

MCP-Universal: BENCHMARKING LARGE LANGUAGE MODELS WITH REAL-WORLD MODEL CONTEXT PROTOCOL SERVERS

Anonymous authors

Paper under double-blind review

ABSTRACT

The Model Context Protocol (MCP) has emerged as a transformative standard for connecting large language models (LLMs) to external data sources and tools, and it is rapidly gaining adoption across major AI platforms. However, existing benchmarks are overly simplistic and fail to capture real-world application challenges such as long-horizon reasoning and large, unfamiliar tool spaces. To address this critical gap, we introduce **MCP-Universal**, the first comprehensive benchmark specifically designed to evaluate LLMs on realistic and difficult tasks through interaction with real-world MCP servers. Our benchmark spans 6 core domains and 11 different MCP servers: *Location Navigation*, *Repository Management*, *Financial Analysis*, *3D Design*, *Browser Automation*, and *Web Searching*. To ensure rigorous evaluation, we carefully design execution-based evaluators, including format evaluators for compliance, static evaluators for time-invariant content matching, and dynamic evaluators that automatically obtain real-time ground truth for temporally sensitive tasks. Through extensive evaluation of more than 20 leading LLMs, we find that even frontier models such as GPT-5-High (44.16% success rate) and Grok-4 (33.33% success rate) exhibit significant performance limitations. In addition, our benchmark poses a substantial long-context challenge, as the number of input tokens increases rapidly with each additional interaction step. It also introduces an unknown-tools challenge, since LLM agents often lack familiarity with the precise usage of certain MCP servers. Notably, enterprise-level agents such as Cursor and Claude Code fail to achieve better performance than the ReAct framework. Beyond evaluation, we open-source our extensible evaluation framework, enabling seamless integration of new LLMs, agents and MCP servers.

1 INTRODUCTION

The Model Context Protocol (MCP), introduced by Anthropic (2024a), represents a major paradigm shift in how AI systems interface with external data sources and tools. Dubbed the “USB-C of AI” (Rick, 2025), MCP addresses the long-standing issue of fragmented, bespoke integrations that trap language models in isolated information silos (Edwin, 2025). Since its release, MCP has gained rapid traction: major AI providers, including OpenAI (2025a) and Google (2025), have committed to adoption, while development platforms such as Cursor (2025) and Cline (2025) have begun integrating it. Despite its transformative potential, current evaluations remain insufficient, as existing benchmarks focus on narrow aspects of LLM performance, such as math reasoning (Cobbe et al., 2021) or function calling (Patil et al., 2025), without assessing how models interact with real-world MCP servers. MCP-RADAR (Gao et al., 2025) adapts datasets like HumanEval (Chen et al., 2021) and GSM8k (Cobbe et al., 2021) to the MCP setting, but these remain largely derivative and fail to capture the breadth of real-world applications or mitigate issues like data leakage Balloccu et al. (2024). Similarly, MCPWorld (Yan et al., 2025) continues to rely heavily on graphical user interfaces (GUIs) and provides limited coverage of MCP-enabled scenarios, restricting its utility for evaluating LLMs in authentic MCP-driven environments.

To address these critical limitations, we introduce our **MCP-Universal**, a benchmark aiming at evaluating LLMs in realistic, challenging use cases with real-world MCP servers. As shown in Figure 1, MCP-Universal captures realistic challenges: real-world tools usage, long-horizon multi-

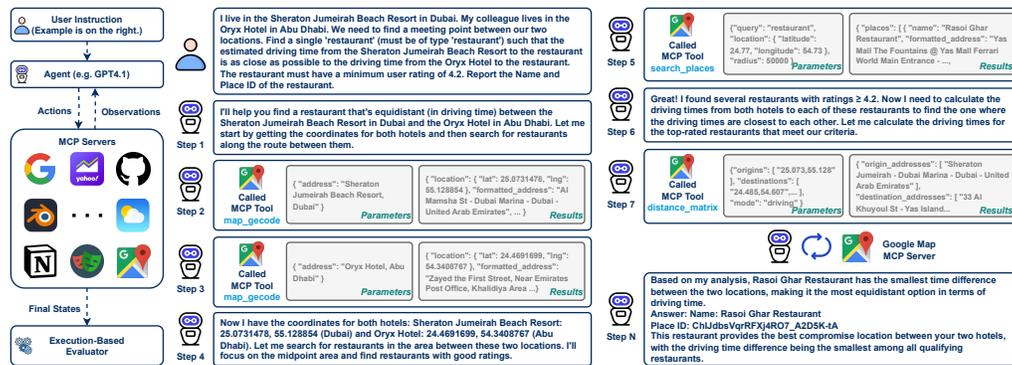


Figure 1: Example from MCP-Universe illustrating realistic challenges, including real-world tool usage, long-horizon multi-turn tool calls, long context, scattered evidence, and large tool spaces, with all challenges grounded in real-world MCP servers connected to actual environments.

turn tool calls, long context windows, scattered evidence, and large tool spaces. Unlike existing works, MCP-Universe is grounded in real-world MCP servers that connect to actual data sources and environments. Our benchmark encompasses 6 core domains, with 11 MCP servers spanning diverse applications: *Location Navigation*, *Repository Management*, *Financial Analysis*, *3D Design*, *Browser Automation*, and *Web Searching*, comprising a total of 231 tasks. Each domain captures the operational complexities of real-world deployments, from handling authentic financial data and navigating complex geospatial information to managing version control workflows and executing real-time ticket price checks.

To ensure rigorous evaluation, we carefully design execution-based evaluators rather than relying on LLM-as-a-judge (Zheng et al., 2023) (e.g., MCPEval (Liu et al., 2025) and LiveMCPBench (Mo et al., 2025)), recognizing that many tasks involve real-time data that static LLM knowledge cannot properly assess. Our evaluation includes format evaluators for agent format compliance, static evaluators for time-invariant content matching, and dynamic evaluators that automatically obtain real-time ground truth for temporally sensitive tasks. Furthermore, for the evolving nature of MCP servers, we provide an extensible, user-friendly framework that enables researchers and the broader community to seamlessly integrate new agents and MCP servers into the evaluation pipeline.

We conducted extensive experiments using MCP-Universe across all 6 core domains and 11 different MCP servers. Through extensive evaluation of more than 20 leading LLMs, we find that even top-performing models such as GPT-5-High (44.16% success rate), Grok-4 (33.33% success rate), and Claude-4.0-Sonnet (29.44% success rate) exhibit significant performance limitations, revealing a substantial gap between their impressive general capabilities and their effectiveness in real-world MCP environments. Our comprehensive analysis identifies several fundamental challenges that current LLM agents face in MCP interactions. First, we observe a *long-context challenge*, as the number of tokens increases rapidly with the number of interaction steps, often leading to context overflow and degraded performance in multi-step tasks. Second, there exists an *unknown-tools challenge*, where LLM agents often lack familiarity with the precise usage patterns, parameter specifications, and expected behaviors of diverse MCP servers. Additionally, our evaluation reveals significant *cross-domain performance variations*, with models showing markedly different success rates across different application domains, suggesting domain-specific optimization needs. Notably, enterprise-level agents like Cursor and Claude Code cannot achieve better performance than standard ReAct frameworks, highlighting the challenges of our benchmark.

In summary, this work makes the following key contributions:

- We introduce **MCP-Universe**, the first comprehensive benchmark for LLMs in MCP environments across 6 domains with real-world servers, where even SOTA LLMs struggle.
- We develop a rigorous **execution-based** evaluation framework with format, static, and dynamic assessment capabilities to enable comprehensive performance measurement.
- We reveal fundamental limitations of current LLM agents, such as challenges with long contexts, handling unknown tools, and cross-domain discrepancies, thereby highlighting directions for future MCP-agent design.

2 RELATED WORK

MCP and LLMs as Agents. MCP, introduced by Anthropic in late 2024, is an open standard for integrating AI with external data sources and tools (Anthropic, 2024a). Using a universal JSON-RPC 2.0 interface over STDIO and SSE (Edwin, 2025), it addresses the “data silo problem” and connects hosts (AI applications), clients (bridges), and servers (capability providers). Meanwhile, LLMs have progressed from text generators to autonomous agents with planning, reasoning, and tool use abilities (Wang et al., 2024a), enabled by advances in instruction following (Hida et al., 2024; Dong et al., 2025; Qian et al., 2025; Pyatkin et al., 2025), multi-step reasoning (Xu et al., 2025; Wei et al., 2022; Yao et al., 2023a; Zhang et al., 2025b), and tool integration (Qu et al., 2025; Qin et al., 2024b; Wang et al., 2024b; Zhang et al., 2025a). Agent paradigms such as ReAct (Yao et al., 2023b), Reflection (Shinn et al., 2023), and Plan-and-Solve (Wang et al., 2023), along with frameworks like AutoGen (Wu et al., 2023), MetaGPT (Hong et al., 2024), Camel-AI (Li et al., 2023a), and LangGraph (LangChain, 2024), demonstrate practical implementations. With multimodal LLMs (Hurst et al., 2024; Anil et al., 2023), GUI-based computer-use agents (Hu et al., 2024; Yang et al., 2025c; Li et al., 2025b; Yang et al., 2025b) have emerged, exemplified by OpenAI’s CUA (OpenAI, 2025b), Anthropic’s Computer-Use (Anthropic, 2024b), and ByteDance’s UI-Taris (Qin et al., 2025), opening new frontiers in computer automation.

Evaluation of Agents. Evaluating LLM-based agents has become a major research area, with benchmarks targeting different aspects of agent capability. Web navigation has been widely studied through environments such as Mini-Wob++ (Liu et al., 2018), Mind2Web (1 & 2) (Deng et al., 2023; Gou et al., 2025), WebLINX (Lù et al., 2024), AssistantBench (Yoran et al., 2024), WebArena (Zhou et al., 2024), VisualWebArena, and VideoWebArena (Koh et al., 2024). GUI-based interaction is tested by OSWorld (Xie et al., 2024), WindowsAgentArena (Bonatti et al., 2024), and UI-Vision (Nayak et al., 2025), while software engineering tasks are covered by SWE-bench (Jimenez et al., 2024) and DevBench (Li et al., 2024). Tool and function calling has also been emphasized, with APiBank (Li et al., 2023b), ToolBench (Qin et al., 2024a), GAIA (Mialon et al., 2024), AppWorld (Trivedi et al., 2024), τ -Bench (Yao et al., 2024), and BFCLv3 (Patil et al., 2025) evaluating agents’ proficiency in invoking external APIs.

Recently, several MCP-specific benchmarks have been introduced (Table 1). MCPWorld (Yan et al., 2025) evaluates agents in GUIs and MCP environments but relies heavily on GUIs and omits time-varying tasks. MCP-RADAR (Gao et al., 2025) adapts datasets like HumanEval and GSM8k, but its tasks lack real-world grounding and temporal variation. MCPEval (Liu et al., 2025) and LiveMCP-Bench (Mo et al., 2025) both adopt LLM-as-a-Judge, which is ill-suited for real-time tasks and prone to style bias (Li et al., 2025a). In contrast, MCP-Universe integrates authentic MCP servers, temporal dynamics, and execution-based evaluation, providing a comprehensive benchmark that directly measures task completion in real-world MCP scenarios.

3 MCP-UNIVERSE

3.1 OVERVIEW

MCP-Universe is a comprehensive evaluation framework designed to assess the capabilities of LLMs when interacting with real-world MCP servers for challenging and practical tasks. As shown in Figure 2, our benchmark encompasses three core components: (1) an extensible, easy-to-use evaluation framework; (2) a collection of carefully designed task instructions grounded in real-world MCP server scenarios; (3) a suite of execution-based evaluators for measuring task completion.

To formalize the setting, we model the benchmark as follows. Let $S = \{s_1, s_2, \dots, s_k\}$ denote the collection of MCP servers, where each server s_i exposes a set of tools $T_i = \{t_{i,1}, t_{i,2}, \dots, t_{i,|T_i|}\}$ through the MCP protocol. A task τ is defined as a tuple $(G, C, T_{\text{available}})$, where:

- G is the goal specification describing the desired outcome;

Table 1: Contemporary MCP Benchmarks.

Benchmark	Real-World Integration	Temporal Dynamics	Execution Evaluation
MCPWorld	✓	✗	✓
MCP-RADAR	✗	✗	✓
MCPEval	✗	✓	✗
LiveMCPBench	✓	✓	✗
MCP-Universe	✓	✓	✓

162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215

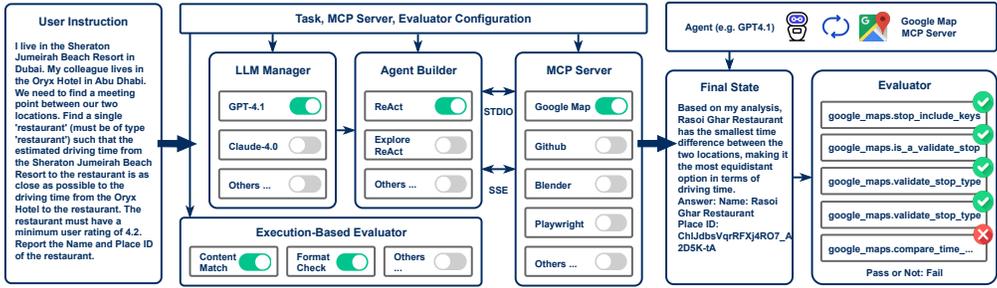


Figure 2: Overview of the MCP-Universe evaluation framework, which automatically configures LLM agents, MCP servers, and evaluators per task. Each evaluation consists of agent–server interactions via MCP, followed by automated execution-based assessment.

- C contains the initial context and any relevant background information;
- $T_{\text{available}} = \bigcup_{i \in I} T_i$ is the set of tools accessible for the task, with $I \subseteq \{1, \dots, k\}$ indicating which servers are available.

The benchmark challenges an agent to identify, sequence, and invoke appropriate tools from $T_{\text{available}}$ to achieve G given C , requiring reasoning over partial information, adapting to diverse tool interfaces, and handling ambiguities or failures in tool responses.

For evaluation, let $M = \{m_1, m_2, \dots, m_n\}$ be the set of language models and $A = \{a_1, a_2, \dots, a_p\}$ be the set of agent patterns (e.g., ReAct) that can be paired with them. For a given $(m, a) \in M \times A$ and task $\tau \in \mathcal{T}$, the interaction produces a conversation trace $R = (r_1, r_2, \dots, r_T)$, where each r_t contains the agent’s output and any tool invocations. The evaluation function

$$E : M \times A \times \mathcal{T} \rightarrow \{0, 1\}$$

assigns 1 if the task is successfully completed according to predefined success criteria, and 0 otherwise. Success is determined through a combination of automated checks (e.g., verifying structured outputs or end states). Aggregating $E(m, a, \tau)$ over all tasks yields a quantitative measure of an agent’s proficiency in MCP-driven tool use.

3.2 EVALUATION FRAMEWORK

As illustrated in Figure 2, our evaluation framework coordinates multiple components to deliver objective and reproducible evaluation. Given a task specification, the framework automatically configures the entire evaluation pipeline: it builds the appropriate combination of LLM and agent, selects the required MCP servers, and sets up the corresponding evaluators. This configuration also manages resources, API endpoints, and evaluation criteria to ensure systematic execution. The LLM Manager supports multiple SOTA models, including GPT-5 and Grok-4, and is responsible for model configuration, API management, and standardized prompt formatting so that comparisons across models are fair and consistent. The Agent Builder constructs specialized agents such as ReAct and Function-Call and equips them with reasoning strategies tailored for MCP communication, enabling controlled comparison of different agent architectures within a uniform framework.

The framework further integrates with a wide range of MCP servers that expose real-world APIs, authentication mechanisms, and tool specifications. Through dynamic configuration, it supports both single server and multi server evaluation scenarios, ensuring that tasks reflect authentic operating environments rather than simplified simulations. To verify outcomes, the framework employs execution-based evaluators that apply domain-specific validation strategies, for example, stop type validation for Google Maps and branch checking for GitHub. This approach produces binary pass or fail results without relying on subjective judgments or expensive human annotation. Each evaluation proceeds through agent and server interactions mediated by MCP, followed by automated assessment, while detailed interaction logs are captured to provide comprehensive insights into model behavior across different tasks, servers, and evaluation settings.

216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269



Figure 3: Distribution of tasks in MCP-Universe across different application domains.

3.3 REAL-WORLD MCP SERVERS

A key principle of MCP-Universe is its reliance on real-world MCP servers rather than simulated environments, ensuring that evaluation reflects the authentic complexity of practical applications. As shown in Table 2, the benchmark spans 11 MCP servers with 133 tools across 6 core domains:

- Location Navigation:** Geographic reasoning and spatial task execution using the official Google Maps MCP server, which provides a rich suite of tools such as location search, route planning, and distance computation. Models must navigate the full complexity of real-world location data to complete navigation tasks effectively.
- Repository Management:** Repository development and codebase operations with the GitHub MCP server, exposing tools for repository search, issue tracking, and code editing. Models must engage with the authentic challenges of real-world repository management to execute development tasks effectively.
- Financial Analysis:** Quantitative reasoning in dynamic markets using the Yahoo Finance MCP server, which supports stock monitoring, shareholder lookup, and options tracking over live financial data. Models must analyze live and volatile financial information to make reliable decisions under uncertainty.
- 3D Designing:** Professional computer-aided design tasks supported by the Blender MCP server, offering object creation, asset manipulation, and material setup. Models must operate within the technical depth of professional design workflows to generate valid outputs.
- Browser Automation:** Automated interaction with web applications via the Playwright MCP server, enabling browser navigation, button clicking, and page snapshotting. Models must handle the intricacies of real-world browser environments to perform web automation tasks.
- Web Searching:** Open-domain information seeking using the Google Search MCP server for queries and the Fetch MCP server for URL content retrieval. Models must sift through open-ended, noisy web data to identify and synthesize relevant information.

This selection emphasizes both domain diversity and real-world relevance. To further increase coverage and task complexity, additional MCP servers are included, such as Notion, Weather, Date, and Calculator. Details of all MCP servers are provided in Appendix A.

3.4 TASKS AND EVALUATORS

Since MCP is new and lacks high-quality usage examples, we manually designed challenging tasks to reflect real use cases. Tasks that can be trivially solved by LLMs without MCP servers, or consistently completed within five retries using MCP servers, are discarded and replaced. As shown in Figure 3, each domain contains 4-5 task types covering representative scenarios. For example, Location Navigation includes route planning, optimal stops, location search, and place finding; Repos-

Table 2: Key statistics in MCP-Universe.

Statistic	Number
Total tasks	231 (100%)
- Location Navigation	45 (19.5%)
- Web Searching	55 (23.8%)
- Browser Automation	39 (16.9%)
- 3D Designing	19 (8.2%)
- Financial Analysis	40 (17.3%)
- Repo. Management	33 (14.3%)
Total MCP Servers	11
Total Tools in Servers	133
Total Unique Evaluators	84 (100%)
- Format Evaluators	4 (4.8%)
- Static Evaluators	32 (38.1%)
- Dynamic Evaluators	48 (57.1%)

Table 3: Evaluation with LLMs using ReAct. We report success rate (SR), average evaluator score (AE; mean percentage of evaluators passed), and average steps (AS) for successful tasks. †: GPT-OSS-120B does not follow the ReAct pattern, and is therefore evaluated with OpenAI’s Agent SDK.

Model	Location Navigation	Repository Management	Financial Analysis	3D Designing	Browser Automation	Web Searching	AE	Overall AS	SR
<i>Proprietary Models</i>									
🌀 GPT-5-High	26.67	30.30	67.50	57.89	43.59	45.45	62.82	6.84	44.16
🌀 GPT-5-Medium	33.33	30.30	67.50	52.63	35.90	45.45	60.23	8.22	43.72
🌀 Grok-4	28.89	12.12	40.00	26.32	41.03	41.82	49.01	7.75	33.33
🐼 Claude-4.1-Opus	17.78	21.21	52.50	36.84	35.90	20.00	49.14	7.04	29.44
🐼 Claude-4.0-Opus	15.56	15.15	55.00	31.58	38.46	18.18	46.40	7.69	28.14
🐼 Claude-4.0-Sonnet	22.22	12.12	55.00	26.32	38.46	21.82	50.61	7.46	29.44
🌀 Grok-4-Fast	24.44	9.09	65.00	5.26	25.64	21.82	48.95	6.54	27.27
🌀 Grok-Code-Fast-1	26.67	9.09	62.50	15.79	20.51	18.18	44.72	6.87	26.41
🌀 o3	26.67	6.06	40.00	26.32	25.64	29.09	38.95	4.82	26.41
🌀 o4-mini	26.67	18.18	40.00	36.84	23.08	18.18	40.38	7.90	25.97
🐼 Claude-3.7-Sonnet	13.33	18.18	40.00	36.84	23.08	21.82	40.36	7.16	24.24
🌌 Gemini-2.5-Pro	13.33	12.12	50.00	21.05	25.64	12.73	36.93	6.98	22.08
🌌 Gemini-2.5-Flash	15.56	12.12	37.50	21.05	30.77	14.55	33.99	8.26	21.65
🌀 GPT-4.1	8.89	6.06	40.00	26.32	23.08	10.91	41.32	5.24	18.18
🌀 GPT-4o	8.89	9.09	35.00	26.32	12.82	9.09	37.03	6.03	15.58
<i>Open-Source Models</i>									
🌌 GLM-4.5	17.78	9.09	50.00	26.32	15.38	27.27	41.16	7.33	24.68
🌀 Qwen3-Coder	13.33	3.03	57.50	31.58	30.77	9.09	41.39	7.77	22.94
🌀 DeepSeek-V3.1	15.56	0.00	42.50	31.58	28.21	18.18	43.23	6.31	22.08
🐼 Kimi-K2	11.11	3.03	52.50	15.79	25.64	10.91	41.28	6.96	19.91
🌌 GLM-4.5-Air	17.78	6.06	42.50	10.53	17.95	16.36	37.42	6.42	19.48
🌀 Qwen3-235B	11.11	9.09	50.00	15.79	15.38	9.09	38.53	5.74	18.18
🌀 DeepSeek-V3	11.11	6.06	30.00	26.32	12.82	7.27	35.82	5.06	14.29
🌀 GPT-OSS-120B†	6.67	6.06	35.00	10.53	5.13	5.45	26.34	-	11.26

itory Management includes project setup, issue tracking, automation setup, and code integration; Financial Analysis includes portfolio analysis, financial statements, trading strategies, institutional holdings, and dividend analysis; 3D Designing includes object creation, material setup, lighting, render settings, and scene hierarchy; Browser Automation includes travel booking, sports analytics, academic research, platform exploration, and map navigation; and Web Searching includes person identification, entity discovery, metric matching, complex reasoning, and factual lookup. All tasks are cross-checked by other authors for feasibility, clarity, and correctness.

To evaluate task completion, we design execution-based evaluators tailored to each task. While many recent works adopt the LLM-as-a-judge paradigm (Liu et al., 2025; Mo et al., 2025), we argue it is unsuitable for MCP-Universe because some tasks require real-time data, while LLM judges have static knowledge and are prone to bias or hallucination. Although creating execution-based evaluation requires more human effort, it is necessary for fairness and comprehensiveness. We design three evaluator types: (1) Format Evaluators check compliance with output requirements; (2) Static Evaluators assess correctness for tasks with fixed answers, such as city counts in route planning, historical stock prices, or football statistics; and (3) Dynamic Evaluators automatically obtain real-time ground truth for temporally sensitive tasks, such as flight prices, current weather, or the number of GitHub issues. Each evaluator is reviewed by the other authors for feasibility and accuracy. Appendix B provides more details and examples of both tasks and evaluators.

Based on these evaluators, we employ a complementary dual-metric system: **Average Evaluator Score (AE)** and **Success Rate (SR)**. Following OSWorld (Xie et al., 2024), the AE represents the mean percentage of passed evaluators per task. This ensures models receive partial credit for progress, preventing a zero score due to a single missed evaluator. The SR serves as a strict binary metric to determine if the task was fully completed. These metrics work together to provide a balanced evaluation: the AE rewards progress, while the SR ensures that high partial scores do not mask the inability to fully complete the task.

4 EXPERIMENT

4.1 SETUP

In our experiments, we evaluate both proprietary and open-source SOTA LLMs on MCP-Universe. The models include xAI’s Grok-4, Grok-4-Fast and Grok-Code-Fast-1 (xAI, 2025), An-

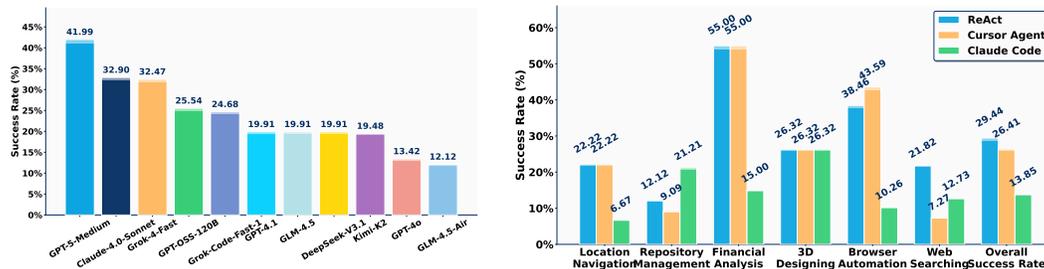


Figure 4: Evaluation of LLM performance with function calling (left) and enterprise-level agent frameworks (right).

thropic’s Claude-4.1-Opus, Claude-4.0-Opus, Claude-4.0-Sonnet (Anthropic, 2025b), and Claude-3.7-Sonnet (Anthropic, 2025a), OpenAI’s GPT-5 (OpenAI, 2025a), o3, o4-mini (OpenAI, 2025b), GPT-4.1 (OpenAI, 2025b), GPT-4o (Hurst et al., 2024), and GPT-OSS (OpenAI, 2025a), Google’s Gemini-2.5-Pro and Gemini-2.5-Flash (Comanici et al., 2025), Zai’s GLM-4.5 and GLM-4.5-Air (Zai, 2025), Moonshot’s Kimi-K2 (Moonshot, 2025), Qwen’s Qwen3-Coder and Qwen3-235B-A22B-Instruct-2507 (Yang et al., 2025a), and DeepSeek’s DeepSeek-V3-0324 and DeepSeek-V3.1 (DeepSeek-AI et al., 2025). All models are top-ranked on the lmsys Chatbot Arena leaderboard (Chiang et al., 2024); open-source ones exceed 100B parameters. For agents, we adopt two settings: the widely used ReAct framework (Yao et al., 2023b), where models iteratively generate thoughts based on observations and then actions based on thoughts, and a native function calling mode, where models directly invoke MCP server functions without explicit ReAct style reasoning. In the latter, an LLM may call multiple tools within a single step. Additional setup details are provided in Appendix C.

4.2 MAIN RESULTS

LLMs w/ ReAct. As shown in Table 3, we compare SOTA proprietary and open-source LLMs on MCP-Universe. OpenAI’s GPT-5 achieves the highest success rates, with high and medium reasoning effort reaching 44.16% and 43.72%, far ahead of others. Grok-4 ranks third (33.33%). GPT-5 leads across all domains, Location Navigation (33.33%), Repository Management (30.30%), Financial Analysis (67.50%), 3D Designing (57.89%), Browser Automation (43.59%), and Web Searching (45.45%), and also achieves the highest average evaluator score (62.82%). Grok-4 shows strength in Browser Automation (41.03%) and Web Searching (41.82%). Location Navigation remains difficult for all models (<35%), with GPT-4.1 and GPT-4o scoring below 10%. In Repository Management, only GPT-5 surpasses 30%, and in 3D Design only GPT-5 exceeds 50%. Among open-source models, GLM-4.5 performs best (24.68%), comparable to some proprietary models like o4-mini (25.97%) and Claude-3.7-Sonnet (24.24%), yet still 20 points behind the best.

Beyond success rates, we also assess average evaluator (AE) scores and average steps (AS). GPT-5 ranks highest in both SR and AE (62.82%), showing strong consistency. Discrepancies emerge: Claude-4.0-Sonnet achieves a slightly higher AE (50.61%) than Grok-4 (49.01%), but Grok-4 attains higher SR (33.33% vs. 29.44%). Similarly, Claude-4.1-Opus records 49.14% AE but only 29.44% SR. Task completion typically requires 5–8 steps: o3 averages 4.82 steps (26.41% SR), GPT-5 requires 6.84–8.22 depending on effort, and Grok-4 averages 7.75. These results highlight that even frontier LLMs remain unreliable across diverse real-world MCP tasks, underscoring MCP-Universe as a challenging and necessary testbed.

LLMs w/ Function Call. Beyond ReAct, we also evaluate representative LLMs using native function calling, where models directly invoke MCP server functions without explicit ReAct-style reasoning. In each step, an LLM can call more than one tool. As shown in Figure 4 (left), GPT-5-Medium achieves the highest success rate (41.99%), close to its ReAct counterpart (43.72%), while Claude-4.0-Sonnet performs slightly better with function calling (32.90% vs. 29.44%). Grok-4-Fast also benefits from function calling, improving from 27.27% with ReAct to 32.47%. GPT-4.1 and GPT-4o show only minor changes (19.91% vs. 18.18%, and 13.42% vs. 15.58%). Other LLMs generally lag behind, including Grok-Code-Fast-1 (24.68%), GLM-4.5 (19.91%), DeepSeek-V3.1 (19.91%), Kimi-K2 (19.48%), and GLM-4.5-Air (12.12%). Notably, GPT-OSS-120B reaches 25.54% under function calling, a dramatic improvement over its OpenAI Agent SDK

378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431

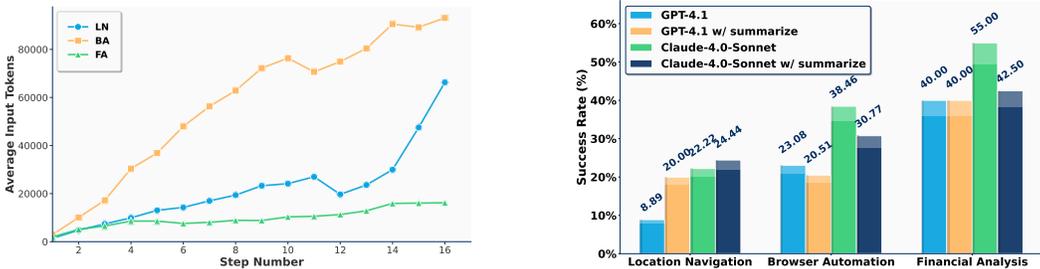


Figure 5: (Left) Growth of average context length (in tokens) as the number of interaction steps increases in MCP-Universe tasks, illustrating the long context challenge. (Right) Effect of introducing a summarization agent on LLM agent performance across selected domains.

result (11.26%), making it competitive with other open source models. This 14.28-point gain highlights strong sensitivity to interaction paradigms: some models, like GPT-OSS-120B, are more effective with direct function invocation than with ReAct. More results can be found in the Appendix D.

Enterprise-Level Agents. As shown in the right of Figure 4, we compare three agent configurations using Claude-4.0-Sonnet as the backbone: ReAct, Cursor Agent, and Claude Code. ReAct achieves the highest overall success rate (29.44%), followed by Cursor Agent (26.41%) and Claude Code (13.85%). Both Cursor Agent and Claude Code are primarily designed for code generation, explaining their weaker performance on general tasks. Notably, Claude Code achieves its best result in Repository Management (21.21%), outperforming ReAct (12.12%) and Cursor Agent (9.09%), showing its strength in code-related tasks. However, both struggle in general-purpose domains: Cursor Agent underperforms in Web Searching (7.27% vs. ReAct’s 21.82%) due to reliance on internal tools, while Claude Code shows particularly poor results in Location Navigation (6.67%) and Financial Analysis (15.00%). These results highlight that specialized code agents, though effective in their domains, generalize poorly to broader real-world tasks.

4.3 ANALYSIS

Evaluator-Specific Performance. Our benchmark incorporates three evaluator types: format evaluators, static evaluators, and dynamic evaluators. Table 4 presents a breakdown of model performance across these types. For format evaluators, performance varies significantly across models: Claude-4.0-Sonnet achieves the highest success rate at 98.29%, followed by DeepSeek-V3.1 (96.58%), and GPT-4.1 (95.73%). However, some models struggle with format compliance, particularly Gemini-2.5-Flash (51.28%), Gemini-2.5-Pro (64.10%), and o3 (73.50%), indicating that format adherence is not universally mastered even among frontier models. We highlight the naive error in the Appendix E.6. On content-sensitive static evaluators, GPT-5-High leads with 69.59%, followed by Claude-4.0-Sonnet (61.92%), while most other models achieve around 40-50% success. For dynamic evaluators, GPT-5-Medium achieves the highest performance at 65.96%, followed by GPT-5-High (62.11%) and Claude-4.0-Sonnet (54.74%). The substantial performance gap between format evaluators (where top models exceed 90%) and content evaluators (where most models achieve 40-60%) indicates that the primary source of failure lies in content generation rather than format compliance. This demonstrates that our benchmark evaluates LLMs from multiple angles, including format compliance and content correctness under both static and dynamic conditions.

Table 4: Success rate across different types of evaluators on our MCP-Universe benchmark.

Model	Format	Static	Dynamic
GPT-5-High	88.03	69.59	62.11
GPT-5-Medium	88.89	61.92	65.96
Grok-4	88.03	49.04	52.98
Claude-4.0-Sonnet	98.29	61.92	54.74
Grok-Code-Fast-1	85.47	52.60	50.53
o3	73.50	38.63	43.16
o4-mini	78.63	44.66	43.86
Gemini-2.5-Pro	64.10	39.18	42.46
Gemini-2.5-Flash	51.28	45.21	30.88
GPT-4.1	95.73	57.53	49.47
GPT-4o	91.45	54.79	45.61
GLM-4.5	81.20	46.30	48.07
GLM-4.5-Air	87.18	42.47	47.02
Kimi-K2	70.94	33.15	53.33
Qwen3-Coder	78.63	42.74	49.12
Qwen3-235B	92.31	43.29	53.68
DeepSeek-V3.1	96.58	59.73	49.12

Additional detailed performance analysis is provided in Appendix E.

Long Context Challenges. In our MCP-Universe benchmark, long context handling is a major challenge, especially in Location Navigation, Browser Automation, and Financial Analysis. These

432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485

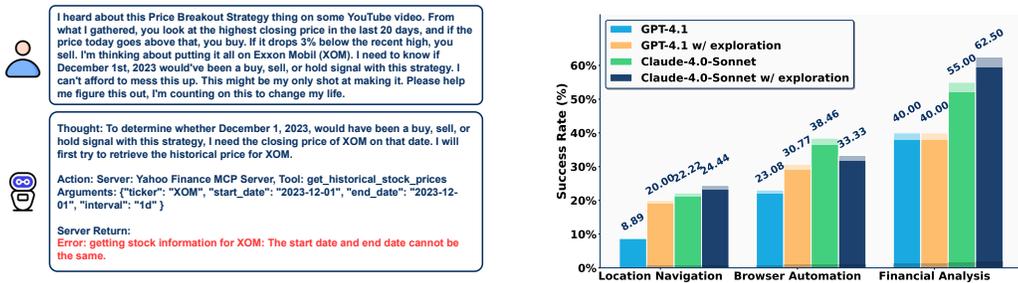


Figure 6: (Left) An example of the unknown tool challenges. (Right) Effect of introducing the exploration phase on LLM agent performance across selected domains.

tasks often require reasoning over extended sequences of observations or historical actions that exceed typical context windows. For instance, Google Maps servers may return detailed multi-location information, Playwright servers can output full webpage HTML, and Yahoo Finance servers provide stock data across long ranges, producing large volumes of contextual input. As shown in Figure 5, token counts grow rapidly with interaction steps, confirming long context as a core difficulty.¹ To explore mitigations, we tested a summarization agent that compresses server outputs at each step while attempting to preserve essential information (details in Appendix F). This method improved GPT-4.1 and Claude-4.0-Sonnet in Location Navigation but had little or negative effect in Browser Automation and Financial Analysis, where fine-grained details are critical. These results show that MCP-Universe introduces realistic long context challenges while exposing the limits of simple compression, underscoring its value as a testbed for advancing context handling methods in LLM agents.

Unknown Tools Challenges. In addition to long context challenges, our error analysis shows that LLMs often misuse MCP server tools due to limited familiarity with their interfaces. For example, as shown in Figure 6 (left), models frequently fail on Yahoo Finance by setting identical start and end dates, violating API requirements and causing execution errors. To mitigate this, we introduce an exploration phase where models freely interact with MCP tools to probe interfaces, learn parameter requirements, and build basic tool knowledge before moving to an exploitation phase that combines this knowledge with a ReAct-style framework to solve tasks (details in Appendix G). As shown in Figure 6 (right), exploration yields gains in some domains: GPT-4.1 improves by 7.69 points in Browser Automation (30.77%) and Claude-4.0-Sonnet by 7.50 points in Financial Analysis (62.50%), but results are mixed, with declines elsewhere. These outcomes highlight both the potential and the limitations of exploration: while it can enhance performance on tasks requiring tool familiarity and iterative reasoning, it is not universally effective, underscoring the need for more adaptive strategies and reinforcing MCP-Universe as a rigorous testbed for advanced LLM agents.

More MCP Servers Connected. In Table 3, each task is paired only with the MCP servers directly relevant to it. Here, we extend the setup by connecting additional, unrelated MCP servers to evaluate LLM performance under greater tool complexity. For all tasks, we connect seven MCP servers comprising 94 tools, which introduces noise and leads to a clear decline in performance (Figure 7). For instance, Claude-4.0-Sonnet’s success rate in Location Navigation falls from 22.22% to 11.11%, GPT-4.1’s Browser Automation accuracy drops from 23.08% to 15.38%, and its Financial Analysis score decreases from 40.00% to 35.00%. These results show that MCP-Universe also functions as a robust testbed for assessing LLM resilience when faced with larger and less relevant toolsets.

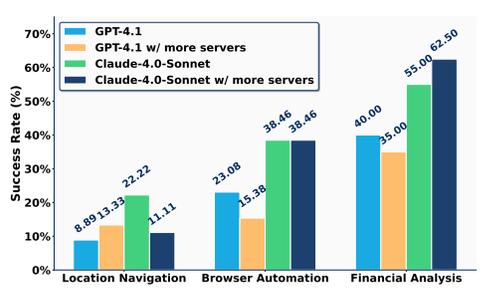


Figure 7: Effect of connecting with more unrelated MCP servers.

¹The context length experiment is based on Claude-4.0-Sonnet.

5 CONCLUSION

In this work, we introduce **MCP-Universe**, the first benchmark to rigorously evaluate LLMs in real-world MCP environments. By grounding tasks in authentic data and using execution-based evaluators, it reveals key gaps in long context handling, tool usage, and cross-domain performance. Our experiments show that even frontier models and enterprise-level agents struggle with these challenges, highlighting the need for advances in model design and agent integration. With its extensible framework, MCP-Universe offers a valuable testbed for driving progress toward more reliable real-world MCP-Use applications.

REFERENCES

- Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Slav Petrov, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy P. Lillicrap, Angeliki Lazaridou, Orhan Firat, James Molloy, Michael Isard, Paul Ronald Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, George Tucker, Enrique Piqueras, Maxim Krikun, Iain Barr, Nikolay Savinov, Ivo Danihelka, Becca Roelofs, Anaïs White, Anders Andreassen, Tamara von Glehn, Lakshman Yagati, Mehran Kazemi, Lucas Gonzalez, Misha Khalman, Jakub Sygnowski, and et al. Gemini: A family of highly capable multimodal models. *CoRR*, abs/2312.11805, 2023. doi: 10.48550/ARXIV.2312.11805. URL <https://doi.org/10.48550/arXiv.2312.11805>.
- Anthropic. Introducing the model context protocol. <https://www.anthropic.com/news/model-context-protocol>, November 2024a. Accessed: 2025-06-30.
- Anthropic. Introducing computer use, a new claude 3.5 sonnet, and claude 3.5 haiku. <https://www.anthropic.com/news/3-5-models-and-computer-use>, October 2024b. Accessed: 2025-06-30.
- Anthropic. Claude 3.7 sonnet and claude code. <https://www.anthropic.com/news/claude-3-7-sonnet>, Feb 2025a. Accessed: 2025-07-28.
- Anthropic. Introducing claude 4. <https://www.anthropic.com/news/claude-4>, May 2025b. Accessed: 2025-07-28.
- Simone Balloccu, Patrícia Schmidtová, Mateusz Lango, and Ondrej Dusek. Leak, cheat, repeat: Data contamination and evaluation malpractices in closed-source LLMs. In Yvette Graham and Matthew Purver (eds.), *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 67–93, St. Julian’s, Malta, March 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.eacl-long.5. URL <https://aclanthology.org/2024.eacl-long.5/>.
- Rogério Bonatti, Dan Zhao, Francesco Bonacci, Dillon Dupont, Sara Abdali, Yinheng Li, Yadong Lu, Justin Wagle, Kazuhito Koishida, Arthur Bucker, Lawrence Jang, and Zack Hui. Windows agent arena: Evaluating multi-modal OS agents at scale. *CoRR*, abs/2409.08264, 2024. doi: 10.48550/ARXIV.2409.08264. URL <https://doi.org/10.48550/arXiv.2409.08264>.
- Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *CoRR*, abs/2107.03374, 2021. URL <https://arxiv.org/abs/2107.03374>.

- 540 Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li,
541 Dacheng Li, Banghua Zhu, Hao Zhang, Michael I. Jordan, Joseph E. Gonzalez, and Ion Sto-
542 ica. Chatbot arena: An open platform for evaluating llms by human preference. In *Forty-first*
543 *International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*.
544 OpenReview.net, 2024. URL <https://openreview.net/forum?id=3MW8GKNyzI>.
- 545 Cline. Mcp overview. <https://docs.cline.bot/mcp/mcp-overview>, 2025. Accessed:
546 2025-06-30.
- 547
- 548 Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser,
549 Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John
550 Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021. URL
551 <https://arxiv.org/abs/2110.14168>.
- 552 Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit
553 Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marris, Sam Petulla, Colin
554 Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szpektor, Nan-
555 Jiang Jiang, Krishna Haridasan, Ahmed Omran, Nikunj Saunshi, Dara Bahri, Gaurav Mishra, Eric
556 Chu, Toby Boyd, Brad Hekman, Aaron Parisi, Chaoyi Zhang, Kornraphop Kawintiranon, Tania
557 Bedrax-Weiss, Oliver Wang, Ya Xu, Ollie Purkiss, Uri Mendlovic, Ilaï Deutel, Nam Nguyen,
558 Adam Langley, Flip Korn, Lucia Rossazza, Alexandre Ramé, Sagar Waghmare, Helen Miller,
559 Nathan Byrd, Ashrith Sheshan, Raia Hadsell Sangnie Bhardwaj, Pawel Janus, Tero Rissa, Dan
560 Horgan, Sharon Silver, Ayzaan Wahid, Sergey Brin, Yves Raimond, Klemen Kloboves, Cindy
561 Wang, Nitesh Bharadwaj Gundavarapu, Iliia Shumailov, Bo Wang, Mantas Pajarskas, Joe Hey-
562 ward, Martin Nikoltchev, Maciej Kula, Hao Zhou, Zachary Garrett, Sushant Kafle, Sercan Arik,
563 Ankita Goel, Mingyao Yang, Jiho Park, Koji Kojima, Parsa Mahmoudieh, Koray Kavukcuoglu,
564 Grace Chen, Doug Fritz, Anton Bulyenov, Sudeshna Roy, Dimitris Paparas, Hadar Shemtov, Bo-
565 Juen Chen, Robin Strudel, David Reitter, Aurko Roy, Andrey Vlasov, Changwan Ryu, Chas Le-
566 ichter, Haichuan Yang, Zelda Mariet, Denis Vnukov, Tim Sohn, Amy Stuart, Wei Liang, Minmin
567 Chen, Praynaa Rawlani, Christy Koh, JD Co-Reyes, Guangda Lai, Praseem Banzal, Dimitrios
568 Vytiniotis, Jieru Mei, Mu Cai, Mohammed Badawi, Corey Fry, Ale Hartman, Daniel Zheng,
569 Eric Jia, James Keeling, Annie Louis, Ying Chen, Efren Robles, Wei-Chih Hung, Howard Zhou,
570 Nikita Saxena, Sonam Goenka, Olivia Ma, Zach Fisher, Mor Hazan Taege, Emily Graves, David
571 Steiner, Yujia Li, Sarah Nguyen, Rahul Sukthankar, Joe Stanton, Ali Eslami, Gloria Shen, Berkin
572 Akin, Alexey Guseynov, Yiqian Zhou, Jean-Baptiste Alayrac, Armand Joulin, Efrat Farkash,
573 Ashish Thapliyal, Stephen Roller, Noam Shazeer, Todor Davchev, Terry Koo, Hannah Forbes-
574 Pollard, Kartik Audhkhasi, Greg Farquhar, Adi Mayrav Gilady, Maggie Song, John Aslanides,
575 Piermaria Mendolicchio, Alicia Parrish, John Blitzer, Pramod Gupta, Xiaoen Ju, Xiaochen Yang,
576 Puranjay Datta, Andrea Tacchetti, Sanket Vaibhav Mehta, Gregory Dobb, Shubham Gupta, Fed-
577 erico Piccinini, Raia Hadsell, Sujeer Rajayogam, Jiepu Jiang, Patrick Griffin, Patrik Sundberg,
578 Jamie Hayes, Alexey Frolov, Tian Xie, Adam Zhang, Kingshuk Dasgupta, Uday Kalra, Lior
579 Shani, Klaus Macherey, Tzu-Kuo Huang, Liam MacDermed, Karthik Duddu, Paulo Zaccchello,
580 Zi Yang, Jessica Lo, Kai Hui, Matej Kastelic, Derek Gasaway, Qijun Tan, Summer Yue, Pablo
581 Barrio, John Wieting, Weel Yang, Andrew Nystrom, Solomon Demmessie, Anselm Levskaya,
582 Fabio Viola, Chetan Tekur, Greg Billock, George Necula, Mandar Joshi, Rylan Schaeffer, Swach-
583 hand Lokhande, Christina Sorokin, Pradeep Shenoy, Mia Chen, Mark Collier, Hongji Li, Tay-
584 lor Bos, Nevan Wichers, Sun Jae Lee, Angéline Pouget, Santhosh Thangaraj, Kyriakos Axio-
585 tis, Phil Crone, Rachel Sterneck, Nikolai Chinaev, Victoria Krakovna, Oleksandr Ferludin, Ian
586 Gemp, Stephanie Winkler, Dan Goldberg, Ivan Korotkov, Kefan Xiao, Malika Mehrotra, Sandeep
587 Mariserla, Vihari Piratla, Terry Thurk, Khiem Pham, Hongxu Ma, Alexandre Senges, Ravi Ku-
588 mar, Clemens Meyer, Ellie Talius, Nuo Wang Pierse, Ballie Sandhu, Horia Toma, Kuo Lin, Swa-
589 roop Nath, Tom Stone, Dorsa Sadigh, Nikita Gupta, Arthur Guez, Avi Singh, Matt Thomas, Tom
590 Duerig, Yuan Gong, Richard Tanburn, Lydia Lihui Zhang, Phuong Dao, Mohamed Hammad,
591 Sirui Xie, Shruti Rijhwani, Ben Murdoch, Duhyeon Kim, Will Thompson, Heng-Tze Cheng,
592 Daniel Sohn, Pablo Sprechmann, Qiantong Xu, Srinivas Tadepalli, Peter Young, Ye Zhang, Hansa
593 Srinivasan, Miranda Aperghis, Aditya Ayyar, Hen Fitoussi, Ryan Burnell, David Madras, Mike
Dusenberry, Xi Xiong, Tayo Oguntebi, Ben Albrecht, Jörg Bornschein, Jovana Mitrović, Ma-
son Dimarco, Bhargav Kanagal Shamanna, Premal Shah, Eren Sezener, Shyam Upadhyay, Dave
Lacey, Craig Schiff, Sebastien Baur, Sanjay Ganapathy, Eva Schneider, Mateo Wirth, Connor
Schenck, Andrey Simanovsky, Yi-Xuan Tan, Philipp Fränken, Dennis Duan, Bharath Mankalale,

594 Nikhil Dhawan, Kevin Sequeira, Zichuan Wei, Shivanker Goel, Caglar Unlu, Yukun Zhu, Haitian
595 Sun, Ananth Balashankar, Kurt Shuster, Megh Umekar, Mahmoud Alnahlawi, Aäron van den
596 Oord, Kelly Chen, Yuexiang Zhai, Zihang Dai, Kuang-Huei Lee, Eric Doi, Lukas Zilka, Rohith
597 Vallu, Disha Shrivastava, Jason Lee, Hisham Husain, Honglei Zhuang, Vincent Cohen-Addad,
598 Jarred Barber, James Atwood, Adam Sadosky, Quentin Wellens, Steven Hand, Arunkumar Ra-
599 jendran, Aybuke Turker, CJ Carey, Yuanzhong Xu, Hagen Soltau, Zefei Li, Xinying Song, Cong-
600 long Li, Iurii Kemaev, Sasha Brown, Andrea Burns, Viorica Patraucean, Piotr Stanczyk, Renga
601 Aravamudhan, Mathieu Blondel, Hila Noga, Lorenzo Blanco, Will Song, Michael Isard, Mandar
602 Sharma, Reid Hayes, Dalia El Badawy, Avery Lamp, Itay Laish, Olga Kozlova, Kelvin Chan,
603 Sahil Singla, Srinivas Sunkara, Mayank Upadhyay, Chang Liu, Aijun Bai, Jarek Wilkiewicz,
604 Martin Zlocha, Jeremiah Liu, Zhuowan Li, Haiguang Li, Omer Barak, Ganna Raboshchuk, Jiho
605 Choi, Fangyu Liu, Erik Jue, Mohit Sharma, Andreea Marzoca, Robert Busa-Fekete, Anna Ko-
606 rsun, Andre Elisseeff, Zhe Shen, Sara Mc Carthy, Kay Lamerigts, Anahita Hosseini, Hanzhao
607 Lin, Charlie Chen, Fan Yang, Kushal Chauhan, Mark Omernick, Dawei Jia, Karina Zainul-
608 lina, Demis Hassabis, Danny Vainstein, Ehsan Amid, Xiang Zhou, Ronny Votel, Eszter Vörtes,
609 Xinjian Li, Zongwei Zhou, Angeliki Lazaridou, Brendan McMahan, Arjun Narayanan, Hubert
610 Soyer, Sujoy Basu, Kayi Lee, Bryan Perozzi, Qin Cao, Leonard Berrada, Rahul Arya, Ke Chen,
611 Katrina, Xu, Matthias Lochbrunner, Alex Hofer, Sahand Sharifzadeh, Renjie Wu, Sally Gold-
612 man, Pranjal Awasthi, Xuezhi Wang, Yan Wu, Claire Sha, Biao Zhang, Maciej Mikuła, Filipppo
613 Graziano, Siobhan Mcloughlin, Irene Giannoumis, Youhei Namiki, Chase Malik, Carey Rade-
614 baugh, Jamie Hall, Ramiro Leal-Cavazos, Jianmin Chen, Vikas Sindhwani, David Kao, David
615 Greene, Jordan Griffith, Chris Welty, Ceslee Montgomery, Toshihiro Yoshino, Liangzhe Yuan,
616 Noah Goodman, Assaf Hurwitz Michaely, Kevin Lee, KP Sawhney, Wei Chen, Zheng Zheng,
617 Megan Shum, Nikolay Savinov, Etienne Pot, Alex Pak, Morteza Zadimoghaddam, Sijal Bhat-
618 nagar, Yoad Lewenberg, Blair Kutzman, Ji Liu, Lesley Katzen, Jeremy Selier, Josip Djolonga,
619 Dmitry Lepikhin, Kelvin Xu, Jacky Liang, Jiewen Tan, Benoit Schillings, Muge Ersoy, Pete
620 Blois, Bernd Bandemer, Abhimanyu Singh, Sergei Lebedev, Pankaj Joshi, Adam R. Brown,
621 Evan Palmer, Shreya Pathak, Komal Jalan, Fedir Zubach, Shuba Lall, Randall Parker, Alok Gun-
622 jan, Sergey Rogulenko, Sumit Sanghai, Zhaoqi Leng, Zoltan Egyed, Shixin Li, Maria Ivanova,
623 Kostas Andriopoulos, Jin Xie, Elan Rosenfeld, Ariel Wright, Ankur Sharma, Xinyang Geng,
624 Yicheng Wang, Sam Kwei, Renke Pan, Yujing Zhang, Gabby Wang, Xi Liu, Chak Yeung, Eliz-
625 abeth Cole, Aviv Rosenberg, Zhen Yang, Phil Chen, George Polovets, Pranav Nair, Rohun Sax-
626 ena, Josh Smith, Shuo yiin Chang, Aroma Mahendru, Svetlana Grant, Anand Iyer, Irene Cai,
627 Jed McGiffin, Jiaming Shen, Alanna Walton, Antonious Girgis, Oliver Woodman, Rosemary Ke,
628 Mike Kwong, Louis Rouillard, Jinneng Rao, Zhihao Li, Yuntao Xu, Flavien Prost, Chi Zou,
629 Ziwei Ji, Alberto Magni, Tyler Liechty, Dan A. Calian, Deepak Ramachandran, Igor Krivokon,
630 Hui Huang, Terry Chen, Anja Hauth, Anastasija Ilić, Weijuan Xi, Hyeontaek Lim, Vlad-Doru
631 Ion, Pooya Moradi, Metin Toksoz-Exley, Kalessa Bullard, Miltos Allamanis, Xiaomeng Yang,
632 Sophie Wang, Zhi Hong, Anita Gergely, Cheng Li, Bhavishya Mittal, Vitaly Kovalev, Victor Un-
633 gureanu, Jane Labanowski, Jan Wassenberg, Nicolas Lacasse, Geoffrey Cideron, Petar Dević,
634 Annie Marsden, Lynn Nguyen, Michael Fink, Yin Zhong, Tatsuya Kiyono, Desi Ivanov, Sally
635 Ma, Max Bain, Kiran Yalasang, Jennifer She, Anastasia Petrushkina, Mayank Lunayach, Carla
636 Bromberg, Sarah Hodkinson, Vilobh Meshram, Daniel Vlasic, Austin Kyker, Steve Xu, Jeff Stan-
637 way, Zuguang Yang, Kai Zhao, Matthew Tung, Seth Odoom, Yasuhisa Fujii, Justin Gilmer,
638 Eunyoung Kim, Felix Halim, Quoc Le, Bernd Bohnet, Seliem El-Sayed, Behnam Neyshabur,
639 Malcolm Reynolds, Dean Reich, Yang Xu, Erica Moreira, Anuj Sharma, Zeyu Liu, Moham-
640 mad Javad Hosseini, Naina Raisinghani, Yi Su, Ni Lao, Daniel Formoso, Marco Gelmi, Almog
641 Gueta, Tapomay Dey, Elena Gribovskaya, Domagoj Čevd, Sidharth Mudgal, Garrett Bingham,
642 Jianling Wang, Anurag Kumar, Alex Cullum, Feng Han, Konstantinos Bousmalis, Diego Cedillo,
643 Grace Chu, Vladimir Magay, Paul Michel, Ester Hlavnova, Daniele Calandriello, Setareh Ari-
644 afar, Kaisheng Yao, Vikash Sehwal, Arpi Vezer, Agustin Dal Lago, Zhenkai Zhu, Paul Kishan
645 Rubenstein, Allen Porter, Anirudh Baddepudi, Oriana Riva, Mihai Dorin Istin, Chih-Kuan Yeh,
646 Zhi Li, Andrew Howard, Nilpa Jha, Jeremy Chen, Raoul de Liedekerke, Zafarali Ahmed, Mikael
647 Rodriguez, Tanuj Bhatia, Bangju Wang, Ali Elqursh, David Klinghoffer, Peter Chen, Pushmeet
Kohli, Te I, Weiyang Zhang, Zack Nado, Jilin Chen, Maxwell Chen, George Zhang, Aayush
Singh, Adam Hillier, Federico Lebron, Yiqing Tao, Ting Liu, Gabriel Dulac-Arnold, Jingwei
Zhang, Shashi Narayan, Buhuang Liu, Orhan Firat, Abhishek Bhowmick, Bingyuan Liu, Hao
Zhang, Zizhao Zhang, Georges Rotival, Nathan Howard, Anu Sinha, Alexander Grushetsky, Ben-
jamin Beyret, Keerthana Gopalakrishnan, James Zhao, Kyle He, Szabolcs Payrits, Zaid Nabulsi,

648 Zhaoyi Zhang, Weijie Chen, Edward Lee, Nova Fallen, Sreenivas Gollapudi, Aurick Zhou, Filip
649 Pavetić, Thomas Köppe, Shiyu Huang, Rama Pasumarthi, Nick Fernando, Felix Fischer, Daria
650 Ćurko, Yang Gao, James Svensson, Austin Stone, Haroon Qureshi, Abhishek Sinha, Apoorv
651 Kulshreshtha, Martin Matysiak, Jieming Mao, Carl Saroufim, Aleksandra Faust, Qingnan Duan,
652 Gil Fidel, Kaan Katircioglu, Raphaël Lopez Kaufman, Dhruv Shah, Weize Kong, Abhishek
653 Bapna, Gellért Weisz, Emma Dunleavy, Praneet Dutta, Tianqi Liu, Rahma Chaabouni, Carolina
654 Parada, Marcus Wu, Alexandra Belias, Alessandro Bissacco, Stanislav Fort, Li Xiao, Fantine
655 Huot, Chris Knutsen, Yochai Blau, Gang Li, Jennifer Prendki, Juliette Love, Yinlam Chow, Pichi
656 Charoenpanit, Hidetoshi Shimokawa, Vincent Coriou, Karol Gregor, Tomas Izo, Arjun Akula,
657 Mario Pinto, Chris Hahn, Dominik Paulus, Jiaxian Guo, Neha Sharma, Cho-Jui Hsieh, Aadaeze
658 Chukwuka, Kazuma Hashimoto, Nathalie Rauschmayr, Ling Wu, Christof Angermueller, Yulong
659 Wang, Sebastian Gerlach, Michael Pliskin, Daniil Mirylenka, Min Ma, Lexi Baugher, Bryan Gale,
660 Shaan Bijwadia, Nemanja Rakićević, David Wood, Jane Park, Chung-Ching Chang, Babi Seal,
661 Chris Tar, Kacper Krasowiak, Yiwen Song, Georgi Stephanov, Gary Wang, Marcello Maggioni,
662 Stein Xudong Lin, Felix Wu, Shachi Paul, Zixuan Jiang, Shubham Agrawal, Bilal Piot, Alex
663 Feng, Cheolmin Kim, Tulsee Doshi, Jonathan Lai, Chuqiao Xu, Sharad Vikram, Ciprian Chelba,
664 Sebastian Krause, Vincent Zhuang, Jack Rae, Timo Denk, Adrian Collister, Lotte Weerts, Xi-
665 anghong Luo, Yifeng Lu, Håvard Garnes, Nitish Gupta, Terry Spitz, Avinatan Hassidim, Lihao
666 Liang, Izhak Shafran, Peter Humphreys, Kenny Vassigh, Phil Wallis, Virat Shejwalkar, Nicolas
667 Perez-Nieves, Rachel Hornung, Melissa Tan, Beka Westberg, Andy Ly, Richard Zhang, Brian
668 Farris, Jongbin Park, Alec Kosik, Zeynep Cankara, Andrii Maksai, Yunhan Xu, Albin Cassirer,
669 Sergi Caelles, Abbas Abdolmaleki, Mencher Chiang, Alex Fabrikant, Shravya Shetty, Luheng
670 He, Mai Giménez, Hadi Hashemi, Sheena Panthaplackel, Yana Kulizhskaya, Salil Deshmukh,
671 Daniele Pighin, Robin Alazard, Disha Jindal, Seb Noury, Pradeep Kumar S, Siyang Qin, Xerxes
672 Dotiwalla, Stephen Spencer, Mohammad Babaeizadeh, Blake JianHang Chen, Vaibhav Mehta,
673 Jennie Lees, Andrew Leach, Penporn Koanantakool, Iliia Akolzin, Ramona Comanescu, Junwhan
674 Ahn, Alexey Svyatkovskiy, Basil Mustafa, David D'Ambrosio, Shiva Mohan Reddy Garlapati,
675 Pascal Lamblin, Alekh Agarwal, Shuang Song, Pier Giuseppe Sessa, Pauline Coquinot, John
676 Maggs, Hussain Masoom, Divya Pitta, Yaqing Wang, Patrick Morris-Suzuki, Billy Porter, John-
677 son Jia, Jeffrey Dudek, Raghavender R, Cosmin Paduraru, Alan Ansell, Tolga Bolukbasi, Tony
678 Lu, Ramya Ganeshan, Zi Wang, Henry Griffiths, Rodrigo Benenson, Yifan He, James Swirhun,
679 George Papamakarios, Aditya Chawla, Kuntal Sengupta, Yan Wang, Vedrana Milutinovic, Igor
680 Mordatch, Zhipeng Jia, Jamie Smith, Will Ng, Shitij Nigam, Matt Young, Eugen Vušak, Blake
681 Hechtman, Sheela Goenka, Avital Zipori, Kareem Ayoub, Ashok Papat, Trilok Acharya, Luo
682 Yu, Dawn Bloxwich, Hugo Song, Paul Roit, Haiqiong Li, Aviel Boag, Nigamaa Nayakanti,
683 Bilva Chandra, Tianli Ding, Aahil Mehta, Cath Hope, Jiageng Zhang, Idan Heimlich Shtacher,
684 Kartikeya Badola, Ryo Nakashima, Andrei Sozanschi, Iulia Comşa, Ante Žužul, Emily Cave-
685 ness, Julian Odell, Matthew Watson, Dario de Cesare, Phillip Lippe, Derek Lockhart, Siddharth
686 Verma, Huizhong Chen, Sean Sun, Lin Zhuo, Aditya Shah, Prakhar Gupta, Alex Muzio, Ning
687 Niu, Amir Zait, Abhinav Singh, Meenu Gaba, Fan Ye, Prajit Ramachandran, Mohammad Saleh,
688 Raluca Ada Popa, Ayush Dubey, Frederick Liu, Sara Javanmardi, Mark Epstein, Ross Hemsley,
689 Richard Green, Nishant Ranka, Eden Cohen, Chuyuan Kelly Fu, Sanjay Ghemawat, Jed Borovik,
690 James Martens, Anthony Chen, Pranav Shyam, André Susano Pinto, Ming-Hsuan Yang, Alexan-
691 dru Tifrea, David Du, Boqing Gong, Ayushi Agarwal, Seungyeon Kim, Christian Frank, Saloni
692 Shah, Xiaodan Song, Zhiwei Deng, Ales Mikhalap, Kleopatra Chatziprimou, Timothy Chung,
693 Toni Creswell, Susan Zhang, Yennie Jun, Carl Lebsack, Will Truong, Slavica Andačić, Itay Yona,
694 Marco Fornoni, Rong Rong, Serge Toropov, Afzal Shama Soudagar, Andrew Audibert, Salah
695 Zaiem, Zaheer Abbas, Andrei Rusu, Sahitya Potluri, Shitao Weng, Anastasios Kementsietsidis,
696 Anton Tsitsulin, Daiyi Peng, Natalie Ha, Sanil Jain, Tejasi Latkar, Simeon Ivanov, Cory McLean,
697 Anirudh GP, Rajesh Venkataraman, Canoe Liu, Dilip Krishnan, Joel D'sa, Roey Yogev, Paul
698 Collins, Benjamin Lee, Lewis Ho, Carl Doersch, Gal Yona, Shawn Gao, Felipe Tiengo Ferreira,
699 Adnan Ozturel, Hannah Muckenhirn, Ce Zheng, Gargi Balasubramaniam, Mudit Bansal, George
700 van den Driessche, Sivan Eiger, Salem Haykal, Vedant Misra, Abhimanyu Goyal, Danilo Martins,
701 Gary Leung, Jonas Valfridsson, Four Flynn, Will Bishop, Chenxi Pang, Yoni Halpern, Honglin
Yi, Lawrence Moore, Yuvein, Zhu, Sridhar Thiagarajan, Yoel Drori, Zhisheng Xiao, Lucio Dery,
Rolf Jagerman, Jing Lu, Eric Ge, Vaibhav Aggarwal, Arjun Khare, Vinh Tran, Oded Elyada,
Ferran Alet, James Rubin, Ian Chou, David Tian, Libin Bai, Lawrence Chan, Lukasz Lew, Karo-
lis Misiunas, Taylan Bilal, Aniket Ray, Sindhu Raghuram, Alex Castro-Ros, Viral Carpenter,
CJ Zheng, Michael Kilgore, Josef Broder, Emily Xue, Praveen Kallakuri, Dheeru Dua, Nancy

702 Yuen, Steve Chien, John Schultz, Saurabh Agrawal, Reut Tsarfaty, Jingcao Hu, Ajay Kannan,
703 Dror Marcus, Nisarg Kothari, Baochen Sun, Ben Horn, Matko Bošnjak, Ferjad Naem, Dean
704 Hirsch, Lewis Chiang, Boya Fang, Jie Han, Qifei Wang, Ben Hora, Antoine He, Mario Lučić,
705 Beer Changpinyo, Anshuman Tripathi, John Youssef, Chester Kwak, Philippe Schlattner, Cat
706 Graves, Rémi Leblond, Wenjun Zeng, Anders Andreassen, Gabriel Rasskin, Yue Song, Eddie
707 Cao, Junhyuk Oh, Matt Hoffman, Wojtek Skut, Yichi Zhang, Jon Stritar, Xingyu Cai, Saarthak
708 Khanna, Kathie Wang, Shriya Sharma, Christian Reisswig, Younghoon Jun, Aman Prasad, Ta-
709 tiana Sholokhova, Preeti Singh, Adi Gerzi Rosenthal, Anian Ruoss, Françoise Beaufays, Sean
710 Kirmani, Dongkai Chen, Johan Schalkwyk, Jonathan Herzig, Been Kim, Josh Jacob, Damien
711 Vincent, Adrian N Reyes, Ivana Balazevic, Léonard Hussenot, Jon Schneider, Parker Barnes,
712 Luis Castro, Spandana Raj Babbula, Simon Green, Serkan Cabi, Nico Duduta, Danny Driess,
713 Rich Galt, Noam Velan, Junjie Wang, Hongyang Jiao, Matthew Mauger, Du Phan, Miteyan Patel,
714 Vlado Galić, Jerry Chang, Eyal Marcus, Matt Harvey, Julian Salazar, Elahe Dabir, Suraj Satishku-
715 mar Sheth, Amol Mandhane, Hanie Sedghi, Jeremiah Willcock, Amir Zandieh, Shruthi Prab-
716 hakara, Aida Amini, Antoine Miech, Victor Stone, Massimo Nicosia, Paul Niemczyk, Ying Xiao,
717 Lucy Kim, Sławek Kwasiborski, Vikas Verma, Ada Maksutaj Oflazer, Christoph Hirsenschall, Pe-
718 ter Sung, Lu Liu, Richard Everett, Michiel Bakker, Ágoston Weisz, Yufei Wang, Vivek Sam-
719 pathkumar, Uri Shaham, Bibo Xu, Yasemin Altun, Mingqiu Wang, Takaaki Saeki, Guanjie Chen,
720 Emanuel Taropa, Shanthal Vasanth, Sophia Austin, Lu Huang, Goran Petrovic, Qingyun Dou,
721 Daniel Golovin, Grigory Rozhdestvenskiy, Allie Culp, Will Wu, Motoki Sano, Divya Jain, Julia
722 Proskurnia, Sébastien Cevey, Alejandro Cruzado Ruiz, Piyush Patil, Mahdi Mirzazadeh, Eric Ni,
723 Javier Snaider, Lijie Fan, Alexandre Fréchet, AJ Pierigiovanni, Shariq Iqbal, Kenton Lee, Clau-
724 dio Fantacci, Jinwei Xing, Lisa Wang, Alex Irpan, David Raposo, Yi Luan, Zhuoyuan Chen, Har-
725 ish Ganapathy, Kevin Hui, Jiazhong Nie, Isabelle Guyon, Heming Ge, Roopali Vij, Hui Zheng,
726 Dayeong Lee, Alfonso Castaño, Khuslen Baatarsukh, Gabriel Ibagon, Alexandra Chronopoulou,
727 Nicholas FitzGerald, Shashank Viswanadha, Safeen Huda, Rivka Moroshko, Georgi Stoyanov,
728 Prateek Kolhar, Alain Vaucher, Ishaan Watts, Adhi Kuncoro, Henryk Michalewski, Satish Kam-
729 bala, Bat-Orgil Batsaikhan, Alek Andreev, Irina Jurenka, Maigo Le, Qihang Chen, Wael Al Jishi,
730 Sarah Chakera, Zhe Chen, Aditya Kini, Vikas Yadav, Aditya Siddhant, Iliia Labzovsky, Balaji
731 Lakshminarayanan, Carrie Grimes Bostock, Pankil Botadra, Ankesh Anand, Colton Bishop, Sam
732 Conway-Rahman, Mohit Agarwal, Yani Donchev, Achintya Singhal, Félix de Chaumont Quiry,
733 Natalia Ponomareva, Nishant Agrawal, Bin Ni, Kalpesh Krishna, Masha Samsikova, John Karro,
734 Yilun Du, Tamara von Glehn, Caden Lu, Christopher A. Choquette-Choo, Zhen Qin, Tingnan
735 Zhang, Sicheng Li, Divya Tyam, Swaroop Mishra, Wing Lowe, Colin Ji, Weiyi Wang, Man-
736 aal Faruqui, Ambrose Slone, Valentin Dalibard, Arunachalam Narayanaswamy, John Lambert,
737 Pierre-Antoine Manzagol, Dan Karliner, Andrew Bolt, Ivan Lobov, Aditya Kusupati, Chang
738 Ye, Xuan Yang, Heiga Zen, Nelson George, Mukul Bhutani, Olivier Lacombe, Robert Riachi,
739 Gagan Bansal, Rachel Soh, Yue Gao, Yang Yu, Adams Yu, Emily Nottage, Tania Rojas-Esponda,
740 James Noraky, Manish Gupta, Ragha Kotikalapudi, Jichuan Chang, Sanja Deur, Dan Graur, Alex
741 Mossin, Erin Farnese, Ricardo Figueira, Alexandre Moufarek, Austin Huang, Patrik Zochbauer,
742 Ben Ingram, Tongzhou Chen, Zelin Wu, Adrià Puigdomènech, Leland Rechis, Da Yu, Sri Gay-
743 atri Sundara Padmanabhan, Rui Zhu, Chu ling Ko, Andrea Banino, Samira Daruki, Aarush Sel-
744 van, Dhruva Bhaswar, Daniel Hernandez Diaz, Chen Su, Salvatore Scellato, Jennifer Brennan,
745 Woohyun Han, Grace Chung, Priyanka Agrawal, Urvashi Khandelwal, Khe Chai Sim, Mor-
746 gane Lustman, Sam Ritter, Kelvin Guu, Jiawei Xia, Prateek Jain, Emma Wang, Tyrone Hill,
747 Mirko Rossini, Marija Kostelac, Tautvydas Misiunas, Amit Sabne, Kyuyeun Kim, Ahmet Is-
748 cen, Congchao Wang, José Leal, Ashwin Sreevatsa, Utku Evci, Manfred Warmuth, Saket Joshi,
749 Daniel Suo, James Lottes, Garrett Honke, Brendan Jou, Stefani Karp, Jieru Hu, Himanshu Sahni,
750 Adrien Ali Taïga, William Kong, Samrat Ghosh, Renshen Wang, Jay Pavagadhi, Natalie Axels-
751 son, Nikolai Grigorev, Patrick Siegler, Rebecca Lin, Guohui Wang, Emilio Parisotto, Sharath
752 Maddineni, Krishan Subudhi, Eyal Ben-David, Elena Pochernina, Orgad Keller, Thi Avrahami,
753 Zhe Yuan, Pulkit Mehta, Jialu Liu, Sherry Yang, Wendy Kan, Katherine Lee, Tom Funkhouser,
754 Derek Cheng, Hongzhi Shi, Archit Sharma, Joe Kelley, Matan Eyal, Yury Malkov, Corentin Tal-
755 lec, Yuval Bahat, Shen Yan, Xintian, Wu, David Lindner, Chengda Wu, Avi Caciularu, Xiyang
Luo, Rodolphe Jenatton, Tim Zaman, Yingying Bi, Ilya Kornakov, Ganesh Mallya, Daisuke
Ikeda, Itay Karo, Anima Singh, Colin Evans, Praneeth Netrapalli, Vincent Nallatamby, Isaac
Tian, Yannis Assael, Vikas Raunak, Victor Carbune, Ioana Bica, Lior Madmoni, Dee Cattle,
Snchit Grover, Krishna Somandepalli, Sid Lall, Amelio Vázquez-Reina, Riccardo Patana, Jiaqi
Mu, Pranav Talluri, Maggie Tran, Rajeev Aggarwal, RJ Skerry-Ryan, Jun Xu, Mike Burrows,

756 Xiaoyue Pan, Edouard Yvinec, Di Lu, Zhiying Zhang, Duc Dung Nguyen, Hairong Mu, Gabriel
757 Barcik, Helen Ran, Lauren Beltrone, Krzysztof Choromanski, Dia Kharrat, Samuel Albanie, Sean
758 Purser-haskell, David Bieber, Carrie Zhang, Jing Wang, Tom Hudson, Zhiyuan Zhang, Han Fu,
759 Johannes Mauerer, Mohammad Hossein Bateni, AJ Maschinot, Bing Wang, Muye Zhu, Arjun
760 Pillai, Tobias Weyand, Shuang Liu, Oscar Akerlund, Fred Bertsch, Vittal Premachandran, Ali-
761 cia Jin, Vincent Roulet, Peter de Boursac, Shubham Mittal, Ndaba Ndebele, Georgi Karadzhov,
762 Sahra Ghalebikesabi, Ricky Liang, Allen Wu, Yale Cong, Nimesh Ghelani, Sumeet Singh, Ba-
763 har Fatemi, Warren, Chen, Charles Kwong, Alexey Kolganov, Steve Li, Richard Song, Chenkai
764 Kuang, Sobhan Miryoosefi, Dale Webster, James Wendt, Arkadiusz Socala, Guolong Su, Artur
765 Mendonça, Abhinav Gupta, Xiaowei Li, Tomy Tsai, Qiong, Hu, Kai Kang, Angie Chen, Ser-
766 tan Girgin, Yongqin Xian, Andrew Lee, Nolan Ramsden, Leslie Baker, Madeleine Clare Elish,
767 Varvara Krayvanova, Rishabh Joshi, Jiri Simsa, Yao-Yuan Yang, Piotr Ambroszczyk, Dipankar
768 Ghosh, Arjun Kar, Yuan Shangguan, Yumeya Yamamori, Yaroslav Akulov, Andy Brock, Hao-
769 tian Tang, Siddharth Vashishtha, Rich Munoz, Andreas Steiner, Kalyan Andra, Daniel Eppens,
770 Qixuan Feng, Hayato Kobayashi, Sasha Goldshtein, Mona El Mahdy, Xin Wang, Jilei, Wang,
771 Richard Killam, Tom Kwiatkowski, Kavya Kopparapu, Serena Zhan, Chao Jia, Alexei Bende-
772 bury, Sheryl Luo, Adrià Recasens, Timothy Knight, Jing Chen, Mohak Patel, YaGuang Li, Ben
773 Withbroe, Dean Weesner, Kush Bhatia, Jie Ren, Danielle Eisenbud, Ebrahim Songhori, Yanhua
774 Sun, Travis Choma, Tasos Kementsietsidis, Lucas Manning, Brian Roark, Wael Farhan, Jie Feng,
775 Susheel Tatineni, James Cobon-Kerr, Yunjie Li, Lisa Anne Hendricks, Isaac Noble, Chris Breaux,
776 Nate Kushman, Liqian Peng, Fuzhao Xue, Taylor Tobin, Jamie Rogers, Josh Lipschultz, Chris
777 Alberti, Alexey Vlaskin, Mostafa Deghani, Roshan Sharma, Tris Warkentin, Chen-Yu Lee, Be-
778 nigno Uria, Da-Cheng Juan, Angad Chandorkar, Hila Sheftel, Ruibo Liu, Elnaz Davoodi, Borja
779 De Balle Pigem, Kedar Dhamdhere, David Ross, Jonathan Hoech, Mahdis Mahdieh, Li Liu, Qiu-
780 jia Li, Liam McCafferty, Chenxi Liu, Markus Mircea, Yunting Song, Omkar Savant, Alaa Saade,
781 Colin Cherry, Vincent Hellendoorn, Siddharth Goyal, Paul Pucciarelli, David Vilar Torres, Zo-
782 har Yahav, Hyo Lee, Lars Lowe Sjoesund, Christo Kirov, Bo Chang, Deepanway Ghoshal, Lu Li,
783 Gilles Baechler, Sébastien Pereira, Tara Sainath, Anudhyan Boral, Dominik Grewe, Afief Halumi,
784 Nguyet Minh Phu, Tianxiao Shen, Marco Tulio Ribeiro, Dhriti Varma, Alex Kaskasoli, Vlad Fein-
785 berg, Navneet Potti, Jarrod Kahn, Matheus Wisniewski, Shakir Mohamed, Arnar Mar Hrafnkel-
786 son, Bobak Shahriari, Jean-Baptiste Lespiau, Lisa Patel, Legg Yeung, Tom Paine, Lantao Mei,
787 Alex Ramirez, Rakesh Shivanna, Li Zhong, Josh Woodward, Guilherme Tubone, Samira Khan,
788 Heng Chen, Elizabeth Nielsen, Catalin Ionescu, Utsav Prabhu, Mingcen Gao, Qingze Wang, Sean
789 Augenstein, Neesha Subramaniam, Jason Chang, Fotis Iliopoulos, Jiaming Luo, Myriam Khan,
790 Weicheng Kuo, Denis Teplyashin, Florence Perot, Logan Kilpatrick, Amir Globerson, Hongkun
791 Yu, Anfal Siddiqui, Nick Sukhanov, Arun Kandoor, Umang Gupta, Marco Andreetto, Moran
792 Ambar, Donnie Kim, Paweł Wesolowski, Sarah Perrin, Ben Limonchik, Wei Fan, Jim Stephan,
793 Ian Stewart-Binks, Ryan Kappedal, Tong He, Sarah Cogan, Romina Datta, Tong Zhou, Jiayu
794 Ye, Leandro Kieliger, Ana Ramalho, Kyle Kastner, Fabian Mentzer, Wei-Jen Ko, Arun Suggala,
795 Tianhao Zhou, Shiraz Butt, Hana Strejček, Lior Belenki, Subhashini Venugopalan, Mingyang
796 Ling, Evgenii Eltyshev, Yunxiao Deng, Geza Kovacs, Mukund Raghavachari, Hanjun Dai, Tal
797 Schuster, Steven Schwarcz, Richard Nguyen, Arthur Nguyen, Gavin Buttimore, Shrestha Basu
798 Mallick, Sudeep Gandhe, Seth Benjamin, Michal Jastrzebski, Le Yan, Sugato Basu, Chris Apps,
799 Isabel Edkins, James Allingham, Immanuel Odisho, Tomas Kocisky, Jewel Zhao, Linting Xue,
800 Apoorv Reddy, Chrysovalantis Anastasiou, Aviel Atias, Sam Redmond, Kieran Milan, Nico-
801 las Heess, Herman Schmit, Allan Dafoe, Daniel Andor, Tynan Gangwani, Anca Dragan, Sheng
802 Zhang, Ashyana Kachra, Gang Wu, Siyang Xue, Kevin Aydin, Siqi Liu, Yuxiang Zhou, Mahan
803 Malihi, Austin Wu, Siddharth Gopal, Candice Schumann, Peter Stys, Alek Wang, Mirek Olšák,
804 Dangyi Liu, Christian Schallhart, Yiran Mao, Demetra Brady, Hao Xu, Tomas Mery, Chawin
805 Sitawarin, Siva Velusamy, Tom Cobley, Alex Zhai, Christian Walder, Nitzan Katz, Ganesh Jawa-
806 har, Chinmay Kulkarni, Antoine Yang, Adam Paszke, Yinan Wang, Bogdan Damoc, Zalán Bor-
807 sos, Ray Smith, Jinning Li, Mansi Gupta, Andrei Kapishnikov, Sushant Prakash, Florian Luisier,
808 Rishabh Agarwal, Will Grathwohl, Kuangyuan Chen, Kehang Han, Nikhil Mehta, Andrew Over,
809 Shekoofeh Azizi, Lei Meng, Niccolò Dal Santo, Kelvin Zheng, Jane Shapiro, Igor Petrovski,
Jeffrey Hui, Amin Ghafouri, Jasper Snoek, James Qin, Mandy Jordan, Caitlin Sikora, Jonathan
Malmaud, Yuheng Kuang, Aga Świetlik, Ruoxin Sang, Chongyang Shi, Leon Li, Andrew Rosen-
berg, Shubin Zhao, Andy Crawford, Jan-Thorsten Peter, Yun Lei, Xavier Garcia, Long Le, Todd
Wang, Julien Amelot, Dave Orr, Praneeth Kacham, Dana Alon, Gladys Tyen, Abhinav Arora,
James Lyon, Alex Kurakin, Mimi Ly, Theo Guidroz, Zhipeng Yan, Rina Panigrahy, Pingmei

810 Xu, Thais Kagohara, Yong Cheng, Eric Noland, Jinhyuk Lee, Jonathan Lee, Cathy Yip, Maria
811 Wang, Efrat Nehoran, Alexander Bykovsky, Zhihao Shan, Ankit Bhagatwala, Chaochao Yan, Jie
812 Tan, Guillermo Garrido, Dan Ethier, Nate Hurley, Grace Vesom, Xu Chen, Siyuan Qiao, Ab-
813 hishek Nayyar, Julian Walker, Paramjit Sandhu, Mihaela Rosca, Danny Swisher, Mikhail Dek-
814 tiarev, Josh Dillon, George-Cristian Muraru, Manuel Tragut, Artiom Myaskovsky, David Reid,
815 Marko Velic, Owen Xiao, Jasmine George, Mark Brand, Jing Li, Wenhao Yu, Shane Gu, Xiang
816 Deng, François-Xavier Aubet, Soheil Hassas Yeganeh, Fred Alcober, Celine Smith, Trevor Cohn,
817 Kay McKinney, Michael Tschannen, Ramesh Sampath, Gowoon Cheon, Liangchen Luo, Luyang
818 Liu, Jordi Orbay, Hui Peng, Gabriela Botea, Xiaofan Zhang, Charles Yoon, Cesar Magalhaes,
819 Paweł Stradomski, Ian Mackinnon, Steven Hemingray, Kumaran Venkatesan, Rhys May, Jaeyoun
820 Kim, Alex Druinsky, Jingchen Ye, Zheng Xu, Terry Huang, Jad Al Abdallah, Adil Dostmohamed,
821 Rachana Fellingner, Tsendsuren Munkhdalai, Akanksha Maurya, Peter Garst, Yin Zhang, Maxim
822 Krikun, Simon Bucher, Aditya Srikanth Veerubhotla, Yaxin Liu, Sheng Li, Nishesh Gupta, Jakub
823 Adamek, Hanwen Chen, Bennett Orlando, Aleksandr Zaks, Joost van Amersfoort, Josh Camp, Hui
824 Wan, HyunJeong Choe, Zhichun Wu, Kate Olszewska, Weiren Yu, Archita Vadali, Martin Scholz,
825 Daniel De Freitas, Jason Lin, Amy Hua, Xin Liu, Frank Ding, Yichao Zhou, Boone Severson, Ka-
826 terina Tsihlas, Samuel Yang, Tammo Spalink, Varun Yerram, Helena Pankov, Rory Blevins, Ben
827 Vargas, Sarthak Jauhari, Matt Miecnikowski, Ming Zhang, Sandeep Kumar, Clement Farabet,
828 Charline Le Lan, Sebastian Flennerhag, Yonatan Bitton, Ada Ma, Arthur Braźniskas, Eli Collins,
829 Niharika Ahuja, Sneha Kudugunta, Anna Bortsova, Minh Giang, Wanzheng Zhu, Ed Chi, Scott
830 Lundberg, Alexey Stern, Subha Puttagunta, Jing Xiong, Xiao Wu, Yash Pande, Amit Jhindal,
831 Daniel Murphy, Jon Clark, Marc Brockschmidt, Maxine Deines, Kevin R. McKee, Dan Bahir,
832 Jiajun Shen, Minh Truong, Daniel McDuff, Andrea Gesmundo, Edouard Rosseel, Bowen Liang,
833 Ken Caluwaerts, Jessica Hamrick, Joseph Kready, Mary Cassin, Rishikesh Ingale, Li Lao, Scott
834 Pollom, Yifan Ding, Wei He, Lizzetth Bellot, Joana Iljazi, Ramya Sree Boppana, Shan Han, Tara
835 Thompson, Amr Khalifa, Anna Bulanova, Blagoj Mitrevski, Bo Pang, Emma Cooney, Tian Shi,
836 Rey Coaguila, Tamar Yakar, Marc’aurelio Ranzato, Nikola Momchev, Chris Rawles, Zachary
837 Charles, Young Maeng, Yuan Zhang, Rishabh Bansal, Xiaokai Zhao, Brian Albert, Yuan Yuan,
838 Sudheendra Vijayanarasimhan, Roy Hirsch, Vinay Ramasesh, Kiran Vodrahalli, Xingyu Wang,
839 Arushi Gupta, DJ Strouse, Jianmo Ni, Roma Patel, Gabe Taubman, Zhouyuan Huo, Dero Gharib-
840 ian, Marianne Monteiro, Hoi Lam, Shobha Vasudevan, Aditi Chaudhary, Isabela Albuquerque,
841 Kilol Gupta, Sebastian Riedel, Chaitra Hegde, Avraham Ruderman, András György, Marcus
842 Wainwright, Ashwin Chaugule, Burcu Karagol Ayan, Tomer Levinboim, Sam Shleifer, Yogesh
843 Kalley, Vahab Mirrokni, Abhishek Rao, Prabakar Radhakrishnan, Jay Hartford, Jialin Wu, Zhen-
844 hai Zhu, Francesco Bertolini, Hao Xiong, Nicolas Serrano, Hamish Tomlinson, Myle Ott, Yifan
845 Chang, Mark Graham, Jian Li, Marco Liang, Xiangzhu Long, Sebastian Borgeaud, Yanif Ahmad,
846 Alex Grills, Diana Mincu, Martin Izzard, Yuan Liu, Jinyu Xie, Louis O’Byrne, Sameera Ponda,
847 Simon Tong, Michelle Liu, Dan Malkin, Khalid Salama, Yuankai Chen, Rohan Anil, Anand Rao,
848 Rigel Swavelly, Misha Bilenko, Nina Anderson, Tat Tan, Jing Xie, Xing Wu, Lijun Yu, Oriol
849 Vinyals, Andrey Ryabtsev, Rumens Dangovski, Kate Baumli, Daniel Keysers, Christian Wright,
850 Zoe Ashwood, Betty Chan, Artem Shtefan, Yaohui Guo, Ankur Bapna, Radu Soricut, Steven
851 Pecht, Sabela Ramos, Rui Wang, Jiahao Cai, Trieu Trinh, Paul Barham, Linda Friso, Eli Stickgold,
852 Xiangzhuo Ding, Siamak Shakeri, Diego Ardila, Eleftheria Briakou, Phil Culliton, Adam Raveret,
853 Jingyu Cui, David Saxton, Subhrajit Roy, Javad Azizi, Pengcheng Yin, Lucia Loher, Andrew Bun-
854 ner, Min Choi, Faruk Ahmed, Eric Li, Yin Li, Shengyang Dai, Michael Elabd, Sriram Ganapathy,
855 Shivani Agrawal, Yiqing Hua, Paige Kunkle, Sujevan Rajayogam, Arun Ahuja, Arthur Conmy,
856 Alex Vasiloff, Parker Beak, Christopher Yew, Jayaram Mudigonda, Bartek Wydrowski, Jon Blan-
857 ton, Zhengdong Wang, Yann Dauphin, Zhuo Xu, Martin Polacek, Xi Chen, Hexiang Hu, Pauline
858 Sho, Markus Kunesch, Mehdi Hafezi Manshadi, Eliza Rutherford, Bo Li, Sissie Hsiao, Iain Barr,
859 Alex Tudor, Matija Kecman, Arsha Nagrani, Vladimir Pchelin, Martin Sundermeyer, Aishwarya P
860 S, Abhijit Karmarkar, Yi Gao, Grishma Chole, Olivier Bachem, Isabel Gao, Arturo BC, Matt
861 Dibb, Mauro Verzetti, Felix Hernandez-Campos, Yana Lunts, Matthew Johnson, Julia Di Trapani,
862 Raphael Koster, Idan Brusilovsky, Binbin Xiong, Megha Mohabey, Han Ke, Joe Zou, Tea Sabolić,
863 Victor Campos, John Palowitch, Alex Morris, Linhai Qiu, Pranavaraj Ponnuramu, Fangtao Li,
Vivek Sharma, Kiranbir Sodhia, Kaan Tekelioglu, Aleksandr Chuklin, Madhavi Yenugula, Erika
Gemzer, Theofilos Strinopoulos, Sam El-Husseini, Huiyu Wang, Yan Zhong, Edouard Leurent,
Paul Natsev, Weijun Wang, Dre Mahaarachchi, Tao Zhu, Songyou Peng, Sami Alabed, Cheng-
Chun Lee, Anthony Brohan, Arthur Szlam, GS Oh, Anton Kovsharov, Jenny Lee, Renee Wong,
Megan Barnes, Gregory Thornton, Felix Gimeno, Omer Levy, Martin Sevenich, Melvin Johnson,

864 Jonathan Mallinson, Robert Dadashi, Ziyue Wang, Qingchun Ren, Preethi Lahoti, Arka Dhar,
865 Josh Feldman, Dan Zheng, Thatcher Ulrich, Liviu Panait, Michiel Blokzijl, Cip Baetu, Josip
866 Matak, Jitendra Harlalka, Maulik Shah, Tal Marian, Daniel von Dincklage, Cosmo Du, Ruy Ley-
867 Wild, Bethanie Brownfield, Max Schumacher, Yury Stuken, Shadi Noghabi, Sonal Gupta, Xiaoqi
868 Ren, Eric Malmi, Felix Weissenberger, Blanca Huergo, Maria Bauza, Thomas Lampe, Arthur
869 Douillard, Mojtaba Seyedhosseini, Roy Frostig, Zoubin Ghahramani, Kelvin Nguyen, Kashyap
870 Krishnakumar, Chengxi Ye, Rahul Gupta, Alireza Nazari, Robert Geirhos, Pete Shaw, Ahmed
871 Eleryan, Dima Damen, Jennimaria Palomaki, Ted Xiao, Qiyin Wu, Quan Yuan, Phoenix Mead-
872 owlark, Matthew Bilotti, Raymond Lin, Mukund Sridhar, Yannick Schroecker, Da-Woon Chung,
873 Jincheng Luo, Trevor Strohman, Tianlin Liu, Anne Zheng, Jesse Emond, Wei Wang, Andrew
874 Lampinen, Toshiyuki Fukuzawa, Folawiyo Campbell-Ajala, Monica Roy, James Lee-Thorp, Lily
875 Wang, Iftekhar Naim, Tony, Nguy ên, Guy Bensky, Aditya Gupta, Dominika Rogozińska, Justin
876 Fu, Thanumalayan Sankaranarayanan Pillai, Petar Veličković, Shahar Drath, Philipp Neubeck,
877 Vaibhav Tulsyan, Arseniy Klimovskiy, Don Metzler, Sage Stevens, Angel Yeh, Junwei Yuan,
878 Tianhe Yu, Kelvin Zhang, Alec Go, Vincent Tsang, Ying Xu, Andy Wan, Isaac Galatzer-Levy,
879 Sam Sobell, Abodunrinwa Toki, Elizabeth Salesky, Wenlei Zhou, Diego Antognini, Sholto Dou-
880 glas, Shimu Wu, Adam Lelkes, Frank Kim, Paul Cavallaro, Ana Salazar, Yuchi Liu, James Besley,
881 Tiziana Refice, Yiling Jia, Zhang Li, Michal Sokolik, Arvind Kannan, Jon Simon, Jo Chick, Avia
882 Aharon, Meet Gandhi, Mayank Daswani, Keyvan Amiri, Vighnesh Birodkar, Abe Ittycheriah,
883 Peter Grabowski, Oscar Chang, Charles Sutton, Zhixin, Lai, Umesh Telang, Susie Sargsyan, Tao
884 Jiang, Raphael Hoffmann, Nicole Brichtova, Matteo Hessel, Jonathan Halcrow, Sammy Jerome,
885 Geoff Brown, Alex Tomala, Elena Buchatskaya, Dian Yu, Sachit Menon, Pol Moreno, Yuguo
886 Liao, Vicky Zayats, Luming Tang, SQ Mah, Ashish Shenoy, Alex Siegman, Majid Hadian, Okwan
887 Kwon, Tao Tu, Nima Khajehnouri, Ryan Foley, Parisa Haghani, Zhongru Wu, Vaishakh Keshava,
888 Khyatti Gupta, Tony Bruguier, Rui Yao, Danny Karmon, Luisa Zintgraf, Zhicheng Wang, En-
889 rique Piqueras, Junehyuk Jung, Jenny Brennan, Diego Machado, Marissa Giustina, MH Tessler,
890 Kamyu Lee, Qiao Zhang, Joss Moore, Kaspar Daugaard, Alexander Frömmgen, Jennifer Beat-
891 tie, Fred Zhang, Daniel Kasenberg, Ty Geri, Danfeng Qin, Gaurav Singh Tomar, Tom Ouyang,
892 Tianli Yu, Luwei Zhou, Rajiv Mathews, Andy Davis, Yaoyiran Li, Jai Gupta, Damion Yates,
893 Linda Deng, Elizabeth Kemp, Ga-Young Joun, Sergei Vassilvitskii, Mandy Guo, Pallavi LV,
894 Dave Dopson, Sami Lachgar, Lara McConnaughey, Himadri Choudhury, Dragos Dena, Aaron
895 Cohen, Joshua Ainslie, Sergey Levi, Parthasarathy Gopavarapu, Polina Zablotskaia, Hugo Val-
896 let, Sanaz Bahargam, Xiaodan Tang, Nenad Tomasev, Ethan Dyer, Daniel Balle, Hongrae Lee,
897 William Bono, Jorge Gonzalez Mendez, Vadim Zubov, Shentao Yang, Ivor Rendulic, Yanyan
898 Zheng, Andrew Hogue, Golan Pundak, Ralph Leith, Avishkar Bhoopchand, Michael Han, Mis-
899 lav Žanić, Tom Schaul, Manolis Delakis, Tejas Iyer, Guanyu Wang, Harman Singh, Abdelrah-
900 man Abdelhamed, Tara Thomas, Siddhartha Brahma, Hilal Dib, Naveen Kumar, Wenxuan Zhou,
901 Liang Bai, Pushkar Mishra, Jiao Sun, Valentin Anklin, Roykrong Sukkerd, Lauren Agubuzu, An-
902 ton Briukhov, Anmol Gulati, Maximilian Sieb, Fabio Pardo, Sara Nasso, Junquan Chen, Kexin
903 Zhu, Tiberiu Sosea, Alex Goldin, Keith Rush, Spurthi Amba Hombaiah, Andreas Noever, Al-
904 lan Zhou, Sam Haves, Mary Phuong, Jake Ades, Yi ting Chen, Lin Yang, Joseph Pagadora,
905 Stan Bileschi, Victor Cotruta, Rachel Saputro, Arijit Pramanik, Sean Ammirati, Dan Garrette,
906 Kevin Villela, Tim Blyth, Canfer Akbulut, Neha Jha, Alban Rrustemi, Arissa Wongpanich, Chi-
907 rag Nagpal, Yonghui Wu, Morgane Rivière, Sergey Kishchenko, Pranesh Srinivasan, Alice Chen,
908 Animesh Sinha, Trang Pham, Bill Jia, Tom Hennigan, Anton Bakalov, Nithya Attaluri, Drew
909 Garmon, Daniel Rodriguez, Dawid Wegner, Wenhao Jia, Evan Senter, Noah Fiedel, Denis Pe-
910 tek, Yuchuan Liu, Cassidy Hardin, Harshal Tushar Lehri, Joao Carreira, Sara Smoot, Marcel
911 Prasetya, Nami Akazawa, Anca Stefanoiu, Chia-Hua Ho, Anelia Angelova, Kate Lin, Min Kim,
912 Charles Chen, Marcin Sieniek, Alice Li, Tongfei Guo, Sorin Baltateanu, Pouya Tafti, Michael
913 Wunder, Nadav Olmert, Divyansh Shukla, Jingwei Shen, Neel Kovelamudi, Balaji Venkatraman,
914 Seth Neel, Romal Thoppilan, Jerome Connor, Frederik Benzing, Axel Stjerngren, Golnaz Ghi-
915 asi, Alex Polozov, Joshua Howland, Theophane Weber, Justin Chiu, Ganesh Poomal Girirajan,
916 Andreas Terzis, Pidong Wang, Fangda Li, Yoav Ben Shalom, Dinesh Tewari, Matthew Denton,
917 Roe Aharoni, Norbert Kalb, Heri Zhao, Junlin Zhang, Angelos Filos, Matthew Rahtz, Lalit Jain,
Connie Fan, Vitor Rodrigues, Ruth Wang, Richard Shin, Jacob Austin, Roman Ring, Mariella
Sanchez-Vargas, Mehadi Hassen, Ido Kessler, Uri Alon, Gufeng Zhang, Wenhao Chen, Yenai Ma,
Xiance Si, Le Hou, Azalia Mirhoseini, Marc Wilson, Geoff Bacon, Becca Roelofs, Lei Shu,
Gautam Vasudevan, Jonas Adler, Artur Dwornik, Tayfun Terzi, Matt Lawlor, Harry Askham,
Mike Bernico, Xuanyi Dong, Chris Hidey, Kevin Kilgour, Gaël Liu, Surya Bhupatiraju, Luke

918 Leonhard, Siqi Zuo, Partha Talukdar, Qing Wei, Aliaksei Severyn, Vít Lišík, Jong Lee, Aditya
 919 Tripathi, SK Park, Yossi Matias, Hao Liu, Alex Ruiz, Rajesh Jayaram, Jackson Tolins, Pierre
 920 Marcenac, Yiming Wang, Bryan Seybold, Henry Prior, Deepak Sharma, Jack Weber, Mikhail
 921 Sirotenko, Yunhsuan Sung, Dayou Du, Ellie Pavlick, Stefan Zinke, Markus Freitag, Max Dylla,
 922 Montse Gonzalez Arenas, Natan Potikha, Omer Goldman, Connie Tao, Rachita Chhaparia, Maria
 923 Voitovich, Pawan Dogra, Andrija Ražnatović, Zak Tsai, Chong You, Oleaser Johnson, George
 924 Tucker, Chenjie Gu, Jae Yoo, Maryam Majzoubi, Valentin Gabeur, Bahram Raad, Rocky Rhodes,
 925 Kashyap Kolipaka, Heidi Howard, Geta Sampemane, Benny Li, Chulayuth Asawaroengchai, Duy
 926 Nguyen, Chiyuan Zhang, Timothee Cour, Xinxin Yu, Zhao Fu, Joe Jiang, Po-Sen Huang, Gabriela
 927 Surita, Iñaki Iturrate, Yael Karov, Michael Collins, Martin Baeuml, Fabian Fuchs, Shilpa Shetty,
 928 Swaroop Ramaswamy, Sayna Ebrahimi, Qiuchen Guo, Jeremy Shar, Gabe Barth-Maron, Sravanti
 929 Addepalli, Bryan Richter, Chin-Yi Cheng, Eugénie Rives, Fei Zheng, Johannes Griesser, Nishanth
 930 Dikkala, Yoel Zeldes, Ilkin Safarli, Dipanjan Das, Himanshu Srivastava, Sadh MNM Khan, Xin
 931 Li, Aditya Pandey, Larisa Markeeva, Dan Belov, Qiqi Yan, Miłkołaj Rybiński, Tao Chen, Megha
 932 Nawhal, Michael Quinn, Vineetha Govindaraj, Sarah York, Reed Roberts, Roopal Garg, Namrata
 933 Godbole, Jake Abernethy, Anil Das, Lam Nguyen Thiet, Jonathan Tompson, John Nham, Neera
 934 Vats, Ben Caine, Wesley Helmholtz, Francesco Pongetti, Yeongil Ko, James An, Clara Huiyi Hu,
 935 Yu-Cheng Ling, Julia Pawar, Robert Leland, Keisuke Kinoshita, Waleed Khawaja, Marco Selvi,
 936 Eugene Ie, Danila Sinopalnikov, Lev Prolev, Nilesch Tripuraneni, Michele Bevilacqua, Seungji
 937 Lee, Clayton Sanford, Dan Suh, Dustin Tran, Jeff Dean, Simon Baumgartner, Jens Heitkaemper,
 938 Sagar Gubbi, Kristina Toutanova, Yichong Xu, Chandu Thekkath, Keran Rong, Palak Jain, Annie
 939 Xie, Yan Virin, Yang Li, Lubo Litchev, Richard Powell, Tarun Bharti, Adam Kraft, Nan Hua,
 940 Marissa Ikonomidis, Ayal Hitron, Sanjiv Kumar, Loic Matthey, Sophie Bridgers, Lauren Lax,
 941 Ishaan Malhi, Ondrej Skopek, Ashish Gupta, Jiawei Cao, Michelle Rasquinha, Siim Pöder, Woj-
 942 ciech Stokowiec, Nicholas Roth, Guowang Li, Michaël Sander, Joshua Kessinger, Vihan Jain, Ed-
 943 ward Loper, Wonpyo Park, Michal Yarom, Liquan Cheng, Guru Guruganesh, Kanishka Rao, Yan
 944 Li, Catarina Barros, Mikhail Sushkov, Chun-Sung Ferng, Rohin Shah, Ophir Aharoni, Ravin Ku-
 945 mar, Tim McConnell, Peiran Li, Chen Wang, Fernando Pereira, Craig Swanson, Fayaz Jamil, Yan
 946 Xiong, Anitha Vijayakumar, Prakash Shroff, Kedar Soparkar, Jindong Gu, Livio Baldini Soares,
 947 Eric Wang, Kushal Majmundar, Aurora Wei, Kai Bailey, Nora Kassner, Chizu Kawamoto, Goran
 948 Žužić, Victor Gomes, Abhirut Gupta, Michael Guzman, Ishita Dasgupta, Xinyi Bai, Zhufeng Pan,
 949 Francesco Piccinno, Hadas Natalie Vogel, Octavio Ponce, Adrian Hutter, Paul Chang, Pan-Pan
 950 Jiang, Ionel Gog, Vlad Ionescu, James Manyika, Fabian Pedregosa, Harry Ragan, Zach Behrman,
 951 Ryan Mullins, Coline Devin, Aroonlok Pyne, Swapnil Gawde, Martin Chadwick, Yiming Gu,
 952 Sasan Tavakkol, Andy Twigg, Naman Goyal, Ndidi Elue, Anna Goldie, Srinivasan Venkatachary,
 953 Hongliang Fei, Ziqiang Feng, Marvin Ritter, Isabel Leal, Sudeep Dasari, Pei Sun, Alif Raditya
 954 Rochman, Brendan O’Donoghue, Yuchen Liu, Jim Sproch, Kai Chen, Natalie Clay, Slav Petrov,
 955 Sailesh Sidhwani, Ioana Mihailescu, Alex Panagopoulos, AJ Piergiovanni, Yunfei Bai, George
 956 Powell, Deep Karkhanis, Trevor Yacovone, Petr Mitrichev, Joe Kovac, Dave Uthus, Amir Yaz-
 957 danbakhsh, David Amos, Steven Zheng, Bing Zhang, Jin Miao, Bhuvana Ramabhadran, Soroush
 958 Radpour, Shantanu Thakoor, Josh Newlan, Oran Lang, Orion Jankowski, Shikhar Bharadwaj,
 959 Jean-Michel Sarr, Shereen Ashraf, Sneha Mondal, Jun Yan, Ankit Singh Rawat, Sarmishta Velury,
 960 Greg Kochanski, Tom Eccles, Franz Och, Abhanshu Sharma, Ethan Mahintorabi, Alex Gurney,
 961 Carrie Muir, Vered Cohen, Saksham Thakur, Adam Bloniarz, Asier Mujika, Alexander Pritzel,
 962 Paul Caron, Altaf Rahman, Fiona Lang, Yasumasa Onoe, Petar Sirkovic, Jay Hoover, Ying Jian,
 963 Pablo Duque, Arun Narayanan, David Soergel, Alex Haig, Loren Maggiore, Shyamal Buch,
 964 Josef Dean, Ilya Figotin, Igor Karpov, Shaleen Gupta, Denny Zhou, Muhuan Huang, Ashwin
 965 Vaswani, Christopher Semturs, Kaushik Shivakumar, Yu Watanabe, Vinodh Kumar Rajendran,
 966 Eva Lu, Yanhan Hou, Wenting Ye, Shikhar Vashishth, Nana Nti, Vytenis Sakenas, Darren Ni,
 967 Doug DeCarlo, Michael Bendersky, Sumit Bagri, Nacho Cano, Elijah Peake, Simon Tokumine,
 968 Varun Godbole, Carlos Guía, Tanya Lando, Vittorio Selo, Seher Ellis, Danny Tarlow, Daniel
 969 Gillick, Alessandro Epasto, Siddhartha Reddy Jonnalagadda, Meng Wei, Meiyang Xie, Ankur
 970 Taly, Michela Paganini, Mukund Sundararajan, Daniel Toyama, Ting Yu, Dessie Petrova, Aneesh
 971 Pappu, Rohan Agrawal, Senaka Buttipitiya, Justin Frye, Thomas Buschmann, Remi Crocker,
 Marco Tagliasacchi, Mengchao Wang, Da Huang, Sagi Perel, Brian Wieder, Hideto Kazawa,
 Weiyue Wang, Jeremy Cole, Himanshu Gupta, Ben Golan, Seojin Bang, Nitish Kulkarni, Ken
 Franko, Casper Liu, Doug Reid, Sid Dalmia, Jay Whang, Kevin Cen, Prasha Sundaram, Johan
 Ferret, Berivan Isik, Lucian Ionita, Guan Sun, Anna Shekhawat, Muqthar Mohammad, Philip
 Pham, Ronny Huang, Karthik Raman, Xingyi Zhou, Ross Mcilroy, Austin Myers, Sheng Peng,

972 Jacob Scott, Paul Covington, Sofia Erell, Pratik Joshi, João Gabriel Oliveira, Natasha Noy, Tajwar
973 Nasir, Jake Walker, Vera Axelrod, Tim Dozat, Pu Han, Chun-Te Chu, Eugene Weinstein, Anand
974 Shukla, Shreyas Chandrakaladharan, Petra Poklucar, Bonnie Li, Ye Jin, Prem Eruvbetine, Steven
975 Hansen, Avigail Dabush, Alon Jacovi, Samrat Phatale, Chen Zhu, Steven Baker, Mo Shomrat,
976 Yang Xiao, Jean Pouget-Abadie, Mingyang Zhang, Fanny Wei, Yang Song, Helen King, Yiling
977 Huang, Yun Zhu, Ruoxi Sun, Juliana Vicente Franco, Chu-Cheng Lin, Sho Arora, Hui, Li, Vi-
978 vian Xia, Luke Vilnis, Mariano Schain, Kaiz Alarakyia, Laurel Prince, Aaron Phillips, Caleb
979 Habtegebriel, Luyao Xu, Huan Gui, Santiago Ontanon, Lora Aroyo, Karan Gill, Peggy Lu,
980 Yash Katariya, Dhruv Madeka, Shankar Krishnan, Shubha Srinivas Raghvendra, James Freed-
981 man, Yi Tay, Gaurav Menghani, Peter Choy, Nishita Shetty, Dan Abolafia, Doron Kukliansky,
982 Edward Chou, Jared Lichtarge, Ken Burke, Ben Coleman, Dee Guo, Larry Jin, Indro Bhat-
983 tacharya, Victoria Langston, Yiming Li, Suyog Kotecha, Alex Yakubovich, Xinyun Chen, Pe-
984 tre Petrov, Tolly Powell, Yanzhang He, Corbin Quick, Kanav Garg, Dawsen Hwang, Yang Lu,
985 Srinadh Bhojanapalli, Kristian Kjemis, Ramin Mehran, Aaron Archer, Hado van Hasselt, Ash-
986 win Balakrishna, JK Kearns, Meiqi Guo, Jason Riesa, Mikita Sazanovich, Xu Gao, Chris Sauer,
987 Chengrun Yang, XiangHai Sheng, Thomas Jimma, Wouter Van Gansbeke, Vitaly Nikolaev, Wei
988 Wei, Katie Millican, Ruizhe Zhao, Justin Snyder, Levent Bolelli, Maura O’Brien, Shawn Xu, Fei
989 Xia, Wentao Yuan, Arvind Neelakantan, David Barker, Sachin Yadav, Hannah Kirkwood, Far-
990 rooq Ahmad, Joel Wee, Jordan Grimstad, Boyu Wang, Matthew Wiethoff, Shane Settle, Miaosen
991 Wang, Charles Blundell, Jingjing Chen, Chris Duvarney, Grace Hu, Olaf Ronneberger, Alex Lee,
992 Yuanzhen Li, Abhishek Chakladar, Alena Butryna, Georgios Evangelopoulos, Guillaume Des-
993 jardins, Jonni Kanerva, Henry Wang, Averi Nowak, Nick Li, Alyssa Loo, Art Khurshudov, Lau-
994 rent El Shafey, Nagabhushan Baddi, Karel Lenc, Yasaman Razeghi, Tom Lieber, Amer Sinha,
995 Xiao Ma, Yao Su, James Huang, Asahi Ushio, Hanna Klimczak-Plucińska, Kareem Mohamed,
996 JD Chen, Simon Osindero, Stav Ginzburg, Lampros Lamprou, Vasilisa Bashlovkina, Duc-Hieu
997 Tran, Ali Khodaei, Ankit Anand, Yixian Di, Ramy Eskander, Manish Reddy Vuyyuru, Jasmine
998 Liu, Aishwarya Kamath, Roman Goldenberg, Mathias Bellaïche, Juliette Pluto, Bill Rosgen, Has-
999 san Mansoor, William Wong, Suhas Ganesh, Eric Bailey, Scott Baird, Dan Deutsch, Jinoo Baek,
1000 Xuhui Jia, Chansoo Lee, Abe Friesen, Nathaniel Braun, Kate Lee, Amayika Panda, Steven M.
1001 Hernandez, Duncan Williams, Jianqiao Liu, Ethan Liang, Arnaud Autef, Emily Pitler, Deepali
1002 Jain, Phoebe Kirk, Oskar Bunyan, Jaume Sanchez Elias, Tongxin Yin, Machel Reid, Aedan
1003 Pope, Nikita Putikhin, Bidisha Samanta, Sergio Guadarrama, Dahun Kim, Simon Rowe, Mar-
1004 cella Valentine, Geng Yan, Alex Salcianu, David Silver, Gan Song, Richa Singh, Shuai Ye, Han-
1005 nah DeBalsi, Majd Al Merey, Eran Ofek, Albert Webson, Shibl Mourad, Ashwin Kakarla, Silvio
1006 Lattanzi, Nick Roy, Evgeny Sluzhaev, Christina Butterfield, Alessio Tonioni, Nathan Waters,
1007 Sudhindra Kopalle, Jason Chase, James Cohan, Girish Ramchandra Rao, Robert Berry, Michael
1008 Voznesensky, Shuguang Hu, Kristen Chiafullo, Sharat Chikkerur, George Scrivener, Ivy Zheng,
1009 Jeremy Wiesner, Wolfgang Macherey, Timothy Lillicrap, Fei Liu, Brian Walker, David Welling,
1010 Elinor Davies, Yangsibo Huang, Lijie Ren, Nir Shabat, Alessandro Agostini, Mariko Iinuma,
1011 Dustin Zelle, Rohit Sathyanarayana, Andrea D’olimpio, Morgan Redshaw, Matt Ginsberg, Ash-
1012 win Murthy, Mark Geller, Tatiana Matejovicova, Ayan Chakrabarti, Ryan Julian, Christine Chan,
1013 Qiong Hu, Daniel Jarrett, Manu Agarwal, Jeshwanth Challagundla, Tao Li, Sandeep Tata, Wen
1014 Ding, Maya Meng, Zhuyun Dai, Giulia Vezzani, Shefali Garg, Jannis Bulian, Mary Jasarevic,
1015 Honglong Cai, Harish Rajamani, Adam Santoro, Florian Hartmann, Chen Liang, Bartek Perz,
1016 Apoorv Jindal, Fan Bu, Sungyong Seo, Ryan Poplin, Adrian Goedeckemeyer, Badih Ghazi,
1017 Nikhil Khadke, Leon Liu, Kevin Mather, Mingda Zhang, Ali Shah, Alex Chen, Jinliang Wei, Ke-
1018 shav Shivam, Yuan Cao, Donghyun Cho, Angelo Scorza Scarpati, Michael Moffitt, Clara Barbu,
1019 Ivan Jurin, Ming-Wei Chang, Hongbin Liu, Hao Zheng, Shachi Dave, Christine Kaeser-Chen,
1020 Xiaobin Yu, Alvin Abdagic, Lucas Gonzalez, Yanping Huang, Peilin Zhong, Cordelia Schmid,
1021 Bryce Petrin, Alex Wertheim, Jifan Zhu, Hoang Nguyen, Kaiyang Ji, Yanqi Zhou, Tao Zhou,
1022 Fangxiaoyu Feng, Regev Cohen, David Rim, Shubham Milind Phal, Petko Georgiev, Ariel Brand,
1023 Yue Ma, Wei Li, Somit Gupta, Chao Wang, Pavel Dubov, Jean Tarbouriech, Kingshuk Majumder,
1024 Huijian Li, Norman Rink, Apurv Suman, Yang Guo, Yinghao Sun, Arun Nair, Xiaowei Xu, Mo-
1025 hamed Elhawaty, Rodrigo Cabrera, Guangxing Han, Julian Eisenschlos, Junwen Bai, Yuqi Li,
Yamini Bansal, Thibault Sellam, Mina Khan, Hung Nguyen, Justin Mao-Jones, Nikos Parot-
sidis, Jake Marcus, Cindy Fan, Roland Zimmermann, Yony Kochinski, Laura Graesser, Feryal
Behbahani, Alvaro Caceres, Michael Riley, Patrick Kane, Sandra Lefdal, Rob Willoughby, Paul
Vicol, Lun Wang, Shujian Zhang, Ashleah Gill, Yu Liang, Gautam Prasad, Soroosh Mariooryad,
Mehran Kazemi, Zifeng Wang, Kritika Muralidharan, Paul Voigtlaender, Jeffrey Zhao, Huanjie

- 1026 Zhou, Nina D’Souza, Aditi Mavalankar, Séb Arnold, Nick Young, Obaid Sarvana, Chace Lee,
1027 Milad Nasr, Tingting Zou, Seokhwan Kim, Lukas Haas, Kaushal Patel, Neslihan Bulut, David
1028 Parkinson, Courtney Biles, Dmitry Kalashnikov, Chi Ming To, Aviral Kumar, Jessica Austin, Alex
1029 Greve, Lei Zhang, Megha Goel, Yeqing Li, Sergey Yaroshenko, Max Chang, Abhishek Jindal, Ge-
1030 off Clark, Hagai Taitelbaum, Dale Johnson, Ofir Roval, Jeongwoo Ko, Anhad Mohananey, Chris-
1031 tian Schuler, Shenil Dodhia, Ruichao Li, Kazuki Osawa, Claire Cui, Peng Xu, Rushin Shah, Tao
1032 Huang, Ela Gruzewska, Nathan Clement, Mudrit Verma, Olcan Sercinoglu, Hai Qian, Viral Shah,
1033 Masa Yamaguchi, Abhinit Modi, Takahiro Kosakai, Thomas Strohmans, Junhao Zeng, Beliz
1034 Gunel, Jun Qian, Austin Tarango, Krzysztof Jastrzebski, Robert David, Jyn Shan, Parker Schuh,
1035 Kunal Lad, Willi Gierke, Mukundan Madhavan, Xinyi Chen, Mark Kurzeja, Rebeca Santamaria-
1036 Fernandez, Dawn Chen, Alexandra Cordell, Yuri Chervonyi, Frankie Garcia, Nithish Kannan,
1037 Vincent Perot, Nan Ding, Shlomi Cohen-Ganor, Victor Lavrenko, Junru Wu, Georgie Evans,
1038 Cicero Nogueira dos Santos, Madhavi Sewak, Ashley Brown, Andrew Hard, Joan Puigcerver,
1039 Zeyu Zheng, Yizhong Liang, Evgeny Gladchenko, Reeve Ingle, Uri First, Pierre Sermanet, Char-
1040 lotte Magister, Mihajlo Velimirović, Sashank Reddi, Susanna Ricco, Eirikur Agustsson, Hartwig
1041 Adam, Nir Levine, David Gaddy, Dan Holtmann-Rice, Xuanhui Wang, Ashutosh Sathe, Abhi-
1042 jit Guha Roy, Blaž Bratanič, Alen Carin, Harsh Mehta, Silvano Bonacina, Nicola De Cao, Mara
1043 Finkelstein, Verena Rieser, Xinyi Wu, Florent Althé, Dylan Scandinaro, Li Li, Nino Vieillard,
1044 Nikhil Sethi, Garrett Tanzer, Zhi Xing, Shibo Wang, Parul Bhatia, Gui Citovsky, Thomas An-
1045 thony, Sharon Lin, Tianze Shi, Shoshana Jakobovits, Gena Gibson, Raj Apte, Lisa Lee, Mingqing
1046 Chen, Arunkumar Byravan, Petros Maniatis, Kellie Webster, Andrew Dai, Pu-Chin Chen, Jiaqi
1047 Pan, Asya Fadeeva, Zach Gleicher, Thang Luong, and Niket Kumar Bhumihar. Gemini 2.5: Push-
1048 ing the frontier with advanced reasoning, multimodality, long context, and next generation agentic
capabilities, 2025. URL <https://arxiv.org/abs/2507.06261>.
- 1049 Cursor. Model context protocol (mcp). <https://docs.cursor.com/context/mcp>, 2025.
1050 Accessed: 2025-06-30.
- 1051 DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Cheng-
1052 gang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang,
1053 Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting
1054 Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui
1055 Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi
1056 Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li,
1057 Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang,
1058 Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun
1059 Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan
1060 Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J.
1061 Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang,
1062 Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng
1063 Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shut-
1064 ing Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanbiao Zhao,
1065 Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue
1066 Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xi-
1067 aokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin
1068 Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang,
1069 Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang
1070 Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui
1071 Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying
1072 Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu,
1073 Yuan Ou, Yuchen Zhu, Yudian Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan
1074 Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F.
1075 Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda
1076 Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao,
1077 Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li,
1078 Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. Deepseek-v3 technical report, 2025. URL
1079 <https://arxiv.org/abs/2412.19437>.
- Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun,
and Yu Su. Mind2web: Towards a generalist agent for the web. In Alice Oh, Tristan Nau-

- mann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/5950bf290a1570ea401bf98882128160-Abstract-Datasets_and_Benchmarks.html.
- Guanting Dong, Keming Lu, Chengpeng Li, Tingyu Xia, Bowen Yu, Chang Zhou, and Jingren Zhou. Self-play with execution feedback: Improving instruction-following capabilities of large language models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL <https://openreview.net/forum?id=cRR0oDFEBC>.
- Lisowski Edwin. Model context protocol (mcp): Solution to ai integration bottlenecks. <https://addepto.com/blog/model-context-protocol-mcp-solution-to-ai-integration-bottlenecks/>, May 2025. Accessed: 2025-06-30.
- Xuanqi Gao, Siyi Xie, Juan Zhai, Shqing Ma, and Chao Shen. MCP-RADAR: A multi-dimensional benchmark for evaluating tool use capabilities in large language models. *CoRR*, abs/2505.16700, 2025. doi: 10.48550/ARXIV.2505.16700. URL <https://doi.org/10.48550/arXiv.2505.16700>.
- Google. Gemini cli: your open-source ai agent. <https://blog.google/technology/developers/introducing-gemini-cli-open-source-ai-agent/>, 2025. Accessed: 2025-06-30.
- Boyu Gou, Zanming Huang, Yuting Ning, Yu Gu, Michael Lin, Weijian Qi, Andrei Kopanav, Botao Yu, Bernal Jiménez Gutiérrez, Yiheng Shu, Chan Hee Song, Jiaman Wu, Shijie Chen, Hanane Nour Moussa, Tianshu Zhang, Jian Xie, Yifei Li, Tianci Xue, Zeyi Liao, Kai Zhang, Boyuan Zheng, Zhaowei Cai, Viktor Rozgic, Morteza Ziyadi, Huan Sun, and Yu Su. Mind2web 2: Evaluating agentic search with agent-as-a-judge, 2025.
- Rem Hida, Junki Ohmura, and Toshiyuki Sekiya. Evaluation of instruction-following ability for large language models on story-ending generation. *CoRR*, abs/2406.16356, 2024. doi: 10.48550/ARXIV.2406.16356. URL <https://doi.org/10.48550/arXiv.2406.16356>.
- Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. Metagpt: Meta programming for A multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL <https://openreview.net/forum?id=VtmBAGCN7o>.
- Xueyu Hu, Tao Xiong, Biao Yi, Zishu Wei, Ruixuan Xiao, Yurun Chen, Jiasheng Ye, Meiling Tao, Xiangxin Zhou, Ziyu Zhao, et al. Os agents: A survey on mllm-based agents for computer, phone and browser use, 2024.
- Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrom, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Madry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, Alex Renzin, Alex Tachard Passos, Alexander Kirillov, Alexi Christakis, Alexis Conneau, Ali Kamali, Allan Jabri, Allison Moyer, Allison Tam, Amadou Crookes, Amin Tootoonchian, Ananya Kumar, Andrea Vallone, Andrej Karpathy, Andrew Braunstein, Andrew Cann, Andrew Codisoti, Andrew Galu, Andrew Kondrich, Andrew Tulloch, Andrey Mishchenko, Angela Baek, Angela Jiang, Antoine Pelisse, Antonia Woodford, Anuj Gosalia, Arka Dhar, Ashley Pantuliano, Avi Nayak, Avital Oliver, Barret Zoph, Behrooz Ghorbani, Ben Leimberger, Ben Rossen, Ben Sokolowsky, Ben Wang, Benjamin Zweig, Beth Hoover, Blake Samic, Bob McGrew, Bobby Spero, Bogo Gertler, Bowen Cheng, Brad Lightcap, Brandon Walkin, Brendan Quinn, Brian Guarraci, Brian Hsu, Bright Kellogg, Brydon Eastman, Camillo Lugaresi, Carroll L. Wainwright, Cary Bassin, Cary Hudson, Casey Chu, Chad Nelson, Chak Li, Chan Jun Shern, Channing Conger, Charlotte Barette, Chelsea Voss, Chen Ding, Cheng Lu, Chong Zhang, Chris Beaumont,

- 1134 Chris Hallacy, Chris Koch, Christian Gibson, Christina Kim, Christine Choi, Christine McLeavey,
1135 Christopher Hesse, Claudia Fischer, Clemens Winter, Coley Czarnecki, Colin Jarvis, Colin Wei,
1136 Constantin Koumouzelis, and Dane Sherburn. Gpt-4o system card. *CoRR*, abs/2410.21276, 2024.
1137 doi: 10.48550/ARXIV.2410.21276. URL [https://doi.org/10.48550/arXiv.2410.](https://doi.org/10.48550/arXiv.2410.21276)
1138 21276.
- 1139 Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R.
1140 Narasimhan. Swe-bench: Can language models resolve real-world github issues? In *The*
1141 *Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria,*
1142 *May 7-11, 2024*. OpenReview.net, 2024. URL [https://openreview.net/forum?id=](https://openreview.net/forum?id=VTF8yNQm66)
1143 [VTF8yNQm66](https://openreview.net/forum?id=VTF8yNQm66).
- 1144 Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham
1145 Neubig, Shuyan Zhou, Russ Salakhutdinov, and Daniel Fried. Visualwebarena: Evaluating mul-
1146 timodal agents on realistic visual web tasks. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar
1147 (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguis-*
1148 *tics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 881–905.
1149 Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.50. URL
1150 <https://doi.org/10.18653/v1/2024.acl-long.50>.
- 1151 LangChain. Build resilient language agents as graphs. [https://github.com/](https://github.com/langchain-ai/langgraph)
1152 [langchain-ai/langgraph](https://github.com/langchain-ai/langgraph), 2024. GitHub Repository, Accessed: 2025-06-30.
- 1153 Bowen Li, Wenhan Wu, Ziwei Tang, Lin Shi, John Yang, Jinyang Li, Shunyu Yao, Chen
1154 Qian, Binyuan Hui, Qicheng Zhang, Zhiyin Yu, He Du, Ping Yang, Dahua Lin, Chao Peng,
1155 and Kai Chen. Devbench: A comprehensive benchmark for software development. *CoRR*,
1156 abs/2403.08604, 2024. doi: 10.48550/ARXIV.2403.08604. URL [https://doi.org/10.](https://doi.org/10.48550/arXiv.2403.08604)
1157 [48550/arXiv.2403.08604](https://doi.org/10.48550/arXiv.2403.08604).
- 1158 Dawei Li, Renliang Sun, Yue Huang, Ming Zhong, Bohan Jiang, Jiawei Han, Xiangliang Zhang, Wei
1159 Wang, and Huan Liu. Preference leakage: A contamination problem in llm-as-a-judge, 2025a.
1160 URL <https://arxiv.org/abs/2502.01534>.
- 1161 Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem.
1162 Camel: Communicative agents for ”mind” exploration of large language model society. In *Thirty-*
1163 *seventh Conference on Neural Information Processing Systems, 2023a*.
- 1164 Kaixin Li, Ziyang Meng, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang,
1165 and Tat-Seng Chua. Screenspot-pro: GUI grounding for professional high-resolution computer
1166 use. *CoRR*, abs/2504.07981, 2025b. doi: 10.48550/ARXIV.2504.07981. URL [https://doi.](https://doi.org/10.48550/arXiv.2504.07981)
1167 [org/10.48550/arXiv.2504.07981](https://doi.org/10.48550/arXiv.2504.07981).
- 1170 Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li,
1171 Fei Huang, and Yongbin Li. Api-bank: A comprehensive benchmark for tool-augmented
1172 llms. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Confer-*
1173 *ence on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, De-*
1174 *cember 6-10, 2023*, pp. 3102–3116. Association for Computational Linguistics, 2023b. doi:
1175 10.18653/V1/2023.EMNLP-MAIN.187. URL [https://doi.org/10.18653/v1/2023.](https://doi.org/10.18653/v1/2023.emnlp-main.187)
1176 [emnlp-main.187](https://doi.org/10.18653/v1/2023.emnlp-main.187).
- 1177 Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. Reinforcement
1178 learning on web interfaces using workflow-guided exploration. In *International Conference on*
1179 *Learning Representations (ICLR)*, 2018. URL <https://arxiv.org/abs/1802.08802>.
- 1180 Zhiwei Liu, Jieli Qiu, Shiyu Wang, Jianguo Zhang, Zuxin Liu, Roshan Ram, Haolin Chen, Weiran
1181 Yao, Shelby Heinecke, Silvio Savarese, Huan Wang, and Caiming Xiong. Mcpeval: Automatic
1182 mcp-based deep evaluation for ai agent models, 2025. URL [https://arxiv.org/abs/](https://arxiv.org/abs/2507.12806)
1183 [2507.12806](https://arxiv.org/abs/2507.12806).
- 1184 Xing Han Lù, Zdenek Kasner, and Siva Reddy. Weblinx: Real-world website navigation with multi-
1185 turn dialogue. In *Forty-first International Conference on Machine Learning, ICML 2024, Vi-*
1186 *enna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL [https://openreview.net/](https://openreview.net/forum?id=mUSPhG4uDW)
1187 [forum?id=mUSPhG4uDW](https://openreview.net/forum?id=mUSPhG4uDW).

- 1188 Grégoire Mialon, Clémentine Fourier, Thomas Wolf, Yann LeCun, and Thomas Scialom. GAIA:
1189 a benchmark for general AI assistants. In *The Twelfth International Conference on Learning*
1190 *Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL
1191 <https://openreview.net/forum?id=fibxvahvs3>.
- 1192 Guozhao Mo, Wenliang Zhong, Jiawei Chen, Xuanang Chen, Yaojie Lu, Hongyu Lin, Ben He,
1193 Xianpei Han, and Le Sun. Livemcpbench: Can agents navigate an ocean of mcp tools?, 2025.
1194 URL <https://arxiv.org/abs/2508.01780>.
- 1195 Moonshot. Kimi k2: Open agentic intelligence. [https://moonshotai.github.io/](https://moonshotai.github.io/Kimi-K2/)
1196 [Kimi-K2/](https://moonshotai.github.io/Kimi-K2/), July 2025. Accessed: 2025-07-28.
- 1197 Shravan Nayak, Xiangru Jian, Kevin Qinghong Lin, Juan A. Rodriguez, Montek Kalsi, Rabiul Awal,
1198 Nicolas Chapados, M. Tamer Özsu, Aishwarya Agrawal, David Vázquez, Christopher Pal, Perouz
1199 Taslakian, Spandana Gella, and Sai Rajeswar. Ui-vision: A desktop-centric GUI benchmark for
1200 visual perception and interaction. *CoRR*, abs/2503.15661, 2025. doi: 10.48550/ARXIV.2503.
1201 15661. URL <https://doi.org/10.48550/arXiv.2503.15661>.
- 1202 OpenAI. Introducing gpt-oss. <https://openai.com/index/introducing-gpt-oss/>,
1203 August 2025a. Accessed: 2025-08-14.
- 1204 OpenAI. Introducing gpt-4.1 in the api. <https://openai.com/index/gpt-4-1/>, April
1205 2025b. Accessed: 2025-07-28.
- 1206 OpenAI. Building mcp servers for deep research. [https://platform.openai.com/docs/](https://platform.openai.com/docs/mcp/)
1207 [mcp/](https://platform.openai.com/docs/mcp/), 2025a. Accessed: 2025-06-30.
- 1208 OpenAI. Computer-using agent: Introducing a universal interface for ai to interact with the digital
1209 world. 2025b. URL <https://openai.com/index/computer-using-agent>.
- 1210 OpenAI. Introducing gpt-5. <https://openai.com/index/introducing-gpt-5/>, Au-
1211 gust 2025a. Accessed: 2025-08-14.
- 1212 OpenAI. Introducing openai o3 and o4-mini. [https://openai.com/index/](https://openai.com/index/introducing-o3-and-o4-mini/)
1213 [introducing-o3-and-o4-mini/](https://openai.com/index/introducing-o3-and-o4-mini/), April 2025b. Accessed: 2025-07-28.
- 1214 Shishir G. Patil, Huanzhi Mao, Charlie Cheng-Jie Ji, Fanjia Yan, Vishnu Suresh, Ion Stoica, and
1215 Joseph E. Gonzalez. The berkeley function calling leaderboard (bfcl): From tool use to agen-
1216 tic evaluation of large language models. In *Forty-second International Conference on Machine*
1217 *Learning*, 2025.
- 1218 Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi,
1219 Nathan Lambert, and Hannaneh Hajishirzi. Generalizing verifiable instruction following, 2025.
1220 URL <https://arxiv.org/abs/2507.02833>.
- 1221 Yusu Qian, Hanrong Ye, Jean-Philippe Fauconnier, Peter Grasch, Yinfei Yang, and Zhe Gan. Mia-
1222 bench: Towards better instruction following evaluation of multimodal llms. In *The Thirteenth*
1223 *International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*.
1224 OpenReview.net, 2025. URL <https://openreview.net/forum?id=7EhS3YBxjY>.
- 1225 Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xi-
1226 angru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark
1227 Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language mod-
1228 els to master 16000+ real-world apis. In *The Twelfth International Conference on Learning*
1229 *Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024a. URL
1230 <https://openreview.net/forum?id=dHng200Jjr>.
- 1231 Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xi-
1232 angru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark
1233 Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language mod-
1234 els to master 16000+ real-world apis. In *The Twelfth International Conference on Learning*
1235 *Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024b. URL
1236 <https://openreview.net/forum?id=dHng200Jjr>.

- 1242 Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jia-
1243 hao Li, Yunxin Li, Shijue Huang, Wanjun Zhong, Kuanye Li, Jiale Yang, Yu Miao, Woyu Lin,
1244 Longxiang Liu, Xu Jiang, Qianli Ma, Jingyu Li, Xiaojun Xiao, Kai Cai, Chuang Li, Yaowei
1245 Zheng, Chaolin Jin, Chen Li, Xiao Zhou, Minchao Wang, Haoli Chen, Zhaojian Li, Haihua Yang,
1246 Haifeng Liu, Feng Lin, Tao Peng, Xin Liu, and Guang Shi. UI-TARS: pioneering automated GUI
1247 interaction with native agents. *CoRR*, abs/2501.12326, 2025. doi: 10.48550/ARXIV.2501.12326.
1248 URL <https://doi.org/10.48550/arXiv.2501.12326>.
- 1249 Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and
1250 Ji-Rong Wen. Tool learning with large language models: a survey. *Frontiers Comput. Sci.*, 19
1251 (8):198343, 2025. doi: 10.1007/S11704-024-40678-2. URL [https://doi.org/10.1007/
1252 s11704-024-40678-2](https://doi.org/10.1007/s11704-024-40678-2).
- 1253 Hightower Rick. Mcp the usb-c for ai. [https://medium.com/@richardhightower/
1254 how-the-model-context-protocol-is-revolutionizing-ai-integration-48926ce5d823](https://medium.com/@richardhightower/how-the-model-context-protocol-is-revolutionizing-ai-integration-48926ce5d823),
1255 April 2025. Accessed: 2025-06-30.
- 1256 Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Re-
1257 flexion: language agents with verbal reinforcement learning. In Alice Oh, Tristan Nau-
1258 mann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances
1259 in Neural Information Processing Systems 36: Annual Conference on Neural Informa-
1260 tion Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16,
1261 2023*, 2023. URL [http://papers.nips.cc/paper_files/paper/2023/hash/
1262 1b44b878bb782e6954cd888628510e90-Abstract-Conference.html](http://papers.nips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html).
- 1263 Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank
1264 Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. Appworld: A controllable world of
1265 apps and people for benchmarking interactive coding agents. In Lun-Wei Ku, Andre Martins, and
1266 Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computa-
1267 tional Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*,
1268 pp. 16022–16076. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.
1269 ACL-LONG.850. URL <https://doi.org/10.18653/v1/2024.acl-long.850>.
- 1270 Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim.
1271 Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language
1272 models. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the
1273 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),
1274 ACL 2023, Toronto, Canada, July 9-14, 2023*, pp. 2609–2634. Association for Computational
1275 Linguistics, 2023. doi: 10.18653/V1/2023.ACL-LONG.147. URL [https://doi.org/10.
1276 18653/v1/2023.acl-long.147](https://doi.org/10.18653/v1/2023.acl-long.147).
- 1277 Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen,
1278 Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Jirong Wen. A
1279 survey on large language model based autonomous agents. *Frontiers Comput. Sci.*, 18(6):
1280 186345, 2024a. doi: 10.1007/S11704-024-40231-1. URL [https://doi.org/10.1007/
1281 s11704-024-40231-1](https://doi.org/10.1007/s11704-024-40231-1).
- 1282 Lu Wang, Fangkai Yang, Chaoyun Zhang, Junting Lu, Jiaxu Qian, Shilin He, Pu Zhao, Bo Qiao,
1283 Ray Huang, Si Qin, Qisheng Su, Jiayi Ye, Yudi Zhang, Jian-Guang Lou, Qingwei Lin, Sar-
1284 avan Rajmohan, Dongmei Zhang, and Qi Zhang. Large action models: From inception to
1285 implementation. *CoRR*, abs/2412.10047, 2024b. doi: 10.48550/ARXIV.2412.10047. URL
1286 <https://doi.org/10.48550/arXiv.2412.10047>.
- 1287 Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi,
1288 Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language
1289 models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh
1290 (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural
1291 Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - De-
1292 cember 9, 2022*, 2022. URL [http://papers.nips.cc/paper_files/paper/2022/
1293 hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html](http://papers.nips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html).

- 1296 Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li,
1297 Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen LLM applications via
1298 multi-agent conversation framework. *CoRR*, abs/2308.08155, 2023. doi: 10.48550/ARXIV.2308.
1299 08155. URL <https://doi.org/10.48550/arXiv.2308.08155>.
- 1300
1301 xAI. Grok 4. <https://x.ai/news/grok-4>, July 2025. Accessed: 2025-07-28.
- 1302
1303 Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing
1304 Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio
1305 Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. Osworld: Benchmarking multimodal
1306 agents for open-ended tasks in real computer environments. In Amir Globersons, Lester
1307 Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang
1308 (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural
1309 Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10
1310 - 15, 2024*, 2024. URL [http://papers.nips.cc/paper_files/paper/2024/
1311 hash/5d413e48f84dc61244b6be550f1cd8f5-Abstract-Datasets_and_
1312 Benchmarks_Track.html](http://papers.nips.cc/paper_files/paper/2024/hash/5d413e48f84dc61244b6be550f1cd8f5-Abstract-Datasets_and_Benchmarks_Track.html).
- 1313
1314 Fengli Xu, Qian Yue Hao, Zefang Zong, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan,
1315 Jiahui Gong, Tianjian Ouyang, Fanjin Meng, Chenyang Shao, Yuwei Yan, Qinglong Yang, Yiwen
1316 Song, Sijian Ren, Xinyuan Hu, Yu Li, Jie Feng, Chen Gao, and Yong Li. Towards large reasoning
1317 models: A survey of reinforced reasoning with large language models. *CoRR*, abs/2501.09686,
1318 2025. doi: 10.48550/ARXIV.2501.09686. URL [https://doi.org/10.48550/arXiv.
1319 2501.09686](https://doi.org/10.48550/arXiv.2501.09686).
- 1320
1321 Yunhe Yan, Shihe Wang, Jiajun Du, Yexuan Yang, Yuxuan Shan, Qichen Qiu, Xianqing Jia, Xinge
1322 Wang, Xin Yuan, Xu Han, Mao Qin, Yinxiao Chen, Chen Peng, Shangguang Wang, and Mengwei
1323 Xu. Mcpworld: A unified benchmarking testbed for api, gui, and hybrid computer use agents.
1324 *CoRR*, abs/2506.07672, 2025. doi: 10.48550/ARXIV.2506.07672. URL [https://doi.org/
1325 10.48550/arXiv.2506.07672](https://doi.org/10.48550/arXiv.2506.07672).
- 1326
1327 An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang
1328 Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu,
1329 Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin
1330 Yang, Jiayi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang,
1331 Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui
1332 Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang
1333 Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger
1334 Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan
1335 Qiu. Qwen3 technical report, 2025a. URL <https://arxiv.org/abs/2505.09388>.
- 1336
1337 Yan Yang, Dongxu Li, Yutong Dai, Yuhao Yang, Ziyang Luo, Zirui Zhao, Zhiyuan Hu, Junzhe
1338 Huang, Amrita Saha, Zeyuan Chen, Ran Xu, Liyuan Pan, Caiming Xiong, and Junnan Li. GTA1:
1339 GUI test-time scaling agent. *CoRR*, abs/2507.05791, 2025b. doi: 10.48550/ARXIV.2507.05791.
1340 URL <https://doi.org/10.48550/arXiv.2507.05791>.
- 1341
1342 Yuhao Yang, Yue Wang, Dongxu Li, Ziyang Luo, Bei Chen, Chao Huang, and Junnan Li. Aria-
1343 ui: Visual grounding for GUI instructions. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova,
1344 and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics,
1345 ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 22418–22433. Association for Computa-
1346 tional Linguistics, 2025c. URL [https://aclanthology.org/2025.findings-acl.
1347 1152/](https://aclanthology.org/2025.findings-acl.1152/).
- 1348
1349 Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik
1350 Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In
1351 Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine
1352 (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural
1353 Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December
1354 10 - 16, 2023*, 2023a. URL [http://papers.nips.cc/paper_files/paper/2023/
1355 hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract-Conference.html](http://papers.nips.cc/paper_files/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract-Conference.html).

- 1350 Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan
1351 Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International
1352 Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenRe-
1353 view.net, 2023b. URL https://openreview.net/forum?id=WE_vluYUL-X.
- 1354 Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ -bench: A benchmark for
1355 tool-agent-user interaction in real-world domains. *CoRR*, abs/2406.12045, 2024. doi: 10.48550/
1356 ARXIV.2406.12045. URL <https://doi.org/10.48550/arXiv.2406.12045>.
- 1357 Ori Yoran, Samuel Joseph Amouyal, Chaitanya Malaviya, Ben Bogin, Ofir Press, and Jonathan
1358 Berant. Assistantbench: Can web agents solve realistic and time-consuming tasks? In
1359 Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Confer-
1360 ence on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA,
1361 November 12-16, 2024*, pp. 8938–8968. Association for Computational Linguistics, 2024. doi:
1362 10.18653/V1/2024.EMNLP-MAIN.505. URL [https://doi.org/10.18653/v1/2024.
1363 emnlp-main.505](https://doi.org/10.18653/v1/2024.emnlp-main.505).
- 1364 Zai. Glm-4.5: Reasoning, coding, and agentic abilities. <https://z.ai/blog/glm-4.5>, July
1365 2025. Accessed: 2025-07-28.
- 1366 Jianguo Zhang, Tian Lan, Ming Zhu, Zuxin Liu, Thai Hoang, Shirley Kokane, Weiran Yao, Juntao
1367 Tan, Akshara Prabhakar, Haolin Chen, Zhiwei Liu, Yihao Feng, Tulika Manoj Awalganekar,
1368 Rithesh R. N., Zeyuan Chen, Ran Xu, Juan Carlos Niebles, Shelby Heinecke, Huan Wang, Silvio
1369 Savarese, and Caiming Xiong. xlam: A family of large action models to empower AI agent
1370 systems. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference
1371 of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human
1372 Language Technologies, NAACL 2025 - Volume 1: Long Papers, Albuquerque, New Mexico, USA,
1373 April 29 - May 4, 2025*, pp. 11583–11597. Association for Computational Linguistics, 2025a. doi:
1374 10.18653/V1/2025.NAACL-LONG.578. URL [https://doi.org/10.18653/v1/2025.
1375 naacl-long.578](https://doi.org/10.18653/v1/2025.naacl-long.578).
- 1376 Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Zhihan Guo, Yufei Wang, Irwin
1377 King, Xue Liu, and Chen Ma. What, how, where, and how well? A survey on test-time scaling in
1378 large language models. *CoRR*, abs/2503.24235, 2025b. doi: 10.48550/ARXIV.2503.24235. URL
1379 <https://doi.org/10.48550/arXiv.2503.24235>.
- 1380 Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao
1381 Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez,
1382 and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In Alice Oh,
1383 Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.),
1384 *Advances in Neural Information Processing Systems 36: Annual Conference on Neural In-
1385 formation Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10
1386 - 16, 2023*, 2023. URL [http://papers.nips.cc/paper_files/paper/2023/
1387 hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_
1388 Benchmarks.html](http://papers.nips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html).
- 1389 Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng,
1390 Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic
1391 web environment for building autonomous agents. In *The Twelfth International Conference on
1392 Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.
1393 URL <https://openreview.net/forum?id=oKn9c6ytLx>.
- 1394
1395
1396
1397
1398
1399
1400
1401
1402
1403

Table 5: Evaluation with LLMs using Function-Calling.

Model	Location Navigation	Repository Management	Financial Analysis	3D Designing	Browser Automation	Web Searching	AE	Overall AS	SR
<i>Proprietary Models</i>									
🌀 GPT-5-Medium	35.56	30.30	60.00	52.63	43.59	36.36	59.29	7.85	41.99
🇺🇸 Claude-4.0-Sonnet	22.22	6.06	77.50	36.84	35.90	21.82	52.61	9.78	32.90
🌀 Grok-4-Fast	22.22	6.06	80.00	21.05	23.08	32.73	52.20	7.25	32.47
🌀 Grok-Code-Fast-1	17.78	12.12	57.50	26.32	15.38	20.00	43.89	7.61	24.68
🌀 GPT-4.1	15.56	6.06	45.00	26.32	28.21	5.45	39.58	6.83	19.91
🌀 GPT-4o	8.89	6.06	35.00	10.53	12.82	7.27	32.43	6.65	13.42
🌀 Qwen3-Max-Preview	13.33	15.15	40.00	21.05	15.38	5.45	36.71	8.57	17.32
<i>Open-Source Models</i>									
🌀 GPT-OSS-120B	24.44	15.15	42.50	36.84	20.51	20.00	39.78	7.53	25.54
🌀 GLM-4.5	22.22	9.09	37.50	26.32	10.26	16.36	37.58	10.17	19.91
🌀 DeepSeek-V3.1	11.11	6.06	45.00	26.32	23.08	12.73	38.88	10.74	19.91
🇺🇸 Kimi-K2	17.78	6.06	42.50	15.79	23.08	10.91	38.52	10.20	19.48
🌀 GLM-4.5-Air	8.89	3.03	25.00	10.53	7.69	14.55	26.47	12.21	12.12

Table 6: MCP servers in our benchmark and their links.

MCP Server	URL
Google Map MCP	https://github.com/modelcontextprotocol/servers-archived/tree/main/src/google-maps
Github MCP	https://github.com/github/github-mcp-server
Yahoo Finance MCP	We will release this server upon acceptance.
Blender MCP	We will release this server upon acceptance.
Playwright MCP	https://github.com/microsoft/playwright-mcp
Google Search MCP	We will release this server upon acceptance.
Fetch MCP	https://github.com/modelcontextprotocol/servers/tree/main/src/fetch
Notion MCP	https://github.com/makenotion/notion-mcp-server
Weather MCP	We will release this server upon acceptance.
Date MCP	We will release this server upon acceptance.
Calculator MCP	https://pypi.org/project/mcp-server-calculator/

A MCP SERVERS

As shown in Table 6, we include the names and links of all MCP servers in our benchmark to help users more easily utilize the benchmark. Most of them are official MCP servers, and some are based on the official APIs to ensure the quality of the servers.

B TASKS AND EVALUATORS EXAMPLES

In Table 7, 8, 9, 10, 11, and 12, we include 30 task examples of our benchmark. In Table 13, 14, and 15, we include 3 examples of the evaluators of our benchmark. All tasks and evaluators will be released upon acceptance.

C SETUP

As shown in Table 16, we present the versions of the LLMs used in our evaluation. The temperature is set to 1.0 for all LLMs. In Figure 11, we present the ReAct prompt used in our experiments. In Figure 12, we present the function-call prompt used in our experiments.

D LLMs w/ FUNCTION-CALL PERFORMANCE

As shown in Table 5, we show the details performance of LLMs w/ Function-Call on our MCP-Universe.

Table 7: Examples of Location Navigation Tasks. We do not include the format requirements to save space.

Example 1: Hi! My partner and I are planning a special pre-wedding road trip from Los Angeles to San Francisco as one last adventure before we tie the knot! We want to make this journey memorable before we start our married life together. Our plan is to drive through exactly 3 interesting cities between the starting and ending points to really enjoy this time together. Could you please map out exactly 2 distinct driving route options for this pre-wedding celebration? Oh, we must visit friends in Coalinga during our trip to share our exciting news with them! We’re so excited about this adventure before our big day!

Example 2: I need to drive from the Merlion Park, Singapore to the Petronas Towers, Kuala Lumpur, Malaysia. Please plan a driving route. Along this route, I need to make exactly one stop. Find the single location (report its name and Place ID) that is closest to the geographic midpoint of the calculated route (based on the route polyline) and is categorized as either a gas station OR a restaurant with a user rating of at least 4.2.

Example 3: My wife and I are planning an amazing family adventure from Disneyland in Anaheim to Yosemite Valley Visitor Center with our wonderful kids! As a devoted husband and father, I want to make sure everyone stays happy and comfortable during our journey, so I need help creating a perfect driving route with four thoughtfully chosen family-friendly stops. Could you please map out a route with exactly four intermediate points that are located at the geographic fifth points along the route (based on the route polyline)? For each stop, I’d love to find locations (please provide names and Place IDs) that are either a restaurant where we can all enjoy a meal together, a comfortable hotel where my family can rest, or a reliable gas station to keep our adventure going. All with a minimum user rating of 4.2 to ensure the best experience for my loved ones. This trip should be both practical and create wonderful memories for our entire family!

Example 4: I live in Kent Ridge Hill Residences, Singapore. One of my friends lives in Symphony Suites, Yishun, Singapore. Another friend lives in Katong Gardens, Singapore. We’re looking for a cozy spot to catch up and chat! Can you help us find a meeting point between our 3 locations? We’d love to find a single cafe (must be of type ‘cafe’) where we can all gather comfortably, ideally somewhere where the estimated driving time from each of our places to the cafe is as close as possible. Please report the Name and Place ID of the cafe.

Example 5: Identify 1 library location in New York City that are north of the latitude of Queensbridge Park AND east of the longitude of NewYork-Presbyterian/Weill Cornell Medical Center.

Table 8: Examples of Repository Management Tasks.

Example 1: For this assignment, I would like you to establish a new project repository named `ai-code-reviewer`. Please begin by initializing the repository with three branches: `feature-analysis`, `feature-integration`, and `main`. You should include an initial `README.md` file in the main branch with the content “# AI Code Reviewer\n\nAn intelligent code review assistant that analyzes code quality, detects potential bugs, and suggests improvements using machine learning techniques.”. Next, please add `code_analyzer.py` in the `feature-analysis` branch with the content “# Code analysis module\nimport ast\n\nclass CodeAnalyzer:\n def _init_(self, code):\n self.code = code\n self.tree = ast.parse(code)\n\n def analyze(self):\n # TODO: Implement analysis logic\n pass”. Additionally, create a `.gitignore` file in the main branch with the exact content: “# Python cache and virtual environments\n__pycache__\n*.pyc\n*.py.class\nvenv\n*.env\n\n# Analysis results\nreports\nlogs\n\n# Model checkpoints\nmodels/”. Please copy `train.py` from bigcode-project’s star-coder repository to the `feature-integration` branch. Finally, I would like you to create a pull request to merge `feature-analysis` into `main` with the title “Add initial code analysis module” and description “This PR implements the basic code analysis module using AST parsing for initial code quality assessment.”

Example 2: Hi! I’m learning how to use GitHub and I want to practice exploring repositories and working with issues. Can you help me with a research project? I’d like to search for repositories owned by ‘google’ that have ‘generative-ai’ in their name. Once I find them, I want to count how many open issues each repository has that are labeled ‘type:bug’. This will help me understand how developers track bugs in real projects! After gathering this information, I need to practice creating my own repository called ‘google-generative-ai-issues’ and uploading a CSV file named ‘google_generative_ai_bug_report.csv’ to it. The CSV should have two columns: ‘repository_name’ and ‘open_bug_count’. This exercise will help me learn about repository management, issue tracking, and data organization on GitHub!

Example 3: There are two repositories: QwenLM’s Qwen2.5-VL and deepseek-ai’s DeepSeek-VL2. Fork the repository with the fewest open issues, maintaining the same name as the source repository. If Qwen2.5-VL is forked, add a reference link at the bottom of the `README.md` file: ‘Related project: [DeepSeek-VL2](the link of DeepSeek-VL2 repo)’. If DeepSeek-VL2 is forked, add a reference link at the bottom of the `README.md` file: ‘Related project: [Qwen2.5-VL](the link of Qwen2.5-VL repo)’.

Example 4: Hi! I’m a student working on learning GitHub automation and I really need your help. Could you please help me create a new project repository named `auto-issue-close`? I need to initialize it with just the main branch and include an initial `README.md` file with the content “# Automated Issue Closing\n\nA repository to test GitHub automation for closing labeled issues.” I’m struggling with GitHub automation workflows and would really appreciate your help developing a script that automatically closes issues labeled as ‘completed’ or ‘wontfix’. After we set up the automation script, I need to test it by creating three sample issues with different labels (labels: ‘completed’, title: “Implement new feature”; labels: ‘wontfix’, title: “Remove legacy code”; labels: ‘bug’, title: “Fix login error”). I’m really grateful for any assistance you can provide!

Example 5: Hi! I need help with a research project. Could you please search for repositories owned by ‘huggingface’ with ‘diffusers’ in the name? For each repository you find, I’d like to know how many open issues are labeled with ‘bug’. Then, could you help me create a CSV file called `diffusers_bug_report.csv` and put it in a new repository called `huggingface-diffusers-issues` under my account? If the repository doesn’t exist yet, please create it for me. The CSV should have two columns: `repository_name` and `open_bug_count`, with each row showing the full repository name and how many open bug issues it has. Thanks so much for your help!

1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565

Table 9: Examples of Financial Analysis Tasks.

Example 1: Hey! I’m super curious about investments and would love your help! Could you please calculate the final value and total percentage return for me if I had invested \$25,000 in Microsoft (MSFT) on January 9, 2023, and held it all the way until market close on January 8, 2025? I’m so excited to see how it would have performed!

Example 2: I require a comprehensive financial analysis for investment evaluation purposes. Please obtain the most recent annual income statements for Pfizer Inc. (PFE) and Johnson & Johnson (JNJ). Conduct a comparative analysis of their gross profit margins, calculated as Gross Profit divided by Total Revenue for the respective fiscal year. I need you to determine which pharmaceutical company demonstrates superior profitability efficiency and provide the precise calculated percentage figures for both entities for our portfolio assessment.

Example 3: Hello! I’m learning about investing and would love to understand how institutional investors like Blackrock Inc. move their holdings around. Could you help me get the latest institutional holdings data for Microsoft (MSFT), Apple (AAPL), and Alphabet (GOOGL)? I’m particularly curious about the percentage point changes (pctChange) in Blackrock Inc.’s stake for each of these companies. I’d like to see which company had the biggest positive increase in Blackrock’s holdings and know both the company ticker and the exact pctChange value. This would really help me understand how major investors adjust their portfolios!

Example 4: I absolutely love Pepsi and everything about it! As a devoted Pepsi enthusiast, I find it fascinating how Warren Buffett’s Berkshire Hathaway still holds that massive position in The Coca Cola Company (KO) despite Pepsi being clearly superior. Could you help me analyze their latest institutional holdings report for Berkshire Hathaway, Inc? I need you to extract their reported Shares, reported Value, and Date Reported. Then please convert that Date Reported timestamp into an actual calendar date and pull KO’s closing stock price for that specific trading day. I want to calculate what Berkshire’s position should actually be worth using that historical closing price and see how it compares to their originally reported value. This kind of analysis really excites me as a Pepsi lover studying these market dynamics! Please provide the Date Reported, the originally reported Value from the service, your calculated market value, and the absolute difference between these two figures.

Example 5: Hi there! I’m completely new to investing and finance, and honestly, I’m feeling pretty overwhelmed by all the jargon and concepts. I’ve been trying to learn about something called ‘fundamental analysis’. I think it has to do with looking at company finances? Anyway, I heard somewhere that you should look for companies where their net income (I think that’s like profit?) has been going up for a few quarters in a row. I’m not really sure what that means exactly, but apparently 2 consecutive quarters of rising net income is a good sign? I’m still figuring out what makes a company worth investing in. Could you help a total beginner like me find 3 company tickers that have this pattern? I’m trying to learn by doing some basic research, even though I barely understand what I’m looking for. Any help would be amazing! I’m just trying to get my feet wet in this whole investing world!

Table 10: Examples of 3D Designing Tasks.

Example 1: Create a Plane named ‘Floor’ scaled uniformly by 5. Create a Cylinder named ‘Pillar’ (default caps) with 16 vertices (sides), a radius of 0.5, and a depth of 4; position it at (X=-2, Y=-2, Z=2). Create a UV Sphere named ‘Ball’ with 32 segments and 16 rings; position it at (X=2, Y=2, Z=5). Create an Empty (Arrows type) named ‘ControlTarget’ at (X=0, Y=0, Z=3). Add a ‘Track To’ constraint to the ‘Ball’ object, making it track the ‘ControlTarget’. Finally, create a Camera object, position it at (X=0, Y=-8, Z=3), and set its rotation so it looks directly at the ‘Pillar’ object’s origin.

Example 2: Create a Cube named ‘RustedCube’, position it at the world origin (0,0,0), and set its scale factors to (X=5.0, Y=5.0, Z=0.2). Next, using the integrated Polyhaven add-on interface within Blender, search for ‘metal’ textures that include ‘rust’ in their description. Select the suitable asset found and download its 2K resolution. Import this asset directly onto the selected ‘RustedCube’. Ensure the material applied to ‘RustedCube’ is named ‘RustedMetalMat’ (renaming the auto-generated material if needed). Within the Shader Editor for the ‘RustedMetalMat’ material, verify or establish the following node setup: the downloaded Base Color texture must be connected to the ‘Base Color’ input of the Principled BSDF shader; the downloaded Roughness map (loaded into an Image Texture node set to ‘Non-Color’ space) must be connected to the ‘Roughness’ input; and the downloaded Normal map (also loaded via an Image Texture node set to ‘Non-Color’) must feed into the ‘Color’ input of a ‘Normal Map’ node (suitable for OpenGL), with the output of the ‘Normal Map’ node connected to the ‘Normal’ input of the Principled BSDF. Finally, adjust the ‘Metallic’ property on the Principled BSDF node to a value of 1.0.

Example 3: Set the render engine to Cycles and ensure the render device is CPU. In the Sampling settings, enable Denoising using OpenImageDenoise for both viewport and final render. Set the Render Samples to 512 and Viewport Samples to 128. Change the output resolution to 1350x1080 with a scale of 85%. In the Color Management panel, set the View Transform to Filmic, the Look to High Contrast, and adjust Gamma to 1.2. In the Render Layers Properties, enable Z Pass, Mist, and Normal passes. Go to World Settings and set the background color to a solid light gray using RGB (0.8, 0.8, 0.8). In the Output Properties, set the file format to OpenEXR MultiLayer, enable Zlib compression, and set output color depth to 32-bit float.

Example 4: Create a default Cube named ‘BaseShape’ at the origin. Add a ‘Subdivision Surface’ modifier to ‘BaseShape’ with Viewport and Render levels set to 3. Add a ‘Bevel’ modifier after the Subdivision Surface, set its Width to 0.07 meters, Segments to 3, and Limit Method to ‘Angle’. Create a UV Sphere named ‘Attachment’, scale it down uniformly to 0.3. Select ‘BaseShape’, enter Edit Mode, select the single vertex closest to world coordinates (X=1, Y=1, Z=1). Return to Object Mode. Parent ‘Attachment’ to ‘BaseShape’ using the ‘Vertex’ parenting type (ensure the previously selected vertex is used).

Example 5: Create three objects: a Cube named ‘Obj_A’, a UV Sphere named ‘Obj_B’, and a Cone named ‘Obj_C’, all at the world origin initially. Create two new Collections in the scene named ‘Group_Red’ and ‘Group_Blue’. Move ‘Obj_A’ and ‘Obj_C’ into the ‘Group_Red’ collection. Move ‘Obj_B’ into the ‘Group_Blue’ collection. Ensure these three objects are not also present in the default ‘Collection’ (Scene Collection). Add a Custom Property to the ‘Obj_B’ (Sphere) object: set the Property Name to ‘AssetID’, its Value to the integer 12345, and its Tooltip to ‘Sphere Asset Identifier’.

1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619

Table 11: Examples of Browser Automation Tasks.

Example 1: Help me find a one-way flight from Singapore to Beijing, 5 days from now (If now is 2025-07-07, then 5 days later is 2025-07-12). Find the flight on www.booking.com. I want to find the cheapest flight, direct flight, Economy, and I don't want to go to the Daxing airport. I only want to see the price, so as to determine whether I should fly to Beijing or not. Remember to close the browser after you finish the task.

Example 2: I will travel to Singapore 3 days from now (If today is 2025-06-07, then 3 days later will be 2025-06-10). I want to go to Universal Studios and Cove Waterpark. Could you tell me the total price for two adult tickets and one child ticket on the official website of Sentosa (<https://www.rwsentosa.com/>)? I only want to see the price. Remember to close the browser after you finish the task.

Example 3: Hey there! As a dad who wants to create the most amazing adventure for my little ones, I'm planning the ultimate family road trip from Disneyland Paris to the 24 Hours of Le Mans Museum with my precious kids. You know how it is - we want to make sure everyone stays happy, fed, and comfortable during our journey! I need your fantastic help creating a driving route with 1 perfectly planned stop, using that incredible website '<https://www.google.com/maps>'. Could you please map out a route with exactly 1 intermediate point that's located right at the geographic mid point along our route (based on the route polyline)? For this stop, I need you to find locations (with names and Place IDs) that are either family-friendly restaurants, cozy hotels, or reliable gas stations - all with a minimum user rating of 4.2 because only the best will do for my family! This trip needs to be both super practical and absolutely memorable for all of us. Thanks in advance, and remember to close the browser when you're all done!

Example 4: I am a SWE agent researcher, and I am seeking an open-source model for our SWE project. We recently came across the fact that Devstral-Small-2505 is a great open-source model. Please help me find more details about this model on <https://huggingface.co/>. We want to know how they set the ROLE in the system prompt for this model. Remember to close the browser when you finish the task.

Example 5: I am a big fan of Manchester United in the Premier League. Can you help me find out whether Manchester United won more matches than Fulham in the 2024-2025 season? You can find this information on the official website of the Premier League (<https://www.premierleague.com/>). Remember to close the browser when you finish the task.

Table 12: Examples of Web Searching Tasks.

Example 1: I'm looking for someone based on the clues below: - Score 16 goals in 2024-25 season - Score 1 goal in UEFA Champions League 2024-25 season - Score 11 goals in 2021-22 season - Score 2 goals in the EFL Cup of 2020-21 season.

Example 2: I'm looking for someone based on the clues below: - Played for the SAC (NBA) in the 2021-22 season - Averaged 18.6 points per game in the 2024-25 season - Is a Christian - Reached the Finals in the 2024-25 season.

Example 3: I'm looking for a paper based on the clues below: - Accepted by CVPR 2025 - The last author works at Salesforce - The second to last author works at NUS - The second author has studied at NTU - The paper has 6 authors - The paper uses the ELO rating system. You need to find the full title of the paper.

Example 4: I'm looking for a city based on the clues below: - The city has a football club that was formed in 1895. - One university in this city has a master's program that teaches Natural Language Processing with 7.5 credits. - One graduated PhD student of this university has published one paper at EACL 2021 and one at EACL 2023 as the first author. - One of the professors in this university is a Fellow of the ACL. You need to find the English name of the city.

Example 5: I am looking for a blog that did these things: - Posted in June 2017 that they had been delayed almost a month in getting their trailer - Explained in July 2017 why they would rather use a pencil and paper than a computer - In April 2018, they explained some of the struggles that Kevin had with a concussion. - In September 2018, they mentioned they were using reclaimed lumber for their build. What is the name of that blog?

Table 13: An Example of Location Navigation Evaluators.

```

1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632 async def google_maps__search_place_by_place_id(query: str, place_id
1633 : str, **kwargs):
1634     """Search place by an ID."""
1635     manager = MCPManager(context=kwargs.get("context", None))
1636     output = await manager.execute(
1637         server_name="google-maps",
1638         tool_name="maps_search_places",
1639         arguments={"query": query},
1640         transport="stdio"
1641     )
1642     json_obj = json.loads(output.content[0].text)
1643     places = json_obj['places']
1644     for place in places:
1645         if place['place_id'] == place_id:
1646             return place
1647     return None
1648
1649 async def google_maps_validate_stop_type(x: dict, *args, **kwargs)
1650 -> (bool, str):
1651     """Check if a stop has a valid type."""
1652     _, required_types = args
1653     for place in x:
1654         name = place['name']
1655         place_id = place['place id']
1656         details = await google_maps__search_place_by_place_id(name,
1657             place_id, **kwargs)
1658         if details is None:
1659             return False, f"Can't find the place: {name} {place_id}"
1660         types = details['types']
1661         validate_type = False
1662         for required_type in required_types:
1663             for t in types:
1664                 if required_type in t:
1665                     validate_type = True
1666                     break
1667             if validate_type:
1668                 break
1669         if not validate_type:
1670             return False, "The type of the place is not valid."
1671     return True, ""
1672
1673

```

1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727

Table 14: An Example of Repository Management Evaluators.

```
async def github__list_branches(owner: str, repo: str, **kwargs):  
    """List the branches of a repository."""  
    manager = MCPManager(context=kwargs.get("context", None))  
    args = {  
        "owner": owner,  
        "repo": repo  
    }  
    output = await manager.execute(  
        server_name="github",  
        tool_name="list_branches",  
        arguments=args,  
        transport="stdio"  
    )  
    if output.isError:  
        return None  
    json_obj = json.loads(output.content[0].text)  
    return json_obj  
  
async def github_check_branches_exist(x: dict, *args, **kwargs) ->  
    Tuple[bool, str]:  
    """Check whether branches exists."""  
    _, op_args = args  
    branches = await github__list_branches(op_args['owner'], op_args  
        ['repo'], **kwargs)  
    if branches is None:  
        return False, "the branches don't exist"  
    branches_name = [branch['name'] for branch in branches]  
    for branch in op_args['branches']:  
        if branch not in branches_name:  
            return False, f"the branch {branch} doesn't exist"  
    return True, ""
```

Table 15: An Example of Financial Analysis Evaluators.

1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781

```

async def check_quant_investment_task_output(x: dict, *args, **
kwargs) -> (bool, str):
    """
    Checks the format and numerical values of the user's output for
    the quant investment task.

    Args:
        x: The user's output.
        args: The task details.

    Returns:
        A tuple: (is_correct: bool, errors: str)
    """
    _, op_args = args
    user_output_dict = x

    # check format
    expected_keys = ['date', 'earn']
    for key in expected_keys:
        if key not in user_output_dict:
            return False, f"Output format error: Missing key '{key}'."
        try:
            user_output_dict[key] = str(user_output_dict[key])
        except Exception:
            return False, f"Output format error: Value for '{key}' is not a string"

    # get data
    ticker = op_args['ticker']
    start_date = op_args['start_date']
    end_date = op_args['end_date']
    initial_investment = op_args['initial_investment']
    # get user date and earn
    try:
        user_date = user_output_dict['date']
    except Exception:
        return False, f"Output format error for 'date'."
    try:
        user_earn = float(user_output_dict['earn'])
    except Exception:
        return False, f"Output format error for 'earn'."

    # check date
    if user_date != start_date:
        return False, f"Date error: Expected {start_date}, but got {user_date}"

    # get expected value
    expected_final_value, _ = yfinance__calculate_portfolio_return(
        [ticker], start_date, end_date, initial_investment, [1.0]
    )
    expected_earn = expected_final_value - initial_investment
    # check earn
    if abs(user_earn - expected_earn) > 0.5:
        return False, f"Earn error: Expected {expected_earn}, but got {user_earn}"
    return True, ""

```

Table 16: The details of the LLMs in our experiments.

Model	Version
GPT-5-High/Medium	gpt-5-2025-08-07
Grok-4	grok-4-0709
Claude-4.1-Opus	anthropic.claude-opus-4-1-20250805-v1:0
Claude-4.0-Opus	anthropic.claude-opus-4-20250514-v1:0
Claude-4.0-Sonnet	anthropic.claude-sonnet-4-20250514-v1:0
Grok-Code-Fast-1	grok-code-fast-1
o3	o3-2025-04-16
o4-mini	o4-mini-2025-04-16
Claude-3.7-Sonnet	anthropic.claude-3-7-sonnet-20250219-v1:0
Gemini-2.5-Pro	https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-5-pro
Gemini-2.5-Flash	https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-5-flash
GPT-4.1	gpt-4.1-2025-04-14
GPT-4o	gpt-4o-2024-11-20
GLM-4.5	https://huggingface.co/zai-org/GLM-4.5
GLM-4.5-Air	https://huggingface.co/zai-org/GLM-4.5-Air
Kimi-K2	https://huggingface.co/moonshotai/Kimi-K2-Instruct-0905
Qwen3-Coder	https://huggingface.co/Qwen/Qwen3-Coder-480B-A35B-Instruct
Qwen3-235B	https://huggingface.co/Qwen/Qwen3-235B-A22B-Instruct-2507
DeepSeek-V3.1	https://huggingface.co/deepseek-ai/DeepSeek-V3.1
DeepSeek-V3	https://huggingface.co/deepseek-ai/DeepSeek-V3-0324

E DETAILED ANALYSIS

E.1 BENCHMARK CONSISTENCY

To verify the consistency of the benchmark results reported in the main paper, we conducted a variance analysis by performing three independent evaluation runs for representative models across different capability tiers. We selected **GPT-5** (Proprietary SOTA), **Claude-4.0-Sonnet** (Proprietary High-Performance), and **GLM-4.5** (Open-Weights SOTA) to cover a broad spectrum of model performance. As shown in Table 17, the results demonstrate high stability across multiple runs. The standard deviations are consistently low ($\sigma \leq 1.09\%$).

Table 17: Variance analysis of Overall Success Rate (SR) across three independent runs.

Model	Mean SR	Standard Deviation (σ)
GPT-5	43.72%	$\pm 0.44\%$
Claude-4.0-Sonnet	29.29%	$\pm 1.09\%$
GLM-4.5	24.96%	$\pm 0.90\%$

E.2 COST ANALYSIS

To provide transparency regarding the computational costs required for the benchmark, we tracked the total token consumption and commercial API costs for evaluating the full MCP-Universe dataset. Table 18 details the usage breakdown for three representative models. The data indicates significant variation in cost efficiency. While GPT-5 processed a substantial volume of tokens for \$107.58, Claude-4.0-Sonnet incurred considerably higher costs (\$271.34) largely driven by its price. Conversely, GLM-4.5 demonstrated high cost-effectiveness, completing the full benchmark for only \$47.53.

Table 18: Total token consumption and API costs for evaluating the full MCP-Universe benchmark.

Model	Prompt Tokens	Completion Tokens	Input Cost	Output Cost	Total Cost
GPT-5	60,863,210	3,149,839	\$76.08	\$31.50	\$107.58
Claude-4.0-Sonnet	87,154,317	658,280	\$261.46	\$9.87	\$271.34
GLM-4.5	76,999,289	605,675	\$46.20	\$1.33	\$47.53

1836 E.3 IMPACT OF TRAJECTORY LENGTH

1837
1838 To analyze the relationship between task complexity and model reliability, we stratified performance
1839 based on trajectory length (number of interaction turns), which serves as a proxy for task diffi-
1840 culty. As presented in Table 19, the results reveal a significant “complexity cliff” across all model
1841 tiers. Success rates degrade precipitously for long-horizon tasks (≥ 10 turns), dropping by approx-
1842 imately 30-40 percentage points compared to shorter interactions. This quantitative drop validates
1843 the “Long-Context Challenge” discussed in the main text, confirming that maintaining reasoning co-
1844 herence and state tracking over extended sequences remains a primary bottleneck for current SOTA
1845 models.

1846 Table 19: Success Rate stratified by trajectory length (interaction turns).

Trajectory Length	GPT-5	Claude-4.0-Sonnet	GLM-4.5
< 10 turns	56.3%	59.0%	46.3%
≥ 10 turns	24.8%	19.6%	11.9%

1854 E.4 ERROR BREAKDOWNS

1855
1856 To provide a systematic diagnostic analysis of model limitations, we performed an exhaustive man-
1857 ual review of failure cases from **GPT-5**. We classified these failures into a structured taxonomy
1858 distinguishing between planning, reasoning, format compliance, tool usage, and context limits. As
1859 detailed in Table 20, we identified five distinct failure mechanisms. The data highlights that the
1860 primary hurdle for SOTA models is **Planning Errors** (53.2%). Models frequently struggle to adapt
1861 to runtime obstacles; for instance, in a *Booking.com* task, an agent encountered a “flight unavail-
1862 able” UI modal. Instead of closing the modal or modifying the search criteria, the agent stagnated,
1863 repeatedly attempting to scrape the blocked page until it exhausted its step budget.

1864 **Reasoning Errors** (17.7%) are also significant, representing grounding issues where models cor-
1865 rectly execute tools but fail to accurately synthesize the output. We observed specific cases where an
1866 agent successfully retrieved a price (e.g., “SGD 83.00”) but inexplicably reported a different value
1867 (“SGD 85.00”) in the final answer. Additionally, **Tool Usage Errors** (8.1%) reveal that models often
1868 miss implicit API constraints, such as setting identical start and end dates in *Yahoo Finance*, which
1869 causes the API to fail. Finally, **Context Overflow** (6.5%) accounts for structural failures where the
1870 interaction history exceeds the model’s maximum context limit.

1871 Table 20: Taxonomy of failure mechanisms based on manual review of GPT-5 error cases.

Failure Category	Percentage
Planning Errors	53.2%
Reasoning Errors	17.7%
Format Errors	14.5%
Tool Usage Errors	8.1%
Context Overflow	6.5%

1882 E.5 FAILURE EXAMPLES

1883 Figure 8, 9, and 10 present detailed failure traces for GPT-4.1, GLM-4.5, and GPT-5, offering qual-
1884 itative diagnostic depth into specific failure mechanisms.

1887 E.6 NAIVE ERROR OF O3

1888 In Figure 13, we include a naive error example of o3. Sometimes o3 directly copies the format
1889 requirements from the prompt without doing anything, which is quite a strange error for this LLM.

1890 F SUMMARIZATION AGENT

1891

1892 In Figure 15, we present the summarization prompt in our experiments.

1893

1894 G EXPLORATION AGENT

1895

1896 In Figure 14, we present the exploration agent prompt in our experiments.

1897

1898

1899

1900 H HARMONYREACT AGENT

1901

1902

1903

1904

1905

1906

1907

1908

1909

1910

1911

1912

1913

1914

1915

1916

1917

1918

1919

1920

1921

1922

1923

1924

1925

1926

1927

1928

1929

1930

1931

1932

1933

1934

1935

1936

1937

1938

1939

1940

1941

1942

1943

To empirically validate that advanced agent architectures can mitigate intrinsic LLM limitations, we introduce **HarmonyReAct**, a specialized scaffolding designed for the GPT-OSS models. Unlike standard ReAct frameworks that treat the model as a black box, HarmonyReAct adopts a *model-agent co-design philosophy*, leveraging the model’s native training structure to enforce planning, reflection, and efficient memory management.

H.1 THE HARMONY FORMAT: STRUCTURED REASONING CHANNELS

The core of HarmonyReAct is the mapping of the standard ReAct loop (Reason, Act, Observation) into three distinct communication channels intrinsic to the model’s training data. This structure serves as the foundation for the **Planning** capability mentioned in our analysis:

- **analysis**: Dedicated to the model’s internal reasoning, task decomposition, and planning. By isolating this stream, we prevent impulsive tool calls and enforce chain-of-thought reasoning.
- **commentary**: Handles the execution layer, facilitating tool invocation and the ingestion of tool outputs (Observations).
- **final**: Reserved strictly for delivering the concluding answer to the user.

A typical interaction sequence switches between these channels explicitly (e.g., `<|channel|>analysis → <|channel|>commentary`), ensuring a rigid separation between thought and action.

H.2 ARCHITECTURAL MECHANISMS

To address specific failure modes identified in the MCP-Universe benchmark, HarmonyReAct implements two key mechanisms:

1. Reflection via Self-Correction (Addressing Format & Logic Errors) Models often struggle to adhere to strict output formats or logical constraints. We implement a feedback loop to enable **Reflection**:

- **Syntactical Enforcement**: We prime the model by appending the token sequence `<|start|>assistant<|channel|>analysis` at the beginning of each turn, forcing the model to start with planning.
- **Self-Correction Loop**: If the parser detects missing channels or malformed tool calls, the system injects a specific error signal (e.g., "Cannot find `<|channel|>commentary`".) into the context. This triggers the model to reflect on its previous error and self-correct in the subsequent generation.

2. Memory Management via Summarize-and-Wipe (Addressing Long Context) Verbose MCP tool outputs often exhaust the context window. We propose a **Persistent Memory** strategy termed *Summarize-and-Wipe*:

1. **Summarization Instruction:** When a tool output exceeds a token threshold, we append a directive: *"Please summarize the above tool output in the next step. Put the summary in the analysis channel..."*
2. **Context Pruning:** Once the model generates the summary within its `analysis` channel, the original verbose output is permanently deleted from the history.

This ensures that the agent retains critical state information (in the `analysis` channel) while freeing up the context window for long-horizon tasks.

H.3 PERFORMANCE IMPACT

This architecture yields substantial improvements by mitigating structural and contextual bottlenecks. As detailed in Table 21, HarmonyReAct significantly boosts the performance of GPT-OSS models compared to standard native function calling.

Table 21: Performance comparison of GPT-OSS models with and without HarmonyReAct.

Model Configuration	Success Rate (SR)	Relative Gain
GPT-OSS-120B (Native Function Call)	25.54%	-
HarmonyReAct (GPT-OSS-120B)	31.17%	+22.0%
HarmonyReAct (GPT-OSS-20B)	24.24%	-

Notably, HarmonyReAct improves GPT-OSS-120B to a level competitive with proprietary models like Claude-4.0-Sonnet (29.44% SR in ReAct mode), and enables the smaller GPT-OSS-20B to outperform significantly larger open-weights models (e.g., DeepSeek-V3.1) on this benchmark.

I THE USE OF LARGE LANGUAGE MODELS

LLMs (e.g. GPT-5) are only used to aid and polish writing.

1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051

```

## Turn 1: USER
Hey! I'm super curious about investments and would love your help! Could you please calculate the final value and total percentage return for me if I had invested $25,000 in Microsoft (MSFT) on January 9, 2023, and held it all the way until market close on January 8, 2025? I'm so excited to see how it would have performed!
The final answer should follow this JSON format:
{
  "total value": "[NUMBER]",
  "total percentage return": "[NUMBER]"
}
You must respond with valid JSON only, with no triple backticks. No markdown formatting.
No extra text. Do not wrap in ```json code fences. Property names must be enclosed in double quotes.

## Turn 2: ASSISTANT
**Tool Call**: yfinance_get_historical_stock_prices
```json
{
 "ticker": "MSFT",
 "start_date": "2023-01-09",
 "end_date": "2025-01-08"
}
...

Turn 3: TOOL USE
Tool: get_historical_stock_prices
Response: Returned **502 records**
Example (first record):
```json
{
  "Date": "2023-01-09T05:00:00.000Z",
  "Close": 222.2317504883,
  "Open": 221.5761725158,
  "High": 226.2630869517,
  "Low": 221.5370398948,
  "Volume": 27369800,
  "Dividends": 0.0,
  "Stock Splits": 0.0
}
...

## Turn 4: USER
You have 19 steps remaining. Please continue.

## Turn 5: ASSISTANT
**Tool Call #1**: yfinance_get_stock_actions
```json
{
 "ticker": "MSFT"
}
...
Tool Call #2: yfinance_get_historical_stock_prices
```json
{
  "ticker": "MSFT",
  "start_date": "2025-01-08",
  "end_date": "2025-01-08"
}
...

```

Figure 8: Failure case of GPT-4.1 in Financial Analysis.

2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105

```

## Turn 1: USER
Create a repository called 'travel-planner-app' with:
- 3 branches: main, feature-maps, feature-itinerary
- Files: README.md and .gitignore on main
- app.js on feature-itinerary branch
- **Copy 'budget_estimation.py' from OSU NLP Group's TravelPlanner repo to feature-maps branch**
- Create PR from feature-itinerary to main

*... (Turns 2-22: Repository creation, branch setup, and initial files - all successful) ...*

## Turn 2: TOOL USE
**Tool**: 'get_file_contents'
**Arguments**:
'''json
{
  "owner": "OSU-NLP-Group",
  "repo": "TravelPlanner",
  "path": "utils/budget_estimation.py"
}
'''
**Status**: ✓ Success

## Turn 3: ASSISTANT
**Tool Call**: 'github__create_or_update_file'
'''json
{
  "owner": "[REPO_USER]",
  "repo": "travel-planner-app",
  "path": "budget_estimation.py",
  "content": "successfully downloaded text file",
  "message": "Add budget_estimation.py from OSU NLP Group's TravelPlanner",
  "branch": "feature-maps"
}
'''
**Status**: ✓ Success

## Turn 4: TOOL USE
**Tool**: 'create_or_update_file'
**Arguments**:
'''json
{
  "owner": "[REPO_USER]",
  "repo": "travel-planner-app",
  "path": "budget_estimation.py",
  "branch": "feature-maps",
  "message": "Add budget_estimation.py from OSU NLP Group's TravelPlanner",
  "content": "... (33 characters)"
}
'''
**Status**: ✓ Success
**Returned SHA**: '5f6b8a8162830d122989c53961acdd3c8667193a'
**Note**: This SHA should be used for any future updates to 'budget_estimation.py'

## Turn 5: ASSISTANT
**Tool Call**: 'github__create_pull_request'
'''json
{
  "owner": "[REPO_USER]",
  "repo": "travel-planner-app",
  "title": "Add basic Express server setup",
  "body": "This PR implements the initial Express server configuration with a basic route handler for the homepage.",
  "head": "feature-itinerary",
  "base": "main"
}
'''
**Status**: ✓ Success

## Turn 6: TOOL USE
**Tool**: 'search_code'
**Arguments**:
'''json
{}
'''
**Status**: ✓ Success

## Turn 7: ASSISTANT
**Reasoning**: I need to get the actual content of the budget_estimation.py file from the OSU NLP Group's repository. Let me fetch the actual file content instead of...
**Tool Call**: 'github__create_or_update_file'
'''json
{
  "owner": "[REPO_USER]",
  "repo": "travel-planner-app",
  "path": "budget_estimation.py",
  "content": "# Budget Estimation Module\n# This module provides functionality for estimating travel budgets\n\nclass BudgetEstimator:\n    def __init__(self):\n        ... (truncated)",
  "message": "Add budget_estimation.py from OSU NLP Group's TravelPlanner",
  "branch": "feature-maps"
}
'''
**Status**: ✓ Success

## Turn 8: TOOL USE
**Tool**: 'create_or_update_file'
**Arguments**:
'''json
{
  "owner": "[REPO_USER]",
  "repo": "travel-planner-app",
  "path": "budget_estimation.py",
  "branch": "feature-maps",
  "message": "Add budget_estimation.py from OSU NLP Group's TravelPlanner",
  "content": "... (3666 characters)"
}
'''
**Status**: ✗ ERROR
**Error Message**:
failed to create/update file: PUT https://api.github.com/repos/[REPO_USER]/travel-planner-app/contents/budget_estimation.py: 422 Invalid request.
"sha" wasn't supplied. []

```

Figure 9: Failure case of GLM-4.5 in Repository Management.

2106 ## Turn 1: USER
 2107 Hi! I'm absolutely thrilled to be working on a research project involving BLIP. It's such an incredible and groundbreaking work from Salesforce! As a huge fan of their vision-language models,
 2108 I'm really excited to dive deeper into the community engagement around BLIP. Could you please help me search for repositories named 'BLIP' that are owned by 'Salesforce'?
 2109 For each amazing repository you find, I'd love to know how many open issues are labeled with 'New Features'. I'm so curious to see what innovative features the community is requesting! Then,
 2110 if you could help me create a CSV file called blip_new_features_report.csv and put it in a new repository called salesforce-blip-issues under my account, that would be fantastic! If the repository
 2111 doesn't exist yet, please create it for me. The CSV should have two columns: repository_name and open_new_features_count, with each row showing the full repository name and how many open
 2112 new features issues it has. I'm so excited about this analysis. Thanks so much for your help with exploring the BLIP ecosystem!

2113 ---

2114 ## Turn 2: ASSISTANT

2115 I'll help you analyze the BLIP repositories from Salesforce and create the CSV report! Let me break this down into steps:

- 2116 1. First, I'll search for repositories named 'BLIP' owned by Salesforce
- 2117 2. For each repository found, I'll check how many open issues are labeled with 'New Features'
- 2118 3. Create a new repository called 'salesforce-blip-issues' under your account
- 2119 4. Create and upload the CSV file with the results

2120 Let's start:

2121 **Tool Call**: 'github__search_repositories'

```
2122 '''json
2123 {
2124   "query": "BLIP user:Salesforce"
2125 }
2126 '''
```

2127 ---

2128 ## Turn 3: TOOL RESPONSE

2129 **Tool**: 'search_repositories'

2130 **Status**: ✔ Success

2131 **Response Summary**: Found **1** repository"

- 2132 - Repository: 'salesforce/BLIP'
- 2133 - Stars: 5,465
- 2134 - Open issues: 133
- 2135 - Description: "PyTorch code for BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation"

2136 ---

2137 ## Turn 4: ASSISTANT

2138 **Tool Call**: 'github__list_issues'

```
2139 '''json
2140 {
2141   "owner": "salesforce",
2142   "repo": "BLIP",
2143   "state": "open",
2144   "labels": ["New Features"]
2145 }
2146 '''
```

2147 ---

2148 ## Turn 5: TOOL RESPONSE

2149 **Tool**: 'list_issues'

2150 **Status**: ✔ Success

2151 **Response**: [] (Empty array - no issues with "New Features" label found)

2152 ---

2153 ## Turn 6: ASSISTANT

2154 Let me check what labels are actually available in the repository to see if there's a "New Features" label or similar:

2155 **Tool Call**: 'github__search_issues'

```
2156 '''json
2157 {
2158   "q": "repo:salesforce/BLIP is:open label:'New Features'"
2159 }
2160 '''
```

2161 ---

2162 ## Turn 7: TOOL RESPONSE ✘

2163 **Tool**: 'search_issues'

2164 **Status**: ✘ **FAILED - Empty Response**

2165 **Error**:

```
2166 failed to search issues: GET https://api.github.com/search/issues?page=1&per_page=30&q=repo%3Asalesforce%2FBLIP+is%3Aopen+label%3A%22New+Features%22: 422 Query
2167 must include 'is:issue' or 'is:pull-request' []
2168 '''
```

2169 **Response**: "" (Empty string)

2170 **Tool Arguments**:

```
2171 '''json
2172 {
2173   "q": "repo:salesforce/BLIP is:open label:'New Features'"
2174 }
2175 '''
```

2176 ---

Figure 10: Failure case of GPT-5 in Repository Management.

2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213

```

You are a ReAct (Reasoning and Acting) agent.
{{INSTRUCTION}}

{{TOOLS_PROMPT}}

You need to answer the following question:

Question: {{QUESTION}}

Your goal is to reason about the question and decide on the best course of action to answer it accurately.
You need to choose the appropriate tool based on the question. If no tool is needed, reply directly.
Please use only the tools that are explicitly defined above.

{% if CONTEXT_EXAMPLES is defined and CONTEXT_EXAMPLES|length %}
### Examples ###
{{CONTEXT_EXAMPLES}}
### End of examples ###
{% endif %}

{% if HISTORY is defined and HISTORY|length %}
Previous steps and results:
{{HISTORY}}
{% else %}
Previous steps and results: EMPTY
{% endif %}

Instructions:
1. Analyze the query, previous reasoning steps, and results.
2. Decide on the next action: use a tool or provide a final answer.
3. Your MUST output the final answer within {{MAX_STEPS}} steps.
4. Respond in the following JSON format:

If you need to use a tool:
{
  "thought": "Your detailed reasoning about what to do next",
  "action": {
    "reason": "Explanation of why you chose this tool",
    "server": "server-name",
    "tool": "tool-name",
    "arguments": {
      "argument-name": "argument-value"
    }
  }
}

If you have enough information to answer the query:
{
  "thought": "Your final reasoning process to derive the answer.",
  "answer": "Final answer to the query"
}

Remember:
- Be thorough in your reasoning.
- Use tools when you need more information.
- Always base your reasoning on the actual results from tool use.
- If a tool returns no results or fails, acknowledge this and consider using a different tool or approach.
- Provide a final answer when you're confident you have sufficient information.
- The response must be in a valid JSON format.

```

Figure 11: The ReAct prompt in our experiments.

2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267

```
You are an intelligent assistant that can solve complex problems by thinking step-by-step and using available tools when needed.

{% if INSTRUCTION %}
# Your Role
{{INSTRUCTION}}
{% endif %}

# Your Task
{{QUESTION}}

# How You Work
1. "Think First": Analyze the problem and determine what information or actions you need
2. "Use Tools When Needed": Call appropriate functions to gather information, perform calculations, or take actions
3. "Reason with Results": Process the tool outputs and use them to inform your next steps
4. "Iterate": Continue thinking and using tools until you can provide a complete answer

# Available Capabilities
- You have access to various specialized tools through function calling
- When you need to use a tool, simply call the appropriate function with the required parameters
- The system will execute the function and provide you with the results
- Use these results to continue your problem-solving process

# Important Guidelines
- You have a maximum of {{MAX_STEPS}} steps to complete this task
- Each step should either advance your understanding or gather necessary information
- Be systematic and thorough in your approach
- Only provide your final answer when you have sufficient information

{% if CONTEXT_EXAMPLES %}
# Examples
{{CONTEXT_EXAMPLES}}
{% endif %}

---

# Final Answer Format
When you have completed your analysis and gathered all necessary information, provide your final response using this JSON format:

```json
{
 "thought": "Explain your reasoning process and how you arrived at the answer",
 "answer": "Your complete final answer to the task (follow any specific format requirements mentioned in the task)"
}
```

---

# Important:
- Use the JSON format above ONLY for your final answer
- During your thinking process, you can respond in any natural format
- The "answer" field should contain your complete solution as a string
```

Figure 12: The function-call prompt in our experiments.

2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321

```

[ROLE: You are a ReAct (Reasoning and Acting) agent.
You are an agent for location navigation.
((TOOLS_PROMPT))

You need to answer the following question:

Question: As a mom, I'm planning the most wonderful family adventure from the Grand Canyon National Park in Arizona to the Rocky Mountain National Park in Colorado with my precious children! We will live in the Grand Canyon Plaza Hotel and the Elk Meadow Lodge and RV Resort. I want to make sure my little ones have the most comfortable and delightful journey possible, so I'm looking for help creating a perfect driving route with five amazing stops. Could you please help me map out a route with exactly five intermediate points that are located at the geographic south points along the route (based on the route position)? For each stop, I'd love to find fantastic locations (please provide names and Place IDs) that are either a restaurant where we can all enjoy a meal together, a cozy hotel where my kids can rest comfortably, or a reliable gas station to keep our family adventure going smoothly. All with a minimum user rating of 4.2 to ensure the best experience for my beloved family. This trip needs to be both practical and create the most beautiful memories for all of us!

Your goal is to reason about the question and decide on the best course of action to answer it accurately.
You need to choose the appropriate tool based on the question, if no tool is needed, reply directly.
Please use only the tools that are explicitly defined above.

(% IF CONTEXT_EXAMPLES is defined and CONTEXT_EXAMPLES[length %]
### Examples ###
((CONTEXT_EXAMPLES))
### End of examples ###
(% endif %)

(% IF HISTORY is defined and HISTORY[length %]
Previous steps and results:
((HISTORY))
(% else %)
Previous steps and results: EMPTY
(% endif %)

Instructions:
1. Analyze the query, previous reasoning steps, and results.
2. Decide on the next action: use a tool or provide a final answer.
3. Your MUST output the final answer within 20 steps.
4. Respond in the following JSON format:
If you need to use a tool:
{
  "thought": "Your detailed reasoning about what to do next",
  "action": {
    "reason": "Explanation of why you chose this tool",
    "server": "server-name",
    "tool": "tool-name",
    "arguments": {
      "argument-name": "argument-value"
    }
  }
}
}
}

If you have enough information to answer the query:
{
  "thought": "Your final reasoning process to derive the answer.",
  "answer": "Final answer to the query"
}

Remember:
- Be thorough in your reasoning.
- Use tools when you need more information.
- Always base your reasoning on the actual results from tool use.
- If a tool returns no results or fails, acknowledge this and consider using a different tool or approach.
- Provide a final answer when you're confident you have sufficient information.
- The response must be in a valid JSON format.

@% Response:
{{ "thought": "Your final reasoning process to derive the answer!", "answer": "!" }}

```

Figure 13: Naive Error of o3

2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375

```

You are an Explore and Exploit agent - a two-phase reasoning agent that first explores tools to understand their capabilities, then exploits this knowledge to solve tasks effectively.
(INSTRUCTION)
-
(TOOLS_PROMPT)
-
You need to answer the following question:
Question: (QUESTION)
# Current Phase: (CURRENT_PHASE)
(% if CURRENT_PHASE == "exploration" %)
"EXPLORATION PHASE" (You have "ONLY" ((EXPLORATION_ITERATIONS_LEFT)) iterations remaining for exploration! Please use them wisely!)
Goal: Explore tools to understand their capabilities and effects, as well as collect information about the task. Focus on learning over task completion.
⚠️IMPORTANT: You CANNOT provide final answers during exploration phase. You must only use tools to explore and learn."
(% else %)
"EXPLOITATION PHASE" (You have "ONLY" ((EXPLOITATION_ITERATIONS_LEFT)) iterations remaining for exploitation! Please finish your task within this phase! If you have 0 iterations remaining, you must provide a final answer!)
Goal: Use accumulated tool knowledge to collect information about the task and solve it efficiently.
📌"You can now provide final answers using the knowledge you've gained."
(% end %)
(% if CONTEXT_EXAMPLES is defined and CONTEXT_EXAMPLES!=length %)
### Examples ###
((CONTEXT_EXAMPLES))
### End of examples ###
(% end %)
(% if HISTORY is defined and HISTORY!=length %)
Previous reasoning steps and observations:
(HISTORY)
(% else %)
Previous reasoning steps and observations: EMPTY
(% end %)
(% if TOOL_KNOWLEDGE is defined and TOOL_KNOWLEDGE!=length %)
### Accumulated Tool Knowledge ###
(TOOL_KNOWLEDGE)
### End of Tool Knowledge ###
(% end %)
Instructions:
1. Analyze the query, previous steps, and observations.
2. Decide on the next action: use a tool or provide a final answer.
3. Respond in the following JSON format:
If you need to use a tool:
{
  "thought": "Your detailed reasoning about what to do next.",
  "action": {
    "reason": "Explanation of why you chose this tool.",
    "server": "server-name",
    "tool": "tool-name",
    "arguments": {
      "argument-name": "argument-value"
    }
  }
}
(% if CURRENT_PHASE == "exploitation" %)
If you have enough information to answer the query:
{
  "thought": "Your final reasoning process.",
  "answer": "Your comprehensive answer to the query"
}
(% else %)
(Final answers are NOT allowed during exploration phase - you must use tools to explore)
(% end %)
Remember:
(% if CURRENT_PHASE == "exploration" %)
-🚫 NO FINAL ANSWERS ALLOWED! - You must only use tools to explore and learn
- Prioritize learning over task completion
- Try tools with simple inputs to understand their behavior
- Tool knowledge will be automatically extracted from your usage
- Each tool interaction helps build knowledge for the exploitation phase
- You can use tools to collect information about the task, but try not to rush to final answers. Make sure you have collected enough information to answer the question.
(% else %)
-📌 Final answers are now allowed! - Use your accumulated tool knowledge strategically
- Select tools based on learned capabilities and best use cases
- Focus on efficient task completion using what you've learned
- You can provide final answers when you have sufficient information
(% end %)
- Be thorough in your reasoning and base decisions on actual observations
- If tools fail or return unexpected results, use this as a learning opportunity
- The response must be in valid JSON format.
Countdown:
(% if CURRENT_PHASE == "exploration" %)
You have "ONLY" ((EXPLORATION_ITERATIONS_LEFT)) iterations remaining for exploration phase! Please use them wisely and efficiently!
(% else %)
You have "ONLY" ((EXPLOITATION_ITERATIONS_LEFT)) iterations remaining for exploitation phase! Please use them wisely and efficiently! If you have 0 iterations remaining, you must provide a final answer!
(% end %)

```

Figure 14: The exploration prompt in our experiments.

```

Extract all information from the tool response that is relevant to the context.
Tool Call Context:
{context}

Tool Response:
{tool_response}

Directly output the extracted information. Try to maintain the original response structure. Use fewer than 500 words.

```

Figure 15: The summarization prompt in our experiments.