

MedCalc-Pro: Solving Complex Medical Calculations with LLM Agents

Siran Zhao^{1*}, Ruihui Hou^{1*}, Ziyue Huai¹, Chennuo Zhang¹, Tong Ruan^{1†}

¹East China University of Science and Technology, Shanghai, China.

Abstract

Current benchmarks for evaluating large language models (LLMs) in medical calculation are largely based on simplified settings, where each patient case corresponds to a single calculator and the required tool is explicitly specified in the query. However, real clinical scenarios often require multiple calculators for joint evaluation, nested-calculator calculation, and fuzzy queries that do not directly specify the target calculator. To this end, we propose a new medical calculation benchmark, MedCalc-Pro, which covers three progressively challenging task settings: single-calculator, multi-calculator, and nested-calculator calculation settings. MedCalc-Pro contains 2,268 real-world clinical cases, covering 77 medical calculators across 14 clinical departments. Meanwhile, to address the limited performance of existing frameworks and methods in complex clinical scenarios, we further propose a more generalizable agent framework that supports multi-calculator selection and nested-calculator calling, while suppressing parameter error propagation through structured validation and evidence audit. We conduct systematic comparisons across open-source, closed-source, and medical-specialized LLMs, and the results show that our framework achieves the best performance across all three task settings. This work provides a new benchmark and method for evaluating and applying LLMs in challenging medical calculation scenarios. The dataset and code are publicly available at <https://github.com/easymoney7774/MedCalc-Pro>.

1 Introduction

Large language models (LLMs) have recently achieved remarkable progress in information understanding (Minaee et al., 2024) and complex reasoning (Hao et al., 2024). However, they still

* Equal Contribution.

† Corresponding Authors.

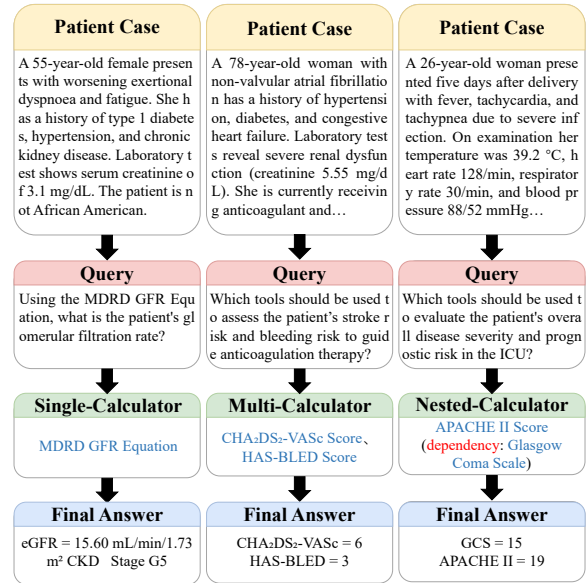


Figure 1: Illustrative examples of three medical calculation scenarios: single-calculator, multi-calculator, and nested-calculator scenarios.

exhibit clear limitations on certain tasks that require precise execution by external tools, such as numerical computation and rule-based clinical calculation (Huang et al., 2024; Chen, 2020). This issue is particularly salient in the medical domain. In clinical practice, medical calculators are widely used for risk assessment and prognosis prediction (Moons et al., 2015). Consequently, there has been growing interest in integrating LLMs with medical calculators.

However, existing medical calculation benchmarks, including MedCalc-Bench (Khandekar et al., 2024), CMedCalc-Bench (Zhang et al., 2025), OpenMedCalc (Goodell et al., 2023), CalcQA (Zhu et al., 2025), and AgentMD (Jin et al., 2025), remain largely restricted to simplified settings. These benchmarks typically pair an explicit target-calculator query with a clinical case description, each case corresponding to only one calculator. Such settings are insufficient for cap-

turing more realistic medical calculation scenarios. In practice, multiple calculators often need to be used jointly, while some complex calculators depend on sub-calculator outputs, forming nested calculator-calling chains.

To address these limitations, we propose a new medical calculation benchmark, **MedCalc-Pro**, for systematically evaluating LLMs in complex medical calculation scenarios. The benchmark covers three task settings with increasing structural complexity: single-calculator, multi-calculator, and nested-calculator calculation. Specifically, the single-calculator task represents the basic setting in which a single patient case requires one calculator. The multi-calculator task requires multiple calculators to be identified and jointly applied to a single case; it constitutes the largest proportion of our dataset and is also the most challenging setting overall. The nested-calculator task further requires the model to recognize and resolve dependencies among calculators. Figure 1 illustrates these three task settings. MedCalc-Pro contains 2,268 real-world clinical cases, covering 77 medical calculators across 14 clinical departments. Compared with existing benchmarks, MedCalc-Pro not only covers task settings that are closer to real clinical usage but also places greater emphasis on clinically goal-driven queries, thereby avoiding direct exposure of the tool name to the model and imposing stricter requirements on query understanding, tool selection, parameter extraction, and multi-step execution. Table 1 summarizes the key differences between existing medical calculator benchmarks and MedCalc-Pro.

Meanwhile, to address the limited generalization of existing methods and frameworks in complex settings, we propose a more generalizable medical calculator agent framework. The framework follows realistic medical calculation workflows and consists of four stages: Query Rewriting, Retrieval and Reranking, Tool Selection, and Tool Execution. Query Rewriting enhances clinical-intent representation through multi-dimensional rewriting of the original query. Retrieval and Reranking identifies relevant candidate calculators from the tool repository. Tool Selection enables the model to choose the appropriate calculators for the clinical task. Finally, Tool Execution performs dependency-aware nested-calculator invocation, while structured validation and evidence audit are introduced to sup-

press error propagation and improve the robustness of the final calculation.

Our main contributions are as follows:

- We construct a new medical calculation benchmark, MedCalc-Pro, to enable more realistic evaluation of LLMs across single-calculator, multi-calculator, and nested-calculator scenarios.
- We propose a generalizable agent framework for complex medical calculation, supporting multi-calculator selection and nested-calculator execution through query rewriting, retrieval and reranking, tool selection, and dependency-aware tool execution with structured validation and evidence audit.
- We compare our method with representative frameworks based on open-source, closed-source, and medical-specialized models. It achieves the best performance across all three task settings, demonstrating strong robustness and generalization in complex scenarios.

2 Related Works

2.1 Medical Calculator Benchmarks

In recent years, as LLMs have been applied to an increasingly wide range of real-world tasks (Morris et al., 2023; Nori et al., 2023), their performance in domain-specific applications has attracted growing attention (Singhal et al., 2023). In the medical domain, many early datasets, such as MedQA (Jin et al., 2021), PubMedQA (Jin et al., 2019), MedMCQA (Pal et al., 2022), and MMLU medical subjects (Hendrycks et al., 2021), primarily adopt multiple-choice and open-ended formats, which assess whether LLMs can recall and reason over medical knowledge. Furthermore, several recent studies have proposed benchmarks specifically targeting medical calculation tasks such as CMedCalc-Bench, CalcQA, and MedCalc-Bench. Despite these efforts, existing benchmarks typically assume a single-calculator setting, and the task description often explicitly specifies which calculator should be used. In real-world clinical scenarios, however, a single patient case often requires evaluation using multiple medical calculators, and clinicians may not explicitly indicate which calculators should be applied. To better reflect realistic clinical workflows, we constructed MedCalc-Pro that supports three types of

Benchmarks	Fuzzy Query	Proc. Eval.	Multi-calculator	Nested-calculator	Size
RiskQA (Jin et al., 2025)	✗	✓	✗	✗	350
MedRaC (Wang et al., 2025)	✗	✓	✗	✗	1,048
CMedCalc-Bench (Zhang et al., 2025)	✗	✓	✗	✗	1,143
OpenMedCalc (Goodell et al., 2023)	✗	✗	✗	✗	287
MedCalc-Bench (Khandekar et al., 2024)	✗	✗	✗	✗	1,047
CalcQA (Zhu et al., 2025)	✓	✓	✗	✗	100
MedCalc-Pro (Ours)	✓	✓	✓	✓	2,268

Table 1: Comparison of medical calculator benchmarks. “Fuzzy Query” indicates whether the benchmark includes goal-driven queries that do not explicitly specify the target calculator. “Proc. Eval.” denotes whether process-level evaluation is provided. “Multi-calculator” indicates whether cases may require multiple calculators, and “Nested-calculator” indicates whether cases involve dependencies between calculators.

settings: single-calculator, multi-calculator, and nested-calculator calculation.

2.2 LLM-based Methods for Medical Calculation

Relying solely on the model for the entire computation process poses a significant limitation. The model’s numerical reasoning and calculation performance are often subpar (Li et al., 2025), reducing transparency and making intermediate steps difficult to verify, which is particularly critical in medical scenarios. To improve reliability and transparency, more recent studies have explored integrating LLMs with external tools (Schick et al., 2023), such as code interpreters (Gao et al., 2023; Chen et al., 2023) and external APIs. This approach has also been widely adopted in medical calculation tasks, such as MeNTi (Zhu et al., 2025), and AgentMD enable LLMs to interact with structured medical calculators, allowing deterministic execution of clinical scoring programs. However, these frameworks also typically assume that each patient case corresponds to a single-calculator calculation, limiting their applicability in realistic clinical scenarios. These challenges motivate our framework design, which supports flexible multi-calculator and nested-calculator execution through a structured agent workflow.

3 Methodology

3.1 Task Definition

Given a patient’s electronic medical record (EMR) and a query Q , the task aims to select appropriate medical calculators from a set of candidates and compute the corresponding answer based on the patient’s clinical information. This task includes three types of calculation: single-calculator, multi-calculator, and nested-calculator. As illustrated in Figure 1, when the query is “Which tools should

be used to assess the patient’s stroke risk and bleeding risk to guide anticoagulation therapy?”, the model first selects the relevant calculators, CHA₂DS₂-VASc Score and HAS-BLED Score. It then applies the scoring rules of these calculators using the patient’s clinical information and finally produces the corresponding scores of 6 and 3.

3.2 Benchmark Creation

To evaluate the capability of LLMs in medical calculation tasks, we construct MedCalc-Pro, a benchmark covering multiple evaluation scenarios.

For the **single-calculator** task, we use MedRaC as the raw dataset. It contains 1,048 patient cases covering 55 medical calculators, including 20 rule-based calculators and 35 numerical calculators. For the **multi-calculator** task, we construct cases from MedCalc-Bench by grouping entries sharing the same patient case and merging their ground-truth calculator annotations, yielding 1,066 multi-calculator candidates. However, this dataset is highly imbalanced, with rule-based calculators appearing much less frequently than numerical calculators. To improve rule-based calculator coverage, we construct cases from PMC-Patients (Zhao et al., 2022). We use regular expression matching to identify clinical records explicitly mentioning multiple calculator names and scores from 180,142 cases, obtaining 3,517 candidate descriptions. After filtering for sufficient clinical information to compute all required calculators, we obtain 92 multi-calculator cases covering 33 rule-based calculators.

For all multi-calculator cases, we use GPT-5.1 to rewrite the original calculator-explicit queries as fuzzy, goal-oriented requests conditioned on the corresponding clinical texts. This design better simulates realistic medical calculation requests,

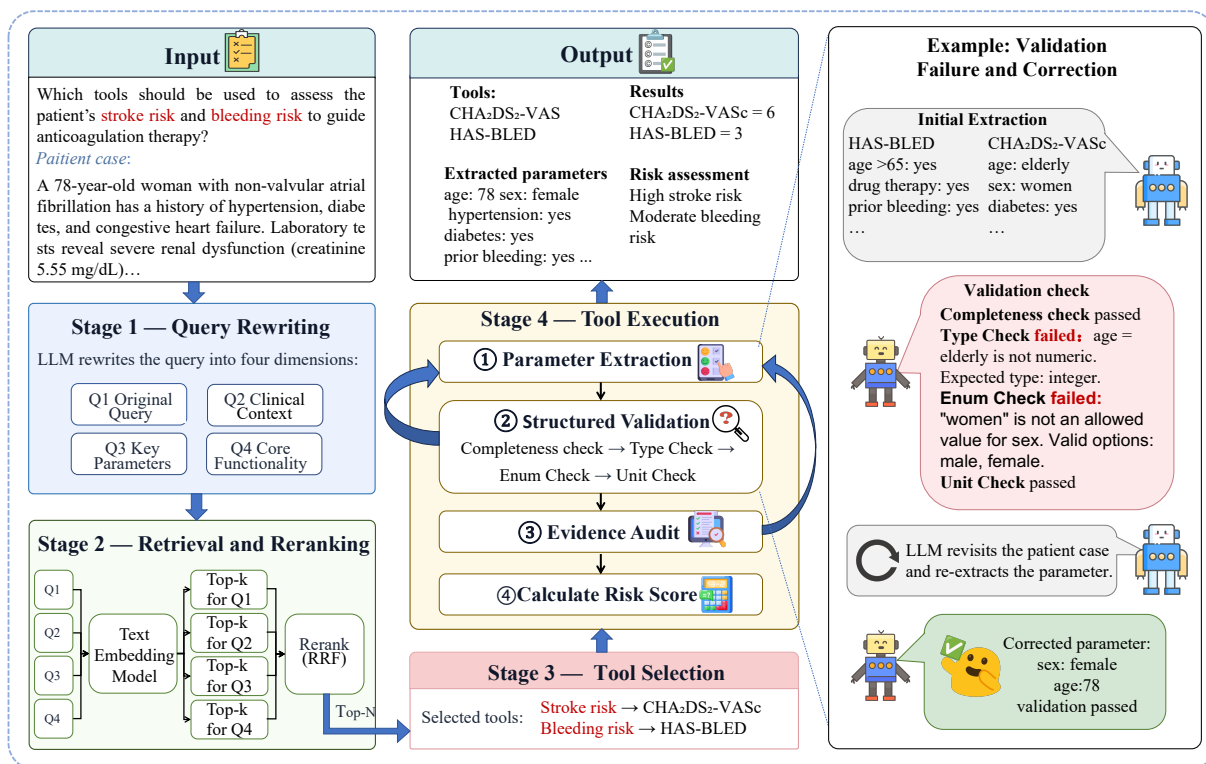


Figure 2: Overview of our framework. This agent contains four stages: Query Rewriting, Retrieval and Reranking, Tool Selection, and Tool Execution.

where physicians typically describe clinical goals rather than explicitly naming all target calculators. Prompts are provided in Appendix B.

For the **nested-calculator** task, we also construct data from PMC-Patients using regular expression matching to identify clinical records that mention calculators with dependency relationships. In addition to the parent calculator names, we require the presence of sub-calculator names together with their corresponding scores and supporting clinical descriptions, ensuring that both parent and sub-calculators can be derived from the clinical text. After filtering for cases with sufficient information to support the full dependency chain, we obtain 62 nested-calculator cases.

Quality Control. We apply a three-stage quality-control process to ensure the reliability and clinical realism of the benchmark. First, to prevent answer leakage, we remove explicit calculator names, aliases, and final-score expressions directly revealing the target calculators or expected outputs, while preserving all clinical evidence required for calculation. Second, three medical students review the full dataset for consistency among the clinical descriptions, queries, gold calculator sets, required parameters, and expected outputs. Third, three clinicians independently au-

dit 20% of the data for query and answer validity, with case-level decisions determined by majority voting.

The first-round clinician-audit pass rate is 93.13%, below the predefined 95% threshold. Based on the audit feedback, the student annotators conduct a second full-data review and revise 83 of the 1,048 single-calculator cases and 115 of the 1,158 multi-calculator cases, covering 198 of the 2,206 cases in these two settings. A subsequent clinician audit achieves a pass rate of 97.33%, exceeding the predefined threshold. Detailed verification criteria, revision statistics, and inter-clinician agreement results are provided in Appendix A.1.

Toolkit Construction. We build the toolkit in two stages. First, we reuse the validated computational logic of 46 calculators from MedCalc-Bench and reorganize them under a unified configuration schema, standardizing parameter definitions and supplementing descriptions from authoritative clinical references such as the MSD Manuals¹. Second, we implement 31 additional calculators to expand coverage in underrepresented areas, including neurological assessment (e.g., NIHSS),

¹<https://www.msdmanuals.com/>

functional evaluation (e.g., ECOG), and psychiatric screening (e.g., PHQ-9). All calculators serve as callable tools within the agent framework.

3.3 Agent Framework

Existing medical-calculation agents are primarily designed for one-to-one settings in which a query corresponds to a single, often explicitly specified calculator. Consequently, they provide limited support for recovering a complete calculator set from fuzzy queries, resolving dependencies between parent and sub-calculators, and preventing parameter-extraction errors from propagating through multi-step execution. To address these limitations, we propose an agent framework composed of four core modules: query rewriting, retrieval and reranking, tool selection, and tool execution. A complete execution trace and detailed prompts are provided in Appendix C.

Query Rewriting. In real clinical practice, physicians’ queries are often expressed in terms of task goals, such as “*assessing stroke risk*” or “*evaluating overall disease severity*”. Directly matching such queries with calculator descriptions often leads to inaccurate retrieval results. To better capture the underlying clinical intent, this module rewrites the original query along three medically meaningful dimensions: clinical context, key parameters, and core functionality. This multi-dimensional rewriting helps the system better align the query with relevant medical calculators and more accurately recover the physician’s intended task.

Retrieval and Reranking. For each rewritten query, we perform dense retrieval (Zhuang et al., 2023; Zhou et al., 2023) against the corresponding calculator description fields in the toolkit. Each query independently retrieves top- K candidate calculators. Candidates retrieved from multiple query routes are then combined and reranked using Reciprocal Rank Fusion (RRF) (Cormack et al., 2009), yielding a final candidate set of N calculators. The calculation formula is:

$$\text{RRF}(d) = \sum_{i=1}^4 \frac{1}{k + r_i(d)} \quad (1)$$

Tool Selection. In multi-calculator scenarios, the number of applicable calculators for a given clinical case is unknown. Therefore, directly selecting the top-1 candidate based on similarity is insufficient. At this stage, the model is provided

with the detailed descriptions and parameter requirements of all candidate calculators, enabling more fine-grained reasoning to determine which calculators should be executed.

Tool Execution. Even after appropriate calculators are selected, successful execution is not guaranteed. Some complex calculators depend on sub-calculator outputs for computation. Therefore, the system first identifies dependencies, determining whether the target calculator requires inputs that cannot be directly extracted from the patient case but must be obtained by executing other calculators. If dependencies exist, relevant sub-calculators are executed first, and their outputs are used as inputs for the parent calculator.

After determining the execution order, the extracted parameters are validated through a pipeline: Completeness Check → Type Check → Enum Check → Unit Check. The system then performs an evidence audit to ensure consistency with the original clinical text and medical logic.

- **Completeness Check:** whether all parameters required for the calculation are present in the returned results;
- **Type Check:** whether the extracted values match the expected parameter types;
- **Enum Check:** whether the values of enumerated variables belong to the allowed options;
- **Unit Check:** whether the units of numerical values are consistent with calculator requirements.

Overall, if an error is detected at any stage, the system returns error information and backtracks to parameter extraction for correction. The corresponding calculators are executed only after all checks pass to produce the final result. This mechanism helps prevent error propagation from incorrect parameters during nested calculator execution.

4 Experiment

In this section, we first conduct extensive experiments to evaluate the effectiveness of our framework. Next, we provide a detailed analysis to offer deeper insights into our framework.

4.1 Evaluation Setup

Baselines. We compare our framework with three representative agent methods: (1) MeNTi (Zhu et al., 2025), using a meta-tool mechanism for tool selection and invocation; (2) MedRaC (Wang et al., 2025), integrating retrieval-augmented gen-

eration with LLM-based code generation for medical calculation; and (3) ReAct (Yao et al., 2022), a reasoning-and-acting framework for iterative step-by-step tool use. For single-calculator baselines, we adapt their selection and execution procedures to support multi-calculator and nested-calculator settings. To ensure controlled comparison, all methods use the same calculator tool library, while retrieval-based methods use the same embedding model. For each method, we evaluate six LLMs to examine model-framework interactions: four open-source general LLMs (Qwen3-235B-A22B (Yang et al., 2025), gpt-oss-120b (Agarwal et al., 2025), Llama-3.3-70B-Instruct (Grattafiori et al., 2024), and DeepSeek-V3.1 (Liu et al., 2024)), one closed-source LLM (GPT-5-mini), and one medical-specialized LLM (MedResearcher-R1-32B (Yu et al., 2025)). Beyond these task-specific baselines, we include two complementary references. We evaluate Codex with GPT-5.4-mini as a strong modern general-purpose agent harness, while the Human row reports the average performance of two medical students who independently completed the benchmark under closed-book conditions, providing an empirical estimate of human-level performance.

Metrics. We evaluate systems across three aspects: tool selection, parameter extraction, and final score computation. We use Tool Selection F1 (T-F1) to measure correct calculator selection, Extraction Accuracy to assess correct extraction of required parameters, and Score Accuracy to assess whether selected calculators produce correct final scores. For nested-calculator tasks, we introduce two dependency-related metrics: Dependency Detection Accuracy, which measures whether the model correctly identifies dependencies between the target calculator and its sub-calculators, and Sub-calculator Score Accuracy, which evaluates whether intermediate results from dependent sub-calculator tools are correct. We conduct case-level bootstrap analysis for Score Accuracy across the three task settings in Appendix A.6.

Implementations. Our system is deployed through an OpenAI-compatible API interface. For the closed-source GPT model, we use the default settings. For all other open-source models, we set the temperature to 0.01 to encourage stable and deterministic outputs. In the semantic retrieval stage, we use m3e-base (Wang et al., 2023) as the embedding model and perform vector retrieval over the calculator toolkit using normalized cosine sim-

ilarity on GPU. For each retrieval route, we retrieve $K_{\text{route}} = 16$ candidate tools. After reranking, the final number of candidates provided to the LLM selector is set to $K_{\text{cand}} = 32$. A detailed hyperparameter analysis is provided in Appendix A.2. All experiments are conducted on a cluster with four NVIDIA A800 80GB GPUs.

4.2 Main Results

To evaluate the effectiveness of the proposed framework, we conduct systematic comparisons with all baseline methods. The experimental results are presented in Table 2.

Under the shared gpt-oss-120b backbone, we conclude that: 1) **Performance across task settings.** Our method consistently outperforms all baselines across the three task settings. On the Single-Calculator task, it surpasses the strongest baseline, MedRaC, by 13.05% in T-F1 and 4.77% in Score Accuracy, indicating that existing agents remain unstable even with an explicitly specified target calculator. On the Multi-Calculator task, our method improves parameter extraction accuracy and Score Accuracy over ReAct by 19.07% and 21.96%, respectively, highlighting growing bottlenecks in parameter extraction and execution. On the Nested-Calculator task, our method improves Sub-calculator Score Accuracy and Parent-calculator Score Accuracy over MeNTi by 32.27% and 14.51%, respectively.

2) **Codex and human references.** We further report Codex and human performance as complementary upper-bound references. Human evaluation is independently conducted by two medical students under closed-book conditions, with the average results reported. Human evaluators achieve the best performance across all metrics, including perfect T-F1 on the single- and nested-calculator tasks and perfect dependency detection on the nested task. Nevertheless, their Multi-Calculator T-F1 (73.63%) remains substantially lower than their performance in the other two settings, demonstrating the intrinsic difficulty of identifying a complete calculator set. Codex consistently ranks second across all single- and multi-calculator metrics and remains near human performance in nested-calculator parameter extraction and execution, but shows larger gaps in tool selection and dependency detection.

3) **Analysis of MedRaC.** MedRaC obtains the same Single-Calculator T-F1 (86.57%) across all base models because its original RAG module

Method	Model	Single-Calculator			Multi-Calculator			Nested-Calculator				
		T-F1	Ext.	Score	T-F1	Ext.	Score	T-F1	Dep.	Ext.	Par.	Sub.
MeNTi (Zhu et al., 2025)	gpt-oss-120b	86.11	37.56	58.40	29.06	15.12	15.61	36.52	56.45	61.68	61.29	54.83
	Qwen3-235B-A22B	82.91	43.50	57.92	31.95	19.26	17.47	37.96	54.84	59.53	45.16	58.06
	Llama-3.3-70B-Instruct	84.53	38.35	58.49	33.83	21.11	26.68	21.12	33.90	15.50	16.12	11.29
	GPT-5-mini	82.35	45.66	56.11	32.38	24.62	28.28	31.78	66.13	56.08	54.83	53.22
	DeepSeek-V3.1	73.56	39.09	49.71	34.99	23.17	21.11	34.19	48.39	53.28	51.61	45.16
	MedResearcher-R1-32B	85.51	41.25	55.73	29.55	17.73	15.79	42.86	11.29	40.15	35.43	19.35
MedRaC (Wang et al., 2025)	gpt-oss-120b	86.57	43.92	75.38	20.38	5.29	13.71	0.00	0.00	0.00	0.00	0.00
	Qwen3-235B-A22B	86.57	42.35	73.85	18.10	5.53	11.26	0.00	0.00	0.00	0.00	0.00
	Llama-3.3-70B-Instruct	86.57	34.05	69.37	20.72	6.22	14.49	0.00	0.00	0.00	0.00	0.00
	GPT-5-mini	86.57	44.70	76.72	18.00	6.97	12.30	0.00	0.00	0.00	0.00	0.00
	DeepSeek-V3.1	86.57	39.05	69.08	21.12	6.86	12.89	0.00	0.00	0.00	0.00	0.00
	MedResearcher-R1-32B	86.57	40.42	64.50	20.38	6.33	12.78	0.00	0.00	0.00	0.00	0.00
ReAct (Yao et al., 2022)	gpt-oss-120b	79.50	48.01	55.06	54.32	34.80	37.57	80.62	83.87	66.42	11.74	77.41
	Qwen3-235B-A22B	75.51	39.36	51.34	50.21	30.38	33.59	74.02	75.81	58.45	19.35	56.45
	Llama-3.3-70B-Instruct	72.28	23.26	40.84	47.01	25.21	27.20	55.76	71.00	25.19	11.29	12.90
	GPT-5-mini	77.90	35.47	49.05	50.79	31.99	35.12	77.78	90.30	72.01	25.80	82.30
	DeepSeek-V3.1	78.03	46.65	55.06	48.58	28.91	30.84	53.23	51.61	38.11	20.97	24.19
	MedResearcher-R1-32B	80.21	40.42	56.49	44.99	31.82	34.41	67.20	67.74	49.95	24.19	27.42
Ours	gpt-oss-120b	99.62	53.56	80.15	59.97	53.87	59.53	87.22	<u>93.50</u>	88.48	75.80	87.10
	Qwen3-235B-A22B	98.12	52.84	79.58	54.44	45.58	44.37	78.26	87.10	81.05	75.81	80.65
	Llama-3.3-70B-Instruct	86.63	37.03	65.94	53.53	44.53	46.15	55.03	58.66	67.21	71.67	80.33
	GPT-5-mini	93.13	41.19	70.13	57.73	44.70	48.94	80.00	<u>93.50</u>	88.91	<u>83.90</u>	85.50
	DeepSeek-V3.1	98.53	56.19	76.04	52.63	39.65	43.18	58.73	59.68	52.53	51.61	56.45
	MedResearcher-R1-32B	98.03	40.76	77.00	50.33	40.14	37.50	61.54	64.52	57.27	56.45	<u>91.94</u>
Others	Codex	<u>99.76</u>	<u>63.50</u>	<u>89.22</u>	<u>68.52</u>	<u>58.03</u>	<u>60.27</u>	<u>89.22</u>	88.71	<u>93.86</u>	83.87	<u>91.94</u>
	Human	100.00	83.85	96.00	73.63	62.61	67.05	100.00	100.00	95.98	87.50	92.50

Table 2: Main results on the single-calculator, multi-calculator, and nested-calculator tasks (%). “Ext.” denotes parameter extraction accuracy; “Dep.”, “Par.”, and “Sub.” denote dependency detection accuracy, parent-calculator score accuracy, and sub-calculator score accuracy, respectively. The best and second-best results are highlighted in **bold** and underline.

directly returns the top-1 calculator, bypassing LLM-based tool selection. For the multi- and nested-calculator tasks, we retrieve the top-10 candidates, allow the base LLM to select multiple tools, and supplement the knowledge base with missing formulas and descriptions. Thus, its poor nested-task performance mainly reflects the difficulty of generalizing direct formula retrieval to implicit clinical-goal queries, rather than an incomplete candidate space.

4.3 Ablation Study

To analyze the contribution of each core component, we conduct ablation experiments by removing Query Rewriting, Retrieval and Reranking, Dependency Detection, Structured Validation, and Evidence Audit individually. Dependency Detection is applicable only to the nested-calculator task. We use gpt-oss-120b as the base model in all analysis experiments. The results presented in Table 3 show that:

- 1) In the single-calculator task, Retrieval and

Reranking is the most important module. Removing it causes the largest overall degradation, including decreases of 19.96% in T-F1 and 12.40% in Score Accuracy. This indicates that reliable calculator retrieval remains the primary determinant of performance even when only one calculator is required.

- 2) In the multi-calculator task, Retrieval and Reranking remains the most influential module. Its removal leads to the largest decreases in Extraction Accuracy and Score Accuracy, by 12.01% and 13.45%, respectively. This suggests that identifying the complete set of relevant calculators is the main bottleneck in multi-calculator reasoning.

- 3) In the nested-calculator task, Structured Validation has the broadest impact, with its removal causing the largest decrease in Extraction Accuracy at 13.35%. Dependency Detection is particularly important for sub-calculator execution: removing it causes the largest decrease in Sub-calculator Score Accuracy, by 9.70%. These results show that nested-calculator reasoning de-

Method	Single-Calculator			Multi-Calculator			Nested-Calculator				
	T-F1	Ext.	Score	T-F1	Ext.	Score	T-F1	Dep.	Ext.	Par.	Sub.
Ours	99.62	53.56	80.15	59.97	53.87	59.53	87.22	93.50	88.48	75.80	87.10
w/o Query Rewriting	89.70	41.69	77.67	57.02	46.52	49.81	79.39	83.87	75.57	72.58	77.42
w/o Retrieval & Reranking	79.66	35.95	67.75	55.59	41.86	46.08	85.04	87.10	82.13	72.58	85.48
w/o Structured Validation	96.92	43.52	76.81	58.72	50.26	52.86	85.65	80.65	75.13	69.35	77.42
w/o Evidence Audit	97.40	42.86	77.96	58.64	50.68	52.46	83.20	83.87	78.50	66.13	79.03
w/o Dependency Detection	–	–	–	–	–	–	86.50	0.00	84.82	74.20	77.40

Table 3: Ablation results on the single-calculator, multi-calculator, and nested-calculator tasks (%). Dependency Detection is not applicable to non-nested tasks.

Error Type	Single-calculator	Multi-calculator	Nested-calculator
E1: Retrieval Miss	7	321	2
E2: Selection Error	0	460	2
E3: Over-Selection	0	102	8
E4: Param-Value Error	468	116	27
E5: Unit Conversion Error	75	55	0
E6: Enum Mapping Error	386	66	23

Table 4: Error type distribution across three task settings.

depends not only on accurate parameter validation but also on correctly resolving dependencies between calculators.

4.4 Error Analysis

To identify the dominant residual failure modes, we first manually inspect 50 erroneous cases and derive six categories: **Retrieval Miss**, **Selection Error**, **Over-selection**, **Parameter Value Error**, **Unit Conversion Error**, and **Enum Mapping Error**. We then apply this taxonomy to all residual error instances produced by our framework. Table 4 reports their distribution across task settings.

The dominant bottleneck differs across tasks. In the single-calculator setting, parameter-related errors account for 929 of 936 residual errors (99.3%), indicating that the main challenge is grounding clinical text into normalized values, units, and categorical inputs. In the multi-calculator setting, tool-identification errors account for 883 of 1,120 errors (78.8%), showing that complete tool-set identification remains the primary bottleneck. In the nested-calculator setting, Parameter Value and Enum Mapping Errors account for 50 of 62 errors (80.6%); such errors are particularly consequential because incorrect sub-calculator inputs may propagate to the parent calculator. Appendix A.4 provides three fully annotated cases that collectively cover all six error categories.

Method	O1	O2	O3	O4	O5	Avg.
ReAct	1.22	1.40	1.19	1.65	0.97	1.22
MeNTi	1.22	1.06	1.24	1.16	1.10	1.16
MedRaC	0.85	1.06	0.59	0.00	0.69	0.75
Ours	1.50	1.56	1.61	1.82	1.23	1.49

Table 5: Rubric-based manual evaluation results. Dimension scores range from 0 to 2. Avg. is micro-averaged over all applicable case–dimension ratings.

4.5 Manual Evaluation

To complement the score-accuracy metrics, we conduct a rubric-based manual evaluation on 262 cases, comprising 100 single-calculator cases, 100 multi-calculator cases, and all 62 nested-calculator cases. The rubric evaluates clinical-goal coverage and completeness (O1), parameter extraction and evidence consistency (O2), result presentation and interpretability (O3), nested-calculator dependency handling (O4) which applies only to nested-calculator cases, and clinical plausibility and usability (O5). Detailed criteria are provided in Appendix A.4. Table 5 reports the overall score micro-averaged over all applicable dimension ratings.

Manual evaluation shows that our framework achieves the highest scores across O1–O5 and the highest overall average among the evaluated methods. It performs particularly strongly on O3 (1.61) and O4 (1.82), while also leading on O1, O2, and O5. These results demonstrate advantages beyond numerical correctness, including completeness, evidence grounding, interpretability, dependency handling, and clinical usability.

5 Conclusion

In this work, we introduce MedCalc-Pro, a benchmark consisting of 2,268 cases and 77 calculators, incorporating three task types: single-

calculator, multi-calculator, and nested-calculator calculation, which better reflect real-world clinical scenarios. To address the challenges posed by these tasks, we propose a more generalized framework that integrates dependency identification and multi-tool nested invocation. Extensive experiments demonstrate that our framework outperforms existing approaches across all task types, achieving the best performance in each case. We hope that our framework will help facilitate the application of LLMs in the medical field, advancing clinical decision support systems.

Limitations and Future Work

Although the proposed framework substantially improves LLM performance on medical calculator tasks through dependency identification, nested tool execution, and structured validation, several limitations still remain. First, the data for nested-calculator tasks is still relatively limited. Although we intentionally expanded the coverage of rule-based calculators and specifically constructed nested-calculator cases, the number of nested cases remains much smaller than that of the other two settings.

Second, the current benchmark is still mainly based on relatively clean clinical text and does not sufficiently capture the noise commonly encountered in real medical settings. Examples include OCR recognition errors, non-standard abbreviations, spelling mistakes, formatting irregularities, and cross-source document concatenation, all of which are common in real electronic medical records and scanned documents.

Third, the current benchmark mainly adopts a single-turn query-response setting. In real clinical scenarios, however, the input is often incomplete, and physicians or systems may need to actively ask follow-up questions, conduct multi-turn clarification, and confirm missing parameters before the calculation can be completed. As a result, the present benchmark does not assess the model's ability to proactively interact under incomplete information, nor does it capture performance in collaborative multi-turn clinical calculation.

Overall, these limitations suggest that there is still a gap between the current benchmark setting and realistic deployment conditions. Future work may therefore focus on expanding nested cases, incorporating noisier clinical text, and moving from single-turn evaluation toward more inter-

active multi-turn clinical calculation settings.

Ethics Consideration

Because the medical domain is inherently high-stakes, any errors in tool selection, parameter extraction, or calculator execution may lead to incorrect clinical assessments if such systems are used without appropriate oversight. Therefore, our proposed benchmark and agent framework are intended for research purposes only and should not be used as a standalone basis for real-world diagnosis, prognosis, or treatment decisions.

Our benchmark is constructed from real-world clinical cases collected from existing benchmark resources and PMC-patient open-access dataset. The benchmark is intended for methodological evaluation rather than patient-level decision support. Although we do not intentionally include personally identifiable information, even de-identified or publicly available clinical text may still raise privacy and data governance concerns. We therefore emphasize that future extensions of this work should continue to follow applicable data protection standards and institutional review requirements.

Another ethical concern is that improved medical calculator agents may be over-interpreted as reliable clinical assistants. Although our framework improves performance in complex settings, the results also show that substantial errors remain, especially in parameter extraction and nested execution scenarios. Accordingly, these systems should be viewed as assistive research prototypes rather than clinically deployable tools, and human oversight by qualified medical professionals remains essential.

Acknowledgments

This work was supported by the Shanghai Science and Technology Innovation Action Plan in Computational Biology (No. 24JS2840200).

References

- Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. 2025. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*.
- Lingxiao Chen. 2020. Overview of clinical prediction models. *Annals of translational medicine*, 8(4):71.

- Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. 2023. [Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks](#). *Transactions on Machine Learning Research*.
- Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. 2009. [Reciprocal rank fusion outperforms condorcet and individual rank learning methods](#). In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '09*, page 758–759, New York, NY, USA. Association for Computing Machinery.
- Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In *International conference on machine learning*, pages 10764–10799. PMLR.
- 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.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.
- Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. [Measuring massive multitask language understanding](#). In *International Conference on Learning Representations*.
- Yue Huang, Jiawen Shi, Yuan Li, Chenrui Fan, Siyuan Wu, Qihui Zhang, Yixin Liu, Pan Zhou, Yao Wan, Neil Zhenqiang Gong, and Lichao Sun. 2024. [Meta-tool benchmark for large language models: Deciding whether to use tools and which to use](#). In *The Twelfth International Conference on Learning Representations*.
- Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421.
- Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. [PubMedQA: A dataset for biomedical research question answering](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577, Hong Kong, China. Association for Computational Linguistics.
- Qiao Jin, Zhizheng Wang, Yifan Yang, Qingqing Zhu, Donald Wright, Thomas Huang, Nikhil Khandekar, Nicholas Wan, Xuguang Ai, W John Wilbur, et al. 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, et al. 2024. Medcalc-bench: Evaluating large language models for medical calculations. *Advances in Neural Information Processing Systems*, 37:84730–84745.
- Haoyang Li, Xuejia Chen, Zhanchao Xu, Darian Li, Nicole Hu, Fei Teng, Yiming Li, Luyu Qiu, Chen Jason Zhang, Li Qing, and Lei Chen. 2025. [Exposing numeracy gaps: A benchmark to evaluate fundamental numerical abilities in large language models](#). In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20004–20026, Vienna, Austria. Association for Computational Linguistics.
- Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.
- Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.
- Karel GM Moons, Douglas G Altman, Johannes B Reitsma, John PA Ioannidis, Petra Macaskill, Ewout W Steyerberg, Andrew J Vickers, David F Ransohoff, and Gary S Collins. 2015. Transparent reporting of a multivariable prediction model for individual prognosis or diagnosis (tripod): explanation and elaboration. *Annals of internal medicine*, 162(1):W1–W73.
- Meredith Ringel Morris, Jascha Sohl-Dickstein, Noah Fiedel, Tris Warkentin, Allan Dafoe, Aleksandra Faust, Clement Farabet, and Shane Legg. 2023. Levels of agi for operationalizing progress on the path to agi. *arXiv preprint arXiv:2311.02462*.
- Harsha Nori, Nicholas King, Scott Mayer McKinney, Dean Carignan, and Eric Horvitz. 2023. Capabilities of gpt-4 on medical challenge problems. *arXiv preprint arXiv:2303.13375*.
- Ankit Pal, Logesh Kumar Umapathi, and Malaikanan Sankarasubbu. 2022. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on health, inference, and learning*, pages 248–260. PMLR.
- 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.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, et al. 2023. Large language models encode clinical knowledge. *Nature*, 620(7972):172–180.

Benlu Wang, Iris Xia, Yifan Zhang, Junda Wang, Feiyun Ouyang, Shuo Han, Arman Cohan, Hong Yu, and Zonghai Yao. 2025. [From scores to steps: Diagnosing and improving LLM performance in evidence-based medical calculations](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 10809–10833, Suzhou, China. Association for Computational Linguistics.

Yuxin Wang, Qingxuan Sun, and Sicheng He. 2023. M3e: Moka massive mixed embedding model. <https://huggingface.co/moka-ai/m3e-base>. Model card.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

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*.

Ailing Yu, Lan Yao, Jingnan Liu, Zhe Chen, Jiajun Yin, Yuan Wang, Xinhao Liao, Zhiling Ye, Ji Li, Yun Yue, et al. 2025. Medresearcher-r1: Expert-level medical deep researcher via a knowledge-informed trajectory synthesis framework. *arXiv preprint arXiv:2508.14880*.

Yunyan Zhang, Zhihong Zhu, and Xian Wu. 2025. [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 25650–25659, Suzhou, China. Association for Computational Linguistics.

Zhengyun Zhao, Qiao Jin, Fangyuan Chen, Tuorui Peng, and Sheng Yu. 2022. Pmc-patients: A large-scale dataset of patient summaries and relations for benchmarking retrieval-based clinical decision support systems. *arXiv preprint arXiv:2202.13876*.

Shuyan Zhou, Uri Alon, Frank F. Xu, Zhengbao Jiang, and Graham Neubig. 2023. [Docprompting: Generating code by retrieving the docs](#). In *The Eleventh International Conference on Learning Representations*.

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, Albuquerque, New Mexico. Association for Computational Linguistics.

Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. 2023. [ToolQA: A dataset for LLM question answering with external tools](#). In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

References

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. 2025. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*.

Lingxiao Chen. 2020. Overview of clinical prediction models. *Annals of translational medicine*, 8(4):71.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. 2023. [Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks](#). *Transactions on Machine Learning Research*.

Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. 2009. [Reciprocal rank fusion outperforms condorcet and individual rank learning methods](#). In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '09*, page 758–759, New York, NY, USA. Association for Computing Machinery.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In *International conference on machine learning*, pages 10764–10799. PMLR.

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.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*.

- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. [Measuring massive multitask language understanding](#). In *International Conference on Learning Representations*.
- Yue Huang, Jiawen Shi, Yuan Li, Chenrui Fan, Siyuan Wu, Qihui Zhang, Yixin Liu, Pan Zhou, Yao Wan, Neil Zhenqiang Gong, and Lichao Sun. 2024. [Meta-tool benchmark for large language models: Deciding whether to use tools and which to use](#). In *The Twelfth International Conference on Learning Representations*.
- Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421.
- Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. [PubMedQA: A dataset for biomedical research question answering](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577, Hong Kong, China. Association for Computational Linguistics.
- Qiao Jin, Zhizheng Wang, Yifan Yang, Qingqing Zhu, Donald Wright, Thomas Huang, Nikhil Khandekar, Nicholas Wan, Xuguang Ai, W John Wilbur, et al. 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, et al. 2024. Medcalc-bench: Evaluating large language models for medical calculations. *Advances in Neural Information Processing Systems*, 37:84730–84745.
- Haoyang Li, Xuejia Chen, Zhanchao Xu, Darian Li, Nicole Hu, Fei Teng, Yiming Li, Luyu Qiu, Chen Jason Zhang, Li Qing, and Lei Chen. 2025. [Exposing numeracy gaps: A benchmark to evaluate fundamental numerical abilities in large language models](#). In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20004–20026, Vienna, Austria. Association for Computational Linguistics.
- Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.
- Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.
- Karel GM Moons, Douglas G Altman, Johannes B Reitsma, John PA Ioannidis, Petra Macaskill, Ewout W Steyerberg, Andrew J Vickers, David F Ransohoff, and Gary S Collins. 2015. Transparent reporting of a multivariable prediction model for individual prognosis or diagnosis (tripod): explanation and elaboration. *Annals of internal medicine*, 162(1):W1–W73.
- Meredith Ringel Morris, Jascha Sohl-Dickstein, Noah Fiedel, Tris Warkentin, Allan Dafoe, Aleksandra Faust, Clement Farabet, and Shane Legg. 2023. Levels of agi for operationalizing progress on the path to agi. *arXiv preprint arXiv:2311.02462*.
- Harsha Nori, Nicholas King, Scott Mayer McKinney, Dean Carignan, and Eric Horvitz. 2023. Capabilities of gpt-4 on medical challenge problems. *arXiv preprint arXiv:2303.13375*.
- Ankit Pal, Logesh Kumar Umapathi, and Malaikanan Sankarasubbu. 2022. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on health, inference, and learning*, pages 248–260. PMLR.
- 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.
- Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, et al. 2023. Large language models encode clinical knowledge. *Nature*, 620(7972):172–180.
- Benlu Wang, Iris Xia, Yifan Zhang, Junda Wang, Feiyun Ouyang, Shuo Han, Arman Cohan, Hong Yu, and Zonghai Yao. 2025. [From scores to steps: Diagnosing and improving LLM performance in evidence-based medical calculations](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 10809–10833, Suzhou, China. Association for Computational Linguistics.
- Yuxin Wang, Qingxuan Sun, and Sicheng He. 2023. M3e: Moka massive mixed embedding model. <https://huggingface.co/moka-ai/m3e-base>. Model card.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.
- 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*.
- Ailing Yu, Lan Yao, Jingnan Liu, Zhe Chen, Jiajun Yin, Yuan Wang, Xinhao Liao, Zhiling Ye, Ji Li,

Yun Yue, et al. 2025. Medresearcher-r1: Expert-level medical deep researcher via a knowledge-informed trajectory synthesis framework. *arXiv preprint arXiv:2508.14880*.

Yunyan Zhang, Zhihong Zhu, and Xian Wu. 2025. 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 25650–25659, Suzhou, China. Association for Computational Linguistics.

Zhengyun Zhao, Qiao Jin, Fangyuan Chen, Tuorui Peng, and Sheng Yu. 2022. Pmc-patients: A large-scale dataset of patient summaries and relations for benchmarking retrieval-based clinical decision support systems. *arXiv preprint arXiv:2202.13876*.

Shuyan Zhou, Uri Alon, Frank F. Xu, Zhengbao Jiang, and Graham Neubig. 2023. *Docprompting: Generating code by retrieving the docs*. In *The Eleventh International Conference on Learning Representations*.

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, Albuquerque, New Mexico. Association for Computational Linguistics.

Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. 2023. ToolQA: A dataset for LLM question answering with external tools. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

A Appendix

A.1 Dataset Verification and Inter-Clinician Agreement

Full-data verification. After fuzzy-query generation, all cases are checked and revised by three trained student annotators. For each case, the annotators compare the generated query, gold calculator set, gold parameters, and expected calculator outputs against the original clinical record.

Independent clinician audit. Following the full-data review, three clinicians independently audit 20% of the data. Each sampled case is evaluated for both query validity and answer validity according to the criteria in Table 6.

Each clinician assigns a binary true/false judgment to both dimensions. A clinician-level judgment is considered valid only when both query validity and answer validity are labeled true. A case

Dim.	Result Criterion	
Query validity	Pass	The query preserves the original clinical goal, is natural and clear, and can be answered using the case.
	Fail	The query deviates from the original clinical goal, is unclear, or cannot be answered using the case.
Answer validity	Pass	The required parameters are supported by the case and do not contradict the clinical evidence.
	Fail	Unsupported or contradictory parameters are included, or key parameters conflict with the case.

Table 6: Criteria used by clinicians to evaluate query and answer validity.

passes the clinician audit when at least two of the three clinicians judge it as valid.

Revision process and revision rate. Under our predefined protocol, an audit pass rate below 95% triggers another full-data review, followed by a second clinician audit. As shown in Table 7, the first-round overall pass rate is 93.13%, below the predefined threshold. Therefore, the three trained student annotators conduct a second full-data review and revise 83 of the 1,048 single-calculator cases and 115 of the 1,158 multi-calculator cases. In total, 198 of the 2,206 single- and multi-calculator cases are revised during this stage. The second-round clinician audit achieves an overall pass rate of 97.33%, exceeding the predefined quality threshold.

Audit Round	Overall Pass Rate (%)
Round 1	93.13
Round 2	97.33

Table 7: Overall pass rates in the two clinician-audit rounds.

Inter-clinician agreement. To quantify agreement among the three clinicians, we compute Gwet’s AC1 for query validity, answer validity, and case-level validity. For inter-clinician agreement analysis, we select a common subset of 262 audited cases, each independently evaluated by all three clinicians. In Table 8, 0T–3T denote the number of clinicians who assign a valid (true) judgment to a case. Cases receiving 2T or 3T are considered valid under majority voting. All values are reported as percentages.

The majority-vote pass rates reach 99.62%, 97.33%, and 97.33% for query validity, answer validity, and case-level validity, respectively. The

Dimension	0T	1T	2T	3T	MV Pass	Gwet's AC1
Query validity	0.38	0.00	4.96	94.66	99.62	96.55
Answer validity	1.15	1.53	4.20	93.13	97.33	95.90
Case-level validity	1.53	1.15	9.16	88.17	97.33	92.36

Table 8: Inter-clinician agreement and majority-vote pass rates (%). MV Pass denotes the majority-vote pass rate.

corresponding Gwet’s AC1 values of 96.55%, 95.90%, and 92.36% indicate very high inter-clinician agreement. These results show that the revised data satisfy the predefined quality threshold and that the clinician judgments are highly consistent.

A.2 Hyperparameter Analysis

The retrieval stage involves two key hyperparameters: K_{route} and K_{cand} . K_{route} denotes the retrieval depth of each route, namely the number of candidates recalled by each retrieval branch, while K_{cand} denotes the maximum size of the final candidate pool passed to the downstream Tool Selection stage. We perform three independent grid searches on the multi-calculator tasks with $K_{\text{cand}} \in \{8, 16, 32, 64\}$ and $K_{\text{route}} \in \{8, 16, 32, 77\}$.

Firstly, we determine the optimal K_{cand} by comparing the average best T-F1 across runs (Table 9). Although T-F1 increases monotonically with larger candidate budgets, the marginal gain quickly diminishes: when K_{cand} increases from 32 to 64, the average T-F1 only improves from 64.57% to 65.73% (+1.16%). Meanwhile, the prompt length in the tool selection stage grows almost proportionally with the number of candidate tool descriptions, making the token consumption with $K_{\text{cand}} = 64$ nearly twice that of $K_{\text{cand}} = 32$. Considering the minimal performance gain and substantially higher inference cost, we adopt $K_{\text{cand}} = 32$.

With K_{cand} fixed, we further analyze the effect of the retrieval depth K_{route} (Table 10). The best performance occurs at $K_{\text{route}} = 16$, while larger values bring no additional benefit and slightly degrade performance. Overall, $K_{\text{cand}} = 32$ and $K_{\text{route}} = 16$ provide a good balance between candidate coverage and quality.

A.3 Manual Evaluation Rubric

Each response is scored on five dimensions using a three-level rubric. Scores of 2, 1, and 0 indicate

K_{cand}	run1	run2	run3	avg	Δavg
8	59.03	59.13	59.03	59.06	−6.67
16	62.77	62.46	62.37	62.53	−3.20
32	64.66	64.63	64.43	64.57	−1.16
64 [†]	65.95	65.73	65.53	65.73	−

Table 9: Validation T-F1 under different K_{cand} values. Δavg denotes the difference from the $K_{\text{cand}} = 64$ setting. [†] Near-unconstrained candidate retention setting.

K_{route}	Precision	Recall	T-F1
8	62.52	66.61	64.50
16	62.46	66.93	64.62
32	62.17	66.43	64.23
77	62.31	67.02	64.62

Table 10: Effect of K_{route} when $K_{\text{cand}} = 32$.

full satisfaction, partial satisfaction, and failure to satisfy the corresponding criterion, respectively.

A.4 Fully Annotated Error-Analysis Cases

We select three cases whose errors can be reproduced directly from the clinical record and the current retrieval, selection, and execution traces. We exclude discrepancies caused only by tool or parameter aliases, value-format differences, or inconsistent reference annotations. Together, the cases cover the six semantic categories in Table 4: Case 1 covers Retrieval Miss and Over-selection, Case 2 covers Selection Error and Unit Conversion Error, and Case 3 covers Parameter Value Error and Enum Mapping Error. Downstream calculation changes are reported as consequences rather than as an additional error category.

A.5 Calculator-Type Analysis

We further categorize calculators as equation-based or rule-based. Equation-based calculators derive numerical results from explicit mathematical formulas, whereas rule-based calculators map clinical findings to predefined categorical rules. Table 12 reports performance by calculator type, where gold-tool occurrences count the number of annotated calculator appearances across cases.

Rule-based calculators achieve 4.54 percentage points higher Tool Selection F1 than equation-based calculators, but their Extraction Accuracy and Score Accuracy are lower by 20.35 and 6.83 points, respectively. This contrast indicates that rule-based calculators are generally identifiable from their clinical purpose, while their execution is more sensitive to mapping free-text evidence,

ID	Dimension	2 Points	1 Point	0 Points
O1	Clinical-goal coverage and completeness	Fully addresses all requested calculation goals and reports all required results.	Addresses the main goal but omits some calculators, results, or intermediate outputs.	Misses the core calculation goal or essential results.
O2	Parameter extraction and evidence consistency	All key parameters are supported by the case, with correct values, units, categories, and timing.	Most parameters are supported, with only minor or non-critical issues.	Key parameters are missing, incorrect, contradictory, or unsupported.
O3	Result presentation and interpretability	Clearly reports the score, unit, label or risk category, and necessary explanation.	Reports the main result but provides incomplete units, labels, evidence, or explanation.	Omits essential results or provides an unclear or inconsistent explanation.
O4	Nested-calculator dependency handling	Correctly identifies and executes the parent-sub dependency and propagates the intermediate result.	Generally correct, but the execution order, intermediate result, or dependency explanation is incomplete.	Fails to identify or execute the dependency correctly.
O5	Clinical plausibility and usability	Clinically reasonable and useful without overinterpreting the calculation as diagnosis or treatment.	Generally reasonable but lacks context, limitations, uncertainty, or review guidance.	Clinically implausible, misleading, or potentially unsafe.

Table 11: Rubric used for manual response-quality evaluation. O1, O2, O3, and O5 are applied to all task settings, while O4 is applied only to nested-calculator cases.

Case 1: Retrieval Miss and Over-Selection

Physician query. For this patient with possible atrial-fibrillation-related retinal artery occlusion, which tools should be used to assess stroke risk and blood-pressure management?

Clinical record. An 88-year-old man presented with acute painless monocular vision loss. His history included hypertension, coronary artery disease, and atrial fibrillation treated with apixaban. Admission vital signs included blood pressure 123/53 mmHg and heart rate 62 beats/min. Neurological examination yielded an NIHSS score of 2 for complete unilateral vision loss. Ophthalmologic examination showed a retinal plaque and ischemia, while carotid imaging showed no severe stenosis or thrombus. Echocardiography showed an ejection fraction of 40%. The workup found no evidence of carotid disease or vasculitis, and the event was considered cardioembolic because of atrial fibrillation and multiple missed anticoagulant doses.

Representative retrieved candidates. The gold tools are `chads2_vasc` and `mean_arterial_pressure_map`. The 32-tool candidate pool contains `chads2_vasc` at rank 1 and `abcd2` at rank 3, but does not contain `mean_arterial_pressure_map`. The record directly provides the blood pressure required for MAP calculation, and the query separately requests atrial-fibrillation-related stroke-risk and blood-pressure assessment.

Final selection and execution. The model selects `chads2_vasc` and `abcd2`. CHA₂DS₂-VASC is executed with heart failure = no, hypertension = yes, age = 88, diabetes = no, prior stroke = no, vascular disease = yes, and male sex, producing a score of 4. ABCD2 is executed with age 88, blood pressure 123/53 mmHg, clinical presentation = other, symptom duration = 1,440 minutes, and diabetes = no, producing a score of 3.

Verified model error.

- **E1—Retrieval Miss.** MAP is a gold tool supported by the explicit blood-pressure-management goal and the recorded value of 123/53 mmHg, but it is absent from all 32 retrieved candidates and is therefore unavailable to the selector.
- **E3—Over-Selection.** The model additionally selects ABCD2, which estimates short-term risk after transient ischemic attack. It does not serve the requested atrial-fibrillation stroke-risk or blood-pressure-management goals and is not a gold tool.

Root cause. The retrieval stage fails to represent the blood-pressure subgoal, while the selector overgeneralizes “stroke risk” and adds a TIA-specific calculator. A stale automatic enum-mapping label for vascular disease is not retained because the current trace correctly maps coronary artery disease to vascular disease = yes and produces the correct CHA₂DS₂-VASC score.

Calculator Type	Gold Occ.	T-F1	Extract	Score
Equation-based	2,941	73.38	62.09	68.99
Rule-based	883	77.92	41.74	62.16

Table 12: Performance breakdown by calculator type (%). Gold Occ. denotes gold-tool occurrences.

negations, and severity descriptions into categorical or enumerated inputs. Equation-based calculators rely more heavily on explicit numerical variables, which are comparatively easier to identify and normalize. These results further motivate structured enum validation and evidence audit, particularly for rule-based tools.

A.6 Bootstrap Confidence-Interval Analysis

We conduct case-level bootstrap analysis and report 95% confidence intervals for Score Accuracy

across the four agent frameworks and three task settings. All values in Table 13 are percentages, with point estimates followed by confidence intervals in brackets.

Method	Single-Calculator	Multi-Calculator	Nested-Calculator
Ours	80.15 [77.67, 82.54]	59.53 [58.08, 61.50]	75.80 [64.52, 85.58]
MeNTi	58.40 [55.77, 62.69]	15.61 [14.25, 17.27]	61.29 [50.35, 71.58]
ReAct	55.06 [52.10, 58.21]	37.57 [35.30, 39.94]	11.74 [3.64, 19.42]
MedRaC	75.38 [62.85, 78.00]	13.71 [12.61, 15.32]	0.00 [0.00, 0.00]

Table 13: Case-level bootstrap 95% confidence intervals for Score Accuracy (%).

Among the four agent frameworks, our method achieves the highest point estimate in all three settings. In the multi-calculator setting, its confidence interval is fully separated from those of all baselines, indicating a stable advantage under resampling. In the single-calculator setting,

Case 2: Selection and Unit-Conversion Errors

Physician query. For this patient with non-resolving pneumonia and severely impaired pulmonary function, which tools should be used to assess infection severity, prognosis, and possible acid–base disturbance?

Clinical record. A 29-year-old woman presented with a month of progressive dyspnea, productive cough, fever, chills, and body aches despite prior treatment for pneumonia and influenza. Admission findings included heart rate 118 beats/min, blood pressure 111/72 mmHg, respiratory rate 24 breaths/min, and oxygen saturation 88% on room air. Laboratory values included sodium 136 mEq/L, chloride 98 mEq/L, bicarbonate 21 mEq/L, blood urea nitrogen 15 mg/dL, creatinine 1.0 mg/dL, and glucose 113 mg/dL. Imaging showed infectious or inflammatory tree-in-bud changes. The patient had persistent hypoxia and non-resolving pneumonia despite antimicrobial and steroid therapy. Pulmonary function testing later showed combined severe obstructive and restrictive disease with markedly reduced diffusion capacity.

Representative retrieved candidates. The gold tools are `curb65`, `delta_gap`, and `delta_ratio`. All are retrieved: CURB-65 at rank 1, Delta Ratio at rank 8, and Delta Gap at rank 11. The record supplies the electrolyte values needed for the acid–base calculations and reports blood urea nitrogen as 15 mg/dL.

Final selection and execution. The model selects `curb65`, `albumin_corrected_anion_gap`, and `anion_gap`, but omits both retrieved Delta tools. For CURB-65, it passes $\text{urea} = 15 \text{ mmol/L}$ directly from the recorded blood urea nitrogen value of 15 mg/dL and returns a score of 1. It also computes an anion gap of 17 and an albumin-corrected anion gap of 17 using an assumed albumin value of 4.0 g/dL.

Verified model error.

- **E2—Selection Error.** Delta Gap and Delta Ratio are both present in the candidate pool and directly match the acid–base-assessment goal, but the selector omits them and chooses anion-gap alternatives instead.
- **E5—Unit Conversion Error.** CURB-65 requires urea in mmol/L, whereas the record reports blood urea nitrogen in mg/dL. The correct conversion is approximately $15/2.8 = 5.36 \text{ mmol/L}$; the model instead passes 15 as mmol/L without conversion.
- **Downstream consequence.** The conversion error incorrectly activates the CURB-65 urea criterion, changing the total score from 0 to 1. This score change is not counted as a separate error category.

Root cause. The selector fails to map the acid–base goal to the two retrieved Delta calculators, and the parameter-normalization stage conflates a BUN value in mg/dL with a urea value in mmol/L.

Case 3: Parameter-Value and Enum-Mapping Errors

Physician query. Which tools should be used to assess consciousness and hemorrhage severity in this patient with subarachnoid hemorrhage?

Clinical record. A 60-year-old woman presented with thunderclap headache, lightheadedness, and neck stiffness. Her initial neurological examination was normal, but over the next 24 hours she became drowsy while remaining arousable to voice, without focal deficits. CT showed diffuse subarachnoid hemorrhage with intraventricular hemorrhage and mild hydrocephalus. She subsequently became obtunded, required intubation, and underwent external ventricular drainage and aneurysm coiling. After extubation she initially recovered without neurological deficit. Later examinations during readmission were unremarkable.

Gold tools and retrieval. The gold tools are `hunt_hess_scale` and `gcs`, both of which are retrieved and selected. Hunt–Hess is ranked first and GCS fifth. Evaluation is anchored to the deterioration after the hemorrhage, when the patient was drowsy but arousable to voice, rather than to the normal initial examination or subsequent recovery.

Final selection and execution. The model executes Hunt–Hess with grade = 2 and returns 2. It executes GCS with eye opening = spontaneous, verbal response = oriented, and motor response = obeys commands, returning 15. These values reflect an earlier or later normal examination rather than the target deterioration period.

Verified model errors.

- **E4—Parameter Value Error.** The model assigns Hunt–Hess grade 2 from the earlier headache-and-neck-stiffness presentation. The benchmark target is grade 3, corresponding to the subsequent drowsy state, so the extracted grade is anchored to the wrong clinical time point.
- **E6—Enum Mapping Error.** The target GCS state maps eye opening to sound, confused verbal response, and obeying commands. The model instead maps recovery-period language to spontaneous eye opening and oriented verbal response, producing a normal GCS of 15.

Root cause. The model does not maintain a consistent temporal anchor across the clinical trajectory. It uses evidence from the initial or recovery examinations instead of mapping the neurological state during acute deterioration to the calculator’s numerical and categorical inputs. The values reported here follow the current execution trace; superseded automatic-log values are not used.

our method also achieves the highest point estimate, although its interval narrowly overlaps with that of MedRaC, making the separation less decisive than in the multi-calculator setting. In the nested-calculator setting, our method exceeds the strongest baseline, MeNTi, by 14.51 percentage points. The intervals overlap, however, and are relatively wide because the nested evaluation contains only 62 cases; this highlights the greater uncertainty associated with the smaller test set. MedRaC’s degenerate interval at zero reflects its failure to produce any correct nested-calculator

scores.

Difficulty	Parameters	Tools	Extract	Score
Easy	≤ 3	2750	65.22	68.91
Medium	4–7	817	47.52	53.98
Hard	> 7	299	36.32	42.81

Table 14: Tool-level performance by difficulty.

A.7 Tool-Level Analysis by Difficulty.

To better understand how calculator complexity affects system performance, we conduct a tool-level complexity analysis by grouping calcula-

tors according to the number of required input parameters. The corresponding tool-level results are reported in Table 14. The results show a clear performance decline as tool complexity increases. For Easy tools, Extract and Score reach 65.22% and 68.91%, respectively. When the difficulty increases to Medium, these metrics drop by 17.70% and 14.93%, and further decrease by 28.90% and 26.10% for Hard tools relative to the Easy group. This trend indicates that as the number of required parameters grows, the model faces increasing difficulty in accurate parameter extraction, and extraction errors are more likely to propagate through downstream computation, leading to lower final score accuracy. These findings highlight tool complexity as a key factor affecting system performance and further emphasize the importance of structured validation and auditing mechanisms for stabilizing execution in complex tools.

A.8 Cost-Efficiency Analysis

To further analyze the computational cost of different methods, we conduct a component-level efficiency analysis. Specifically, we randomly sample 100 cases from the single-calculator dataset and 100 cases from the multi-calculator dataset, and include all 62 cases from the nested-calculator dataset. For each case, we record the end-to-end running time as well as the total number of input and output tokens consumed during inference. The averaged results are reported in Table 15.

Method	Avg Time	Score Acc.	Avg Input Tokens	Avg Output Tokens
MeNTi	38.76	30.86	8,940	4,439
MedRaC	13.53	24.77	3,149	1,711
ReAct	53.29	31.31	59,136	3,382
Ours	37.14	63.06	22,946	4,857

Table 15: Comparison of performance and inference cost across methods. Avg Time is measured in seconds, Score Acc. is reported in percentage, and Avg Input/Output Tokens denote the average number of input/output tokens.

B Prompt Design for Key Components

This section presents the detailed prompts used in several key components of our framework, including query rewriting, tool selection, dependency identification, and fuzzy query generation during data construction. These prompts are provided to enhance transparency and facilitate reproducibility of our system.

C Inference Example

This section presents a representative clinical case to illustrate the end-to-end execution process of the proposed framework. The example demonstrates how the system handles a multi-calculator scenario, including query rewriting, multi-route retrieval, tool selection, and dependency-aware execution, providing a detailed view of each stage in the pipeline.

B.1 Prompt: Query Rewriting

You are assisting a retrieval system for medical scales and clinical calculators.

Task:

Based on the [Physician Query] and [Patient Case], generate exactly 3 search phrases for semantic retrieval, each corresponding to one of three index dimensions:

- Core Function
- Clinical Context
- Key Input Variables.

Goal: The phrases must enable retrieval of the correct scale/calculator.

Input:

Physician Query: {INSERT_QUERY_HERE} **Patient Case:** {INSERT_CASE_HERE}

Index Dimensions:

Core clinical function (what clinical state, risk, severity, or outcome is assessed)

Target population and clinical context (which patient group, care setting, or stage)

Key input variables (core variables, risk factors, symptom items, or formula inputs typically required by this class of tool)

Mandatory Rules

[Master Principle: Intent Lock (Focus Lock)]

A) First extract the single assessment target from the physician query (e.g., pneumonia severity / AF stroke risk / VTE risk stratification / GFR estimation / upper GI bleeding triage / depression severity / pre-operative cardiac risk / MAP calculation).

B) All three phrases must center on this target. Other diagnoses, tests, or scales in the case that do not change "which tool should be used" must **NOT** be included.

[Forbidden]

1) No scale, score, or rule names in any of the three phrases (e.g., APACHE, Caprini, Wells, PERC, NIHSS, GCS, PHQ are all forbidden).

2) Phrase 3 (Key Variables) must **NOT** simply list prominent variables from the case; list only variables that are strongly tied to the assessment target and typically used by this class of tool.

[Task Type Templates — for guiding Phrase 3 content]

- T1 Risk Prediction/Stratification: Phrase 3: risk factors, comorbidities, exposures, surgical/activity status.
- T2 Severity/Prognosis Scoring: Phrase 3: vital signs + key lab values + mental status / organ function.
- T3 Symptom Scales: Phrase 3: symptom items, frequency, duration, functional impact.
- T4 Physiological/Lab Calculators: Phrase 3: formula input variables (e.g., SBP/DBP, serum glucose/sodium, creatinine/age/sex, last menstrual period).
- T5 Diagnostic Probability/Exclusion Rules: Phrase 3: key discriminating features + clinical signs/lab results.

[Phrase Generation Requirements]

Output **ONLY** a JSON array of exactly 3 strings, in the following fixed order:

- **(1) Core Function Query:**
Describe only the assessment target (risk/severity/probability/calculated quantity).
Include common abbreviations/synonyms (e.g., CAP/pneumonia; GFR/glomerular filtration rate; VTE/DVT/PE), but **NO scale names**.
Do not include patient background or specific variables.
- **(2) Clinical Context Query:**
Describe the population and care stage/setting (e.g., emergency / ICU / pre-operative / post-operative / inpatient / outpatient; adult / pediatric / pregnant).
No specific variables, no score values, no scale names.
Include only context strongly relevant to tool selection.
- **(3) Key Variables Query:**
List 5-10 variable types/risk factors/symptom items/formula inputs, comma-separated.
Prioritize variables that appear in the case AND are strongly relevant to the assessment target; standard tool variables may also be included.
Do **NOT** list variables or imaging findings irrelevant to the assessment target.

Output format:

A single JSON array only, in the following format:

["Core Function Query here", "Clinical Context Query here", "Key Variables Query here"]

B.2 Prompt : Tool Selection

You are a rigorous Clinical Decision Support System (CDSS) tool selector.

Task:

Based on the physician's query intent and patient record, select the most appropriate medical scale(s)/calculator(s) from the [Candidate Tool List].

[Candidate Tool List]

(selection must come exclusively from this list; copy strings verbatim)

INSERT_TOOL_LIST_HERE

[Physician Query]

INSERT_QUERY_HERE

[Patient Case]

INSERT_CASE_HERE

INSERT_MAX_TOOLS_CONSTRAINT_HERE

Selection Rules (mandatory):

1. **[Highest Priority — Exact Name Match]** If the physician query explicitly names a specific tool or scale, you **MUST** identify the tool whose name exactly corresponds in the candidate list. Do **NOT** substitute a similarly named but functionally different tool.
2. **[Computability Check]** For each candidate tool, check whether its Required Inputs can be reliably supported by the patient case text (including summary, lab results, physical exam, and explicitly negated findings).
3. **[Tool Selection]** Output only one tool for single-tool queries. For multi-tool queries, only select tools that are necessary and applicable.
4. **[Nested Scale Detection]** If a selected tool depends on another tool's output (child tool), declare the dependency in the "dependencies" field.

Output Requirements (strict JSON; no additional text):

- Output a single JSON object. No Markdown, no explanatory prefix or suffix.
- "selected_tools" must be a string array (may be empty); contains only parent tools.
- "dependencies" must be an object: { "parent_tool_id": { "param": "child_tool_id" } }; use {} if no dependencies.
- "analysis" should be brief (less than 80 words); do not enumerate every candidate.

Output JSON structure:

```
{
  "analysis": "brief rationale here...",
  "selected_tools": [
    "Tool ID A",
    "Tool ID B"
  ],
  "dependencies": {
    "Tool ID A": {
      "param_name_in_A":
        "sub_tool_id"
    }
  }
}
```

B.3 Prompt: Dependency Identification

You are a strict clinical tool dependency checker.

Task:

For each selected parent tool, examine its Required Inputs parameter list. Determine whether any input parameter represents a composite score that must be COMPUTED by another medical calculator (rather than directly extracted from the patient record).

[Doctor Query]

INSERT_QUERY_HERE

[Patient Case]

INSERT_CASE_HERE

[Selected Parent Tools (with their Required Inputs)]

INSERT_SELECTED_TOOLS_HERE

[Candidate Tool List]

INSERT_TOOL_LIST_HERE

[Existing Dependencies (may be empty)]

INSERT_EXISTING_DEPENDENCIES_HERE

Rules:

1. Only output dependencies where parent tool is in Selected Parent Tools.
2. Child tool does NOT need to be from Candidate Tools — it can be any medical scale/calculator that exists.
3. Do not add child tools into selected tools; only express them in dependencies.
4. For each selected parent tool, examine its Required Inputs. If any input parameter represents a composite score or scale total, that parameter has a dependency on the corresponding sub-tool.
5. A parameter is a dependency if its value must be COMPUTED via another scoring system, not directly read from the patient record. Raw vitals (temperature, blood pressure, heart rate) are NOT dependencies; scale totals (GCS score, NIHSS score) ARE dependencies.
6. If no dependency is needed, return {}.
7. Keep exact tool IDs and exact parameter names.

Output format (strict JSON only):

```
{
  "analysis": "short reason",
  "dependencies": {
    "Parent Tool ID": {
      "parameter_name_in_parent": "child_tool_id"
    }
  }
}
```

B.4 Prompt: Fuzzy Query Generation

You are a medical benchmark dataset curator.

Task:

Given a patient case and the ground-truth medical scale(s) that should be applied, write one physician query that naturally leads a clinician to select the specified scale(s). Do **NOT** mention the scale name(s) or acronym(s) explicitly.

[Patient Case]

{INSERT_CASE_HERE}

[Ground-Truth Scale(s)]

{INSERT_TOOLS_HERE}

[Scale Description(s)]

{INSERT_TOOL_DESC_HERE}

[Key Parameters the Scale Requires]

{INSERT_MEASUREMENTS_HERE}

Requirements:

1. **Clinical authenticity** — The query must sound like a real clinical communication (e.g., attending instruction, referral question, or bedside decision prompt); avoid academic or test-like phrasing.
2. **No scale name leakage** — Do **NOT** mention the scale/tool name, acronym, or any sub-score name.
3. **Answerability** — The query must be answerable using the patient case and the ground-truth scale(s); do not introduce information needs absent from the case.
4. **Language** — Output in English.

Output format:

A single plain string only, with no JSON wrapping and no extra text.

<query>Your physician query here.</query>

C.Full Execution Trace of a multi-calculator Case

Doctor Query:

For this patient with a history of atrial fibrillation, which tools should be used to assess stroke risk, the degree of neurological deficits, and functional recovery?

1. Query Rewriting:

The system rewrote the original query into three auxiliary retrieval queries:

- (1) *stroke risk stratification, neurological deficit severity, and functional recovery assessment*
- (2) *adult patients with atrial fibrillation presenting with acute neurological symptoms in emergency or inpatient setting*
- (3) *age, hypertension, diabetes mellitus, prior stroke/TIA, heart failure, sex, systolic blood pressure, level of consciousness, motor strength, speech impairment*

Together with the original query, these formed four retrieval routes.

2. Multi-Route Retrieval:

For each route, the system retrieved a ranked list of candidate tools.

Route 1 top-5: NIHSS, mRS Score, ECOG Performance Status, GCS, Trauma Index

Route 2 top-5: PHQ-9, CURB-65, NIHSS, ABCD2, MEWS

Route 3 top-5: NYHA, Framingham Risk, CURB-65, APACHE II Score, PHQ-9

Route 4 top-5: NIHSS, ABCD2, GCS, mRS Score, Hunt–Hess Scale

After fusion and reranking, the final candidate set contained 32 tools, including NIHSS, mRS Score, CHA₂DS₂-VASc, ABCD2, GCS, and Framingham Risk.

3. Tool Selection:

The system finally selected the following tools:

- CHA₂DS₂-VASc
- NIHSS
- mRS Score

Selection Rationale:

The query asks for stroke risk, neurological deficit severity, and functional recovery. These correspond naturally to CHA₂DS₂-VASc, NIHSS, and mRS, respectively.

4. Parameter Extraction:

NIHSS (initial extraction):

The first NIHSS extraction produced a complete parameter set, including normal consciousness, normal gaze, no visual loss, no motor drift, no sensory deficit, no aphasia, and dysarthria = severe.

mRS Score:

The system extracted:

- mrs_grade = 0

Supporting evidence:

During the outpatient neurology evaluation, the patient continued to be free of dysarthria, dysphagia, and the stigmata of TIA or CVA.

CHA₂DS₂-VASc:

The extracted parameters were:

- congestive heart failure = no
- hypertension = yes
- age = 73
- diabetes = no
- stroke/TIA history = no
- vascular disease = no
- sex category = female

5. Final Calculator Execution:

NIHSS:

- Final score: 1
- Label: *Minor Stroke*

mRS Score:

- Final score: 0
- Label: *mRS Grade 0*

CHA₂DS₂-VASc:

- Final score: 3
- Label: *High Risk*
- Recommendation: *Oral anticoagulation (OAC) is recommended*