MedMCP-Calc: Benchmarking LLMs for Realistic Medical Calculator Scenarios via MCP Integration

Yakun Zhu[1,2*] **Yutong Huang**[1*] **Shengqian Qin**[1*] **Zhongzhen Huang**[1]
Shaoting Zhang[1†] **Xiaofan Zhang**[1,2†]

[1]Shanghai Jiao Tong University, [2]Shanghai Innovation Institute

Abstract

Medical calculators are fundamental to quantitative, evidence-based clinical practice. However, their real-world use is an adaptive, multi-stage process, requiring proactive EHR data acquisition, scenario-dependent calculator selection, and multi-step computation, whereas current benchmarks focus only on static single-step calculations with explicit instructions. To address these limitations, we introduce MedMCP-Calc, the first benchmark for evaluating LLMs in realistic medical calculator scenarios through Model Context Protocol (MCP) integration. MedMCP-Calc comprises 118 scenario tasks across 4 clinical domains, featuring fuzzy task descriptions mimicking natural queries, structured EHR database interaction, external reference retrieval, and process-level evaluation. Our evaluation of 23 leading models reveals critical limitations: even top performers like Claude Opus 4.5 exhibit substantial gaps, including difficulty selecting appropriate calculators for end-to-end workflows given fuzzy queries, poor performance in iterative SQL-based database interactions, and marked reluctance to leverage external tools for numerical computation. Performance also varies considerably across clinical domains. Building on these findings, we develop CalcMate, a fine-tuned model incorporating scenario planning and tool augmentation, achieving state-of-the-art performance among open-source models[1].

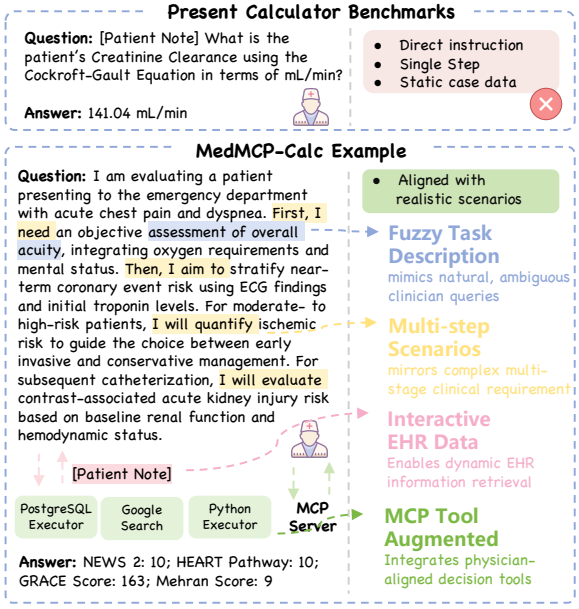

Figure 1: Existing benchmarks typically provide direct instructions with static case data for single-step computation. In contrast, MedMCP-Calc presents fuzzy task descriptions mimicking natural clinician queries, requires multi-step decision making across complex scenarios, enables dynamic interaction with structured EHR databases, and integrates MCP-based tools for physician assistants.

1 Introduction

Medical calculators are foundational to quantitative, evidence-based clinical practice. They transform patient-specific data into risk estimates, diagnostic thresholds, and treatment recommendations, thereby supporting diagnosis, triage, and therapeutic decision-making across diverse clinical settings. In practice, the use of medical calculators is rarely a single, isolated computation. Instead, it unfolds as an adaptive, multi-stage process embedded within broader clinical workflows. For example, in emergency chest pain triage, clinicians may first apply an initial stability screening (e.g., NEWS2), followed by risk stratification (e.g., HEART Pathway). These results subsequently inform downstream decisions, such as mortality prediction (e.g., GRACE) or procedure-related risk assessment (e.g., Mehran) prior to angiography. Throughout this process, clinicians must gather fragmented patient information from electronic health records (EHRs), determine which calculators are clinically appropriate as context evolves, and reconcile multiple quantitative

[*] Co-first authors
[†] Corresponding author
[1] https://github.com/SPIRAL-MED/MedMCP-Calc

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 4845–4873

outputs into actionable decisions.

Recent advances in large language models (LLMs) have sparked growing interest in the potential to support such complex, tool-intensive workflows (Mialon et al., 2023; Patil et al.; Fan et al., 2024; Xie et al., 2024a; Qiao et al., 2024). A key enabler is the Model Context Protocol (MCP) (Anthropic, 2024), often described as the "USB-C of AI," (Hightower, 2025), which standardizes how LLM-based agents interact with heterogeneous external systems, including tools, APIs, databases, and other resources. By replacing bespoke, task-specific integrations with a uniform protocol, MCP enables a more extensible ecosystem in which agents can effectively operate with "eyes and hands" in real-world environments (Hou et al., 2025; Yang et al., 2025; Hasan et al., 2025; Wang et al., 2025b). In the medical domain, MCP is particularly promising for enabling "clinical agentic workflows", allowing LLMs to seamlessly interface with EHR databases and auxiliary tools such as search engines and medical calculators.

Despite the rapid adoption of MCP and the emerging body of MCP-based benchmarks, most existing applications focus on general-purpose domains rather than healthcare, and rarely address the stringent requirements of medical calculator use (Wang et al., 2025b; Luo et al., 2025; Wu et al., 2025; Liu et al., 2025). Current medical calculator benchmarks typically evaluate models under highly simplified conditions that deviate substantially from real-world clinical practice (Khandekar et al., 2024; Zhu et al., 2025; Zhang et al., 2025b). Specifically, (1) test cases are often presented as clean, pre-packaged narratives processed in a single turn, rather than requiring iterative evidence gathering from noisy, structured EHR; (2) agents are explicitly instructed to invoke specific calculators, instead of responding to ambiguous, goal-driven queries that reflect natural clinical interactions; and (3) data and tool logic are tightly entangled, making it costly to integrate and maintain evolving calculators. These limitations create a substantial gap in assessing LLMs' ability to perform realistic, multi-step clinical computations, hindering the development of trustworthy medical agents.

To address this gap, we introduce MedMCP-Calc, the first benchmark designed to evaluate LLMs on realistic medical calculator workflows integrated with MCP servers. MedMCP-Calc comprises 118 clinically grounded scenarios spanning four medical domains. As illustrated in Figure 1, the benchmark incorporates three MCP-based environments to support realistic simulation: (1) a `PostgreSQL` server for iterative evidence acquisition from EHR databases, (2) `Google Search` for retrieving up-to-date references (e.g., calculator definitions and clinical guidelines), and (3) a `Python Executor` to enable robust and precise numerical computation. Task prompts in MedMCP-Calc are phrased as fuzzy, goal-oriented questions rather than explicit calculator instructions, better reflecting naturalistic clinician intent. The benchmark also offers an evaluation method for assessing calculator selection, evidence acquisition, and quantitative accuracy capabilities.

We conduct extensive experiments on 23 leading models. Across all models, including top-performing systems such as GPT-5 and Gemini-3-Pro-Preview, we observe substantial performance limitations, underscoring the inherent difficulty of solving tasks within realistic medical calculator workflows. Detailed analyses reveal several key challenges faced by current LLM agents. First, LLMs struggle to select all appropriate calculators required to complete an end-to-end workflow when confronted with fuzzy queries. Second, they exhibit poor performance in iterative database interactions involving SQL. Third, LLMs show a marked reluctance to leverage external tools for numerical computation. In addition, performance varies considerably across clinical domains, indicating the need for domain-specific optimization, especially for complex scoring systems and cognitive assessment scales. Collectively, these findings demonstrate the value of MedMCP-Calc in surfacing critical limitations and open challenges in realistic medical calculator workflows. Building on these, we develop **CalcMate**, a fine-tuned model incorporating scenario planning and aggressive tool augmentation, which achieves state-of-the-art performance among open-source models.

Our contributions can be summarized as follows:

- We introduce MedMCP-Calc, the first MCP benchmark for evaluating LLMs on realistic medical calculator workflow tasks.
- We conduct comprehensive evaluations of 23 leading models, revealing critical limitations in real-world calculator task performance.
- We propose CalcMate, demonstrating that scenario planning combined with tool-augmented training significantly improves performance on complex medical calculator tasks.

2 Related Works

LLM Agents. Modern LLMs (Comanici et al., 2025; Qwen Team, 2025; Team et al., 2025b; Anthropic, 2025b) have evolved from instruction following to autonomous agents capable of planning and tool use, further enhanced by reinforcement learning (Shao et al., 2024; Putta et al., 2024; Huang et al., 2025, 2026b,a; Zhang et al., 2026). Early prompting strategies such as ReAct (Yao et al., 2022) and reflection (Shinn et al., 2023) strengthened single-agent reasoning, while multi-agent frameworks like MetaGPT (Hong et al., 2023) and AutoGen (Wu et al., 2024) coordinate complex workflows via explicit role assignment. Capabilities have expanded from API-based tool use (Schick et al., 2023; Qin et al., 2023; Li et al., 2023; Qin et al., 2025; Liu et al., 2026) to multimodal interactions in digital environments (Xie et al., 2024b; Zheng et al., 2024a; Wang et al., 2024; Dong et al., 2025b; Zhang et al., 2025a), including coding agents (Yang et al., 2024) and GUI-based systems such as OpenAI's CUA (OpenAI, 2025) and Anthropic's Computer Use (Anthropic, 2024). This trend toward general-purpose agents motivates standardized protocols like the Model Context Protocol (MCP) (Anthropic, 2024). Yet such agents struggle with the precision demanded in medical settings (Masayoshi et al., 2025), lacking designs that reliably couple structured EHR retrieval with accurate clinical computation, and often hallucinating in data extraction and calculation.

MCP Benchmark. Recent works have introduced various benchmarks for evaluating agent performance within MCP systems. Early efforts like MCP-AgentBench (Guo et al., 2025) and MCP-Verse (Lei et al., 2025) prioritized tool coverage and scalability, while MCP-RADAR (Gao et al., 2025) adapted static datasets such as HumanEval to MCP scenarios. However, these approaches often overlook the dynamic nature of real-world applications where ground truth evolves over time. To better capture task complexity, MCP-Bench (Wang et al., 2025b) targets complex real-world tasks, and OSWorld-MCP (Jia et al., 2025) extends evaluation to operating system tool invocations. A methodological gap persists in evaluation metrics. While MCPEval (Liu et al., 2025) and LiveMCP-Bench (Mo et al., 2025) adopt the "LLM-as-a-Judge" for scalability, they often suffer from evaluation biases and lack rigorous ground-truth validation for real-time outcomes. Recent advances address these limitations through high-fidelity execution environments. MCP-Universe (Luo et al., 2025) integrates authentic MCP servers with time-sensitive scenarios for long-horizon dynamic workflows , while MCPMark (Wu et al., 2025) employs containerized settings with diverse CRUD operations and programmatic verification to ensure safety and reproducibility. Despite these advances, current MCP benchmarks remain largely domain-agnostic or focused on OS-level interactions, lacking frameworks tailored to vertical medical applications—encompassing long-horizon clinical decision-making, EHR database interactions, and authentic clinical tool use for literature retrieval and dosage calculation.

Medical Calculators. LLM capabilities in quantitative reasoning (Chen et al., 2025b,a; Zhang et al., 2023) have evolved from general mathematical problem-solving to specialized clinical calculations. Foundational benchmarks such as GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) established the viability of chain-of-thought reasoning for structured logic (Dong et al., 2025a; Wang et al., 2025a; Li et al., 2025; Wei et al., 2025), but translating this proficiency to medicine requires handling lengthy clinical contexts and strict adherence to domain-specific formulas. OpenMedCalc (Goodell et al., 2023) first addressed this gap by demonstrating that augmenting LLMs with external calculation APIs significantly reduces errors. Building on this tool-use paradigm, AgentMD (Jin et al., 2025) introduced an autonomous framework for curating and applying thousands of clinical calculators. MedCalc-Bench (Khandekar et al., 2024) subsequently provided the first large-scale, human-verified evaluation dataset , revealing LLM deficiencies in parameter extraction and formula selection. Recent works have further expanded this landscape: CalcQA (Zhu et al., 2025) addresses an end-to-end clinical process involving nested tool calling within lengthy patient histories, while CMedCalc-Bench (Zhang et al., 2025b) extends assessment to Chinese medical environments. Despite these advances, existing studies predominantly formulate medical calculation as reading comprehension over static text, overlooking two critical aspects of real-world clinical workflows: (1) scenario task planning, requiring analysis of clinical scenarios and multi-step reasoning to orchestrate multiple calculators rather than executing isolated single-calculator instructions; and (2) proactive data acquisition, demanding dynamic interaction with clin-

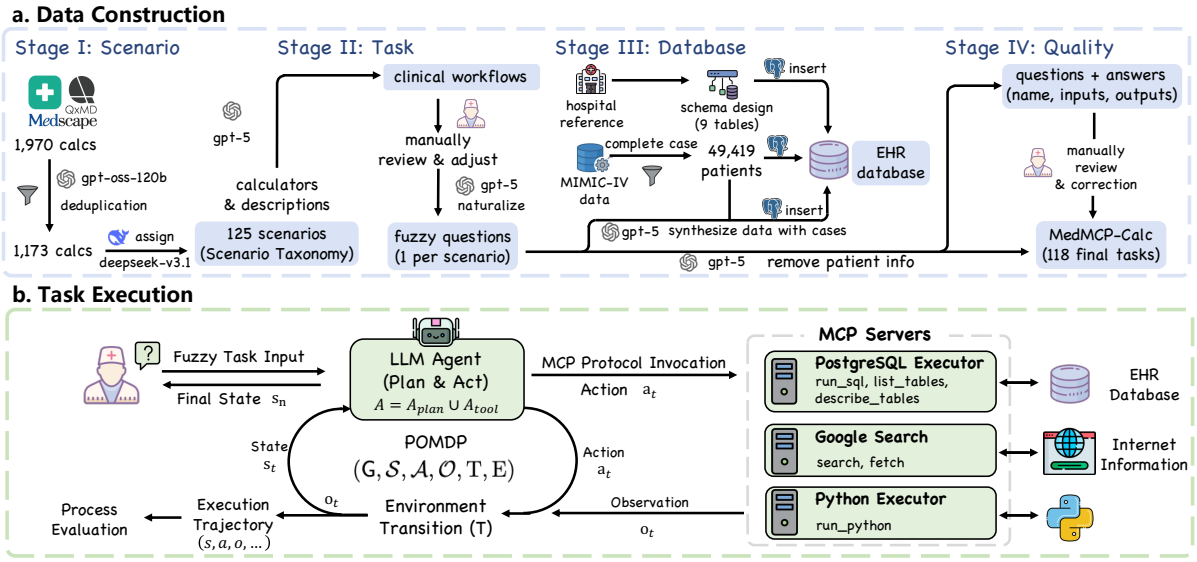

Figure 2: Overview of the MedMCP-Calc benchmark. (a) The data construction pipeline comprises four stages: Scenario Instantiation, Task Creation, Database Construction, and Quality Verification. (b) Task execution is formulated as a POMDP, where an LLM agent interacts with MCP servers to process clinical calculator tasks over realistic EHR databases.

ical databases rather than passive extraction from pre-provided text.

3 MedMCP-Calc Benchmark

To rigorously evaluate LLMs' capabilities in solving complex, dynamic medical calculator tasks within realistic clinical scenarios via MCP servers, we introduce MedMCP-Calc.

3.1 Formalization

The task formulation and evaluation in MedMCP-Calc can be formally characterized as a Partially Observable Markov Decision Process (POMDP). Under multi-turn planning, execution, and observation, each task is represented as a POMDP tuple $(\mathrm{G}, \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathrm{T}, \mathrm{E})$, where G denotes the task instruction and target outcome. For an LLM m that executes a total of n steps, the state space can be defined as $\mathcal{S} = \{s_0, s_1, \ldots, s_n\}$ and the observation space as $\mathcal{O} = \{o_1, \ldots, o_n\}$.

The action space $\mathcal{A}$ of the model m encompasses both planning and tool invocation, formally expressed as $\mathcal{A} = \mathcal{A}_{\text{plan}} \cup \mathcal{A}_{\text{tool}}$. We adopt the MCP, where each tool action is defined as $\mathcal{A}_{\text{tool}} = \{(a_{\text{server\_name}}, a_{\text{tool\_name}}, a_{\text{args}}), \ldots\}$, specifying the MCP server, tool name, and tool parameters, respectively. At each round t, the agent determines the next action a_t based on the current state s_{t-1} and the preceding observation o_{t-1}. The execution of a_t triggers a state transition and yields a new observation, which is formally captured by the transition function $\mathrm{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S} \times \mathcal{O}$. This iterative process continues until either the maximum number of rounds is exceeded or a final answer is produced.

For the final evaluation, given any LLM $\forall m \in \mathcal{M} = \{m_1, \ldots\}$ and agent framework $\forall f \in \mathcal{F} = \{f_1, \ldots\}$, we assess performance based on their trajectories. Specifically, we evaluate the accuracy of achieving task objectives through $\mathrm{E}_{\text{final}} : \mathrm{G} \times \mathcal{S} \to \{0, 1\}$. Beyond final outcomes, we also assess process accuracy, the correctness of intermediate steps during execution $\mathrm{E}_{\text{step}} : \mathcal{S} \times \mathcal{A} \times \mathcal{O} \to \{0, 1\}$. This involves rigorous validation of intermediate parameters by comparing the execution results of the tool calling against the ground-truth.

3.2 Component

MedMCP-Calc comprises four core components: a scenario taxonomy defining clinical task categories, a physician workstation simulating the clinical environment, naturalistic tasks reflecting real-world user behavior, and evaluation metrics for comprehensive assessment.

Scenario Taxonomy. We collaborated with physicians to develop a classification taxonomy encompassing four major clinical domains: *Critical Care & Perioperative Medicine*, *Internal Medicine & Organ Systems*, *Neurology & Psychiatry*, and *Special Populations & Universal Tools*. This taxon-

omy identifies 125 distinct clinical use categories for medical calculators (e.g., "Chest Pain and Acute Coronary Syndrome"), each involving multi-step, interdependent decision-making processes.

Physician Workstation. To replicate the clinical working environment, we designed a series of MCP servers: (1) a `PostgreSQL Executor` server providing `run_sql`, `list_tables`, and `describe_tables` tools for EHR database interaction (we constructed an EHR database from MIMIC-IV data); (2) a `Google Search` server with `search` and `fetch` tools for retrieving up-to-date medical references; and (3) a `Python Executor` server with a `run_python` tool for numerical calculations. This design mirrors how physicians retrieve patient data from hospital databases and consult medical resources during calculator use.

Naturalistic Task. Each task instance is anchored to a specific scenario and patient, accompanied by a fuzzy question that prompts contextual clinical reasoning rather than explicit calculator instructions. For example, instead of "calculate the HEART score," the task asks for risk stratification based on clinical findings. Tasks encompass multiple procedural steps and branching decisions, requiring the LLM to iteratively select actions until reaching a final answer.

Evaluation Metrics. We evaluate both final outputs and intermediate processes: *Task Fulfillment (TF)* measures the proportion of tasks that yield valid final outputs out of all evaluated tasks; *Calculator Selection (CS)* measures the accuracy of calculator choices, computed as the average ratio of correctly selected calculators to the total number of calculators required per task; *Quantitative Precision (QP)* measures the numerical accuracy of computed results, calculated as the average ratio of calculators with correct outputs to the total number of calculators invoked per task; and *Evidence Acquisition (EA)* assesses the LLM's capability to proactively acquire relevant data from databases, computed as the average ratio of correctly extracted data items to the total number of required inputs across all calculators per task. Details are presented in Appendix A.2.

3.3 Pipeline

Based on the benchmark design, we present a detailed construction pipeline for generating tasks in the following parts.

Stage I: Scenario Instantiation. Following the established scenario taxonomy, we instantiated specific scenarios by collecting medical calculators and mapping them to corresponding categories. We gathered all publicly available calculators from three widely-used medical platforms—MDCalc, Medscape, and QxMD—yielding 1,970 entries. To eliminate redundancy, we employed `GPT-OSS-120B` for two rounds of semantic similarity screening based on calculator names and descriptions, followed by manual removal of invalid entries, resulting in 1,173 unique calculators. Guided by the taxonomy, we then used `DeepSeek-V3.1` to assign applicable calculators to each of the 125 scenarios.

Stage II: Task Creation. For each scenario, we first established the corresponding clinical workflow. Specifically, we provided `GPT-5` with the assigned calculators and their functional descriptions, instructing it to construct coherent calculator sequences based on the scenario context. The generated workflows were then manually reviewed and adjusted to ensure alignment with real-world clinical practices, while the results demonstrated strong consistency. Subsequently, we used `GPT-5` to convert these workflows into naturalistic task instructions reflecting authentic user behavior. This process simulates real-world situations where users operate under incomplete information without explicit knowledge of procedural details, issuing underspecified instructions in a conversational tone and requiring the LLM to infer appropriate calculator sequences and execution strategies. Through this process, we constructed one task per scenario. Full prompt templates are provided in Appendix C.4.

Stage III: Database Construction. To enable realistic EHR interactions, we developed a database infrastructure under physician guidance (see Appendix C.5). We designed a relational schema modeled based on real hospital EHR systems, comprising nine tables (e.g., `patient_information`), with detailed specifications provided in Appendix C.3. We then extracted and filtered data from MIMIC-IV, mapping records to conform to our schema, and selected patients with complete records across all tables, yielding a cohort of 49,419 patients. To support task execution, we linked scenario tasks from Stage II with clinically similar existing patients, synthesized corresponding patient data with `GPT-5`, and validated the results (Corbeil et al., 2025). Finally, we employed rule-based code generation to produce SQL statements for inserting task-specific records and removed patient information from task questions, thereby establishing the interactive plat-

Benchmark	Interactive EHR Data	Fuzzy Task Description	Multi-step Scenarios	Process Evaluation	Tool Augmented	MCP Ecosystem
OpenMedCalc (Goodell et al., 2023)	✘	✘	✘	✘	✔	✘
MedCalc-Bench (Khandekar et al., 2024)	✘	✘	✘	✘	✘	✘
CalcQA (Zhu et al., 2025)	✘	✔	✘	✔	✔	✘
CMedCalc-Bench (Zhang et al., 2025b)	✘	✘	✘	✔	✘	✘
MedMCP-Calc	✔	✔	✔	✔	✔	✔

Table 1: **Comparison with existing medical calculator benchmarks.** Existing benchmarks primarily evaluate models' abilities to memorize and apply static medical knowledge for calculator computations, making it difficult to assess their practical capabilities in real-world clinical scenarios that require proactive EHR data retrieval, user intent recognition, multi-step workflow resolution, and real-time tool integration. In contrast, MedMCP-Calc aligns with real-world complex calculator tasks and comprehensively evaluates the practical task-solving capabilities.

form for MedMCP-Calc.

Stage IV: Quality Verification. We conducted multiple rounds of validation to ensure benchmark quality and enable rigorous evaluation. For generated task questions, we performed manual verification to ensure alignment with real-world clinical workflows and practical requirements. For task answers, calculator names, outputs, and input parameters, we referenced all elements against patient information through item-by-item review and correction, using the three aforementioned medical calculator websites as ground-truth references. After excluding data that did not meet quality standards (e.g., insufficient available calculators, inability for EHR data), we obtained a final set of 118 tasks.

3.4 Characteristics

Table 1 presents a comparative analysis between MedMCP-Calc and existing medical calculator benchmarks. MedMCP-Calc exhibits the following distinctive characteristics.

Interactive EHR Data. MedMCP-Calc incorporates a dynamic EHR database requiring active data retrieval, rather than providing static case information. This design enables realistic simulation of clinical data acquisition and evaluates models' capabilities in proactively identifying relevant patient information.

Fuzzy Task Description. Instead of explicit computational instructions, MedMCP-Calc employs naturalistic clinical queries that require models to interpret user intent and autonomously select appropriate calculators—reflecting how clinicians actually formulate requests in practice.

Multi-step Scenarios. MedMCP-Calc is constructed around comprehensive clinical workflows involving staged subtasks and logical reasoning chains, moving beyond single-calculator computations to evaluate authentic multi-step decision-making procedures.

Process Evaluation. Beyond final-answer accuracy, MedMCP-Calc assesses the information acquisition phase by analyzing database interactions and comparing extracted data against ground-truth calculator inputs—a dimension largely overlooked by existing benchmarks.

MCP Tool Augmented. Rather than relying on limited, calculator-specific tools, MedMCP-Calc adopts the MCP protocol with general-purpose servers for database interaction, web search, and computation. This approach enables access to over one thousand publicly available online calculators, supports real-time retrieval of current medical guidelines, and mirrors clinicians' actual workflows of information retrieval and computation.

4 Experiments

4.1 Settings

To comprehensively evaluate the capabilities of existing models in our MedMCP-Calc, we benchmarked a diverse set of state-of-the-art models, encompassing proprietary LLMs, open-source LLMs, and domain-specific medical models. For proprietary models, we evaluated Claude Opus 4.5 (Anthropic, 2025c), Claude Sonnet 4.5 (Anthropic, 2025a), GPT-5 (OpenAI, 2025b), Gemini-3-Pro-Preview (Deepmind, 2025) and Gemini-2.5-Pro (Comanici et al., 2025). For open-source models, we included GLM-4.7 (Team et al., 2025a), DeepSeek-V3.1 (DeepSeek-AI, 2024), Kimi-K2-Instruct (Team et al., 2025b), Qwen3-Series (Team, 2025), Llama-3.3 (Dubey et al., 2024) and Llama-3.1 (Dubey et al., 2024). For domain-specific models, we evaluated Baichuan-M2-32B (Dou et al., 2025) and MedGemma-27B-IT (Sellergren et al., 2025).

Model	Param.	Crit. Care & Periop.			Int. Med. & Organ Sys.			Neuro. & Psych.			Spec. Pop. & Univ. Tools			Overall			
		CS	EA	QP	CS	EA	QP	CS	EA	QP	CS	EA	QP	TF	CS	EA	QP
Proprietary LLMs																	
Claude Opus 4.5	/	71.30	73.51	36.00	69.63	73.89	37.35	60.02	65.83	30.37	56.77	63.41	23.70	100	**66.66**	71.08	**33.92**
Claude Sonnet 4.5	/	69.55	53.32	23.64	67.77	56.64	33.10	55.98	41.26	31.57	60.97	55.14	14.72	100	65.60	53.86	27.80
GPT-5	/	61.87	46.58	26.61	62.77	56.64	29.64	60.20	49.15	27.87	48.24	53.70	17.33	100.00	59.81	53.12	26.70
Gemini-3-Pro	/	57.59	58.68	24.70	56.37	59.10	31.17	48.42	47.07	21.92	46.71	46.49	18.10	97.46	54.05	55.45	26.49
Gemini-2.5-Pro	/	59.61	41.98	20.83	58.89	41.35	21.76	50.38	35.00	18.95	46.64	39.70	19.08	100.00	55.96	40.45	20.77
Open Source LLMs																	
GLM-4.7	355 B	50.86	46.41	11.74	51.03	52.14	17.57	49.46	51.19	17.10	57.43	47.72	13.49	96.61	51.89	50.06	15.59
Deepseek-V3.1	671 B	41.85	37.28	10.77	45.35	39.70	14.22	43.79	44.44	7.42	44.78	47.29	12.38	99.15	44.33	41.04	12.37
Qwen3-235B-Instruct-2507	235 B	48.83	34.95	11.38	41.31	32.62	10.29	34.60	39.49	11.03	39.41	34.42	11.26	100.00	41.74	34.24	10.76
Qwen3-235B-Thinking-2507	235 B	39.74	28.28	13.77	39.81	27.70	12.90	33.63	16.39	6.87	38.37	26.79	5.64	100.00	38.82	26.33	11.14
Kimi-K2-Instruct	1 T	40.54	26.73	7.50	43.52	28.62	9.07	38.00	33.55	11.08	41.85	24.68	4.92	96.61	41.95	28.14	8.27
Qwen3-235B non-thinking	235 B	32.34	41.55	8.17	32.40	38.50	7.22	26.84	33.83	7.50	32.43	36.55	7.11	100.00	31.73	38.26	7.44
Qwen3-235B thinking	235 B	30.94	29.64	5.60	34.67	31.41	9.61	32.25	17.58	6.07	33.71	39.17	7.19	100.00	33.42	30.60	7.91
Qwen3-VL-235B	235B	32.69	26.13	7.87	37.16	27.31	11.76	22.45	18.17	4.46	32.88	25.52	10.99	99.15	33.74	25.67	9.94
Qwen3-Next-80B-Instruct	80 B	39.89	18.19	7.27	42.62	24.57	9.99	32.57	7.39	2.86	29.20	27.27	3.83	99.15	38.57	21.64	7.53
Qwen3-Next-80B-Thinking	80 B	35.79	33.57	10.23	34.09	35.97	13.33	33.08	33.23	10.77	28.02	29.84	5.40	99.15	33.30	34.10	11.03
Llama-3.3-70B	70 B	8.04	24.56	2.40	11.57	23.49	2.61	15.03	12.85	4.46	13.75	24.08	2.89	100.00	11.60	22.55	2.83
Llama-3.1-70B	70 B	9.22	16.37	0.73	12.55	23.24	1.19	16.12	14.03	0.00	10.19	15.03	1.50	99.15	11.87	19.30	1.01
Qwen2.5-72B	72 B	29.93	41.66	8.04	33.21	38.81	8.45	17.75	33.06	4.14	25.46	37.38	5.83	97.46	29.37	38.49	7.41
Qwen3-30B-Instruct-2507	30 B	29.99	23.53	6.07	32.41	24.74	5.94	18.44	17.25	2.86	24.59	26.77	7.98	100.00	28.91	23.94	5.95
Qwen3-30B-Thinking-2507	30 B	34.21	18.92	6.93	29.87	22.95	10.08	22.46	13.25	4.37	28.87	12.84	3.40	100.00	29.74	19.23	7.60
Qwen3-4B-Instruct-2507	4 B	20.70	15.14	7.47	19.30	8.09	1.45	12.83	6.54	3.73	25.94	7.74	1.94	100.00	19.95	9.34	3.08
Medical LLMs																	
Baichuan-M2-32B	32 B	29.79	32.13	8.77	36.31	36.72	9.23	36.02	26.44	7.40	30.73	33.46	2.46	79.66	33.95	33.98	7.77
MedGemma-27B-IT	27 B	31.75	10.31	2.93	32.45	20.49	4.54	28.85	15.47	1.69	23.35	24.68	3.08	97.46	30.33	18.45	3.61
Ours																	
CalcMate-4B	4 B	73.90	78.19	20.14	54.00	69.74	14.05	49.08	76.87	11.76	49.94	61.93	13.70	**100**	56.97	71.06	15.02
CalcMate-30B	30 B	73.60	73.99	20.95	62.76	76.14	20.73	60.97	75.48	17.95	67.57	74.38	18.46	**100**	**65.66**	**75.31**	**20.07**

Table 2: **Model Performance on MedMCP-Calc.** The evaluation encompassed model performance across four clinical domains (*Critical Care & Perioperative Medicine*, *Internal Medicine & Organ Systems*, *Neurology & Psychiatry*, and *Special Populations & Universal Tools*) and three evaluation metrics (Calculator Selection, Evidence Acquisition, and Quantitative Precision), along with the overall performance with Task Fulfillment. Best results are highlighted in **bold** (best) and underlined (second-best) under two settings, all models and open-source models only.

We adopted the widely recognized ReAct agent framework (Yao et al., 2022) for evaluation and employed DeepSeek-V3.1 for structured extraction of results. GPT-OSS (OpenAI, 2025a) and MedGemma-4B-IT were excluded from this experiment due to poor instruction-following capabilities—GPT-OSS struggled to adhere to the ReAct framework, consistent with findings in MCP-Universe (Luo et al., 2025), while MedGemma-4B-IT showed similar limitations likely attributable to its smaller model size. Additionally, we included our fine-tuned model, **CalcMate**, derived from Qwen3-4B-Instruct-2507 through specialized training on medical workflow scenarios with enhanced tool-use capabilities.

4.2 Main Results

Table 2 presents the main evaluation results of LLMs on MedMCP-Calc. Based on these results, we draw the following conclusions. We present only the main findings here; further detailed analysis can be found in Appendix B.1.

Overall Observation. Regarding the Task Fulfillment, experimental results confirm that mainstream models have achieved maturity in instruction-following capabilities, with near-perfect task fulfillment after minimal post-processing, demonstrating readiness for multi-turn ReAct-based interactions in complex clinical scenarios.

Proprietary models outperform open-source counterparts. Claude Opus 4.5 achieves the best overall performance, ranking first in both CS and QP and demonstrating exceptional capabilities across all metrics. Claude Sonnet 4.5 also shows competitive results with strong comprehensive task-handling ability. GPT-5 and Gemini-3-Pro follow closely but slightly lag behind. Among open-source models, GLM-4.7 delivers impressive results, closely trailing proprietary models on CS and EA while even surpassing Gemini-2.5-Pro on EA. DeepSeek-V3.1 also demonstrates robust performance.

Models struggle with calculator selection in fuzzy questions. Although Claude Opus 4.5 (66.66) and Claude Sonnet 4.5 (65.6) achieve strong Calculator Selection scores alongside GPT-5 (59.81), most other models—including medical-specialized ones—cluster around 30. This reflects two key factors. First, calculators have been overlooked in certain model development, leaving specialized models like MedGemma with limited exposure to calculator-related content. Second, real-world calculator scenarios pose additional challenges: while Baichuan-M2 achieves respectable CS performance, its low TF score reveals difficulties with agent frameworks requiring tool interaction and long-context reasoning. Top-performing models demonstrate more consistent behavior, though with notable deviations on less common calculators.

Models exhibit poor performance in iterative database interactions. Models are required to autonomously compose SQL queries to retrieve patient data, yet they exhibit poor planning capabilities and low robustness when handling complex task intents. To enhance SQL robustness, we incorporated `list_tables` and `describe_tables` tools into the experimental setup, as all models—particularly smaller ones—exhibit hallucination without explicit schema inspection: fabricating queries based on unfounded assumptions and persisting in erroneous approaches despite failure signals. This behavior may stem from the disproportionate representation of certain database schemas in training corpora, which biases models' default assumptions.

LLMs show marked reluctance to leverage external tools for numerical computation. Success requires models to select the correct calculator, retrieve accurate data, comprehend computation rules, and perform numerical calculations. The suboptimal results stem from multiple factors. First, limited performance on CS and EA creates compounding effects. Second, models fail to reliably internalize calculator-specific rules; while high-performing models handle common calculators adequately, performance degrades substantially for less frequent ones and among other models. Third, models rarely invoke external tools or consult calculator references, instead relying predominantly on internal knowledge. Fourth, computational hallucinations occur across small and medium-sized models, with smaller models exhibiting frequent arithmetic errors. And despite these limitations, models show minimal inclination to leverage Python tools for computational assistance.

Model	SQL	Python	Search	Fetch
Claude Opus 4.5	5.19	3.93	0.08	0.08
Claude Sonnet 4.5	5.16	1.68	0.1	0.08
GPT-5	3.85	0.06	0.25	0.25
Gemini-3-Pro	4.41	0.65	0.92	0.14
Gemini-2.5-Pro	4.45	0.99	0.14	0.14
GLM-4.7	4.31	0.82	0.18	0.2
DeepSeek-V3.1	3.71	0.94	0.64	0.04
Qwen3-235B-Thinking-2507	4.08	0.13	0.01	0
Qwen3-235B-Instruct-2507	3.92	0.53	0	0
Kimi-K2-Instruct	2.61	0.7	0.09	0.02
Qwen3-235B thinking	2.94	0.22	0.2	0.2
Qwen3-235B non-thinking	3.16	0.84	0.06	0.02
Qwen3-Next-80B-Thinking	4.05	0.07	0.07	0.03
Qwen3-Next-80B-Instruct	5.18	0.39	0.1	0.01
Llama-3.3-70B	3.5	2.45	0.02	0
Llama-3.1-70B	3.42	3.41	0.08	0
Qwen3-30B-Thinking-2507	2.81	0.18	0.02	0
Qwen3-30B-Instruct-2507	3.81	0.93	0.07	0.03
Qwen3-4B-Instruct-2507	1.9	0.13	0.02	0.01
MedGemma-27B-IT	3.54	0.27	0.19	0
Baichuan-M2-32B	4.46	0.24	0.03	0.04
CalcMate-4B	7.97	7.32	8.16	8.07
CalcMate-30B	8.34	8.21	8.49	8.42

Table 3: Comparison of average tool usage frequency and types across different models.

Extended thinking improves computational precision but may impair tool utilization. All thinking models achieve higher QP than non-thinking counterparts, as deliberation enhances numerical calculation accuracy. However, thinking mode degrades EA (e.g., Qwen3-235B: 38.26 to 30.60), likely due to over-reliance on internal knowledge rather than external tools. Dedicated thinking models (e.g., Qwen3-Next-80B) show larger but less balanced gains compared to hybrid models.

Domain Variability. Performance varies across clinical domains, with all models achieving higher scores in *Critical Care & Perioperative Medicine* and *Internal Medicine & Organ Systems* than in *Neurology & Psychiatry* and *Special Populations & Universal Tools*. The latter domains involve complex scoring systems and cognitive scales that prove more challenging than standardized physiological formulas, with QP scores often falling below 10% even for top-performing models.

4.3 CalcMate

Through experimental analysis, we identify two primary limitations affecting model performance: (1) unfamiliarity with clinical decision-making and planning, and (2) limited tool utilization within agent frameworks. To address these limitations, we

investigate whether fine-tuning with scenario planning and tool augmentation can more effectively solve calculator tasks in clinical settings.

Scenario Planning. Experiments reveal that most models, except top-tier ones, fail to plan and execute tasks step by step when facing complex clinical workflows. This limitation likely stems from insufficient reasoning capabilities and limited exposure to complex clinical scenarios. To address this, we leverage existing calculators and use `GPT-5` to construct 1,000 scenario tasks and generate the corresponding global planning reasoning.

Tool Augmentation. Experiments reveal a pronounced tendency toward tool avoidance across all models. However, tool usage is crucial for mitigating knowledge and computational hallucinations, particularly for smaller models. Therefore, we adopt an aggressive tool augmentation strategy. Each calculator computation follows the ReAct format, incorporating complete calculator reference operations, database exploration and querying, and code-assisted computation. While `run_sql` data is generated by `GPT-5`, all other steps are constructed using `Qwen3-235B-A22B-Instruct-2507-FP8`.

Evaluation Results. CalcMate achieves comprehensive improvements across all metrics. CS gains validate scenario planning for superior task decomposition; EA improvements (9.34 to 71.06) confirm the effectiveness of our tool augmentation strategy; QP gains demonstrate code-assisted tools' contribution. Notably, CalcMate achieves SOTA among open-source models and surpasses both source models used for training data construction, confirming that gains stem from methodological advances rather than data distillation alone.

4.4 Tool Utilization

To comprehensively analyze tool utilization strategies across models and examine the relationship between tool usage and task performance in clinical calculator scenarios, we investigated the usage patterns of primary functional tools (`run_sql`, `run_python`, `search`, and `fetch`) during task execution, as presented in Table 3. We present only the main findings here; further detailed analysis can be found in Appendix B.2.

Current models exhibit conservative tool utilization: while SQL invocation is relatively frequent (3–5 calls) due to data retrieval requirements, auxiliary tools are significantly underutilized—Python averages below 1 call for most models, with notable exceptions like Claude Opus 4.5 (3.93), and Search/Fetch usage approaches zero for several models (e.g., Qwen3-235B-Instruct-2507 records zero for both).

Tool usage correlates with specific performance metrics: SQL frequency positively correlates with EA (e.g., Gemini-3-Pro: SQL=4.41, EA=55.45), while Python usage associates with QP improvements (e.g., Gemini-2.5-Pro: Python=0.99, QP=20.77). Notably, thinking models reduce tool invocation without performance degradation, suggesting enhanced reasoning partially compensates for reduced tool usage.

Our tool augmentation strategy yields substantial gains. CalcMate exhibits dramatically higher utilization across all tools compared to baselines, which show pronounced tool avoidance. This tool engagement directly improves performance.

5 Conclusion

We introduce MedMCP-Calc, the first benchmark for evaluating LLMs on realistic medical calculator tasks through MCP tool integration, featuring fuzzy task descriptions, interactive EHR databases, and multi-step clinical scenarios across 118 tasks spanning 4 clinical domains. Our evaluation of 23 models reveals critical limitations: even Claude Opus 4.5 achieves only 33.92% calculation accuracy, with models exhibiting knowledge hallucinations, computational errors, and tool avoidance tendencies. We develop CalcMate through scenario planning training and tool augmentation, achieving state-of-the-art performance and demonstrating that targeted training on clinical workflow planning effectively bridges the gap between current capabilities and real-world demands. We hope this work advances both clinical AI applications and model development.

Limitations

Despite its rigorous design and careful consideration of clinical task characteristics, MedMCP-Calc has several limitations. First, while the benchmark references real hospital EHR systems to simulate authentic clinical environments, the current data association patterns may exhibit region-specific biases and cannot fully capture the heterogeneity of EHR systems across diverse global healthcare settings. Second, although CalcMate's "proactive tool augmentation" strategy yields substantial performance improvements, it significantly increases context length, resulting in elevated inference latency

and computational overhead. Future work could explore optimization techniques such as knowledge distillation or long-context compression to enhance efficiency, particularly for deployment in resource-constrained clinical environments.

Ethical Considerations

This study presents MedMCP-Calc, a benchmark for evaluating LLMs on medical calculator tasks in simulated clinical environments. Patient data were derived from MIMIC-IV, a publicly available, de-identified dataset compliant with HIPAA regulations and approved by an institutional review board. Synthesized records were generated to supplement task-specific scenarios, containing no real patient identifiers. We acknowledge that LLMs may exhibit knowledge hallucinations and computational errors in clinical calculations, as our experimental results demonstrate. The observed tool avoidance tendencies and calculation inaccuracies highlight the risks of deploying such systems without appropriate safeguards. MedMCP-Calc serves as an academic evaluation framework for assessing medical AI capabilities in calculator-based clinical workflows, not as a deployable clinical decision support system. Any practical application of models evaluated or trained on this benchmark requires rigorous clinical validation with human oversight. Decision-making authority must remain with qualified healthcare professionals, with model outputs serving only as assistive references subject to professional verification.

References

Anthropic. 2024. Introducing computer use, a new claude 3.5 sonnet, and claude 3.5 haiku. Accessed: 2025-06-30.

Anthropic. 2024. Introducing the model context protocol.

Anthropic. 2024. Introducing the model context protocol. Accessed: 2025-06-30.

Anthropic. 2025a. Claude sonnet 4.5.

Anthropic. 2025b. Introducing claude 4. Accessed: 2025-07-28.

Anthropic. 2025c. Introducing claude opus 4.5.

Kedi Chen, Zhikai Lei, Xu Guo, Xuecheng Wu, Siyuan Zeng, Jianghao Yin, Yinqi Zhang, Qin Chen, Jie Zhou, Liang He, and 1 others. 2025a. Code-driven number sequence calculation: Enhancing the inductive reasoning abilities of large language models. *arXiv preprint arXiv:2510.14620*.

Kedi Chen, Dezhao Ruan, Yuhao Dan, Yaoting Wang, Siyu Yan, Xuecheng Wu, Yinqi Zhang, Qin Chen, Jie Zhou, Liang He, and 1 others. 2025b. A survey of inductive reasoning for large language models. *arXiv preprint arXiv:2510.10182*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Jean-Philippe Corbeil, Asma Ben Abacha, George Michalopoulos, Phillip Swazinna, Miguel Del-Agua, Jérôme Tremblay, Akila Jeeson Daniel, Cari Bader, Kevin Cho, Pooja Krishnan, and 1 others. 2025. Empowering healthcare practitioners with language models: Structuring speech transcripts in two real-world clinical applications. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 859–870.

Google Deepmind. 2025. Gemini 3 pro.

DeepSeek-AI. 2024. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Shuai Dong, Siyuan Wang, Xingyu Liu, and Zhongyu Wei. 2025a. Interleaved latent visual reasoning with selective perceptual modeling. *ArXiv*, abs/2512.05665.

Shuai Dong, Jie Zhang, Guoying Zhao, Shiguang Shan, and Xilin Chen. 2025b. Semantic mismatch and perceptual degradation: A new perspective on image editing immunity. *ArXiv*, abs/2512.14320.

Chengfeng Dou, Chong Liu, Fan Yang, Fei Li, Jiyuan Jia, Mingyang Chen, Qiang Ju, Shuai Wang, Shunya Dang, Tianpeng Li, and 1 others. 2025. Baichuan-m2: Scaling medical capability with large verifier system. *arXiv preprint arXiv:2509.02208*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Shengda Fan, Xin Cong, Yuepeng Fu, Zhong Zhang, Shuyan Zhang, Yuanwei Liu, Yesai Wu, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2024. Workflowllm: Enhancing workflow orchestration capability of large language models. *arXiv preprint arXiv:2411.05451*.

Xuanqi Gao, Siyi Xie, Juan Zhai, Shiqing Ma, and Chao Shen. 2025. Mcp-radar: A multi-dimensional benchmark for evaluating tool use capabilities in large language models. *arXiv preprint arXiv:2505.16700*.

Alex J Goodell, Simon N Chu, Dara Rouholiman, and Larry F Chu. 2023. Augmentation of chatgpt with clinician-informed tools improves performance on medical calculation tasks. *MedRxiv*, pages 2023–12.

Zikang Guo, Benfeng Xu, Chiwei Zhu, Wentao Hong, Xiaorui Wang, and Zhendong Mao. 2025. Mcp-agentbench: Evaluating real-world language agent performance with mcp-mediated tools. *arXiv preprint arXiv:2509.09734*.

Mohammed Mehedi Hasan, Hao Li, Emad Fallahzadeh, Gopi Krishnan Rajbahadur, Bram Adams, and Ahmed E Hassan. 2025. Model context protocol (mcp) at first glance: Studying the security and maintainability of mcp servers. *arXiv preprint arXiv:2506.13538*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Rick Hightower. 2025. Mcp the usb-c for ai.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, and 1 others. 2023. Metagpt: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*.

Xinyi Hou, Yanjie Zhao, Shenao Wang, and Haoyu Wang. 2025. Model context protocol (mcp): Landscape, security threats, and future research directions. *arXiv preprint arXiv:2503.23278*.

Zixuan Huang, Yikun Ban, Lean Fu, Xiaojie Li, Zhongxiang Dai, Jianxin Li, and Deqing Wang. 2025. Adaptive batch-wise sample scheduling for direct preference optimization. *arXiv preprint arXiv:2506.17252*.

Zixuan Huang, Xin Xia, Yuxi Ren, Jianbin Zheng, Xuanda Wang, Zhixia Zhang, Hongyan Xie, Songshi Liang, Zehao Chen, Xuefeng Xiao, and 1 others. 2026a. Does your reasoning model implicitly know when to stop thinking? *arXiv preprint arXiv:2602.08354*.

Zixuan Huang, Xin Xia, Yuxi Ren, Jianbin Zheng, Xuefeng Xiao, Hongyan Xie, Li Huaqiu, Songshi Liang, Zhongxiang Dai, Fuzhen Zhuang, and 1 others. 2026b. Real-time aligned reward model beyond semantics. *arXiv preprint arXiv:2601.22664*.

Hongrui Jia, Jitong Liao, Xi Zhang, Haiyang Xu, Tianbao Xie, Chaoya Jiang, Ming Yan, Si Liu, Wei Ye, and Fei Huang. 2025. Osworld-mcp: Benchmarking mcp tool invocation in computer-use agents. *arXiv preprint arXiv:2510.24563*.

Qiao Jin, Zhizheng Wang, Yifan Yang, Qingqing Zhu, Donald Wright, Thomas Huang, Nikhil Khandekar, Nicholas Wan, Xuguang Ai, W John Wilbur, and 1 others. 2025. Agentmd: Empowering language agents for risk prediction with large-scale clinical tool learning. *Nature Communications*, 16(1):9377.

Nikhil Khandekar, Qiao Jin, Guangzhi Xiong, Soren Dunn, Serina Applebaum, Zain Anwar, Maame Sarfo-Gyamfi, Conrad Safranek, Abid Anwar, Andrew Zhang, and 1 others. 2024. Medcalc-bench: Evaluating large language models for medical calculations. *Advances in Neural Information Processing Systems*, 37:84730–84745.

Fei Lei, Yibo Yang, Wenxiu Sun, and Dahua Lin. 2025. Mcpverse: An expansive, real-world benchmark for agentic tool use. *arXiv preprint arXiv:2508.16260*.

Chenglin Li, Feng Han, Feng Tao, Ruilin Li, Qianglong Chen, Jingqi Tong, Yin Zhang, and Jiaqi Wang. 2025. Videopro: Adaptive program reasoning for long video understanding.

Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. 2023. Api-bank: A comprehensive benchmark for tool-augmented llms. *arXiv preprint arXiv:2304.08244*.

Zhiwei Liu, Jielin Qiu, Shiyu Wang, Jianguo Zhang, Zuxin Liu, Roshan Ram, Haolin Chen, Weiran Yao, Shelby Heinecke, Silvio Savarese, and 1 others. 2025. Mcpeval: Automatic mcp-based deep evaluation for ai agent models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 373–402.

Zhizhao Liu, Zhihua Wen, Zhiliang Tian, Zhen Huang, Miaorong Zhu, Zimian Wei, Yifu Gao, Liang Ding, and Dongsheng Li. 2026. Rethinking the hidden risk of reranking: Achieving risk-aware reranking with

information gain for rag with llms. In *Proceedings of the ACM Web Conference 2026*, pages 1887–1898.

Ziyang Luo, Zhiqi Shen, Wenzhuo Yang, Zirui Zhao, Prathyusha Jwalapuram, Amrita Saha, Doyen Sahoo, Silvio Savarese, Caiming Xiong, and Junnan Li. 2025. Mcp-universe: Benchmarking large language models with real-world model context protocol servers. *arXiv preprint arXiv:2508.14704*.

Kanato Masayoshi, Masahiro Hashimoto, Ryoichi Yokoyama, Naoki Toda, Yoshifumi Uwamino, Shogo Fukuda, Ho Namkoong, and Masahiro Jinzaki. 2025. Ehr-mcp: Real-world evaluation of clinical information retrieval by large language models via model context protocol. *arXiv preprint arXiv:2509.15957*.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. Gaia: a benchmark for general ai assistants. In *The Twelfth International Conference on Learning Representations*.

Guozhao Mo, Wenliang Zhong, Jiawei Chen, Xuanang Chen, Yaojie Lu, Hongyu Lin, Ben He, Xianpei Han, and Le Sun. 2025. Livemcpbench: Can agents navigate an ocean of mcp tools? *arXiv preprint arXiv:2508.01780*.

OpenAI. 2025a. gpt-oss-120b gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

OpenAI. 2025b. Introducing gpt-5.

OpenAI. 2025. Introducing operator: An agent that can use a computer. The underlying model is the Computer-Using Agent (CUA).

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E Gonzalez. The berkeley function calling leaderboard (bfcl): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*.

Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. 2024. Agent q: Advanced reasoning and learning for autonomous ai agents. *arXiv preprint arXiv:2408.07199*.

Shuofei Qiao, Runnan Fang, Zhisong Qiu, Xiaobin Wang, Ningyu Zhang, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. 2024. Benchmarking agentic workflow generation. *arXiv preprint arXiv:2410.07869*.

Shengqian Qin, Yakun Zhu, Linjie Mu, Shaoting Zhang, and Xiaofan Zhang. 2025. Meta-tool: Unleash open-world function calling capabilities of general-purpose large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 30653–30677.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, and 1 others. 2023. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*.

Qwen Team. 2025. Qwen3 technical report. Technical Report; arXiv preprint.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36:68539–68551.

Andrew Sellergren, Sahar Kazemzadeh, Tiam Jaroensri, Atilla Kiraly, Madeleine Traverse, Timo Kohlberger, Shawn Xu, Fayaz Jamil, Cían Hughes, Charles Lau, and 1 others. 2025. Medgemma technical report. *arXiv preprint arXiv:2507.05201*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652.

GLM Team, Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, Kedong Wang, Lucen Zhong, Mingdao Liu, Rui Lu, Shulin Cao, Xiaohan Zhang, Xuancheng Huang, Yao Wei, and 152 others. 2025a. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *Preprint*, arXiv:2508.06471.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, and 1 others. 2025b. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*.

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

Qwen Team. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Dianyi Wang, Wei Song, Yikun Wang, Siyuan Wang, Kaicheng yu, Zhongyu Wei, and Jiaqi Wang. 2025a. Autoregressive semantic visual reconstruction helps vlms understand better. *ArXiv*, abs/2506.09040.

Junyang Wang, Haiyang Xu, Jiabo Ye, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. 2024. Mobile-agent: Autonomous multi-modal mobile device agent with visual perception. *arXiv preprint arXiv:2401.16158*.

Zhenting Wang, Qi Chang, Hemani Patel, Shashank Biju, Cheng-En Wu, Quan Liu, Aolin Ding, Alireza Rezazadeh, Ankit Shah, Yujia Bao, and 1 others. 2025b. Mcp-bench: Benchmarking tool-using llm agents with complex real-world tasks via mcp servers. *arXiv preprint arXiv:2508.20453*.

Shaohang Wei, Wei Li, Feifan Song, Wen Luo, Tianyi Zhuang, Haochen Tan, Zhijiang Guo, and Houfeng Wang. 2025. TIME: A multi-level benchmark for temporal reasoning of llms in real-world scenarios. *arXiv preprint arXiv:2505.12891*.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, and 1 others. 2024. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First Conference on Language Modeling*.

Zijian Wu, Xiangyan Liu, Xinyuan Zhang, Lingjun Chen, Fanqing Meng, Lingxiao Du, Yiran Zhao, Fanshi Zhang, Yaoqi Ye, Jiawei Wang, and 1 others. 2025. Mcpmark: A benchmark for stress-testing realistic and comprehensive mcp use. *arXiv preprint arXiv:2509.24002*.

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

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, and 1 others. 2024b. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 37:52040–52094.

John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. 2024. Swe-agent: Agent computer interfaces enable software engineering language models. *arXiv preprint arXiv.2405.15793*.

Yingxuan Yang, Huacan Chai, Yuanyi Song, Siyuan Qi, Muning Wen, Ning Li, Junwei Liao, Haoyi Hu, Jianghao Lin, Gaowei Chang, and 1 others. 2025. A survey of ai agent protocols. *arXiv preprint arXiv:2504.16736*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.

Hongmei Zhang, Songshi Liang, Luke A Matkovic, Shadab Momin, Kai Wang, Xiaofeng Yang, and Michael F Insana. 2023. Deep q-learning to globally optimize a kd parameter search for medical imaging. *Quantitative Imaging in Medicine and Surgery*, 13(8):4879.

Jie Zhang, Shuai Dong, Shiguang Shan, and Xilin Chen. 2025a. Dual attention guided defense against malicious edits. *ArXiv*, abs/2512.14333.

Yunyan Zhang, Zhihong Zhu, and Xian Wu. 2025b. Cmedcalc-bench: A fine-grained benchmark for chinese medical calculations in llm. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 25661–25670.

Zhixia Zhang, Zixuan Huang, Xin Xia, Deqing Wang, Fuzhen Zhuang, Shuai Ma, Ning Ding, Yaodong Yang, Jianxin Li, and Yikun Ban. 2026. Heterogeneous agent collaborative reinforcement learning. *arXiv preprint arXiv:2603.02604*.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. 2024a. Gpt-4v (ision) is a generalist web agent, if grounded. *arXiv preprint arXiv:2401.01614*.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Livia Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, and 1 others. 2024b. Sglang: Efficient execution of structured language model programs. *Advances in neural information processing systems*, 37:62557–62583.

Yakun Zhu, Shaohang Wei, Xu Wang, Kui Xue, Shaoting Zhang, and Xiaofan Zhang. 2025. Menti: Bridging medical calculator and llm agent with nested tool calling. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5097–5116.

A Experiments Details

A.1 Models

Table 4 provides the full names, abbreviations, and references for all models evaluated in our experiments.

Closed-source Models. For proprietary models, we conducted all evaluations through their official APIs provided by the respective vendors. We used the default hyperparameters unless otherwise specified.

Open-source Models. For open-source models, we deployed all models using SGLang (Zheng et al., 2024b) as the inference framework. The experiments were conducted on servers equipped with 1-32 NVIDIA H100 GPUs or 1-8 NVIDIA H200 GPUs, depending on the model size and memory requirements.

A.2 Metrics

We evaluate both final outputs and intermediate processes through four complementary metrics:

Task Fulfillment (TF) measures the percentage of successfully completed quantitative assessments, reflecting the model's end-to-end capability to complete valid clinical calculation tasks.

$$\text{TF} = \frac{N_{\text{success}}}{N_{\text{total}}} \times 100\%$$

where N_{success} denotes the number of tasks with valid final outputs, and N_{total} represents the total number of evaluation tasks.

Calculator Selection (CS) measures the percentage of appropriate calculator choices, evaluating whether the model can correctly identify the required medical calculator based on clinical context.

$$\text{CS} = \frac{1}{N_{\text{total}}} \sum_{i=1}^{N_{\text{total}}} \frac{N^i_{\text{correct\_calc}}}{N^i_{\text{calc}}} \times 100\%$$

where i represents the i-th task, $N^i_{\text{correct\_calc}}$ indicates the number of correctly selected calculators in the i-th task, and N^i_{calc} represents the number of calculators required in the i-th task.

Quantitative Precision (QP) measures the numerical accuracy of computed results, assessing the model's ability to perform precise calculations.

$$\text{QP} = \frac{1}{N_{\text{total}}} \sum_{i=1}^{N_{\text{total}}} [\frac{1}{N^i_{\text{calc}}} \sum_{j=1}^{N^i_{\text{calc}}} \mathbb{1}(|y_{i,j} - \hat{y}_{i,j}| \leq \epsilon)]$$

where $y_{i,j}$ and $\hat{y}_{i,j}$ denote the ground-truth and predicted values of the j-th calculator in the i-th task, respectively, and ϵ is the tolerance threshold for numerical equivalence.

Evidence Acquisition (EA) assesses the LLM's capability to proactively acquire relevant data from databases, measuring information retrieval completeness.

$$\text{EA} = \frac{1}{N_{\text{total}}} \sum_{i=1}^{N_{\text{total}}} [\frac{|E^{\text{extracted}}_i \cap E^{\text{gt}}_i|}{|E^{\text{gt}}_i|}] \times 100\%$$

where E^{gt}_i represents the ground-truth set of required patient data inputs for task i, and $E^{\text{extracted}}_i$ denotes the set of data fields successfully extracted by the model.

A.3 ReAct Configuration

All models are evaluated under the ReAct (Yao et al., 2022) agent framework. We adopt ReAct as it is the standard framework for MCP benchmarks, widely used in recent work such as MCP-Universe (Luo et al., 2025) and MCP-Bench (Wang et al., 2025b), which ensures direct comparability with the broader MCP community and isolates model capability from the confounds of framework design.

Setup. The maximum number of interaction rounds is set to 80. Each round consists of three steps: (1) **Thought**—the model analyzes the current state and reasons about the next action; (2) **Action**—the model issues an MCP tool call in JSON format with fields `server`, `tool`, and `arguments`; (3) **Observation**—the model receives the structured tool output and incorporates it into subsequent reasoning.

ReAct Prompt:

```
You are an expert Clinical Assessment Assistant
    and ReAct (Reasoning and Acting) agent.

Your primary role is to perform accurate
    medical calculations and clinical
    assessments based on patient-specific data
    retrieved from the EHR database.

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 below.
```

Table 4: Model information and abbreviations used in experiments.

Model Full Name	Abbreviation
Proprietary Models	
Claude Opus 4.5 (Anthropic, 2025c)	Claude Opus 4.5
Claude Sonnet 4.5 (Anthropic, 2025a)	Claude Sonnet 4.5
GPT-5 (OpenAI, 2025b)	GPT-5
Gemini-3-Pro-Preview (Deepmind, 2025)	Gemini-3-Pro
Gemini-2.5-Pro (Comanici et al., 2025)	Gemini-2.5-Pro
Open-Source Models	
GLM-4.7 (Team et al., 2025a)	GLM-4.7
DeepSeek-V3.1 (DeepSeek-AI, 2024)	DeepSeek-V3.1
Kimi-K2-Instruct (Team et al., 2025b)	Kimi-K2-Instruct
Qwen3-235B-A22B (Thinking) (Team, 2025)	Qwen3-235B thinking
Qwen3-235B-A22B (Non-Thinking) (Team, 2025)	Qwen3-235B non-thinking
Qwen3-235B-A22B-Thinking-2507 (Team, 2025)	Qwen3-235B-Thinking-2507
Qwen3-235B-A22B-Instruct-2507 (Team, 2025)	Qwen3-235B-Instruct-2507
Qwen3-VL-235B-A22B-Instruct (Team, 2025)	Qwen3-VL-235B
Qwen3-Next-80B-A3B-Instruct (Team, 2025)	Qwen3-Next-80B-Instruct
Qwen3-Next-80B-A3B-Thinking (Team, 2025)	Qwen3-Next-80B-Thinking
Llama-3.3-70B-Instruct (Dubey et al., 2024)	Llama-3.3-70B
Llama-3.1-70B-Instruct (Dubey et al., 2024)	Llama-3.1-70B
Qwen2.5-72B-Instruct (Team, 2024)	Qwen2.5-72B
Qwen3-30B-A3B-Instruct-2507 (Team, 2025)	Qwen3-30B-Instruct-2507
Qwen3-30B-A3B-Thinking-2507 (Team, 2025)	Qwen3-30B-Thinking-2507
Qwen3-4B-Instruct-2507 (Team, 2025)	Qwen3-4B-Instruct-2507
Baichuan-M2-32B (Dou et al., 2025)	Baichuan-M2-32B
MedGemma-27B-IT (Sellergren et al., 2025)	MedGemma-27B-IT

```
Here are the tools you have:
{{TOOLS_PROMPT}}


{% 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 in the JSON when
    you're confident you have sufficient
    information.
- The response must be in a valid JSON format.
    Only JSON output is required, with no
    additional content.
```

A.4 CalcMate Training

Training Data. The training set consists of 1,005 scenario tasks with a total of 9,097 calculator invocations (an average of 9.05 per task), spanning 637 unique calculators—54.31% coverage of the full 1,173-calculator pool used in MedMCP-Calc. Tasks are distributed across the four clinical domains as *Critical Care & Perioperative Medicine* (440), *Internal Medicine & Organ Systems* (345), *Neurology & Psychiatry* (66), and *Special Populations & Universal Tools* (154), providing balanced coverage of clinical workflows. Within each trajectory, `GPT-5` is used only to produce the global planning trace (approximately 10% of the data), while the remaining agent-execution content—`search`/`fetch` for calculator references, `describe_tables` and SQL queries for database exploration, and `run_python` for computation—is generated by `Qwen3-235B-A22B-Instruct-2507-FP8`. The training signal therefore primarily teaches systematic tool-use behavior rather than distilling GPT-5's medical knowledge, which is consistent with our observation that CalcMate surpasses GPT-5 by invoking external tools far more aggressively (cf. Python 8.21 vs. 0.06, Search 8.49 vs. 0.25 in Table 3) instead of relying on parametric knowledge.

Training Setup. CalcMate is obtained via full-parameter supervised fine-tuning on `Qwen3-4B-Instruct-2507` (CalcMate-4B) and `Qwen3-30B-A3B-Instruct-2507` (CalcMate-30B) using the LLaMA-Factory framework. We train for 1 epoch with a learning rate of 1×10^{-5}, a cosine learning-rate scheduler, and a warmup ratio of 0.1. The per-device batch size is 1 with 8 gradient accumulation steps. To accommodate the long multi-turn tool-use trajectories produced by our tool-augmentation strategy, we set the maximum sequence length to 150,000 tokens.

System and Hardware. Training uses bf16 mixed precision with FlashAttention-2 and gradient checkpointing for memory efficiency. Distributed training is conducted via DeepSpeed ZeRO Stage 3 with sequence parallelism (size = 8). CalcMate-4B is trained on 8× NVIDIA H100 GPUs, and CalcMate-30B on 32× NVIDIA H100 GPUs.

B Further Analysis

B.1 Main Results

Table 2 presents the main evaluation results of LLMs on MedMCP-Calc. In Section 4.2, we provide an analysis of the principal findings. In this appendix, we further extend our analysis through a detailed examination of more fine-grained model behaviors.

Overall Observation. Regarding the Task Fulfillment, experimental results indicate that mainstream models have achieved maturity in instruction-following capabilities. Although certain models (e.g., GPT-OSS) do not adhere to the ReAct framework, and others (e.g., Gemini-2.5-Pro) occasionally exhibit JSON format deviations, the vast majority achieve near-perfect task fulfillment after regular expression cleaning. This demonstrates that current models possess the fundamental capability for multi-turn interactions following the ReAct framework or tool-use protocols in complex clinical scenarios.

Regarding reasoning modes, extended thinking generally enhances computational precision; however, for agentic tasks and tool utilization, thinking models do not necessarily yield positive gains. Our experiments include the hybrid thinking model Qwen3-235B operating in different modes, along with three dedicated thinking models: Qwen3-235B-2507, Qwen3-Next-80B, and Qwen3-30B-2507. Results reveal that on Quantitative Precision, thinking models demonstrate a consistent advantage—all models achieve higher QP than their non-thinking counterparts, suggesting that extended deliberation enables more accurate handling of complex medical formula transformations and numerical calculations. Despite these improvements, all thinking models exhibit degradation in either CS or EA. For instance, the EA of Qwen3-235B decreases from 38.26 to 30.60 when the thinking mode is enabled, possibly reflecting a tendency to over-rely on internal knowledge during deep reasoning, thereby weakening external tool interaction capabilities. Furthermore, performance variance among hybrid thinking models is smaller than that observed in dedicated thinking models. When switching to thinking mode, Qwen3-235B exhibits more balanced yet conservative gains, with modest improvements in both Calculator Selection and Quantitative Precision. In contrast, dedicated thinking models such as Qwen3-Next-80B achieve more substantial improvements in both QP and EA.

Domain Variability. Performance varies notably across the four clinical domains. All models perform better in *Critical Care & Perioperative Medicine* and *Internal Medicine & Organ Systems* than in *Neurology & Psychiatry* and *Special Pop-*

ulations & Universal Tools. For instance, models such as GPT-5 and Gemini-3-Pro maintain relatively high CS and EA in the former two domains, likely due to the standardized nature of physiological parameters and diagnostic protocols. In contrast, *Neurology & Psychiatry* poses a notable challenge, particularly in QP. Even high-performing models such as Deepseek-V3.1 and GLM-4.7 show sharp QP declines in this domain, often falling below 10%. This suggests that clinical calculations involving complex scoring systems and cognitive scales are more difficult for LLMs to execute accurately than direct physiological formulas.

Our Model. CalcMate achieves comprehensive improvements across all metrics. Even at the 4B parameter scale, it substantially outperforms the base model Qwen3-4B-Instruct-2507 and approaching state-of-the-art performance among open-source models. CalcMate-30B further establishes a clear lead over open-source alternatives, surpassing or approaching Gemini-2.5-Pro.

First, CalcMate-4B surpasses all models except GPT-5 in CS, while CalcMate-30B even exceeds GPT-5, demonstrating the effectiveness of scenario planning training and indicating superior task planning capabilities for calculator utilization. Second, EA improves substantially from 9.34 to 71.06, with the model invoking the `run_sql` tool more than twice as frequently as other models, confirming that our aggressive tool augmentation strategy significantly enhances database interaction capability. Third, QP improves from 3.08 to 15.02—approaching GLM-4.7's 15.59—demonstrating the contribution of code-assisted tools. The relatively modest QP gains compared to CS and EA may reflect the inherent limitations of a 4B-parameter model in mathematical computation and coding; while code assistance improves calculation outcomes, these gains remain constrained by the model's fundamental capacity. The CalcMate-30B achieves substantially higher results (20.07), approaching proprietary model performance and validating that our method yields significant improvements in quantitative precision.

Notably, although our training data is constructed partially from GPT-5 and predominantly from Qwen3-235B-A22B-Instruct-2507-FP8, CalcMate surpasses both source models in performance, demonstrating that our improvements stem from methodological advances rather than inherited capabilities. Overall, these results validate our training strategy combining scenario planning with tool augmentation.

B.2 Tool Utilization

Table 3 presents tool usage patterns across different models. Section 4.4 analyzes the principal findings, while this appendix extends the analysis to more fine-grained model behaviors.

Current models generally exhibit a conservative tendency toward tool underutilization. While all models demonstrate relatively high SQL invocation frequency (averaging 3–5 calls)—driven by the inherent task requirement of retrieving patient data from databases—they significantly underutilize auxiliary tools. For instance, most models invoke the Python tool fewer than once on average, with Qwen3-Next-80B-Thinking and GPT-5 averaging merely 0.07 and 0.06 calls, respectively. Search and Fetch tool usage is even sparser: Qwen3-235B-Instruct-2507 records zero invocations for both, and Qwen3-VL-235B shows zero Fetch calls. Even DeepSeek-V3.1, which exhibits relatively superior performance, achieves only 0.64 Search invocations. These findings suggest that current models tend to rely on parametric knowledge rather than proactively leveraging external tools for information retrieval or numerical computation, reflecting conservative tool utilization strategies.

Model performance exhibits certain correlations with tool usage strategies, while thinking models reduce tool invocation without sacrificing performance. First, SQL invocation frequency demonstrates a positive correlation with the EA metric. High-EA models such as GPT-5 (SQL=3.85, EA=53.12) and Gemini-3-Pro (SQL=4.41, EA=55.45) maintain relatively high SQL invocation frequencies, whereas Llama-3.3-70B (SQL=3.50, EA=22.55) and Qwen3-4B-Instruct-2507 (SQL=1.90, EA=9.34) exhibit lower EA corresponding to reduced SQL usage. This pattern is intuitive, as SQL serves as the primary mechanism for data retrieval and database interaction. Second, Python invocation frequency is associated with the QP metric. Gemini-2.5-Pro (Python=0.99, QP=20.77) and Qwen2.5-72B (Python=1.85, QP=7.41) suggest that Python invocation contributes to improved computational accuracy. And, most models exhibit relatively low reliance on Search and Fetch tools. Notably, comparing thinking models with their non-thinking counterparts reveals that the former generally demonstrate reduced frequencies of Python and external tool invocations without significant performance

degradation. This suggests that enhanced internal reasoning capabilities can partially compensate for reduced tool usage—particularly for mathematical computations, where significantly reduced Python invocation does not compromise calculation performance.

Our tool augmentation strategy yields substantial performance gains. CalcMate-30B exhibits significantly higher tool utilization across all categories—SQL (8.34), Python (8.21), Search (8.49), and Fetch (8.42)—compared to baseline models, which show pronounced tool avoidance beyond basic SQL queries (e.g., Qwen3-235B-Instruct-2507: SQL 3.92, Python 0.53, Search 0, Fetch 0). This comprehensive tool engagement translates directly into performance improvements. Elevated SQL usage correlates with CalcMate's superior EA scores (75.31 vs. 34.24 for base Qwen3-235B), as thorough database exploration enables accurate clinical data extraction. Similarly, intensive Python utilization (8.21 vs. 0.53) corresponds to substantial QP gains (20.07 vs. 10.76), confirming that code-assisted computation effectively mitigates calculation errors. The dramatic increase in Search and Fetch operations (8.49/8.42 vs. near-zero for most baselines) demonstrates that our complete calculator reference pipeline substantially addresses knowledge hallucination—a critical limitation of smaller models. Therefore, even with only 4B parameters, CalcMate-4B achieves state-of-the-art performance among open-source models by leveraging tool assistance to reduce both computational and knowledge hallucinations, validating that systematic tool augmentation can compensate for limited model capacity in clinical calculator tasks.

Efficiency of Tool Engagement. A natural follow-up question is whether CalcMate's elevated tool-invocation counts reflect efficient behavior or wasteful looping on the same turns. Inspecting trajectories, we find that the base Qwen3-4B-Instruct-2507 frequently exhibits a trial-and-error pattern: it issues repeated near-duplicate SQL queries because of schema hallucination, guesses field names instead of calling `describe_tables`, performs shallow `search` without subsequent `fetch`, and tries to self-correct arithmetic errors through more internal reasoning rather than invoking `run_python`. These loops inflate round counts without producing useful evidence and often fail to converge. After tool-augmented training, CalcMate instead follows a structured execution pattern: when facing an unfamiliar calculator it first `search`es and then `fetch`es to acquire authoritative reference knowledge; it then calls `describe_tables` to confirm schema before issuing targeted SQL; and for complex arithmetic,, it systematically routes to `run_python` rather than relying on internal computation. The joint gains in EA and QP (Table 2) confirm that the increased tool frequency corresponds to purposeful, coordinated tool use rather than redundant looping, indicating that our augmentation teaches how to use tools rather than merely encouraging more calls.

B.3 Extra Calculators Found Analysis.

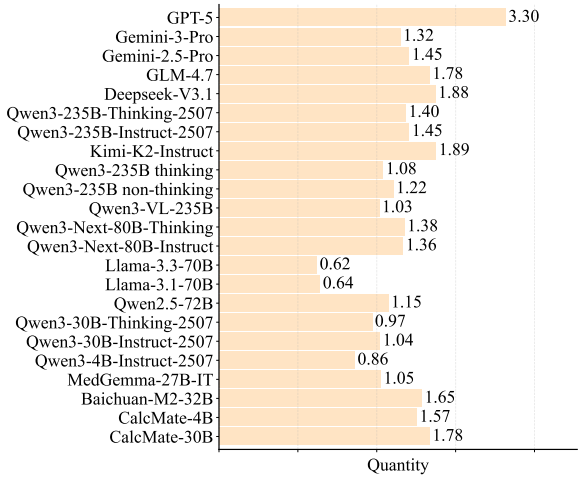

Figure 3: **Extra Calculators Found.**

During evaluation, we measure the average number of times a model selects calculators other than the ground truth calculator per task, as shown in Figure 3.

Overall distribution. For most models, the number of extra calculator invocations falls within a relatively narrow and stable range. On average, models trigger approximately one to two extra calculator calls per task, with only minor differences across models. For example, core models from the Gemini, Qwen, and DeepSeek families mostly exhibit extra invocation counts in the range of 1.0–1.8, indicating a restrained and balanced level of exploration when using calculator tools.

Outlier models and potential causes. GPT-5 exhibits an extra calculator invocation count of 3.3, making it the only model exceeding three calls and placing it significantly above all other models. In contrast, Llama-3.3-70B and Llama-3.1-70B show markedly lower values of 0.62 and 0.64, respectively, forming the lowest-performing cluster in this evaluation.

Given GPT-5's strong overall performance among proprietary models, its substantially higher extra invocation rate likely reflects a more proactive strategy of exploring multiple calculator constructions or computation paths during task execution. This behavior may help cover diverse task requirements and increase the likelihood of selecting the correct calculator. Although such an aggressive strategy may involve some degree of redundant or unnecessary attempts, GPT-5 still achieves competitive performance on CS, QP, and EA, suggesting that increased extra invocations do not come at the cost of computational accuracy or data extraction quality.

The Llama-3 series ranks near the bottom on both CS and QP. Its lower frequency of extra calculator invocations may indicate a stronger reliance on internal parametric knowledge or fixed reasoning patterns, with fewer explicit selections of medical calculators. Combined with the relatively high frequency of Python tool usage observed during evaluation, this suggests that Llama models tend to convert parts of the computation directly into code execution rather than explicitly constructing or switching among multiple calculator logics.

B.4 Failure Pattern Analysis

To understand where in the pipeline errors occur, we present a detailed failure analysis of a representative case — an ICU critical care scenario involving ARDS diagnosis, ventilator parameter optimization, pulmonary function assessment, and ECMO decision-making. In this case, GPT-5 successfully computed 2 of 10 ground-truth calculators (A-a O_2 Gradient = 434.9 and P/F Ratio = 85.0). The remaining failures fall into three categories:

Calculator Omission (4 calculators, ∼52.5% of errors). Despite being explicitly mentioned in the workflow, GPT-5 directly ignored four calculators: the Arterial Blood Gas Analyzer, Berlin Criteria for ARDS, Recruitment to Inflation Ratio, and PRESET Score. The first two are commonly used in ICU and emergency settings and are critical for ARDS scenarios. The omission likely stems from its limited five-step decision process, which lacks comprehensive planning and full workflow awareness. The latter two are relatively specialized, suggesting gaps in the model's domain-specific knowledge. Overall, the primary causes are the lack of global planning and knowledge omission.

Correct Calculator, Incorrect Result (3 calculators). This affected SpO_2/FiO_2 Ratio, Static Lung Compliance, and Murray Score. We identified three underlying causes: (1) the model simply forgot to perform the calculation despite strong context-handling capabilities; (2) most frequently, GPT-5 failed to retrieve the necessary values from the database — the model tends to query data for multiple calculators simultaneously, leading to missed values, and upon encountering missing data, it hallucinated that the data was unavailable rather than re-querying; (3) arithmetic errors, which were relatively rare for GPT-5 but occur frequently in smaller models. Additionally, during database interactions, the model often guessed field names instead of first performing exploratory queries — another form of overconfident hallucination. In summary, forgetting and hallucination are the main drivers of these failures.

Wrong Calculator Selection (1 calculator). The expected calculator was ETT Depth, but the model selected Tidal Volume instead. These two serve fundamentally different purposes; the model likely chose Tidal Volume due to lexical similarity while overlooking the key term "Depth." An incorrect ETT depth can result in single-lung ventilation or vocal cord injury, making this a clinically serious error.

Calculator omission accounts for approximately 52.5% of errors, while incorrect results and wrong selection together account for approximately 47.2%, with the remaining 0.3% attributed to other errors. How to enable LLM foundation models to better learn agentic workflow reasoning and action remains an important open question.

B.5 End-to-End Data Example

To illustrate how our benchmark evaluates complete model behavior—spanning fuzzy natural-language input, multi-step SQL retrieval, calculator selection, arithmetic, and final interpretation—we present one full data point corresponding to the failure analysis case in Appendix B.4. This task requires a 10-step clinical workflow for ARDS assessment and ventilator optimization, drawn from Critical Care & Perioperative Medicine.

Fuzzy Query. The task is presented as a natural-language physician query, deliberately omitting explicit calculator names. A condensed version is shown below; the full text is included in our released dataset:

```
I'm seeing an adult in the ICU with diffuse
    pneumonia who's sliding into acute
    hypoxemic respiratory failure, and I'm
```

```
worried this could be evolving ARDS. Oxygen
needs have been climbing over the past
several hours - I want a quick, bedside
sense of how bad the oxygenation really is
just from the pulse ox and the current
oxygen setting. Once we have the [arterial
blood] gas, I need to understand whether
this is mostly shunt or V/Q mismatch by
looking at the alveolar-arterial oxygen
difference. [Confirm ARDS,] intubate and
run a lung-protective strategy... look at
static compliance... formal bedside check
of recruitability... if severe hypoxemia
persists, open the ECMO conversation. I
need proper calculations on this.
```

This query requires the model to: (1) identify which calculators are needed, (2) retrieve patient data via SQL from an EHR-structured database, (3) execute each calculation, and (4) make data-driven clinical recommendations.

Patient Data. The benchmark provides an EHR database for a de-identified 58-year-old male patient (P006_efad2a48) admitted to the MICU for severe hypoxemic respiratory failure due to bilateral pneumonia (ICD-10: J80, J18.9). Key parameters stored across nine tables are summarized in Table 5.

Source	Parameter	Value
`vital_signs`	Height	178 cm (70 in)
	SpO_2	91%
	MAP	75 mmHg
`laboratory_result`	ABG: pH	7.31
	ABG: $PaCO_2$	54 mmHg
	ABG: PaO_2	68 mmHg
	ABG: HCO_3^-	26 mEq/L
	ABG: FiO_2	80%
	Lactate	3.8 mmol/L
	...	...

Table 5: Key patient parameters for the end-to-end example (subset).

Ground-Truth Annotation. The ground-truth defines a 10-step conditional workflow. Table 6 shows a subset of the steps.

Step	Calculator	Key Inputs	Expected
1	SpO_2/FiO_2 Ratio	SpO_2=91%, FiO_2=80%	113.8
2	ABG Analyzer	pH=7.31, $PaCO_2$=54, HCO_3^-=26	Resp. Acidosis
3	A-a O_2 Gradient	PaO_2=68, $PaCO_2$=54, FiO_2=80%	434.9 mmHg
	...(7 additional steps omitted for brevity)		

Table 6: Ground-truth workflow (subset) for the case.

GPT-5 Response. GPT-5 completed this task in 5 agent steps, using the `postgres_executor` MCP server for SQL access and Python for computation.

Step 1 — Schema exploration. The model first called `list_tables`, discovering 9 tables, then `describe_tables` to inspect column names and types. No patient data was retrieved yet.

Steps 2–3 — SQL data retrieval. The model issued a multi-table SQL query joining `vital_signs`, `laboratory_result`, `examination_report`, and `order_inpatient`. The initial ABG sub-query used exact lowercase matches and returned only Lactate—the ABG items were stored as `pH (arterial)`, `PaCO2`, etc., which did not match. The model recognized the failure, adapted its query to use `LIKE '%arterial%' OR ... LIKE '%blood gas%'`, and successfully retrieved the full ABG panel. Ventilator parameters (PEEP, P_{plat}, VT) were not found, as they are stored in free-text fields rather than structured columns.

Steps 4–5 — Calculator selection and computation. Using the retrieved data, the model computed the results shown in Table 7.

Calculator	Model Output	GT	Name	Result
P/F Ratio	85 mmHg	85.0	✔	✔
A-a O_2 Gradient	434.9 mmHg	434.9	✔	✔
SpO_2/FiO_2	*SpO_2 not found*	113.8	✔	✗
Berlin Criteria	Partial	Positive	✗	✗
ETT Depth & VT	No ETT depth	22 cm	✗	✗
Static Compliance	*not found*	24.4	✔	✗
R/I Ratio	*not attempted*	0.39	✗	✗
Murray Score	Partial	3.8	✔	✗
PRESET Score	*not attempted*	7	✗	✗
ABG Analyzer	*not performed*	Resp. Acid.	✗	✗
Total			**5/10**	**2/10**

Table 7: GPT-5 results on the ARDS case. ✔ = correct match; ✗ = mismatch or missing.

The model demonstrated accurate SQL schema exploration, adaptive query reformulation on failure, and correct arithmetic for the calculators it could complete. Its clinical narrative—recognizing severe ARDS, recommending low-tidal-volume ventilation, and flagging ECMO candidacy—was clinically appropriate despite incomplete computation.

C Implementation Details

C.1 Stage I Model Selection

Our data construction pipeline processes 1,970 initial calculators across 125 scenarios, involving substantial API calls. We selected models at each stage based on task characteristics and cost-performance trade-offs. Using `GPT-5` for all stages would increase the total cost by approximately $9\times$ and, being closed-source, would hinder community reproducibility. We therefore use well-established

open-source models for Stage I and validate that their outputs are statistically equivalent to those of `GPT-5.2`.

Calculator Deduplication with `GPT-OSS-120B`. This step is a semantic similarity judgment task, requiring the identification of functionally identical calculators with different names. `GPT-OSS-120B` demonstrates strong medical semantic understanding at far lower cost than `GPT-5`. To validate this choice, we sampled 100 positive and 100 negative pairs and re-ran them with `GPT-5.2` under identical settings. As shown in Table 8, the two models reach 95% overall agreement, with a Cohen's Kappa of 0.9 (where $\kappa > 0.8$ indicates near-equivalence). An MCC of 0.9045 and F1 of 0.9474 further corroborate the reliability of `GPT-OSS-120B` for this task and confirm its statistical equivalence with `GPT-5.2`.

	Positive	Negative	Overall
Consistent Rate	90%	100%	95%
Cohen's Kappa	–	–	0.9

Table 8: Agreement between `GPT-OSS-120B` and `GPT-5.2` on calculator deduplication (100 positive and 100 negative pairs).

Scenario Assignment with `DeepSeek-V3.1`. Assigning calculators to the 125 clinical scenarios is essentially a classification-matching task. We randomly sampled 600 calculator–scenario pairs and compared `DeepSeek-V3.1` with `GPT-5.2`, obtaining a Cohen's Kappa of 0.855 (> 0.8) and a Jaccard coefficient of 0.817, again confirming high agreement and statistical equivalence with `GPT-5.2`.

C.2 Result Extraction Validation

After each ReAct rollout, we use `DeepSeek-V3.1` to parse the agent's execution trajectory and extract the final calculator names, numerical results, and retrieved SQL fields into a standardized JSON format for automated evaluation. This step is a structured information extraction task that does not involve complex medical reasoning.

To validate this choice, we randomly sampled 20 tasks and re-ran the same extraction with `GPT-5.2` under identical prompts. Table 9 reports per-dimension agreement rates,, and Cohen's Kappa, and Table 10 reports cross-model correlations of per-task scores.

On the two dimensions most central to evaluation—calculator name recognition and calculator result matching—the two extractors reach near-perfect agreement ($\kappa > 0.84$). SQL field matching shows substantial agreement ($\kappa = 0.6454$), reflecting the higher granularity and occasional ambiguity of field-level judgments (e.g., synonymous column names). At the overall task-ranking level, Spearman $\rho = 0.874$ confirms that both extractors yield highly consistent model rankings. `DeepSeek-V3.1` is marginally stricter on average, but this bias is small and does not alter the ordering of evaluated models. These results support using `DeepSeek-V3.1` as a cost-effective and reproducible extractor in our evaluation pipeline.

Dimension	Nums.	Agreement	κ
Calculator Name	127	93.7%	0.8423
Calculator Result	127	92.1%	0.8403
SQL Field	652	82.5%	0.6454
Overall	906	**85.4%**	**0.7038**

Table 9: Per-dimension agreement between `DeepSeek-V3.1` and `GPT-5.2` on result extraction across 20 sampled tasks.

Dimension	Pearson r	Spearman ρ	MAD
Calculator Name	0.957	0.955	0.045
Calculator Result	0.919	0.864	0.061
SQL Field	0.565	0.558	0.116
Overall	**0.872**	**0.874**	**0.074**

Table 10: Cross-model correlation between per-task scores produced by `DeepSeek-V3.1` and `GPT-5.2`.

C.3 EHR Database

To ensure the assessment tasks are based on real-world clinical scenarios, we constructed a dataset containing nine core tables based on MIMIC-IV v3.1. The structure of each table is shown in Tables 11, 12, 13, 14, 15, 16, 17, 18 and 19.

Table 11: patient_information

Field Name	Type	Description
medical_institution_code	TEXT	Medical institution code
patient_id	TEXT	Primary key, unique patient identifier
campus_name	TEXT	Campus name of the medical institution
patient_name	TEXT	Patient's full name
gender	TEXT	Patient's gender
birth_date	DATE	Patient's date of birth
visit_card_no	TEXT	Internal visit card number
marriage_status	TEXT	Patient's marital status
ethnicity	TEXT	Patient's ethnic group
nationality	TEXT	Patient's nationality
abo_blood_type	TEXT	ABO blood group
rh_blood_type	TEXT	Rh blood group
source_system	TEXT	Source system of the data
is_valid	BOOLEAN	Validity indicator of the record

Table 12: visit_inpatient

Field Name	Type	Description
medical_institution_code	TEXT	Medical institution code
patient_id	TEXT	Patient identifier (NOT NULL)
visit_id	TEXT	Visit identifier (NOT NULL, PRIMARY KEY)
visit_type	TEXT	Visit type (reference code) (NOT NULL)
campus_name	TEXT	Campus name (NOT NULL)
inpatient_no	TEXT	Source inpatient number
visits_num	INTEGER	Visit count
patient_name	TEXT	Patient name (NOT NULL)
gender	TEXT	Gender (NOT NULL)
patient_age	INTEGER	Patient age
patient_age_unit	TEXT	Age unit
admission_time	TIMESTAMP	Admission time
admission_way	TEXT	Admission way e.g. outpatient, emergency
admission_dept_name	TEXT	Admission department name
admission_dept_code	TEXT	Admission department source code
admission_ward_name	TEXT	Admission ward name
admission_ward_code	TEXT	Admission ward source code
major_diagnosis	TEXT	Main diagnosis
bed_no	TEXT	Bed number
current_dept_name	TEXT	Current department name
current_dept_code	TEXT	Current department source code
current_ward_name	TEXT	Current ward name
current_ward_code	TEXT	Current ward source code
discharge_dept_name	TEXT	Discharge department name
discharge_dept_code	TEXT	Discharge department source code
discharge_ward_name	TEXT	Discharge ward name
discharge_ward_code	TEXT	Discharge ward source code
discharge_time	TIMESTAMP	Discharge time
create_time	TIMESTAMP	Record create time
update_time	TIMESTAMP	Record update time
is_valid	BOOLEAN	Is valid

Table 13: examination_report

Field Name	Type	Description
medical_institution_code	TEXT	Medical institution code
patient_id	TEXT	Patient identifier (NOT NULL)
visit_id	TEXT	Visit identifier (NOT NULL)
report_id	TEXT	Report identifier (NOT NULL, PRIMARY KEY)
campus_name	TEXT	Campus name (NOT NULL)
visit_type	TEXT	Visit type (NOT NULL)
patient_name	TEXT	Patient name (NOT NULL)
gender	TEXT	Gender (NOT NULL)
patient_age	INTEGER	Patient age
patient_age_unit	TEXT	Age unit
bed_no	TEXT	Bed number
clinic_diag_name	TEXT	Clinical diagnosis
medical_history	TEXT	Medical history summary
exam_type	TEXT	Examination type
exam_item_name	TEXT	Exam item name
exam_site	TEXT	Exam site
exam_method	TEXT	Exam method
exam_tech	TEXT	Exam technique
report_name	TEXT	Report name
enhance_scan_flag	TEXT	Enhancement flag
is_anesthesia	TEXT	Anesthesia flag
exam_time	TIMESTAMP	Exam time
exam_abnormal_flag	TEXT	Abnormal flag (1 normal, 2 abnormal, 3 uncertain)
exam_findings	TEXT	Findings
exam_conclusion	TEXT	Conclusion
radiology_note	TEXT	Radiology notes
intact_exam_report	TEXT	Full report text
summary_note	TEXT	Summary description
comment	TEXT	Comment
review_note	TEXT	Review notes
application_no	TEXT	Application number
review_time	TIMESTAMP	Review time
report_time	TIMESTAMP	Report time
report_state	TEXT	Report state (0 void, 1 normal)
create_time	TIMESTAMP	Create time
update_time	TIMESTAMP	Update time
is_valid	BOOLEAN	Is valid

Table 14: laboratory_result

Field Name	Type	Description
medical_institution_code	TEXT	Medical institution code
patient_id	TEXT	Patient identifier (NOT NULL)
visit_id	TEXT	Visit identifier (NOT NULL)
report_id	TEXT	Report identifier (NOT NULL)
report_item_id	TEXT	Report item identifier (NOT NULL, PRIMARY KEY)
campus_name	TEXT	Campus name (NOT NULL)
visit_type	TEXT	Visit type (NOT NULL)
test_report_name	TEXT	Test package name
sort_no	TEXT	Display order
test_method	TEXT	Test method
test_item_name	TEXT	Test item name
test_result	TEXT	Test result
unit	TEXT	Result unit
sample_name	TEXT	Sample type
normal_low	FLOAT	Normal low
normal_high	FLOAT	Normal high
reference_range	TEXT	Reference range
abnormal_flag	TEXT	Abnormal flag
critical_low	FLOAT	Critical low
critical_high	FLOAT	Critical high
critical_flag	TEXT	Critical flag
absurd_low	FLOAT	Absurd low
absurd_high	FLOAT	Absurd high
report_time	TIMESTAMP	Report time
comment	TEXT	Comment
create_time	TIMESTAMP	Create time
update_time	TIMESTAMP	Update time
is_valid	BOOLEAN	Is valid

Table 15: diagnostic_record

Field Name	Type	Description
medical_institution_code	TEXT	Medical institution code
campus_name	TEXT	Campus name (NOT NULL)
patient_id	TEXT	Patient identifier (NOT NULL)
visit_id	TEXT	Visit identifier (NOT NULL)
report_id	TEXT	Document id
report_item_id	TEXT	Diagnosis id (NOT NULL, PRIMARY KEY)
diagnosis_class_name	TEXT	Diagnosis class name
diagnosis_code	TEXT	Diagnosis source code
diagnosis_seq	TEXT	Diagnosis sequence
diagnosis_name	TEXT	Diagnosis name
main_flag	TEXT	Main diagnosis flag
diagnosis_source	TEXT	Source of diagnosis
diagnosis_time	TIMESTAMP	Diagnosis time
create_time	TIMESTAMP	Create time
update_time	TIMESTAMP	Update time
is_valid	BOOLEAN	Is valid

Table 16: anethesia_record

Field Name	Type	Description
medical_institution_code	TEXT	Medical institution code
campus_name	TEXT	Campus name (NOT NULL)
patient_id	TEXT	Patient identifier (NOT NULL)
visit_id	TEXT	Visit identifier (NOT NULL)
report_id	TEXT	Anesthesia record id (NOT NULL, PRIMARY KEY)
visit_type	TEXT	Visit type (NOT NULL)
height	FLOAT	Height in cm
weight	FLOAT	Weight in kg
urine_volume	FLOAT	Urine volume in ml
blood_lossed	FLOAT	Blood loss in ml
asa_class_code	TEXT	ASA class
surgical_position	TEXT	Surgical position
whether_fast	TEXT	Fasting flag
preoperative_diagnosis	TEXT	Preoperative diagnosis
propose_surgery_name	TEXT	Proposed surgery name
intraoperative_surgical_name	TEXT	Intraoperative surgery name
breath_type	TEXT	Breath type
tracheal_intubation_type	TEXT	Intubation type
drug_before_anesthesia	TEXT	Pre-anesthesia drugs
anesthesia_start_time	TIMESTAMP	Start time
anesthesia_end_time	TIMESTAMP	End time
anesthesia_type	TEXT	Anesthesia type
anesthesia_real_duration	FLOAT	Real duration in hours
go_after_operation	TEXT	Post-op disposition
create_time	TIMESTAMP	Create time
update_time	TIMESTAMP	Update time
is_valid	BOOLEAN	Is valid

Table 17: admission_record

Field Name	Type	Description
medical_institution_code	TEXT	Medical institution code
patient_id	TEXT	Patient identifier (NOT NULL)
visit_id	TEXT	Visit identifier (NOT NULL)
report_id	TEXT	Document id (NOT NULL, PRIMARY KEY)
inpatient_no	TEXT	Inpatient number (NOT NULL)
campus_name	TEXT	Campus name (NOT NULL)
patient_name	TEXT	Patient name (NOT NULL)
gender	TEXT	Gender (NOT NULL)
admission_time	TIMESTAMP	Admission time
admission_dept_name	TEXT	Admission department
admission_dept_code	TEXT	Admission department code
admission_ward_name	TEXT	Admission ward
admission_ward_code	TEXT	Admission ward code
record_title	TEXT	Record title
document_content	TEXT	Document content
chief_complaints	TEXT	Chief complaints
present_illness	TEXT	Present illness
past_medical_history	TEXT	Past medical history
personal_history	TEXT	Personal history
marital_obstetrical_history	TEXT	Marital/obstetrical history
family_history	TEXT	Family history
physical_examination	TEXT	Physical exam
accessory_examination	TEXT	Auxiliary exams
special_examination	TEXT	Specialty exams
intact_physical_examination	TEXT	Combined physical exam text
admission_diagnosis	TEXT	Admission diagnosis
initial_diagnosis	TEXT	Initial diagnosis
modified_diagnosis	TEXT	Modified diagnosis
supplementary_diagnosis	TEXT	Supplementary diagnosis
discharge_diagnosis	TEXT	Discharge diagnosis
record_time	TIMESTAMP	Record time
record_state_name	TEXT	Record state
creator_name	TEXT	Creator name
create_time	TIMESTAMP	Create time
update_time	TIMESTAMP	Update time
is_valid	BOOLEAN	Is valid

Table 18: order_inpatient

Field Name	Type	Description
medical_institution_code	TEXT	Medical institution code
patient_id	TEXT	Patient identifier (NOT NULL)
visit_id	TEXT	Visit identifier (NOT NULL)
report_id	TEXT	Order id (NOT NULL, PRIMARY KEY)
inpatient_no	TEXT	Inpatient number
campus_name	TEXT	Campus name (NOT NULL)
patient_name	TEXT	Patient name (NOT NULL)
group_no	TEXT	Group number
group_seq	TEXT	Group sequence
order_class	TEXT	Order class
order_type	TEXT	Order type
drug_flag	TEXT	Is drug
order_name	TEXT	Order name
specification	TEXT	Specification
dose_form	TEXT	Dose form
unit_price	FLOAT	Unit price
quantity	FLOAT	Quantity
quantity_unit	TEXT	Quantity unit
once_dose	FLOAT	Dose per time
dose_unit	TEXT	Dose unit
frequency	TEXT	Frequency
special_execution_time	TEXT	Special execution time
special_execution_dose	TEXT	Special execution dose
usage	TEXT	Usage
procedure_name	TEXT	Procedure name
herbal_payments	INTEGER	Herbal payments
execution_dept	TEXT	Execution department
skin_test	TEXT	Skin test flag
urgent_flag	TEXT	Urgent flag (0 normal, 1 urgent)
is_discharge_medicine	TEXT	Discharge medicine flag
order_advice	TEXT	Order advice
order_begin_time	TIMESTAMP	Order begin time
order_end_time	TIMESTAMP	Order end time
dept_name	TEXT	Department name
dept_code	TEXT	Department code
ward_name	TEXT	Ward name
ward_code	TEXT	Ward code
submit_dept_name	TEXT	Submit department name
submit_dept_code	TEXT	Submit department code
submit_time	TIMESTAMP	Submit time
body_site_name	TEXT	Body site name
sample_name	TEXT	Sample name
confirm_time	TIMESTAMP	Confirm time
cancel_time	TIMESTAMP	Cancel time
execution_time	TIMESTAMP	Execution time
order_state_name	TEXT	Order state name
create_time	TIMESTAMP	Create time
update_time	TIMESTAMP	Update time
is_valid	BOOLEAN	Is valid

Table 19: vital_signs

Field Name	Type	Description
medical_institution_code	TEXT	Medical institution code
patient_id	TEXT	Patient identifier (NOT NULL)
visit_id	TEXT	Visit identifier (NOT NULL)
vs_id	TEXT	Vital sign id (NOT NULL, PRIMARY KEY)
campus_name	TEXT	Campus name (NOT NULL)
patient_name	TEXT	Patient name (NOT NULL)
dept_name	TEXT	Department name
dept_code	TEXT	Department code
ward_name	TEXT	Ward name
ward_code	TEXT	Ward code
bed_no	TEXT	Bed number
plan_time	TIMESTAMP	Planned time
measure_time	TIMESTAMP	Measured time
breathe	TEXT	Respiration (per min)
pulse	TEXT	Pulse (per min)
heart_rate	TEXT	Heart rate (per min)
temperature	TEXT	Temperature (°C)
systolic_pressure	TEXT	Systolic pressure (mmHg)
diastolic_pressure	TEXT	Diastolic pressure (mmHg)
height	TEXT	Height (cm)
weight	TEXT	Weight (kg)
defecate_frequency	TEXT	Defecation frequency (per day)
urine_volume	TEXT	Urine volume (ml)
sputum_volume	TEXT	Sputum volume (ml)
drainage_volume	TEXT	Drainage volume (ml)
emesis_volume	TEXT	Emesis volume (ml)
output_total_volume	TEXT	Total output volume (ml)
income_volume	TEXT	Intake volume (ml)
postoperative_days	INTEGER	Postoperative days
after_delivery_days	INTEGER	Days after delivery
days_in_hospital	INTEGER	Days in hospital
create_time	TIMESTAMP	Create time
update_time	TIMESTAMP	Update time
is_valid	BOOLEAN	Is valid

C.4 Task Construction

We use GPT-5 to generate evaluation tasks through a multi-stage process, and we designed a set of prompt templates that are applied at different steps.

Task Generation

```
Purpose: Generate ONE clinical workflow task
    within a coherent clinical scenario that
    requires the use of multiple medical
    calculators.

Context:
- Core Clinical Scenario: {major_scenario} _
    {sub_category}
- Available calculators (exact names and short
    descriptions):
{calculators}

Requirements (CRITICAL):
1) Selection rules:
   - Prefer selecting calculators from the
       provided list. Use calculators outside
       the provided list **only if** the
       provided calculators are too few and
       clearly cannot cover a coherent,
       complete clinical workflow for the
       scenario. If you include any calculators
       not in the provided list, explicitly
       name them in "required_calculators"
       **and** provide a clear, one-line
       clinical justification for each in
       "scenario_analysis.workflow_logic". Do
       NOT invent or fabricate calculator
       namesonly use established, widely-known
       calculators when adding external ones.
   - Aim to select **4 to 10** calculators when
       possible to create a comprehensive
       workflow. If you select 8-10, include a
       short justification in
       `scenario_analysis.workflow_logic`
       explaining why a larger set was needed
       for clinical completeness. If fewer than
       4 relevant calculators exist in the
       provided list, you are **not required**
       to force the count up to 4; instead,
       produce a clinically coherent workflow
       using the appropriate number of
       calculators and explain the rationale in
       `scenario_analysis.workflow_logic`.
   - Choose calculators that collectively
       support a comprehensive clinical
       assessment or planning process for the
       given scenario. The primary criterion is
       **clinical appropriateness and workflow
       coherence**: prioritize selections that
       make the sequence logical and useful for
       decision-making, even if that means
       selecting fewer calculators than the
       target range.

2) Workflow Organization:
   - Organize the selected calculators into a
       logical sequence that reflects a typical
       clinical workflow for this scenario.
   - The connection between calculators should
       be based on clinical reasoning, ensuring
       a natural progression from one step to
       the next.
   - Multiple calculators' outputs (at least
       one) must serve as a required input or
       deterministic decision trigger for the
       subsequent calculator(s).
   - Include multiple conditional
       branches(if/else) (at least one) based
       on an intermediate result. This will
       demonstrate how the outcome of one step
       informs or leads to the next.

3) Content constraints:
   - DO NOT include any patient-specific
       numeric values, lab names, demographics,
       or external resource references (no
       URLs, files, DBs, APIs).
   - Describe a unified clinical scenario and
       then specify the sequence of calculator
       uses within it.
   - Focus on the purpose of each calculator in
       the context of the overall clinical
       workflow.

4) Output format (MUST be the only output; no
    extra text):
Return exactly one valid JSON object that
    matches the schema below. If you cannot
    produce the required JSON, return this
    exact error object instead:
    {"error":"unable_to_comply"}.

Schema:
{
  "task_id":
      "medical_workflow_[UNIQUE_ID_HERE]",
```

```
  "task_title": "short title reflecting the
      core scenario",
  "task_description": "Full natural-language
      task: Start with a cohesive clinical
      scenario description for
      '{major_scenario} _ {sub_category}'.
      Then, detail the step-by-step process of
      using the selected calculators in a
      logical order to address this scenario.
      Describe the clinical rationale for the
      sequence.",
  "required_calculators": ["Exact Name of
      Selected Calc A", "Exact Name of Selected
      Calc B", ...],
  "scenario_analysis": {
    "scenario_rationale": "Explain why this set
        of calculators provides comprehensive
        coverage for the core clinical
        scenario.",
    "workflow_logic": "Describe the clinical
        logic behind the order of steps
        (assessment, decision points, planning
        stages).",
    "clinical_goal": "State the overall clinical
        objective achieved by completing this
        workflow."
  }
}

Strict rules to follow:
- Only use calculator names exactly as provided
    in the {calculators} list.
- The "required_calculators" array must contain
    only the selected calculators, ordered
    logically.
- Output MUST be valid JSON parseable by a
    strict JSON parser (no comments, no
    trailing commas, use double quotes).
- Output ONLY the JSON object and nothing else.
```

Task Fuzzy

```
Purpose: Convert detailed clinical calculation
    tasks into natural, conversational medical
    requests that still implicitly but
    completely encompasses each step of the
    original task flow and the required
    intermediate deliverables.
This requires the executor to solve the problem
    step by step according to the complete
    workflow in order to arrive at a
    justifiable conclusion.
Convert this detailed clinical calculation task
    into a NATURAL, CONVERSATIONAL MEDICAL
    REQUEST that truly tests the agent's
    clinical reasoning ability.

Original detailed task: {detailed_task}

Available tools: {len_tools} clinical
    calculators (but DON'T mention them in the
    fuzzy version)

CRITICAL: CREATE A GENUINELY NATURAL CLINICAL
    REQUEST

ABSOLUTELY FORBIDDEN:
 ANY structured language that looks like a
    clinical protocol or algorithm
 Phrases like "I need to calculate", "Please
    compute", "Determine the value of"
 ANY specific calculator names or technical
    tool references
 Formal medical terminology used in isolation

INSTEAD, CREATE A NATURAL CLINICAL CONVERSATION:
 Start with patient context or clinical
    uncertainty: "I'm seeing a patient who...",
    "We have a clinical situation where..."
 Use conversational medical openers: "I'm
    trying to figure out the best approach
    for...", "Been reviewing a case where...",
    "Got a patient where we're not sure
    about..."
 Include clinical uncertainty: "not certain
    if", "wondering whether", "could be",
    "might need to consider"
 Add clinical context: "for my patient
    management", "in our clinic we're
    discussing", "I'm reviewing this case for"
 Express the clinical need through a patient
    story or scenario, not a calculation
    checklist

REQUIRED: STRUCTURED REASONING, BUT DISGUISED
    AS CLINICAL THINKING
You MUST encode every step from the original
    task, but expressed as:
 Clinical doubts
 Sequential concerns
 If this is the case, then style reasoning
 Management-oriented thinking
Think in terms of clinical dependency, not math
    steps.

STRICT REQUIREMENTS: The output must cover
    every step within task_description (even if
    there are multiple sub-steps in the
    original text, they must be presented as
    clinical task points in the fuzzy request),
    so that the executor cannot skip
    intermediate results and directly draw
    conclusions. But never tell them directly
    which calculator or tool to use.

HIDE THE CALCULATION STRUCTURE COMPLETELY:

Don't say: "Calculate BMI, then BSA, then
    adjust chemotherapy dose based on renal
    function"
Say instead: "I am managing a cancer patient
    who requires chemotherapy, but I want to
    ensure precise dosing. I need to consider
    the patient's level of obesity and physical
    condition, and adjust the chemotherapy dose
    based on renal function."

Don't say: "Please calculate the patient's
    CHADS-VASc score"
Say instead: "Given the patient's age,
    comorbidities, and vascular history,
    systematic assessment of his thromboembolic
    risk is required."


PRESERVE CLINICAL CONTEXT NATURALLY:
 Embed patient factors conversationally:
    "middle-aged", "somewhat overweight",
    "kidney function isn't perfect"
 Use approximate clinical language: "roughly in
    this range", "about this level", "somewhere
```

```
    around"
 Keep exact clinical concepts when necessary
    but phrase naturally

MAKE IT SOUND LIKE A REAL CLINICAL DISCUSSION:
 Use natural medical dialogue: "What's your
    clinical thinking on this?", "How would you
    approach this case?"
 Include realistic clinical hesitation: "I'm
    torn between...", "Part of me thinks X, but
    then there's Y to consider"
 Show appropriate clinical concern: "really
    want to avoid complications", "been
    worrying about this aspect"
 Ask for clinical reasoning naturally: "What's
    your take?", "How would you reason through
    this?", "Am I missing anything here?"

CRITICAL: End naturally with clinical evidence
    requirements:
Instead of: "Please provide evidence-based
    calculations"
Say: "I really need proper calculations on this
    - can't make this decision based on gut
    feeling alone. Whatever approach you
    suggest, make sure it's backed by solid
    calculations, okay? This needs to hold up
    to scrutiny."

ALWAYS USE clinical timeframes naturally:
 "recent labs show", "over the past few
    visits", "looking ahead to their next
    appointment"
 NOT specific dates like "January 15th" or "in
    2024"

Return ONLY the natural, conversational fuzzy
    description - make it sound like a real
    clinician discussing a case, not a robot
    executing calculations.
```

Remove Patient Data

```
Please process the following medical question
    text by removing all direct personal
    information and specific data about the
    patient, while maintaining the core content
    of the question and the completeness of the
    medical discussion.

Processing requirements:
1. Remove all information that could identify
    the patient
2. Remove specific test values, dates, ages,
    and other numerical data
3. Convert specific medical history
    descriptions to general descriptions
4. Preserve the essence of the medical question
    and the logic of clinical decision-making
5. Maintain the original tone and purpose of
    the inquiry

Example input:
I'm reviewing a case for a patient with
    cirrhosis who we just found has a liver
    tumor, an HCC. We're considering surgery to
    remove it, but I'm really hesitant. His
    liver function isn't great to begin with,
    and I'm worried a big operation could push
    him over the edge into full decompensation.
    I need to get a solid handle on whether he
    can even tolerate a hepatectomy. Part of me
    thinks we should just go for it, but then I
    keep worrying about his hearthe's got some
    risk factors there too, and major surgery
    is a huge stressor. And on top of that, I'm
    concerned about his lungs post-op; he's a
    former smoker. We absolutely need to avoid
    a situation where we cause more problems
    than we solve. I'm trying to figure out the
    best path forward. Is resection even a safe
    option for him, or are we looking at a
    transplant evaluation instead? I really
    need some evidence-based risk
    stratification heresomething more than just
    my clinical gut feeling. Can you walk me
    through how you'd assess his overall
    surgical risk? Whatever approach you take,
    make sure it's backed by proper
    calculations. This decision has to be
    rock-solid.

Expected output:
I'm reviewing a case for a patient with
    cirrhosis who has a liver tumor (HCC).
    We're considering surgery, but I'm
    hesitant. The patient's liver function is
    compromised, and I'm worried a major
    operation could lead to decompensation. I
    need to determine surgical tolerance for
    hepatectomy. I'm torn between proceeding
    with surgery and being cautious. The
    patient has cardiac risk factors, and major
    surgery poses significant stress.
    Additionally, there's a history of smoking,
    raising concerns about post-operative
    pulmonary complications. We need to avoid
    causing more harm than benefit. I'm trying
    to determine the optimal approach. Is
    resection a safe option, or should we
    consider transplant evaluation? I need
    evidence-based risk stratification beyond
    clinical intuition. Can you guide me
    through a comprehensive surgical risk
    assessment? The approach should be
    supported by proper calculations for a
    well-founded decision.

Please process the following text:
{INSERT_TEXT_HERE}
```

Create Patient Data

```
Your task is to generate realistic, consistent,
    and comprehensive patient data required to
    execute the complete clinical calculator
    task described below.
The data must align with the clinical scenario
    and information in the task.
You must output clear numerical values or
    descriptions, not ambiguous terms.
If relevant data does not exist, you may design
    it appropriately.

Context:
- Clinical Task Description: {task_description}
- Fuzzy Description: {fuzzy_description}
- Available Medical Calculators:
{tool_descriptions}

Output the necessary inputs for calculating ALL
    calculators mentioned in the
```

```
    task_description, in sequential order.
Ensure the generated data is consistent with
    all contextual clues and falls within
    clinically plausible ranges.

Output ONLY the JSON object, with no
    explanatory text.
Output Format:
{
  "metadata": {
    "scenario_reference": "brief text
        summarizing how data matches the
        scenario",
    "note": "de-identified, synthetic or derived
        data  not real PII"
  },
  "calculators": [
    {
      "order": 1,
      "name": "Exact Calculator Name (must match
          provided list)",
      "inputs": [
        { "field": "field_name", "value": "note
            or value (with unit)" }
        // ... (Add more inputs for this
            calculator)
      ],
      "notes": "optional short note about
          assumptions or mapping from scenario"
    }
    // ... (Follow the calculators order
        exactly. Add more objects for all
        calculators)
  ]
}
```

Create Case Report

```
You are a senior attending physician. The task
    is to generate a concise and realistic
    clinical report based on the provided
    partial patient data. You should include
    the chief complaint, physical examination
    findings, and diagnostic reports.

CRITICAL REQUIREMENTS:

DO NOT mention any calculator names, calculator
    results, or scoring systems.
Present all values as if they were obtained
    from real hospital systems (EHR notes, lab
    results, imaging reports, vital signs,
    etc.).
Focus only on the available data, and
    supplement with reasonable and medically
    accurate details where appropriate to
    complete the clinical picture.
The report should be concise, including only
    the most essential sections, and limited to
    the data that has been provided up until
    this point.
Maintain clinical coherence, using realistic
    medical language and making sure that all
    data is consistent with standard medical
    practice.

INPUT INFORMATION:
Clinical Scenario: {scenario}
Clinical Task: {task_description}
Patient Data: {patient_data}
Reference Cases: {reference_cases}

REPORT REQUIREMENTS:
Produce a full clinical case report based on
    the current available data. This should
    include only the following sections:
- PATIENT DEMOGRAPHICS(Name, sex, age, relevant
    background information)
- CHIEF COMPLAINT(The main issue or concern
    that led the patient to seek care)
- PHYSICAL EXAMINATION
- Vital signs, system-by-system examination
    findings, as available at the moment.
- DIAGNOSTIC REPORTS
- Relevant laboratory test results, imaging
    findings, or other diagnostics available at
    this time.

ADDITIONAL GUIDELINES:
- Ensure the language used is appropriate for a
    medical report, with clear headings for
    each section.
- The report should be concise and medically
    plausible, integrating the data provided
    with realistic values and observations.
- No unnecessary sections such as treatment
    plans or discharge instructions should be
    included.
- Ensure no contradictions with the provided
    data or clinical scenario.

Now, generate the clinical report based on the
    provided data.
```

Data Reformat

```
Your task is to reorganize patient data from a
    calculator-based structure into a
    structured format that fits nine database
    tables.
You need to map the input data and the clinical
    context to the appropriate fields in each
    table, ensuring no data is lost and
    resolving any conflicts by selecting the
    most appropriate value.

***IMPORTANT CRITICAL REQUIREMENT:***
**The values MUST mimic those on a normal
    hospital laboratory report or clinical
    documentation.**
**You MUST NOT directly include any calculator
    names, calculator results, or any
    references to calculators in the output.**
**Transform all data into standard hospital
    record format as if it came from actual
    clinical systems (EHR, lab reports, nursing
    notes, etc.).**

TASK_ID: {task_id}
SCENARIO: {scenario}
INPUT DATA: {data}
CLINICAL CONTEXT: {context}

DATABASE TABLES STRUCTURE:
You have nine tables to populate.
SCHEMA CODE FOR DATABASE: {schema}
**Conflict and Selection Principles**:
   - If an input can be mapped to multiple
       tables, fill it in all relevant tables
       while maintaining data consistency
       across them.
   - For synonymous field names (e.g.,
```

```
       `sbp`/`systolic_bp`/`systolic`),
       uniformly choose the **most detailed and
       explicit name** (e.g.,
       `systolic_blood_pressure` if available,
       otherwise `systolic_bp`).

**Rules For Primary Keys Generation**:
1) Derive patient_id from the provided task_id
     using this exact logic:
   - Use the suffix of task_id to create the
       patient_id, set patient_id = "P" +
       suffix.
     Example: task_id =
         "medical_workflow_001_644ebd11" ->
         suffix = "001_644ebd11" -> patient_id
         = "P001_644ebd11"
   - Otherwise, set patient_id = "P_task_id".
       Example: task_id = "task123" ->
       patient_id = "P_task123".
2) Set visit_id = patient_id + "_V01" for every
     visit_inpatient row. (Example: patient_id =
     "P001_644ebd11" -> visit_id =
     "P001_644ebd11_V01")
3) For every non-empty table (except
     patient_information and visit_inpatient
     already handled), ensure the table-specific
     primary-key exists:
   - Use patient_id as prefix and append
       "(table_name)_n" where n is the 1-based
       row index in that table
       (order-preserving).
   - Example for laboratory_result: first row
       -> report_item_id = ""patient_id"+
       "_RI_1", second row ->
       ""patient_id"+"_RI_2". If the primary
       key is report_id, use patient_id +
       "_R_n" instead.
   - If an existing primary-key already
       **starts with** patient_id + "_" keep it
       unchanged.
4) Do not modify any other fields. Return only
     the modified `reformat_data` JSON object.

OUTPUT FORMAT: The output should be a JSON
     object with nine arrays, each representing
     a table in the database. Each array should
     contain objects with the appropriate fields
     and values.
The values should mimic those on a normal
     hospital laboratory report and must not
     directly include the results from any
     calculators.
Strictly follow the format, output only the
     JSON, without any explanations or
     additional content.
```json
{
  "patient_information": [ { ... } ],
  "visit_inpatient": [ { ... } ],
  "vital_signs": [ { ... } ],
  "examination_report": [ { ... } ],
  "laboratory_result": [ { ... } ],
  "admission_record": [ { ... } ],
  "diagnostic_record": [ { ... } ],
  "order_inpatient": [ { ... } ],
  "anethesia_record": [ { ... } ]
}
```

C.5 Recruitment

This study was conducted under continuous medical supervision. Licensed physicians were recruited to provide professional proofreading and participate in experimental procedures, with compensation at $40 per hour.
```