BGE-M3 (Chen et al. 2024) for sentence embedding, with cosine distance computing embedding similarity.

Models We evaluated 21 mainstream LLMs on the Talk2Code benchmark, encompassing diverse parameter scales and architectural designs. Specifically, our evaluation includes 3 closed-source LLMs (GPT-4o (Hurst et al. 2024), Claude-4-Sonnet (Anthropic 2025), and o1-mini (Jaech et al. 2024)) and 18 open-source LLMs (DeepSeek-V3 (Liu et al. 2024), DeepSeek-R1 (Guo et al. 2025), GLM-4-9B (GLM et al. 2024), LLama-3(1B, 3B, 8B, 70B) (Dubey et al. 2024), Mistral-7B (Jiang et al. 2023), Gemma-3-27B (Team et al. 2025), Qwen2.5(0.5B, 1.5B, 3B, 7B, 14B, 32B) (Qwen et al. 2025), Qwen3(8B, 14B, 235B-A22B) (Yang et al. 2025)).

Evaluation of the performance of multi-turn dialogue code generation

Since novice users represent the core demographic for AI programming tools and are most sensitive to interactive experiences, we present an analysis on the novice-user subset of the Talk2Code benchmark (Figure 5) for illustration. Experimental results are examined across three primary dimensions: code generation quality, interactive experience quality, and objective semantic evaluation, while the complete results for all user groups are provided in the Appendix.

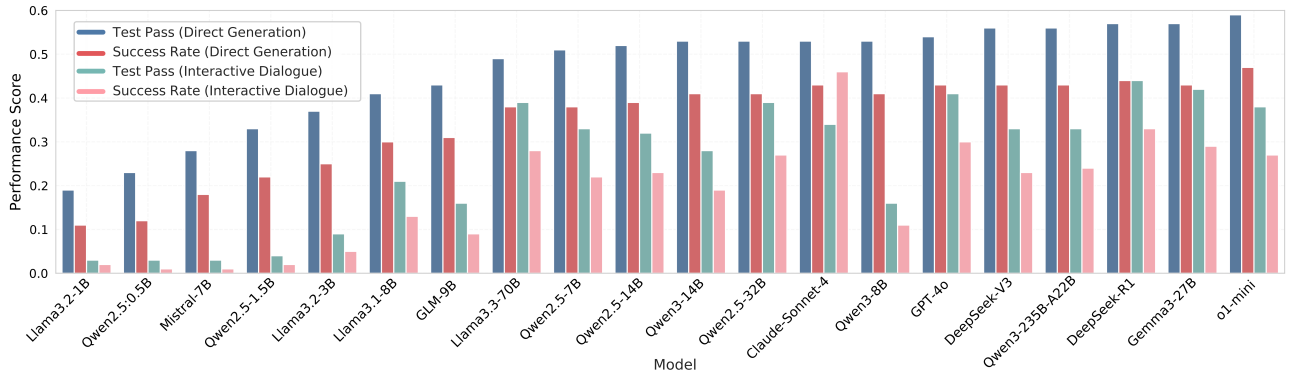
Performance Analysis of Code Generation Quality Figure 5a compares code generation performance across 21 LLMs using our dual-track evaluation methodology, including models renowned for code generation capabilities (DeepSeek-V3, o1-mini, DeepSeek-R1) and interactive conversational capabilities (GPT-4o, Claude-Sonnet-4, GLM-9B). In multi-round dialogue scenarios, traditional model advantages are not fully reflected, revealing inconsistent performance patterns. Even in the Direct Generation Track with complete dialogue context, current LLMs show clear limitations: the best-performing o1-mini reaches only 0.59 Test Pass Rate and 0.47 Success Rate, meaning over 40% of novice requests still fail to produce executable code.

Performance degradation in the Interactive Dialogue Track is even more concerning, directly impacting real-world user experience. In realistic dynamic interaction scenarios where novice users struggle to articulate precise requirements and rely on AI assistants for proactive clarification, all models experience substantial performance drops. GPT-4o, traditionally stronger in interactive capabilities, drops from 0.54/0.43 to 0.41/0.30, while o1-mini plummets from 0.59/0.47 to 0.38/0.27. These results indicate that regardless of whether a model's core strength lies in code generation or dialogue interaction, it remains inadequate for handling multi-turn clarification demands from novice users.

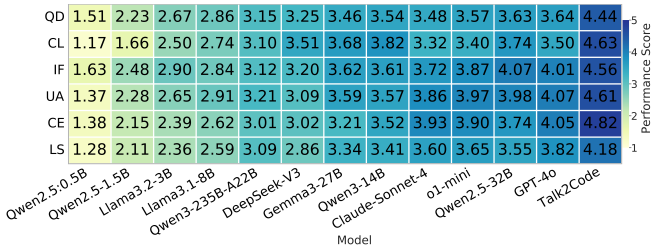
This experimental data highlights the core deficiency of current models: they struggle to identify implicit requirements and gradually drift from true user intent in dynamic interactions. Consequently, novices experience frustration instead of effective assistance, representing a key technical bottleneck for AI coding assistants in educational and entry-level scenarios. The consistent performance gaps across both evaluation tracks underscore the urgent need for more adaptive and user-aware code generation capabilities.

Dialogue Interaction Experience Analysis Figures 5b and 5c present subjective interaction experience and objective assessment metrics for representative models (full results in Appendix), highlighting shortcomings in current LLMs' dialogue capabilities. Across all models, interaction experiences are suboptimal, with the best-performing GPT-4o scoring only 3.85 versus the Talk2Code benchmark average of 4.54. Key deficiencies appear in areas most critical for novice users: GPT-4o reaches 3.82 in learning support and 3.50 in cognitive load management compared to benchmark scores of 4.63, while user adaptability is 4.07, still below the 4.61 benchmark, reflecting limited capacity to adjust interaction strategies across knowledge levels.

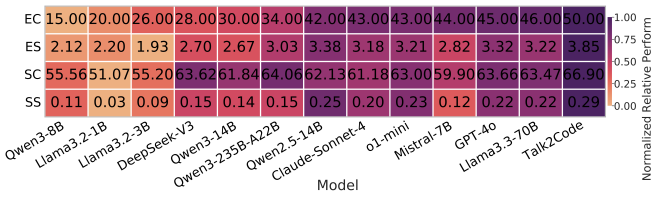
Objective metrics highlight significant issues in user intent understanding. Entity consistency remains consistently low across all models, with top performers achieving below-average levels, causing novice users to encounter mostly irrelevant key elements and increased cognitive load. Semantic consistency performs moderately but falls below benchmark standards, leading to misinterpretations and communication overhead. Satisfaction metrics reveal fundamental limitations in progressive learning capabilities, with both en-



(a) Code generation quality for direct generation and interactive dialogue tracks.



(b) Subjective interaction experience across six dimensions with scores from 1-5.



(c) Objective semantic evaluation measuring entity and semantic consistency/satisfaction across dialogue turns. All values are multiplied by 100 for visualization.

Figure 5: Performance evaluation of LLMs on Talk2Code benchmark for novice users.

tity and semantic satisfaction scores remaining extremely low relative to benchmark levels, indicating rare perceived incremental understanding improvements for novice users.

These findings reveal that current LLMs subject novice users to a "triple dilemma" of substandard code quality, frustrating dialogue experiences, and insufficient learning support. Each dialogue turn resembles a restart rather than coherent progression, directly undermining learning motivation and threatening AI coding assistant sustainability.

Model Adaptability Across Multiple User Types

Table 1 presents the overall performance of different LLM families on the Talk2Code benchmark. For clarity and readability, we report only the largest representative model from each family, as these typically reflect the highest technical sophistication and best achievable performance within their series. Talk2Code serves as the evaluation benchmark in this table, with "-" indicating that code generation metrics (TP and SR in both tracks) are not applicable to the benchmark itself, while interaction experience metrics reflect the reference standards for optimal performance.

Cross-User-Type Variations in Code Generation Quality From the perspective of code generation quality, current LLMs exhibit complex performance patterns across different user types. In the Direct Generation Track, most models display an inverted U-shaped trend, achieving optimal performance in intermediate user scenarios. For instance, DeepSeek-R1 improves its Test Pass Rate from 0.57 with novice users to 0.63 with intermediate users, before slightly dropping to 0.59 for expert users. This phenomenon suggests that current models are better suited for moderately complex requirement descriptions, avoiding ambiguities common with novice users while not yet meeting the highly granular demands of expert users.

The Interactive Dialogue Track highlights performance gaps across user types. Models like DeepSeek-R1 and Gemma-3-27B excel with expert users, who respond effectively to clarification queries, creating quality interactive loops. However, most models drop sharply with novice users. GLM-4-9B falls from 0.43 in Direct Generation to 0.16 in Interactive Dialogue, with Success Rate dropping to 0.09, indicating models struggle to guide novices in clarifying requirements through multi-turn dialogue.

The gap between Test Pass Rate and Success Rate varies by user type. For novices, this gap is typically large, reflecting models' ability to partially understand intent but difficulty delivering complete, executable solutions. For experts, some models show smaller gaps, suggesting clear, structured requirements facilitate more complete code generation.

Cross-User-Type Variations in Interactive Experience Interactive experience performance patterns across user types generally mirror code generation quality but reveal more nuanced variations. Most models achieve highest scores with intermediate users, then decline with experts—GPT-4o drops from 3.78 with intermediate users to 3.66 with experts, while LLaMA-3.3-70B shows a sharper decline from 3.42 to 2.77. Dimension-specific analysis highlights significant adaptability limitations: in communication efficiency, most models exhibit notable drops in expert scenarios, indicating that existing clarification strategies fail to

Level	Model	Direct Generation		Interactive Dialogue												
		Code Quality		Code Quality		Subjective Evaluation							Entity		Semantic	
		TP	SR	TP	SR	QD	CL	IF	UA	CE	LS	Avg	EC	ES	SC	SS
Novice	DeepSeek-V3	0.56	0.43	0.33	0.23	3.25	3.51	3.20	3.09	3.02	2.86	3.16	0.28	0.0270	0.6362	0.0015
	DeepSeek-R1	0.57	0.44	0.44	0.33	3.09	2.63	3.25	3.19	3.57	3.05	3.13	0.46	0.0305	0.6360	0.0020
	GLM-4-9B	0.43	0.31	0.16	0.09	3.60	3.46	3.71	3.74	3.49	3.66	3.61	0.48	0.0282	0.6169	0.0018
	LLaMA-3.3-70B	0.49	0.38	0.39	0.28	3.25	2.97	3.63	3.59	3.54	3.26	3.37	0.46	0.0322	0.6347	0.0022
	Mistral-7B	0.28	0.18	0.03	0.01	2.36	1.92	2.61	2.31	2.09	2.19	2.25	0.44	0.0282	0.5990	0.0012
	Gemma-3-27B	0.57	0.43	0.42	0.29	3.46	3.68	3.62	3.59	3.21	3.34	3.48	0.43	0.0280	0.6355	0.0020
	Qwen2.5-32B	0.53	0.41	0.39	0.27	3.63	3.74	4.07	3.98	3.74	3.55	3.79	0.34	0.0319	0.6264	0.0023
	Qwen3-235B-A22B	0.56	0.43	0.33	0.24	3.15	3.10	3.12	3.21	3.01	3.09	3.11	0.34	0.0303	0.6406	0.0015
	GPT-4o	0.54	0.43	0.41	0.30	3.64	3.50	4.01	4.07	4.05	3.82	3.85	0.46	0.0332	0.6366	0.0022
	Claude-Sonnet-4	0.53	0.43	0.34	0.46	3.48	3.32	3.72	3.86	3.93	3.60	3.65	0.44	0.0318	0.6118	0.0020
	o1-mini	0.59	0.47	0.38	0.27	3.57	3.40	3.87	3.97	3.90	3.65	3.73	0.45	0.0321	0.6300	0.0023
	Talk2Code	-	-	-	-	**4.44**	**4.63**	**4.56**	**4.61**	**4.82**	**4.18**	**4.54**	**0.50**	**0.0385**	**0.6690**	**0.0029**
Intermediate	DeepSeek-V3	0.60	0.48	0.37	0.26	3.40	3.69	3.37	3.36	3.14	3.16	3.35	0.33	0.0243	0.6639	0.0015
	DeepSeek-R1	0.63	0.50	0.49	0.37	3.44	3.09	3.53	3.63	3.82	3.59	3.52	0.48	0.0291	0.7078	0.0018
	GLM-4-9B	0.52	0.38	0.16	0.10	3.66	3.39	3.67	3.84	3.50	3.77	3.64	0.51	0.0239	0.6240	0.0014
	LLaMA-3.3-70B	0.57	0.44	0.44	0.32	3.37	2.96	3.61	3.57	3.56	3.42	3.42	0.49	0.0288	0.6923	0.0019
	Mistral-7B	0.34	0.23	0.06	0.04	2.46	1.95	2.67	2.44	2.13	2.41	2.34	0.48	0.0228	0.6049	0.0012
	Gemma-3-27B	0.64	0.50	0.45	0.33	3.64	3.68	3.65	3.75	3.32	3.51	3.59	0.47	0.0287	0.6844	0.0018
	Qwen2.5-32B	0.59	0.46	0.42	0.29	3.56	3.62	3.95	3.81	3.70	3.59	3.71	0.39	0.0281	0.6745	0.0020
	Qwen3-235B-A22B	0.64	0.51	0.37	0.26	3.70	3.63	3.56	3.76	3.50	3.60	3.62	0.42	0.0273	0.6749	0.0019
	GPT-4o	0.61	0.48	0.45	0.33	3.61	3.41	3.87	3.94	4.00	3.84	3.78	0.47	0.030	0.6942	0.0021
	Claude-Sonnet-4	0.63	0.52	0.40	0.31	3.44	3.27	3.71	3.78	3.85	3.71	3.63	0.46	0.0283	0.6823	0.0019
	o1-mini	0.63	0.51	0.42	0.31	3.44	3.30	3.76	3.81	3.90	3.74	3.66	0.45	0.0285	0.6911	0.0020
	Talk2Code	-	-	-	-	**4.54**	**4.56**	**4.62**	**4.77**	**4.81**	**4.32**	**4.60**	**0.54**	**0.0335**	**0.7259**	**0.0027**
Expert	DeepSeek-V3	0.58	0.45	0.38	0.26	3.40	3.58	3.09	3.38	3.12	2.97	3.26	0.31	0.0220	0.6546	0.0026
	DeepSeek-R1	0.59	0.48	0.49	0.39	3.71	3.45	3.71	3.78	3.71	3.53	3.65	0.40	0.0263	0.6728	0.0022
	GLM-4-9B	0.45	0.33	0.21	0.14	3.61	3.55	3.56	3.77	3.45	3.44	3.56	0.48	0.0259	0.6081	0.0017
	LLaMA-3.3-70B	0.52	0.40	0.40	0.29	2.85	2.62	2.98	2.84	2.83	2.48	2.77	0.46	0.0283	0.6470	0.0026
	Mistral-7B	0.30	0.20	0.03	0.01	2.32	1.96	2.37	2.15	1.93	1.99	2.12	0.46	0.0238	0.6019	0.0016
	Gemma-3-27B	0.57	0.43	0.47	0.35	3.51	3.72	3.42	3.67	3.22	3.26	3.46	0.43	0.0272	0.6631	0.0026
	Qwen2.5-32B	0.56	0.43	0.42	0.30	3.33	3.64	3.61	3.39	3.60	2.88	3.41	0.37	0.0323	0.6572	0.0025
	Qwen3-235B-A22B	0.59	0.45	0.39	0.27	3.60	3.77	3.52	3.72	3.35	3.36	3.55	0.33	0.0264	0.6579	0.0025
	GPT-4o	0.57	0.45	0.48	0.36	3.57	3.62	3.80	3.76	3.88	3.35	3.66	0.45	0.0323	0.6748	0.0027
	Claude-Sonnet-4	0.53	0.42	0.40	0.31	3.47	3.47	3.68	3.63	3.77	3.35	3.56	0.42	0.0313	0.6432	0.0026
	o1-mini	0.59	0.45	0.43	0.31	3.47	3.48	3.62	3.65	3.75	3.27	3.54	0.44	0.0312	0.6381	0.0026
	Talk2Code	-	-	-	-	**4.32**	**4.58**	**4.35**	**4.52**	**4.75**	**3.78**	**4.38**	**0.52**	**0.0382**	**0.6961**	**0.0029**

Table 1: Overall performance of different families of LLMs on the Talk2Code benchmark.

meet expert users' expectations.

Objective evaluation metrics reinforce this distribution pattern. For entity consistency, most models follow the Intermediate $>$ Expert $>$ Novice trend—DeepSeek-V3 rises from 0.28 with novices to 0.33 with intermediate users, then slightly drops to 0.31 with experts. Semantic consistency is more stable but still peaks in intermediate scenarios, with GPT-4o reaching 0.6942 for intermediate users, higher than 0.6366 for novices and 0.6748 for experts. However, satisfaction metrics remain low across all user types with minimal variability, suggesting that regardless of expertise, current models lack effective mechanisms for progressive understanding and adaptive interaction.

These findings show that current models struggle to tailor strategies dynamically to different user profiles, particularly with novice guidance and expert precision requirements. In contrast, our Talk2Code benchmark consistently delivers high-quality and stable interactive experiences across all user types, serving as a valuable reference standard for enhancing LLM-based coding assistants.

Conclusion

In this paper, we introduce Talk2Code, the first user-stratified benchmark for multi-turn code generation dialogues, addressing a key gap in evaluating LLMs' interactive capabilities. Evaluations on 21 models reveal performance drops in realistic interactions, especially disadvantaging novice users. Talk2Code shifts the focus from static code correctness to an integrated framework assessing both code quality and interaction experience. We envision our benchmark driving research toward more adaptive, user-aware AI coding assistants.

References

Anthropic. 2025. Claude Sonnet 4.

Athiwaratkun, B.; Gouda, S. K.; Wang, Z.; Li, X.; Tian, Y.; Tan, M.; Ahmad, W. U.; Wang, S.; Sun, Q.; Shang, M.; Gonugondla, S. K.; Ding, H.; Kumar, V.; Fulton, N.; Farahani, A.; Jain, S.; Giaquinto, R.; Qian, H.; Ramanathan, M. K.; Nallapati, R.; Ray, B.; Bhatia, P.; Sengupta, S.; Roth, D.; and Xiang, B. 2023. Multi-lingual Evaluation of Code Generation Models. In *The Eleventh International Conference on Learning Representations*.

Austin, J.; Odena, A.; Nye, M.; Bosma, M.; Michalewski, H.; Dohan, D.; Jiang, E.; Cai, C.; Terry, M.; Le, Q.; et al. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Byrne, B.; Krishnamoorthi, K.; Sankar, C.; Neelakantan, A.; Duckworth, D.; Yavuz, S.; Goodrich, B.; Dubey, A.; Cedilnik, A.; and Kim, K.-Y. 2019. Taskmaster-1: Toward a realistic and diverse dialog dataset. *arXiv preprint arXiv:1909.05358*.

Chen, J.; Xiao, S.; Zhang, P.; Luo, K.; Lian, D.; and Liu, Z. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; Pinto, H. P. D. O.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Duan, H.; Wei, J.; Wang, C.; Liu, H.; Fang, Y.; Zhang, S.; Lin, D.; and Chen, K. 2024. BotChat: Evaluating LLMs' Capabilities of Having Multi-Turn Dialogues. In Duh, K.; Gomez, H.; and Bethard, S., eds., *Findings of the Association for Computational Linguistics: NAACL 2024*, 3184–3200. Mexico City, Mexico: Association for Computational Linguistics.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv e-prints*, arXiv–2407.

Dubois, Y.; Li, X.; Taori, R.; Zhang, T.; Gulrajani, I.; Ba, J.; Guestrin, C.; Liang, P.; and Hashimoto, T. 2023. AlpacaFarm: A Simulation Framework for Methods that Learn from Human Feedback. In *Thirty-seventh Conference on Neural Information Processing Systems*.

GLM, T.; Zeng, A.; Xu, B.; Wang, B.; Zhang, C.; Yin, D.; Zhang, D.; Rojas, D.; Feng, G.; Zhao, H.; et al. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Hendrycks, D.; Basart, S.; Kadavath, S.; Mazeika, M.; Arora, A.; Guo, E.; Burns, C.; Puranik, S.; He, H.; Song, D.; and Steinhardt, J. 2021. Measuring Coding Challenge Competence With APPS. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Howarth, B.; and Anderson, A. H. 2007. Introducing objects in spoken dialogue: The influence of conversational setting and cognitive load on the articulation and use of referring expressions. *Language and Cognitive Processes*, 22(2): 272–296.

Hurst, A.; Lerer, A.; Goucher, A. P.; Perelman, A.; Ramesh, A.; Clark, A.; Ostrow, A.; Welihinda, A.; Hayes, A.; Radford, A.; et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Jaech, A.; Kalai, A.; Lerer, A.; Richardson, A.; El-Kishky, A.; Low, A.; Helyar, A.; Madry, A.; Beutel, A.; Carney, A.; et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Jain, N.; Han, K.; Gu, A.; Li, W.-D.; Yan, F.; Zhang, T.; Wang, S.; Solar-Lezama, A.; Sen, K.; and Stoica, I. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*.

Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D. S.; de las Casas, D.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; Lavaud, L. R.; Lachaux, M.-A.; Stock, P.; Scao, T. L.; Lavril, T.; Wang, T.; Lacroix, T.; and Sayed, W. E. 2023. Mistral 7B. arXiv:2310.06825.

Khalid, B.; Alikhani, M.; and Stone, M. 2020. Combining cognitive modeling and reinforcement learning for clarification in dialogue. In *Proceedings of the 28th International Conference on Computational Linguistics*, 4417–4428.

Kurt, A. A. 2011. Personalization principle in multimedia learning: Conversational versus formal style in written word. *Turkish Online Journal of Educational Technology-TOJET*, 10(3): 185–192.

Li, Y.; Choi, D.; Chung, J.; Kushman, N.; Schrittwieser, J.; Leblond, R.; Eccles, T.; Keeling, J.; Gimeno, F.; Dal Lago, A.; et al. 2022. Competition-level code generation with alphacode. *Science*, 378(6624): 1092–1097.

Liu, A.; Feng, B.; Xue, B.; Wang, B.; Wu, B.; Lu, C.; Zhao, C.; Deng, C.; Zhang, C.; Ruan, C.; et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Lopes, J.; Lohan, K.; and Hastie, H. 2018. Symptoms of cognitive load in interactions with a dialogue system. In *Proceedings of the workshop on modeling cognitive processes from multimodal data*, 1–5.

Naszadi, K.; Oliehoek, F. A.; and Monz, C. 2024. Communicating with Speakers and Listeners of Different Pragmatic Levels. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 21777–21783. Miami, Florida, USA: Association for Computational Linguistics.

Palta, S.; Chandrasekaran, N.; Rudinger, R.; and Counts, S. 2025. Speaking the right language: The impact of expertise alignment in user-ai interactions. *arXiv preprint arXiv:2502.18685*.

Peng, Q.; Chai, Y.; and Li, X. 2024. HumanEval-XL: A Multilingual Code Generation Benchmark for Cross-lingual

Natural Language Generalization. In Calzolari, N.; Kan, M.-Y.; Hoste, V.; Lenci, A.; Sakti, S.; and Xue, N., eds., *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, 8383–8394. Torino, Italia: ELRA and ICCL.

Qwen; :; Yang, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Li, C.; Liu, D.; Huang, F.; Wei, H.; Lin, H.; Yang, J.; Tu, J.; Zhang, J.; Yang, J.; Yang, J.; Zhou, J.; Lin, J.; Dang, K.; Lu, K.; Bao, K.; Yang, K.; Yu, L.; Li, M.; Xue, M.; Zhang, P.; Zhu, Q.; Men, R.; Lin, R.; Li, T.; Tang, T.; Xia, T.; Ren, X.; Ren, X.; Fan, Y.; Su, Y.; Zhang, Y.; Wan, Y.; Liu, Y.; Cui, Z.; Zhang, Z.; and Qiu, Z. 2025. Qwen2.5 Technical Report. arXiv:2412.15115.

Raihan, N.; Anastasopoulos, A.; and Zampieri, M. 2025. mHumanEval - A Multilingual Benchmark to Evaluate Large Language Models for Code Generation. In Chiruzzo, L.; Ritter, A.; and Wang, L., eds., *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)*, 11432–11461. Albuquerque, New Mexico: Association for Computational Linguistics. ISBN 979-8-89176-189-6.

Shi, Z.; Feng, Y.; and Lipani, A. 2022. Learning to Execute Actions or Ask Clarification Questions. In Carpuat, M.; de Marneffe, M.-C.; and Meza Ruiz, I. V., eds., *Findings of the Association for Computational Linguistics: NAACL 2022*, 2060–2070. Seattle, United States: Association for Computational Linguistics.

Team, G.; Kamath, A.; Ferret, J.; Pathak, S.; Vieillard, N.; Merhej, R.; Perrin, S.; Matejovicova, T.; Ramé, A.; Rivière, M.; et al. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.

Vaithilingam, P.; Zhang, T.; and Glassman, E. L. 2022. Expectation vs. Experience: Evaluating the Usability of Code Generation Tools Powered by Large Language Models. In *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI EA '22. New York, NY, USA: Association for Computing Machinery. ISBN 9781450391566.

Wang, P.; Zhang, L.; Liu, F.; Shi, L.; Li, M.; Shen, B.; and Fu, A. 2025a. Codeif-bench: Evaluating instruction-following capabilities of large language models in interactive code generation. *arXiv preprint arXiv:2503.22688*.

Wang, S.; Wang, Z.; Ma, D.; Yu, Y.; Ling, R.; Li, Z.; Xiong, F.; and Zhang, W. 2025b. CodeFlowBench: A Multi-turn, Iterative Benchmark for Complex Code Generation. *arXiv preprint arXiv:2504.21751*.

Wang, X.; Wang, Z.; Liu, J.; Chen, Y.; Yuan, L.; Peng, H.; and Ji, H. 2024. MINT: Evaluating LLMs in Multi-turn Interaction with Tools and Language Feedback. In *The Twelfth International Conference on Learning Representations*.

Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; Zheng, C.; Liu, D.; Zhou, F.; Huang, F.; Hu, F.; Ge, H.; Wei, H.; Lin, H.; Tang, J.; Yang, J.; Tu, J.; Zhang, J.; Yang, J.; Yang, J.; Zhou, J.; Zhou, J.; Lin, J.; Dang, K.; Bao, K.; Yang, K.; Yu, L.; Deng, L.; Li, M.; Xue, M.; Li, M.; Zhang, P.; Wang, P.; Zhu, Q.; Men, R.; Gao, R.; Liu, S.; Luo, S.; Li, T.; Tang, T.; Yin, W.; Ren, X.; Wang, X.; Zhang, X.; Ren, X.; Fan, Y.; Su, Y.; Zhang, Y.; Zhang, Y.; Wan, Y.; Liu, Y.; Wang, Z.; Cui, Z.; Zhang, Z.; Zhou, Z.; and Qiu, Z. 2025. Qwen3 Technical Report. arXiv:2505.09388.

Zheng, L.; Chiang, W.-L.; Sheng, Y.; Zhuang, S.; Wu, Z.; Zhuang, Y.; Lin, Z.; Li, Z.; Li, D.; Xing, E.; et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36: 46595–46623.

Zhou, Y.; Jiang, S.; Tian, Y.; Weston, J.; Levine, S.; Sukhbaatar, S.; and Li, X. 2025. Sweet-rl: Training multi-turn llm agents on collaborative reasoning tasks. *arXiv preprint arXiv:2503.15478*.

Zou, J.; Sun, A.; Long, C.; Aliannejadi, M.; and Kanoulas, E. 2023. Asking Clarifying Questions: To benefit or to disturb users in Web search? *Information Processing & Management*, 60(2): 103176.