



Large language model agents can use tools to perform clinical calculations



Alex J. Goodell 1, Simon N. Chu 2, Dara Rouholiman 1 & Larry F. Chu 1

Large language models (LLMs) can answer expert-level questions in medicine but are prone to hallucinations and arithmetic errors. Early evidence suggests LLMs cannot reliably perform clinical calculations, limiting their potential integration into clinical workflows. We evaluated ChatGPT's performance across 48 medical calculation tasks, finding incorrect responses in one-third of trials ($n = 212$). We then assessed three forms of agentic augmentation: retrieval-augmented generation, a code interpreter tool, and a set of task-specific calculation tools (OpenMedCalc) across 10,000 trials. Models with access to task-specific tools showed the greatest improvement, with LLaMa and GPT-based models demonstrating a 5.5-fold (88% vs 16%) and 13-fold (64% vs 4.8%) reduction in incorrect responses, respectively, compared to the unimproved models. Our findings suggest that integration of machine-readable, task-specific tools may help overcome LLMs' limitations in medical calculations.

Large language models (LLMs) such as ChatGPT and Med-PaLM have demonstrated competency in the use and application of clinical knowledge, recently answering 90% of US medical licensing exam questions correctly^{1–3}. Unfortunately, confabulated material (“hallucination”) remains a persistent problem in both general and specialized domains of knowledge⁴, limiting the ability of LLMs to be deployed safely in clinical environments. Though not intuitive, one common form of hallucination is the misapplication of simple arithmetic, as even sophisticated models tend to have relatively poor performance on tasks such as single-digit addition. Given the importance of quantitative science in medicine, this inadequacy poses an additional obstacle to LLM's implementation.

Despite their challenges, LLMs have experienced a rapid growth in research and development, giving rise to many new approaches to their most common problems. One simple technique has focused on modification of the LLM's prompt, encouraging it to relay its thinking in stepwise fashion, so-called *chain-of-thought prompting* (CoT). This method has proven an effective approach to complex, multi-step tasks including differential diagnosis⁵. For the task of medical calculation, however, a recent evaluation of CoT vs direct prompting saw no incremental improvement⁶.

Another prominent method, retrieval-augmented generation (RAG), employs an information-gathering step prior to prompt submission, whereby a set of documents are gathered from a database and ranked according to their perceived relevance to the prompt. These are appended to the prompt just prior to submission, providing the model with contextual information to bolster its accuracy. Though this has proven promising for

open-ended and knowledge-based medical questions⁷, retrieval-based solutions alone may be constrained by their reliance on the LLM itself to perform the arithmetic.

One alternative solution to improve the calculation abilities of LLMs involves the use of semi-autonomous “agentic” models. The concept of LLMs as autonomous agents emerged in 2022, demonstrating models' abilities to conceive, plan, and execute tasks⁸. Recent developments have shown that they can learn to use tools, such as conducting web searches or interacting with application program interfaces (APIs) for the exchange of knowledge^{9,10}. GeneGPT, likely the first implementation of a tool-using LLM in medicine, is able to interact with APIs hosted by NCBI, allowing a user to explore genetic data without any specialized database knowledge¹¹. These so-called “tool-augmented language models” have shown superior performance in mathematics benchmarks compared to traditional LLMs. Instead of relying on the underlying neural network architecture to answer the question, LLMs generate a script in a common programming language, execute that script, interpret the results, and relay the solution to the user¹². In late 2023, ChatGPT, a large language model developed by OpenAI, was enhanced with some of this functionality, including the ability to write and execute its own programs for solving complex prompts (“code interpreter”).

Not unlike language models, clinicians often require external tools for complex calculations, many of which are specific to their needs. One of the most popular services, MDCalc, has between three and five million visits per month and is used by 65% of physicians in the US¹³. Its appeal lies in its focus

¹Department of Anesthesiology, Pain, and Perioperative Medicine, Stanford University School of Medicine, Stanford, CA, USA. ²Department of Surgery, University of California, San Francisco School of Medicine, San Francisco, CA, USA. e-mail: agoodell@stanford.edu

on usability, simplicity, and safety features¹⁴. However, even the most intuitive applications can become burdensome if multiple calculations are required or if they require the entry of a large amount of data. This is especially true in perioperative medicine, where risk calculators such as the NSQIP or STS Risk model require 20 or more input parameters^{15,16}.

LLMs offer a uniquely usable interface, driving a dramatic rise in popularity. Indeed, a recent informal study found that 76% of 107 surveyed physicians regularly use LLMs for clinical care and that 40% used them to for treatment planning¹⁷. This trend, in part, has prompted institutions to provide secure versions of LLMs to guard against privacy violations, either through specialized secure cloud provision for proprietary models, or with locally-hosted open-weights models, such as LLaMa^{18,19}. As these models become increasingly integrated into clinical practice, LLMs may be able to offer diagnostic assistance, make risk predictions, provide readable documentation to patients, or provide treatment recommendations as a form of basic clinical decision support²⁰⁻²³. Yet many of these tasks depend on models' quantitative skill, underscoring the importance of understanding LLMs capabilities in this domain.

In this study, we examine these topics in two parts. In part one, we describe an exploratory analysis where evaluate the standard, consumer version of ChatGPT on 48 common clinical calculation tasks with 212 vignettes, simulating how a clinician may prompt a language model. We introduce a novel error classification schema to identify missteps in the reasoning process or arithmetic. After uncovering the most common errors made by the language model, we turn to possible solutions. In part two, our focused analysis, we evaluate how different tools could be applied to augment performance, enriching for difficulty by focusing on the 10 lowest-

scoring calculation tasks. We evaluate five permutations of available tools, including RAG, a code interpreter, and a set of task-specific tools developed by the authors, termed OpenMedCalc. To rigorously evaluate the impact of these tools, we assess both a proprietary model (GPT-4o) as well as an open-weights medium-size model (LLaMa-3.1-70b) with 1,000 unique vignettes (Fig. 1). Lastly, the three physician authors review and annotate 612 responses for error classification, which we release as two datasets (ABACUS-212 and ABACUS-409). To our knowledge, this is the first published study to incorporate a functional clinical calculator tool into a language model.

Results

Exploratory analysis

The initial phase of this work entailed a broad assessment of ChatGPT's performance on a variety of calculation tasks in its commercially-available state, which includes access to both a code interpreter and a web search tool. A total of 48 popular calculators were selected from MDCalc and categorized into one of five calculator types by their clinical role (Table 1). For each chosen calculator, clinicians generated five short fictional patient vignettes which included a request that a specific calculation be performed (see examples in Table 2). These vignettes were presented to ChatGPT over the OpenAI web interface, after which clinicians recorded and graded the model's responses.

Of the 48 calculation tasks tested, six tasks generated atypical answers. ChatGPT was unable to give any substantial answer to four tasks: the GRACE, Gupta, PECARN, and ASCVD. For these prompts, the LLM browsed the internet to find human-readable medical calculators such as

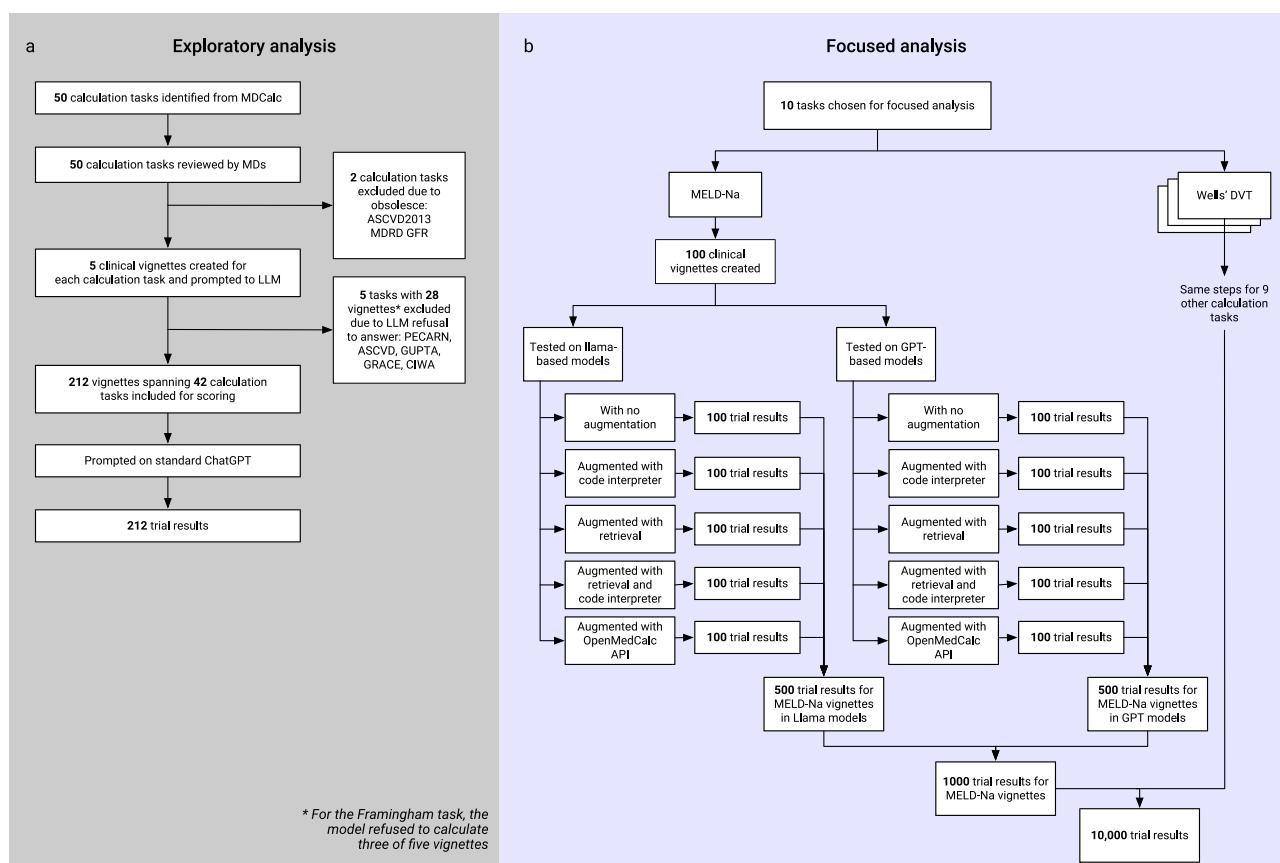


Fig. 1 | Overview of study's approach to assessment of clinical calculation. The schematic delineates the two-phase process. **a** Exploratory analysis of 50 calculation tasks identified from MDCalc. Tasks were reviewed and vetted for suitability, resulting in 212 vignettes spanning 42 tasks after exclusions due to obsolescence or LLM's inability to provide answers. **b** In the focused analysis, in-depth testing of the

LLM's performance on 10 select calculation tasks was performed with both LLaMa and GPT models across five different configurations: base model, code interpreter (CI), retrieval-augmented generation (RAG), RAG + CI, and OpenMedCalc API (OMC), resulting in 10,000 trials.

Table 1 | List of 48 calculators assessed, by calculator type

Calculator Type	Calculators assessed
Descriptive	CrCl (Creatinine Clearance), CKD-EPI Equations (for Glomerular Filtration Rate), FENa (Fractional Excretion of Sodium), NaCorrection (Sodium Correction for Hyperglycemia) MAP (Mean Arterial Pressure), QTc (Corrected QT Interval), LDL Calculator, AG (Anion Gap), BMI/BSA (Body Mass Index and Body Surface Area), IBW (Ideal Body Weight and Adjusted Body Weight), CaCorrection (Calcium Correction for Hypoalbuminemia), FreeWater (Free Water Deficit in Hypernatremia), Fibrosis-4 (for Liver Fibrosis), ABG (Arterial Blood Gas Analyzer), Osm (Serum Osmolality/Osmolarity), HMAIR (Homeostatic Model Assessment for Insulin Resistance), Due Dates (Pregnancy Calculator)
Predictive	CHADS-VASc ^b (CHA ₂ DS ₂ -VASc Score for Atrial Fibrillation Stroke Risk), RCRI (Revised Cardiac Risk Index for Pre-Operative Risk), HEART (HEART Score for Major Cardiac Events), HAS-BLED ^b (HAS-BLED Score for Major Bleeding Risk), CURB-65 (CURB-65 Score for Pneumonia Severity), SOFA ^b (Sequential Organ Failure Assessment Score), Framingham ^a (Framingham Risk Score for Hard Coronary Heart Disease), Caprini ^b (Caprini Score for Venous Thromboembolism), ASCVD ^a (Atherosclerotic Cardiovascular Disease), Gupta ^a (Gupta Perioperative Risk for Myocardial Infarction or Cardiac Arrest), GRACE ^a (GRACE ACS Risk and Mortality Calculator), CCI ^b (Charlson Comorbidity Index), ARISCAT Score ^b (for Postoperative Pulmonary Complications), PSI/PORT Score ^b (Pneumonia Severity Index) CPS (Child-Pugh Score for Cirrhosis Mortality), MELD-Na ^b (Model for End-Stage Liver Disease Sodium)
Speculative	Wells PE (Wells Criteria for Pulmonary Embolism), Wells DVT ^b (Wells Criteria for DVT), Centor (Centor Score for Strep Pharyngitis), PHQ9 (Patient Health Questionnaire), PERC (PERC Rule for Pulmonary Embolism), STOP-BANG (STOP-BANG Score for Obstructive Sleep Apnea)
Summative	GAD7 ^b (General Anxiety Disorder), NIHSS ^b (NIH Stroke Scale/Score), GCS (Glasgow Coma Score), CIWA (CIWA Score for Alcohol Withdrawal), SIRS (SIRS Septic Shock Criteria)
Therapeutic	Steroid Conversion (Steroid Conversion Calculator), mIVF (Maintenance Fluids Calculations), MME (Morphine Milligram Equivalents Calculator), PECARN ^a (Pediatric Head Injury/Trauma Algorithm)

Short names, extended names, and categories of the 48 calculation tasks assessed. Calculators were broken into five types: (A) descriptive, which attempt to estimate the value of an unmeasurable or difficult-to-measure known physical property, (B) speculative, which attempt to estimate the probability of an active disease or process. (C) summative, which combines multiple metrics to summarize the severity of an active, known disease (D) predictive, which estimates the probability or risk of a future event, and (E) therapeutic, which calculates or assists in calculation of a treatment plan. Not shown: Two calculators, ASCVD-2013 and MDRD-GFR, were excluded due to obsolescence.

^aIncomplete responses during exploratory analysis.

^bIncluded in the focused analysis.

Table 2 | Example vignettes

Calculator	Example prompt
PSI/PORT	A 70-year-old male patient is admitted with pneumonia. The patient presents with confusion, disorientation, and inability to recognize family members. They live independently in their own home and manage all personal care. The patient has a history of stage II colon cancer, diagnosed 5 years ago, and is currently in remission. They report no history of liver disease and have normal liver function tests. The patient has congestive heart failure with an ejection fraction of 35% and takes daily diuretics. The patient suffered a stroke 3 years ago, resulting in mild left-sided weakness and aphasia. They have normal kidney function with no history of renal disease. Their vital signs include a respiratory rate of 28, systolic BP of 110, pulse of 130 and temperature of 34.6. Their lab results show a pH of 7.38, BUN of 25, sodium of 135, glucose of 220, hematocrit of 28, PaO ₂ of 55. Chest radiograph reveals bilateral infiltrates and a right-sided pleural effusion. What is their PSI/PORT score?
SOFA	A 58-year-old male is being monitored for organ failure. They are intubated with a 7.0 ETT and are on VCAC. Based on their most recent blood gas, their FiO ₂ is 70%, and their PaO ₂ is 72. Platelets are 110, bilirubin 15, Their MAP is 68, and they are not requiring vasopressor agents. Glasgow Coma Scale is 6, creatinine 6, and urine output 1000 mL/day. Please determine his SOFA score.
MELD-Na	I'm seeking your advice on a liver transplant candidate. The patient is a 40-year-old woman with the following lab results: sodium 125, bilirubin 1, creatinine 1.3, and INR 1. They are on three-times-per-week dialysis. Could you help me determine their MELD-Na score?

Three example vignettes used for prompting during the focused analysis. Examples of full prompts, responses, and error classification are available in the Supplementary Information (Supplementary Note 1 through Supplementary Note 13) as well as the data repository.

MDCalc and offered links to the user as suggestions. For the calculation of CIWA scores, which involve subjective grading of physical exam findings such as perceived anxiety or tremor, the LLM would only provide ranges. Thus, these five tasks were graded “unable to calculate.” For the Framingham Score, three trials resulted in ChatGPT indicating its inability to calculate the score, instead providing links to relevant materials. In two trials, it attempted to provide answers by hallucinating a calculation scheme, one of which resulted in a myocardial infarction risk estimate near 1% and a second with a risk over 100% for similar patients. The first three attempts were coded as “unable to calculate,” while the latter two attempts were scored as incorrect.

Out of the remaining 42 calculators, ChatGPT successfully provided calculations in five of five trials, resulting in a total of 212 scorable answers. Of these, 140 were classified as correct, representing 66% of all trials. There was considerable variation in performance across calculation tasks; in six tasks, none of the provided answers were correct, while in 17 tasks, all provided answers were correct (Fig. 2a). There was a notable difference based on calculator role: for tasks classified as “predictive,” the model was able to accurately answer only 39% of trials, while calculation tasks classified as “descriptive” were answered correctly in 89% of trials (Fig. 2a).

Exploratory analysis error classification

During the exploratory analysis, there were a wide array of errors made by ChatGPT in its attempts to answer clinical calculation tasks; a total of 84 errors were identified across 72 questions. To better understand the nature of these miscalculations, we developed a error classification schema, which separated errors into one of five classes: interpretation error, incorrect criteria, assignment error, incorrect formula, or calculation error (Fig. 3). Each of ChatGPT’s answers was reviewed by a physician and errors were grouped into one of these five classes.

The most common type of error was the *assignment error* (31 of 82 errors, 38%), in which the model correctly identified the relevant criteria but assigned the wrong score. This was more common in longer calculation tasks which had more criteria, making it difficult for the model to “recall” the correct points for each criterion. For example, in the ARISCAT task, the model identified that preoperative SpO₂ was a part of the criteria but was inconsistent on how many points an abnormal value contributed to a final score.

The second most common error class was *incorrect criteria*, where the model either hallucinated new criteria or omitted relevant criteria; this category accounted for 13 of 82 identified errors (16%). Without a clear understanding of criteria, the model would meld variations of criteria from

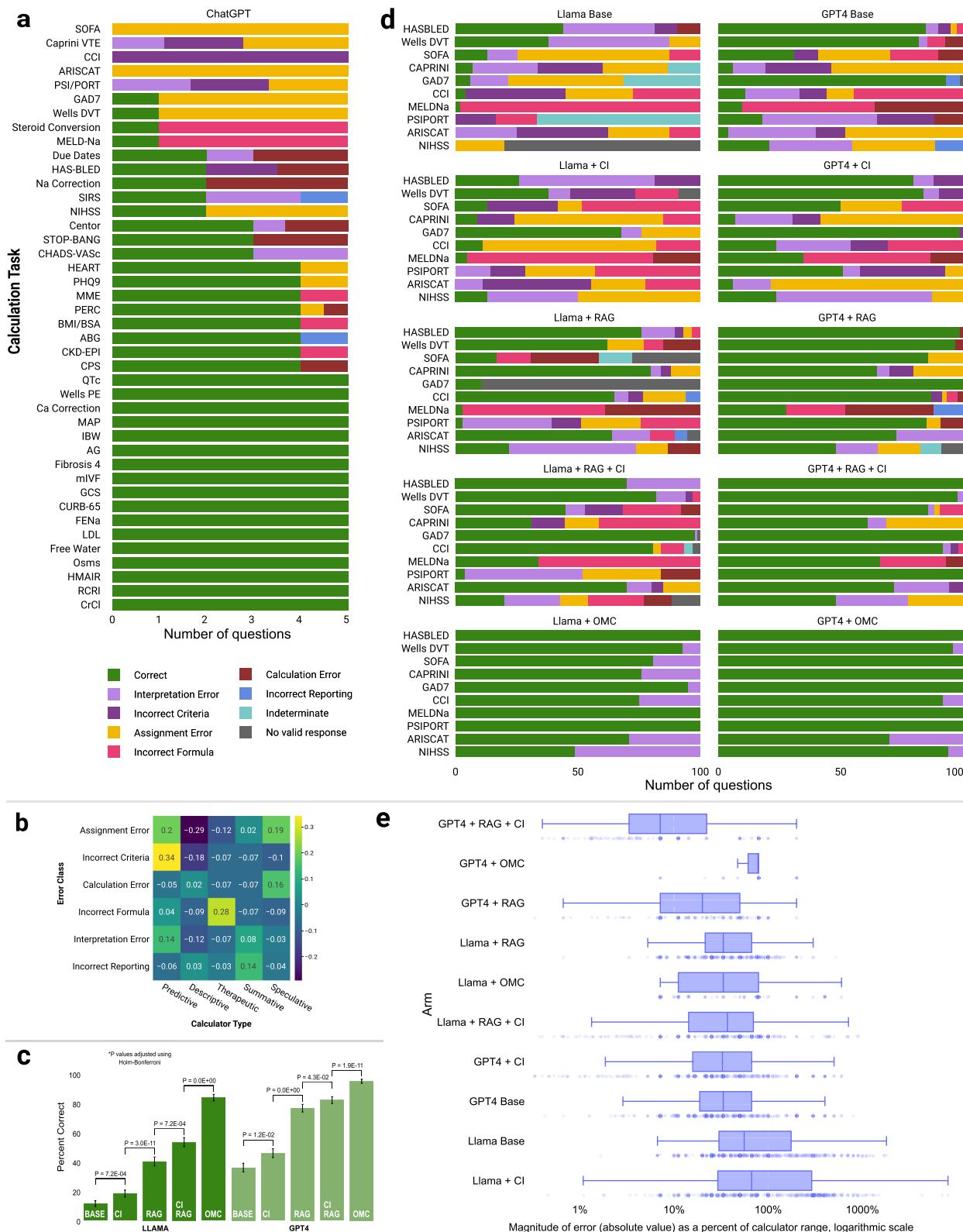


Fig. 2 | Detailed breakdown of ChatGPT's performance on calculation tasks.

a Vertical histogram showing performance on all calculation tasks during exploratory analysis, with correct answers in green and errors color-coded by type.

b Heatmap visualizing correlations between calculator types and error classes, where blue indicates negative correlations and green indicates positive correlations.

c Percentage of correct answers across the 10 selected calculators for each arm of the

focused analysis, showing stepwise improvement with tool augmentation.

d Continuation of vertical histograms from (a) showing performance on all calculation tasks during focused analysis, with correct answers in green and estimated number of errors color-coded by class. e Box plots showing the magnitude of errors when incorrect answers were provided, normalized to each calculator's valid range, demonstrating reduced error severity with tool augmentation.

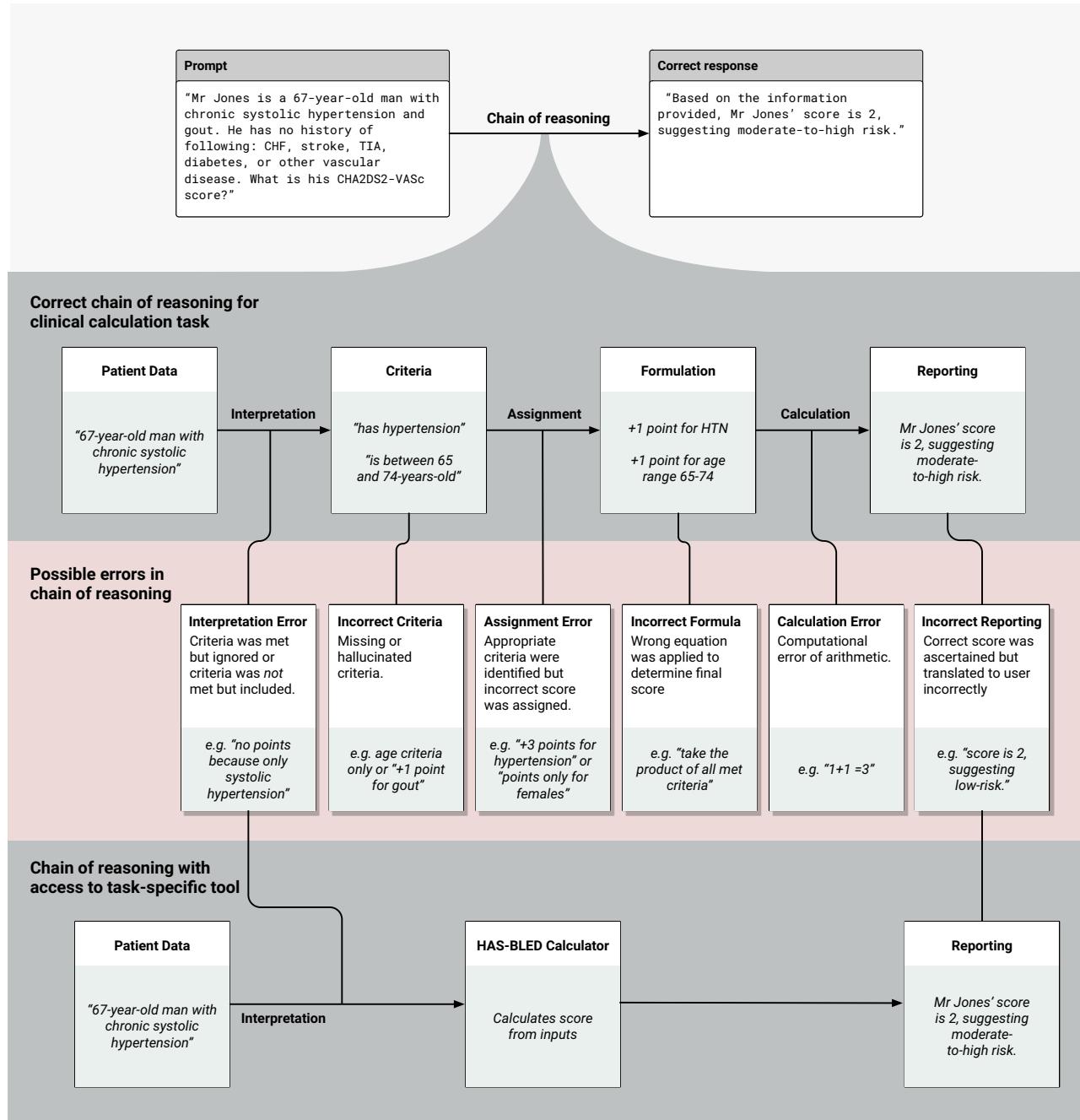


Fig. 3 | Chain of reasoning and error classification framework with example CHA₂DS₂VASC prompt. This diagram provides a detailed overview of the approach used to classify errors encountered in clinical calculation tasks performed by LLMs. It shows an example prompt with the user asking to calculate a CHA2DS2VASC for a patient, and the correct response from the model. The second row delineates the

necessary reasoning steps required to accurately complete a clinical calculation manually. The third row illustrates the possible errors in this chain of thought, with example errors linked to each step. The final row shows the same chain of reasoning with access to a calculation API, demonstrating how tool augmentation bypasses many error-prone steps while remaining susceptible to interpretation errors.

similar calculators, many of which would appear reasonable to an unfamiliar user. A pointed example comes from the model's attempt to assess an individual's Caprini VTE score, where it informed the user that a young overweight patient did not score any points for being overweight because "BMI is only counted as a risk factor in patients aged over 40."

The third and fourth most common error classes involved the application of math within the model. When presented with a calculation task, the base model would either (1) attempt to calculate using the language model alone, or (2) use the "code interpreter" tool. In the former, the model displayed difficulties performing simple calculations and/or counting. In one HAS-BLED prompt, it failed to calculate $1+1$. These were classified as

calculation errors and comprised 13 of 82 errors (16%). The latter, code interpreter functionality, was reserved for more complex math. In general, the model would describe the formulas in plain text or formula notation. Subsequently, it would pause and proceed to write a program in Python to calculate the results. The delivered result was occasionally incorrect even in instances where the model presented a correct version of the formula prior to writing the script. These were classified as having an *incorrect formula*, and represented 12 (15%) of the identified errors (Fig. 4). Interpretation errors (11 of 82, 14%) were also common. In particular, the base ChatGPT model occasionally struggled with interpreting numerical ranges, for example, reporting that a 64-year-old man would be categorized as between

 Anonymous

I have a patient that I'd like to tell you about; I need your help. They are a 82-year-old man who is being evaluated for a liver transplant. Their serum sodium is 125, serum bilirubin is 1, serum creatinine is 1.8, and INR is 1.5. They are on three-times-per-week dialysis. What is their MELD-Na score?

ChatGPT

The Model for End-Stage Liver Disease (MELD) score is a reliable clinical tool used to estimate the severity of chronic liver disease and to prioritize for receipt of a liver transplant. The MELD score incorporates the patient's serum bilirubin, serum creatinine, and INR, and it has been updated to include serum sodium levels, known as the MELD-Na score.

The MELD-Na score is calculated using the following formula:

$$MELD - Na = MELD + 1.32 \times (137 - Na) - \\ [0.033 \times MELD \times (137 - Na)]$$

First, we need to calculate the patient's original MELD score with the following formula:

$$\text{MELD} = 9.57 \times \log_e(\text{Creatinine}) + 3.78 \times \log_e(\text{Bilirubin}) + 11.2 \times \log_e(\text{INR}) + 6.43$$

For patients on dialysis, the maximum serum creatinine used for MELD score calculation is capped at 4.0 mg/dL.

Let's calculate the patient's MELD and MELD-Na scores with the given values.

The patient's original MELD score is approximately 16.6, and their MELD-Na score, which takes into account their serum sodium level, is approximately 25.9. This MELD-Na score can be used to assess the severity of their liver disease and their priority for liver transplantation. [2]

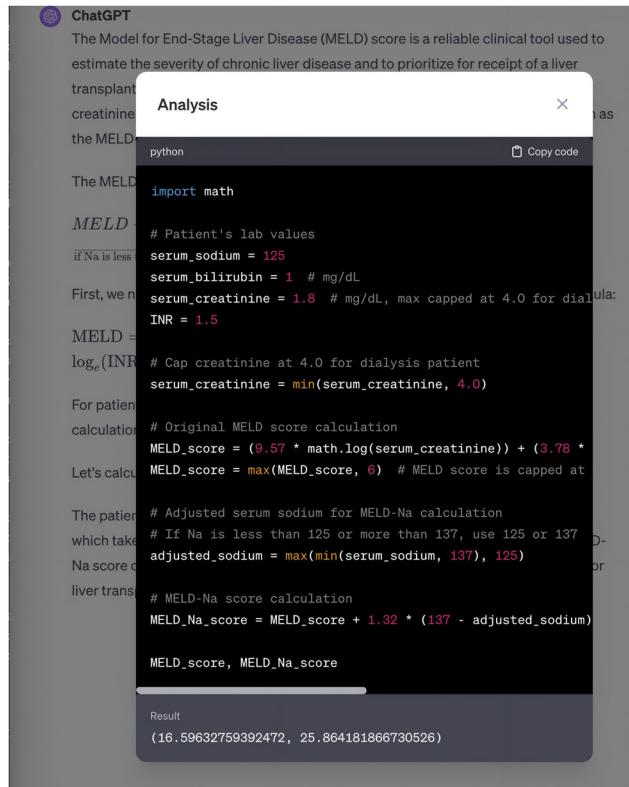
Fig. 4 | Illustration of standard ChatGPT's response to MELD-Na prompt. The left panel shows ChatGPT's initial response with a mathematical formula description, while the right panel shows the Python code it generates and executes. Note the

“65-74” in calculating a CHA₂DS₂-VASc score or that a partial pressure of oxygen of 66 was “less than 60.”

After identification of the errors, the classes of errors were compared to the calculator's functional type (Fig. 2b). Two patterns emerged from the analysis of the errors. Firstly, ChatGPT was less likely to make an assignment error in the *descriptive* models (such as estimating creatinine clearance or correcting hypercalcemia for albumin levels). This was likely due to the simple assignments required by these calculators; there is not a complex scoring system derived from a linear regression. In contrast, ChatGPT was more likely to make assignment errors or use the incorrect formula with *predictive* models such as the SOFA tool. This reflects the higher level of complexity in the criteria of these models, as well as lack of familiarity with some of these calculators.

Focused analysis

Our exploratory analysis revealed that ChatGPT was inaccurate in one-third of clinical calculations, due to errors in both knowledge and logic. To evaluate possible solutions, we developed a set of tools to assist in calculation and tested them with a dataset focused on the most challenging tasks from the exploratory phase. The ten calculators with the lowest accuracy were selected (Table 1), and a templated vignette generator was used to create 100 unique patient vignettes and corresponding ground truth answers for each. This produced 1000 vignettes in total. Concurrently, a set of 10 LLM agents were developed with one of two underlying LLMs (LLaMa3.1 or GPT-4o) and access to one of five tools: no tools (base LLM only), code interpreter, calculation instruction documents (RAG), code interpreter and RAG, or a set of task-specific tools available through an calculation API created by the authors, termed OpenMedCalc (Fig. 5). Agents were programmed to have basic planning and troubleshooting ability, including self-correction. Each of the agents was then presented with the 1000 unique vignettes, in addition to a custom system/example prompt, resulting in 10,000 total trials. These



incorrect handling of dialysis patients (creatinine capped at 4.0 rather than set to 4.0), demonstrating how even seemingly sophisticated responses can contain subtle but clinically significant errors.

trials were programmatically executed and their results were evaluated for accuracy, summarized in Table 3 and Fig. 2c.

GPT outperformed LLaMa in all arms, with cumulative accuracy of 67% vs 42%. Both LLaMa and GPT models saw significant improvements with the addition of tools. For LLaMa, the base model was only able to answer 11% of questions correctly, suggesting medium-sized models alone are likely inadequate for any quantitative medical task. The addition of code interpreter alone saw modest improvement (increase to 18%), while RAG alone showed a larger increase, answering 40% correct. With access to a set of generic tools (code interpreter and RAG), the model answered 53% of trials correctly, representing a 42% absolute improvement compared to the base model. However, the OpenMedCalc toolkit further improved performance by an absolute 30%, an incremental odds ratio of 4.6. The addition of OpenMedCalc resulted in correctly answering 84% of prompts correctly, representing a 73% absolute improvement, or 5.5-fold reduction in error rate compared to the unimproved LLaMa model.

For GPT, tools were also associated with higher accuracy, though absolute improvements were smaller compared to LLaMa due to GPT's substantially higher base performance. Unimproved, GPT answered 36% of prompts correctly. Addition of the code interpreter was again associated with modest improvements in accuracy (9.9% when added to the base model or only 5.5% when added to a model with RAG). RAG provided a stronger boost in performance than in LLaMa: a 41% absolute improvement in accuracy compared to the 29% absolute improvement in LLaMa, suggesting that the smaller model was not adequately using the knowledge provided through RAG. Despite the high performance of GPT with RAG and CI (82%), OpenMedCalc still showed incremental improvement, correctly answering 95% of prompts, a 13% absolute difference. Compared to its base model, this was a 59% absolute improvement in accuracy or a 13-fold reduction in the error rate. Notably, OpenMedCalc's incremental improvements remained highly statistically significant. In fact, all incremental comparisons met a p value threshold of 0.05, except for the addition

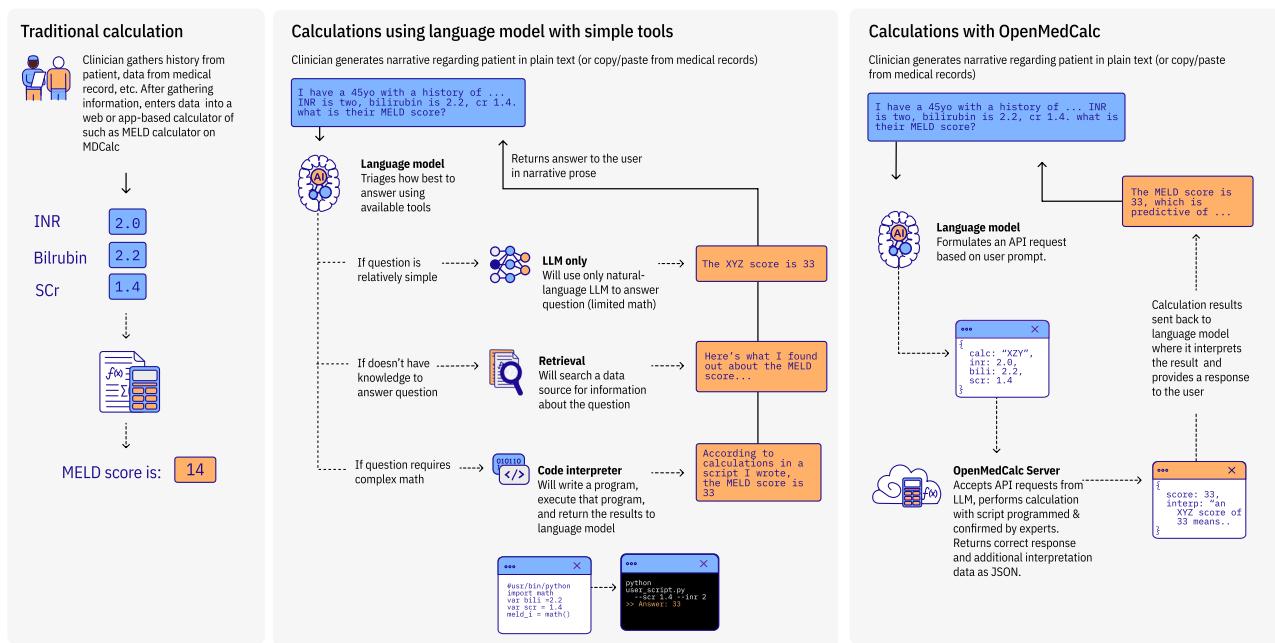


Fig. 5 | Overview of workflow for clinical calculation across three methods. This schematic compares (left) traditional manual calculation through web-based calculators, (center) calculation using LLMs with general-purpose tools like code

interpreter and web search, and (right) calculation using task-specific tools. The workflows demonstrate how the OpenMedCalc approach combines the efficiency of LLMs with the reliability of expert-validated computation.

Table 3 | Performance of models by tool configuration

LLM	Tool configuration	# Correct	% Correct (95% CI)	Δ_{abs}	Δ_{inc}	OR	P_1	P_2	P_3
LLaMa	Base	114	11.4 (9.6–13.5)	-	-	-	-	-	-
	Code Interpreter	183	18.3 (16.0–20.8)	6.9	6.9	1.7	3.6×10^{-4}	7.2×10^{-4}	1.4×10^{-3}
	RAG	403	40.3 (37.3–43.4)	29	22	3	1.0×10^{-11}	3.0×10^{-11}	4.0×10^{-11}
	RAG + Code Interpreter	535	53.5 (50.4–56.6)	42	13	1.7	7.2×10^{-4}	7.2×10^{-4}	2.9×10^{-3}
	OpenMedCalc	840	84.0 (81.6–86.1)	73	30	4.6	$<1 \times 10^{-12}$	$<1 \times 10^{-12}$	$<1 \times 10^{-12}$
GPT	Base	361	36.1 (33.2–39.1)	-	-	-	-	-	-
	Code Interpreter	460	46.0 (42.9–49.1)	9.9	9.9	1.5	6.2×10^{-3}	0.012	0.025
	RAG	768	76.8 (74.1–79.3)	41	31	3.9	$<1 \times 10^{-12}$	$<1 \times 10^{-12}$	$<1 \times 10^{-12}$
	RAG + Code Interpreter	823	82.3 (79.8–84.5)	46	5.5	1.4	0.043	0.043	0.17 ^a
	OpenMedCalc	952	95.2 (93.7–96.4)	59	13	4.3	6.4×10^{-12}	1.9×10^{-11}	2.5×10^{-11}

The table includes total correct out of 10,000 trials, percentage of trials answered correctly with 95% confidence intervals, and three measures of improvement: difference in accuracy compared to the base LLM with no tools (Δ_{abs}), an incremental comparison of difference in accuracy compared to the previous configuration (Δ_{inc}), and the odds ratio of a correct answer compared to the previous configuration (OR). In the last three columns, p values are shown as the raw p value (P_1), p value adjusted using Holm-Bonferroni correction (P_2), and p value adjusted with the Bonferroni correction (P_3). P values and odds ratios represent comparisons with the previous model in the list within each LLM.

LLaMa LLaMa-3.1-70b bf16, GPT GPT-4o, RAG retrieval-augmented generation, CI confidence interval, OR odds ratio.

^aNot significant to a of 0.05.

of code interpreter to a RAG system in the GPT model under the most conservative P value correction for multiple comparisons.

Though tool use was clearly associated with improved performance, there was notable heterogeneity. The most accurate arm, GPT with OpenMedCalc, achieved only 70% in the ARISCAT tasks while scoring remarkably well on the others: 92% for the CCI, 94% for NIHSS, 96% for Wells DVT and 100% on the remaining six calculation tasks. Indeed, if ARISCAT was excluded, accuracy of this arm would have been 98%. Surprisingly, the 70% accuracy in the ARISCAT task was nominally lower than three other arms: LLaMa with OpenMedCalc (71%), GPT with RAG (72%) and GPT with both RAG and CI (71%). This was due in part to a peculiarity of the ARISCAT task, whereby GPT-based models struggled with grouping surgical incisions into the categories of peripheral, upper abdominal, or intrathoracic. The slightly higher accuracy of the RAG arms suggests that additional prompt guidance was helpful, and perhaps could have

supplemented even the OpenMedCalc arm. However, for the same task, the base models of GPT and LLaMa had an accuracy of 0% and 4%, respectively, supporting the idea that knowledge of this calculator was lacking. On other tasks, OpenMedCalc made an outsized positive impact. On the NIH stroke scale, GPT with OpenMedCalc achieved 94% accuracy, while the next-best GPT model only achieved 48% despite access to both RAG and the code interpreter. The NIHSS is a demanding task, requiring detailed attention to the clinical specifics of the vignette for proper interpretation, which may have consumed the attention needed for score assignment, formulation, etc. OpenMedCalc's task decomposition allows the model to attend to interpretation alone, which may explain the outsized improvement in this task.

Focused analysis error classification

In the focused analysis, we evaluated a subset of the responses for error classification. We began by evaluating 409 trials with incorrect responses,

sampled all permutations of LLM, tool configuration, and calculation task. This evaluation yielded a total of 492 identified errors (222 in GPT and 273 in LLaMa), and these errors' classes were extrapolated to the remaining incorrect answers. These estimates are presented in Fig. 2d.

The cause of errors in the LLaMa-based models differed significantly from those seen in the exploratory analysis. Because we utilized a medium-sized LLaMa model, these arms often struggled with basic tasks such as properly formatting the response. Many LLaMa responses ended mid-sentence or mid-calculation leading to an invalid response (5% of cases for LLaMa vs <0.2% for GPT) or resulting in a response with an indeterminate cause of error (3% of cases for LLaMa vs <0.2% for GPT). LLaMa was unreliable in programming tasks, often producing invalid Python code, and frequently failed to attempt self-correction, continuing to produce an answer despite their code interpreter returning only an error. GPT's answers were superior, but even with access to RAG and CI, it had multiple incorrect responses due to inability to add single-digit numbers. It also would ignore tool output, but also had examples of self-correction after submission of invalid input to a tool. Examples of full prompts, responses, and error classification are available in the Supplementary Information (Supplementary Note 1 through Supplementary Note 13).

Numerically, the most common error classes identified in all arms were interpretation errors (estimated 1200 of 4561, 26%) followed closely by assignment errors (estimated 1122 of 4561, 24%) and incorrect formula error (992 of 4561, 21%), demonstrating problems both with the calculation task and with understanding the patient vignette.

Across the different arms, there was a notable decrease in assignment errors (ie, assigning the wrong score for an individual criterion) in arms with RAG or OpenMedCalc compared to those without; the base and CI-only models had an estimated 846 of these errors out of the 2,882 total errors in these arms (29%), while the RAG-containing and OpenMedCalc arms had only 276 of these errors out of a total of 1679 errors (16%). This result is intuitive, as providing the model with a calculator or detailed instructions simplifies the assignment task. Another notable difference between arms was seen in the OpenMedCalc arms, where only interpretation errors were identified; this finding is consistent with our error classification framework. As the OpenMedCalc toolkit does not assist with interpretation of the patient vignette, steps following vignette interpretation would be largely deterministic (Fig. 3).

Although obtaining the correct answer is ideal, in many cases, there is little clinical difference between, for example, a HAS-BLED score of 4 and a score of 5, both of which require anticoagulation. For this reason, we also determined the magnitude of errors across arms by calculating the absolute error for each incorrect answer and normalizing it to each calculator's valid range. Here, we observed an intuitive trend: higher-performing models tended to have smaller errors than those with lower performance (Fig. 2e). The exception to this trend is OpenMedCalc, which had a more narrow range of errors than the other arms, likely due to the deterministic nature of the tool. Additional figures showing error magnitude are available in the Supplementary Information (Supplementary Figs. 2 and 14).

Discussion

LLMs have demonstrated a wide range of capabilities with potential application to medicine and have performed exceedingly well on boards-style questions. However, their skill set is uneven, and, unintuitively, they possess little aptitude in quantitative tasks. In this work, we demonstrate that the commercial version of ChatGPT is an unreliable clinical calculator, providing correct answers in only two-thirds of trials across a diverse group of tasks. However, in our focused analysis, we found that addition of tools improves this performance in an incremental fashion, regardless of LLM size or complexity. With access to task-specific tools, performance showed marked improvement, increasing accuracy to 84% (82–86%) in our LLaMa experiments and 95% (94–96%) in those with GPT.

Augmentation of language models with tools has been explored more fully in fields outside of clinical medicine, revealing that even smaller language models, when supplemented with specialized tools, can yield results

comparable to those of foundation models²⁴. Yet, this approach has not been explored in a medical context until recently. Given the rapid pace of LLM research, our initial findings regarding LLMs' limitations in medical calculations and the potential benefits of tool augmentation have been further supported by concurrent science. Since our initial work and pre-print in December of 2023²⁵, multiple teams have explored the topics of agentic and tool-using models for calculation and have also made their results available through pre-print servers prior to peer review.

Khandekar et al. ²⁶ developed a promising benchmark for medical calculation, MEDCALC-BENCH, to systematically evaluate LLM performance on medical calculations. The benchmark covers 55 calculators taken from MDCalc's most popular list and contains 1047 patient vignettes sourced from publicly-available case reports; it also includes a validated step-by-step explanation. The authors evaluate a variety of models and prompting strategies, finding GPT the most accurate, ranging from 21% accuracy (no prompt engineering) to 51% accuracy (with one-shot exemplar), in step with our observed accuracy of 36% in our base GPT-4o arm, which uses a one-shot exemplar but has a set of calculation tasks enriched for difficulty. The authors also develop an error classification system, similar in spirit to ours, but with reduced granularity; they too find that both misunderstanding of the vignette (error of interpretation), lack of calculator knowledge (incorrect criteria, assignment error), and miscalculations are present. They appropriately conclude that current-generation LLMs are not ready for assisting clinicians with clinical calculation²⁶.

Jin et al. ²⁷ extended the concept of tool-use by building AgentMD. Here, 2164 possible risk calculator tools were identified from PubMed abstracts, where they were used to automatically generate code²⁷. A subset of those tools (the 50 most popular as well as 50 randomly-selected tools) were evaluated. After excluding incorrect tools, GPT was used to generate a set of five vignettes and multiple-choice questions for each calculator. They then evaluate a variety of configurations but find that the tool-augmented arm outperforms their unimproved model (88% vs 40%). Notably, as with our findings, the lower-powered tool-augmented model exceeded the performance of the higher-powered model without tools. However, unlike our study, the large number of available tools required an advanced tool-selection module, adding to the LLM's task complexity.

Exploring tool-selection even further, the MeNTi project²⁸ developed an adaptive, nested tool-calling structure to allow for flexible tool selection. To evaluate their approach, physicians developed vignettes with GPT-4o based on 100 real-life cases, each focused on one of 44 popular medical calculation tasks. As opposed to the other projects, this study included a set of tools for unit conversion functions, numbering 237 in total. The study found that tool-use improved performance, though the maximum final-answer accuracy with GPT-4o remained just under 50%, compared to 22% in the unimproved version. The authors note that incorporation of unit conversion tools complicated the project, however this is likely an essential step for locales that use multiple measurement units.

Lastly, the issue of calculator selection is further investigated by Wan et al. ²⁹, who assessed both LLM and human performance on the task of calculator selection using truncated patient histories from PMC-Patients as vignettes for 35 popular calculators sourced from MDCalc²⁹. Their findings suggest that calculator selection is not trivial.

Compared to these studies, our work built a set of tools focused on a smaller number of calculators, each of which was hand-built, optimized for LLM use, and validated by a physician. Our vignettes were written by two of the physician authors (AG and SC), either one-by-one in the exploratory analysis (212) or with the assistance of templating software in the focused analysis (1000). Our choice of calculators for the focused analysis was enriched for difficulty—ChatGPT had only answered 16% (8 of 50) of these vignettes correctly, and answers were correct only if mathematically equivalent. The referenced work offers a compelling vision of the future but relies on a combination of LLM-generated code, LLM-generated evaluations, evaluations using published cases, and/or multiple-choice questions, limiting some of the generalizability to clinical practice.

However, the convergence of multiple research groups on similar solutions and findings strengthens the evidence for both the limitations of current LLMs in medical calculations and the promise of tool-augmented approaches. For these reasons, we suspect tool-augmented LLMs will play a role in the future applications of AI in medicine. LLMs may work to extract and analyze the relevant text data from the electronic medical record and communicate with a set of tools to provide information on patient's risks and treatment options, using text data that was previously inaccessible. Fields such as preoperative risk assessment have the potential to be transformed as LLMs remove burdensome data entry tasks from clinicians and provide more personalized risk analysis to patients. However, even the highest-performing model configuration in our experiment was inaccurate 5% of the time, which could represent an unacceptable risk depending on the context. From a practical perspective, the implementation of LLM-based calculators would be new regulatory territory and may face hurdles even if its performance consistently exceeded that of physicians. However, pre-programmed tools with well-defined internal functionality may offer a level of transparency that would remove the need for FDA approval as a clinical decision support device, and may be seen as an extension of a tool like MDCalc.

Yet, given the number and variety of errors encountered in our work, there is still significant improvement needed before this promise is to be realized. To better understand these errors, we introduced a novel classification schema based on the errors' position within a chain of logical steps, finding that even augmented models are still prone to misinterpretation of the patient's history. Our framework may assist in the differentiation of errors susceptible to the use of an external tool and those errors which may be intractable (Fig. 3). Given the rapid rise in LLM performance on existing medical benchmarks, more challenging, diverse, and holistic benchmarks such as those produced by the MedAlign and Med-HALT projects, are needed^{30,31}. Future medical LLM evaluations should include a calculation benchmark, such as, and we have contributed our dataset for this purpose under the moniker ABACUS-212 and ABACUS-409; each dataset contains the vignette, ground truth answer, and physician annotation of error class.

There were several notable limitations to this study. As a natural language model, LLMs are unsurprisingly sensitive to variations in question phrasing. Though we established general guidelines for our exploratory analysis, there was variation in the wording of prompts. In the focused analysis, question stems were standardized into templates to reduce this variability. These differences in phrasing likely had unpredictable and difficult-to-characterize impacts on our results. However, since our aim was to simulate how clinicians might interact with an LLM "in the wild," under real-world settings, this variability may be appropriate. To maximize the diversity of prompts, our vignettes represented cases across the spectrum of the input ranges, thereby creating fictionalized patients that were often very ill; this may bias our results in unclear ways. Future studies might use real-world questions posed by clinicians to more accurately capture this task and represent a reasonable patient population.

From a technical limitations standpoint, the exploratory analysis relied on the web version of ChatGPT instead of the more programmer-friendly and reproducible API. The code-interpreter feature was limited to the web version during the study period. Since the exact model configuration parameters used in the web-based ChatGPT are not public information, reproduction of those exploratory results is infeasible. However, our follow-up, focused analysis used the API version and had a set seed for reproducibility. We also did not extensively test different prompt engineering approaches, which might have improved performance of the base case model^{32,33}. Lastly, we did not evaluate the selection of a calculator or tool—our stem explicitly asked for the calculator by name, though others have begun to explore this²⁹.

This work also presents many opportunities for future research. We did not include vignettes with missing or malformed data. Exploring when a model refuses to answer a question outside its estimated abilities (termed "out-of-distribution problems") is essential to LLM safety in the medical context and deserves continued exploration³⁴. Though we classified errors in

a novel framework and evaluated the magnitude of those errors, we did not determine whether those errors would have had a real clinical impact. Future studies could evaluate, for example, what proportion of errors would have led to a change in medical management of the patient in question. Additionally, exploration of clinical calculation by multi-agent models, which simulate medical teams, or high "test time compute" models, which more explicitly imitate reasoning, may also provide insight into this challenge^{35–37}. Lastly, current human performance in this area needs to be better quantified under real-world settings. Though the calculators offered by tools like MDCalc are well-crafted to spot mistakes, it is likely that even the most careful clinicians make data entry errors, or even misinterpret patient data, when under the time restraints inherent in medicine; this baseline is currently unknown.

In summary, this work introduces a novel approach to improving the output of language models through the use of both generic and task-specific tools. Ultimately, LLMs show promise to substantially alter medical practice with a more efficient, equitable, and patient-centered approach. In this evolving landscape, it is imperative for models to exhibit high levels of robustness and reliability. Achieving this will likely necessitate an ensemble of models and tools to effectively meet these demands, ensuring that LLMs can be integrated safely into the healthcare domain.

Methods

Exploratory analysis

The initial phase of this work relied on ChatGPT to provide an overview of LLM capability. Notably, ChatGPT is a proprietary system composed of an LLM, with an unknown, inconstant configuration as well as access to a set of tools that includes a code interpreter. For the exploratory phase, ChatGPT (GPT4, Nov 2023) was used. The following steps were undertaken:

Choice of calculation tasks

We reviewed the 50 most popular tools offered by MDCalc and evaluated their appropriateness for the task of model assessment (Fig. 1). Calculators were categorized into five types: descriptive calculators (which focus on describing a known but difficult-to-measure clinical entity, such as an anion gap or a patient's renal function), predictive calculators (which attempt to predict the risk of some future event such as VTE), speculative (which are primarily diagnostic and focus on determining whether an entity, such as sleep apnea, currently exists in a patient), summative (which provide an overview of disease severity or course), and therapeutic (which are meant to guide treatment in some manner). The specific calculators are shown in Table 1. Each category reflects a specific type of task that the calculator is designed to perform within a medical context. Of the 50 calculators, two were superseded by more recent calculators and were excluded. The remaining 48 calculators underwent assessment.

Vignettes

After review of the calculator, a convenience sample of five clinical vignettes was written by one of the two physician authors. The vignettes were meant to relay all necessary information to perform the calculation with minimal extraneous information and contain a request for a calculation that was unambiguous. To minimize question-answering avoidance (i.e., "as an AI model I am not..."), prompts were written from the perspective of a clinician asking for assistance, using commonplace medical jargon. In cases where multiple common formulas existed for a calculation and no clear gold standard was established, we specified a preferred method (i.e., "calculate the QTc using the Bazett formula"). Data were presented in an objective fashion as much as possible. Where relevant, continuous values were reported within a range that one would reasonably see in clinical practice; units were omitted in most prompts, as is common in clinical communication. For criteria-based calculators, the vignettes provided the underlying data as opposed to whether criteria were met. For example, a vignette for CHA₂DS₂-VASc might report a "history of TIA," not "positive cerebrovascular disease." The intent was to convey information in a manner that was clear from a clinical perspective but indirect, simulating how a clinician

might use the tool or how data might be arranged in a narrative section within a clinician's note.

Presentation and scoring

To evaluate the dynamic programming abilities in ChatGPT, prompts were presented to the web interface of the commercial version of ChatGPT by one of two physician assessors. Each vignette was started in its own conversation to eliminate contamination from prior prompts. If the LLM did not provide a definitive answer but showed progress towards an answer, the user encouraged the conversation with a prompt such as "Please continue." To score the performance of the model, assessors compared ChatGPT's answers with the answer provided by MDCalc with input data from the vignette. For criteria-based models, answers were counted as correct if the reported scores matched. For models with continuous variable outputs, rounding errors were permitted if the calculator did not specify how to round. Some of the calculators involved calculations without a gold standard, such as drug conversions. Responses on these models were considered correct if the underlying math was executed appropriately and the dose given was reasonable. If a model was unable or refused to answer a question even after encouragement with two prompts, a score of "no valid response" was given. The conversation was saved to a database in plain text. When available, a URL for the conversation was saved. These conversations have been provided in an accessible format in our repository.

Classification of errors

After initial trials were complete, one of the three physician authors (AG, SC, LC) inspected the correct and incorrect answers. From these, we formulated a framework for the chain of reasoning required to provide appropriate answers, as well as the various possible missteps within this chain (Fig. 3). When the model offered incorrect solutions, raw output was inspected for these errors in reasoning. Error classification occurred from Nov 14 to Dec 10, 2023 and required an estimated combined 30 h between annotators. Errors were categorized into one or more of the categories: (1) Interpretation Error: Inadequate understanding or misinterpretation of the medical information presented in the question leads to ignored criteria that was met, or inclusion of a criteria that was not met. (2) Incorrect Criteria: Criteria are missing or there is a hallucination of non-existent criteria. (3) Assignment Error: Improper application of correctly-identified criteria. Appropriate criteria were selected but an incorrect score is assigned. (4) Incorrect Formula: An incorrect equation is chosen to represent the scoring mechanism of the calculation task. (5) Calculation Error: The correct formula is chosen, such as taking the sum of all subscores, but the actual mathematical computation carried out was incorrect. (6) Incorrect Reporting: The correct score is calculated, but some component of reporting that score to the user is inaccurate. (7) Indeterminate: Inadequate response information to be able to determine error. (8) No valid response: If the model produced no response that answered the prompt.

Development and evaluation of augmented models

After completion of the exploratory phase, we evaluated the impact of popular LLM augmentation strategies for the task of clinical calculation by building an LLM environment with roughly the same functionality as ChatGPT and evaluating it against the most challenging tasks from the prior analysis. To better understand the contribution of each augmentation strategy, we evaluated the impact of three different components: a code interpreter tool, a retrieval-augmented generation (RAG) system, and a set of task-specific calculation tools (OpenMedCalc toolkit) with two LLMs. The following sections describe this process.

Calculator selection

For the focused analysis, we identified the ten calculation tasks from the exploratory analysis with the lowest performance, while trying to capture a diversity of tasks and organ systems. Tasks with multiple correct answers (ie, steroid conversion) were excluded. Figure 2d shows the ten calculators chosen.

Vignette generation

One of the physician authors then wrote three template vignettes for each calculator. We then decomposed each vignette template into its constituent components that would be required to perform the calculation and created a list of alternatives for each component, as well as what each would contribute to a final score (if relevant). Once complete, we used this information to generate a total of 100 unique vignettes for each calculator, resulting in a total of 1000 vignettes. Concurrently, the ground-truth answer for each was determined programmatically. A discrete randomization seed was used to ensure reproducibility.

LLM choice

We evaluated two different LLMs: GPT-4o, a proprietary state-of-the-art model, and LLaMa 3.1, a free, open-weights model. We utilized a medium-sized (70-billion parameters), instruction-tuned version of LLaMa-3.1 to evaluate the impact of model size on performance. Inference for LLaMa was provided by DeepInfra at bf16 (meta-llama/Meta-Llama-3.1-70B-Instruct). GPT-4o inference was provided by OpenAI via their API. Temperature was set to 0.1 for both models to evaluate the most-likely response³⁸ while allowing for some variability for self-correction (discussed below). An LLM randomization seed was set, ensuring deterministic output for a unique vignette. Total cost was ~400 USD.

Tool and agent architecture

Because LLaMa models do not offer a pre-built code interpreter or function-execution functionality, we implemented a framework to support a simple planning agent able to produce tool-use requests, receive responses, and return a final answer to the user. To allow comparison, this framework was used for both GPT- and LLaMa-based models. The agent was provided a scratchpad field to describe its plan and response. If tools were provided to the agent, it was required to use at least one tool, but there were no explicit limits on the number of calls allowed. To automate evaluation, the agent was required to return the final answer in a specified format using a response tool with structured outputs for models that supported it³⁹. If a response was invalid, an LLM-based formatting prompt was used to coerce a valid format; otherwise, the trial was restarted, allowing the LLM to attempt a different answer up to four additional times. A timeout of 100 s was set for a single trial attempt. After five failures or timeouts, the trial was marked as incorrect with an error class of no valid response. The LangChain package was used to manage the agent's state and responses⁴⁰.

Code interpreter

A Python code-interpretation module was built using a remote sandbox environment tool. The code interpreter was configured to return either: (1) the results of successfully executed code or (2) an error message for invalid code. The agent was allowed to self-correct and re-attempt the calculation if their script produced an error; this process was allowed to continue as needed until 100 s had elapsed or the agent submitted a final response.

Retrieval augmented generation

We evaluated the effect of a simple RAG system which provided instructions for calculating the vignette. For each calculator, the instructions portion of the MDCalc webpage for that calculator was extracted and converted to markdown. They were reviewed by one of the physician authors to ensure that they contained complete instructions required to perform the calculation. Each was then stored in a vector database with their embeddings. For arms with RAG, the agent was automatically provided with the full contents of the document with the highest cosine similarity to the prompt.

OpenMedCalc toolkit

A basic API was created using FastAPI⁴¹ and installed on a remote server at <https://openmedcalc.org>. For each calculation task, an algorithm was written in Python and formatted as a tool that was usable by a large language model. To maximize the usability of the tool by LLMs, the calculation task was decomposed to accept units of irreducible patient information. For example,

if a calculator's inputs required the patient's age in three groupings, the tool would accept the patient's age as an integer and not require the LLM to determine which age group was appropriate. Parameters were given clear names and included a brief description of each input. Where appropriate, upper and lower bounds were included. Validation of inputs was performed on data input using Pydantic. As with the code interpreter, the agent was allowed to self-correct and re-attempt the calculation if a validation error occurred. For those arms which included the OpenMedCalc toolkit, all 10 tools were provided and the agent was free to choose the appropriate tool. The API and its documentation conformed to OpenAPI specification⁴² and are freely accessible online.

Prompting

After creation of the systems described above, we compared the two LLMs (LLaMa-3.1-70b and GPT-4o) across five different configurations, also referred to as experimental arms: (1) the base model, (2) code interpreter, (3) RAG, (4) both code interpreter and RAG, and finally (5) the base model with the OpenMedCalc toolkit, for a total of 10 arms. Each of the ten arms was prompted with each of the 1000 unique vignettes for a total of 10,000 trials. The system prompt used was unique to each arm, where it described the task at hand and provided guidance on the use of any available tools. The prompt also contained a one-shot example of tool use for the available tool type. The process of self-correction was not illustrated. Although we did not iteratively evaluate the effects of different prompt engineering strategies, a simple “reward and punishment” strategy was used. Full examples of the prompts are included in the Supplemental Information (Supplemental Note 1 through Supplemental Note 13).

Error classification

The models' responses were scored as correct only if they were mathematically equivalent to the ground truth answer; no rounding errors were permitted. To classify the errors made in the focused analysis, we randomly sampled up to five errors per calculator-arm pair for a total of 409 incorrect responses. For each of these responses, the full prompt, vignette source components, tool calls, and tool responses were loaded into annotation workflow using the LangSmith platform and each was evaluated by one of the three physician authors (AG, SC, LC). These sampled responses were then classified into one or more of the eight error classes (as above). Error classification for the focused analysis occurred from Nov 1 to Nov 22 and required ~65 h of active annotation time based on browser usage data. Specifically, we began by evaluating 409 trials with incorrect responses, sampled without replacement from 100 bins representing all permutations of LLM, tool configuration, and calculation task. This evaluation yielded a total of 492 identified errors (222 in GPT and 273 in LLaMa). Sparse bins led to models with inferior performance being overrepresented. To assign an error class to those responses not reviewed, we assumed that the proportional errors from each arm-calculator pair were representative of their respective groups. For instance, if 3 of the 5 annotated errors in the LLaMa-base-SOFA group were *interpretation errors*, we assumed a prevalence of 60% in that group's incorrect answers.

Statistics

To compare performance across the study arms, we developed a logistic GEE model to analyze the binary outcome data (correct/incorrect responses) while accounting for repeated measures, whereby each model configuration evaluated the same set of patient cases. The vignette ID was specified as the clustering variable. The model used an independence working correlation structure, and robust standard errors were computed to account for potential misspecification of the correlation structure. To assess the incremental value of each additional tool (code interpreter, RAG, and OpenMedCalc), we performed pairwise comparisons between adjacent configurations within each LLM family (LLaMa and GPT), ordered by their performance. For these post-hoc comparisons, we calculated the difference in coefficients and their standard errors, deriving z-statistics and corresponding *p* values. 95% confidence intervals were calculated using Wilson's

score method. To account for false discovery rate due to multiple comparisons, we report both raw *p* values and two correction methods: the Holm-Bonferroni and traditional Bonferroni corrections. The Holm-Bonferroni was selected as our primary correction approach because it best fits the study design; we test related but distinct hypotheses that each successive tool will augment performance. Effect sizes are reported as odds ratios with 95% confidence intervals, while absolute performance differences are presented as percentage point improvements compared to the unimproved arm and an incremental comparison with the prior arm. All statistical analyses were performed using Python's statsmodels package.

Data availability

Data used in this study, and the resultant datasets (ABACUS-212 and ABACUS-409), are available at <https://github.com/stanfordaimlab/llm-as-clinical-calculator>. The data includes the set of 621 vignettes which underwent error analysis; each entry contains the vignette, ground truth answer, LLM response and the error classification. The data is available in a structured format and is freely accessible.

Code availability

The OpenMedCalc codebase is available on GitHub (<https://github.com/alegoodell/open-med-calc>). API documentation is available at the <https://api.openmedcalc.org/docs>. A basic GPT-based demo agent with access to the OpenMedCalc API is available at <https://openmedcalc.org/chat>. This demo requires a free OpenAI account to use. Experimental code is available at <https://github.com/stanfordaimlab/llm-as-clinical-calculator>. Reproducing our results in full requires a subscription to LangSmith, the annotation and experiment management platform used in this study, OpenAI's API, and an inference provider for LLaMa-3.1-70b running at BF16. Instructions for setting up the environment are provided in the repository.

Received: 12 December 2023; Accepted: 22 January 2025;

Published online: 17 March 2025

References

1. Nori, H. et al. Can generalist foundation models outcompete special-purpose tuning? Case study in medicine. <https://doi.org/10.48550/arXiv.2311.16452> (2023).
2. Kung, T. H. et al. Performance of ChatGPT on USMLE: potential for AI-assisted medical education using large language models. *PLoS Digit. Health* **2**, e0000198 (2023).
3. Singhal, K. et al. Large language models encode clinical knowledge. *Nature* **620**, 172–180 (2023).
4. OpenAI. Gpt-4 technical report. <http://arxiv.org/abs/2303.08774> (2023).
5. Savage, T., Nayak, A., Gallo, R., Rangan, E. & Chen, J. H. Diagnostic reasoning prompts reveal the potential for large language model interpretability in medicine. *NPJ Digit. Med.* **7**, 20 (2023).
6. Patel, D. et al. Evaluating prompt engineering on GPT-3.5's performance in USMLE-style medical calculations and clinical scenarios generated by GPT-4. *Sci. Rep.* **14**, 17341 (2024).
7. Zakka, C. et al. Almanac—retrieval-augmented language models for clinical medicine. *NEJM AI* **1**, Aloa2300068 (2024).
8. Yao, S. et. al. ReAct: Synergizing Reasoning and Acting in Language Models. *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=WE_vluYUL-X (2023).
9. Xi, Z. H. et al. The rise and potential of large language model based agents: a survey. *Sci China Inf Sci.* **68**, 121101 (2025).
10. Yubo, W., Xueguang, M. & Wenhui, C. Augmenting Black-box LLMs with Medical Textbooks for Biomedical Question Answering. <https://doi.org/10.48550/arXiv.2309.02233> (2024).
11. Jin, Q., Yang, Y., Chen, Q. & Lu, Z. Genegpt: augmenting large language models with domain tools for improved access to biomedical information. *Bioinformatics* **40**, btae075 (2024).
12. Imani, S., Du, L. & Shrivastava, H. MathPrompter: Mathematical Reasoning using Large Language Models. In *Proc. 61st Annual*

Meeting of the Association for Computational Linguistics (eds Sitaram, S., Beigman K. B. & Williams, J. D.) Volume 5: Industry Track, 37–42 (Association for Computational Linguistics, Toronto, Canada, 2023).

13. MDCalc. FAQ. <https://web.archive.org/web/20231212045610/https://www.mdcalc.com/faq> (2023).
14. Walker, G. Emergentology: the ups and downs of developing the MDCalc app. *Emerg. Med. News* **38**, 18 (2016).
15. Bilimoria, K. Y. et al. Development and evaluation of the universal ACS NSQIP surgical risk calculator: a decision aid and informed consent tool for patients and surgeons. *J. Am. Coll. Surg.* **217**, 833 (2013).
16. Shahian, D. M. et al. The society of thoracic surgeons 2018 adult cardiac surgery risk models: part 1—background, design considerations, and model development. *Ann. Thorac. Surg.* **105**, 1411–1418 (2018).
17. Gliadkovskaya, A. Some doctors are using public AI chatbots like ChatGPT in clinical decisions. Is it safe? <https://www.fiercehealthcare.com/special-reports/some-doctors-are-using-public-generative-ai-tools-chatgpt-clinical-decisions-it> (2024).
18. Ng, M. Y., Helzer, J., Pfeffer, M. A., Seto, T. & Hernandez-Boussard, T. Development of secure infrastructure for advancing generative ai research in healthcare at an academic medical center. *J. Am. Med. Inform. Assoc.* **32**, 586–588 (2025).
19. Bumgardner, V. K. C. et al. Institutional platform for secure self-service large language model exploration. *arXiv [cs.CR]*. <http://arxiv.org/abs/2402.00913> (2024).
20. Goh, E. et al. Large language model influence on diagnostic reasoning: a randomized clinical trial. *JAMA Netw. Open* **7**, e2440969 (2024).
21. Hong, H. J., Schmiesing, C. A. & Goodell, A. J. Enhancing the readability of preoperative patient instructions using large language models. *Anesthesiology* **141**, 608–610 (2024).
22. Chung, P. et al. Large language model capabilities in perioperative risk prediction and prognostication. *JAMA Surg.* **159**, 928–937 (2024).
23. Kwan, B., Bell, J. F., Longhurst, C. A., Goldhaber, N. H. & Clay, B. Implementation of an electronic health record-integrated instant messaging system in an academic health system. *J. Am. Med. Inform. Assoc.* **31**, 997–1000 (2024).
24. Schick, T. D. et al. Toolformer: Language models can teach themselves to use tools. *Adv. Neural Inf. Process. Syst.* **36**, 68539–68551 (2023).
25. Goodell, A. J., Chu, S. N., Rouholiman, D. & Chu, L. F. Augmentation of ChatGPT with clinician-informed tools improves performance on medical calculation tasks. *medRxiv* 2023-12. <https://doi.org/10.1101/2023.12.13.23299881> (2023).
26. Khandekar, N. et al. Medcalc-bench: Evaluating large language models for medical calculations. *arXiv [cs.CL]*. <http://arxiv.org/abs/2406.12036> (2024).
27. Jin, Q. et al. Agentmd: empowering language agents for risk prediction with large-scale clinical tool learning. *arXiv [cs.CL]*. <http://arxiv.org/abs/2402.13225> (2024).
28. Zhu, Y. et al. Menti: bridging medical calculator and llm agent with nested tool calling. *arXiv [cs.AI]*. <http://arxiv.org/abs/2410.13610> (2024).
29. Wan, N. et al. Humans continue to outperform large language models in complex clinical decision-making: a study with medical calculators. *arXiv [cs.CL]*. <http://arxiv.org/abs/2411.05897> (2024).
30. Fleming, S. L. et al. MedAlign: A Clinician-Generated Dataset for Instruction Following with Electronic Medical Records. In *Proc. AAAI Conference on Artificial Intelligence*, Vol. 38, 22021–22030 (2024).
31. Pal, A. et al. Med-HALT: Medical Domain Hallucination Test for Large Language Models. In *Proc. 27th Conference on Computational Natural Language Learning (CoNLL)* (eds Jiang, J., Reitter, D. & Deng, S) 314–334 (Association for Computational Linguistics, Singapore, 2023) <https://aclanthology.org/2023.conll-1.21/>
32. Huang, J. & Chang, K. C-C. Towards Reasoning in Large Language Models: A Survey. *Findings of the Association for Computational Linguistics: ACL 2023* (eds Rogers, A., Boyd-Graber, J. & Okazaki, N) 1049–1065 (Association for Computational Linguistics, Toronto, Canada, 2023).
33. Qiao, S. et al. Reasoning with Language Model Prompting: A Survey. In *Proc. 61st Annual Meeting of the Association for Computational Linguistics*, Vol. 1 (eds Rogers, A., Boyd-Graber, J. & Okazaki, N) 5368–5393 (Association for Computational Linguistics, Toronto, Canada, 2023).
34. Shashikumar, S. P., Wardi, G., Malhotra, A. & Nemati, S. Artificial intelligence sepsis prediction algorithm learns to say “I don’t know”. *npj Digit. Med.* **4**, 1–9 (2021).
35. Tang, X. et al. MedAgents: Large Language Models as Collaborators for Zero-shot Medical Reasoning. <https://doi.org/10.48550/arXiv.2311.10537> (2024).
36. Liao, Y., Meng, Y., Liu, H., Wang, Y. & Wang, Y. An automatic evaluation framework for multi-turn medical consultations capabilities of large language models. <https://doi.org/10.48550/arXiv.2309.02077> (2023).
37. Schmidgall, S. et al. Agentclinic: a multimodal agent benchmark to evaluate ai in simulated clinical environments. *arXiv [cs.HC]* <http://arxiv.org/abs/2405.07960> (2024).
38. Van Veen, D. et al. Adapted large language models can outperform medical experts in clinical text summarization. *Nat. Med.* **30**, 1134–1142 (2024).
39. Willard, B. T. & Louf, R. Efficient guided generation for large language models. *arXiv [cs.CL]*. <http://arxiv.org/abs/2307.09702> (2023).
40. Chase, H. Langchain. <https://github.com/langchain-ai/langchain> (2022).
41. Ramírez, S. Fastapi. <https://github.com/tiangolo/fastapi> (2023).
42. Miller, D. et al. *OpenAPI Specification v3. 1. 0* (OpenAPI Initiative, The Linux Foundation, 2021). <https://spec.openapis.org/>.

Acknowledgements

This study received no dedicated funding. AG was supported by the Stanford University Research in Anesthesia Training (ReAP) Program and the National Institutes of Health Grant #GM089626. SC was supported by the UCSF Filling a Void of Research Training for Transplant Surgeons (FAVOR) Program and the National Institutes of Health Grant #5T32AI125222-08. Funders played no role in research design or execution. Lastly, the authors thank Graham Walker of MDCalc for discussion of concepts presented in paper.

Author contributions

A.G. and S.C. conceptualized the study, collected data, reviewed LLM responses for error classification, and contributed to the manuscript. A.G. conceived, designed, and implemented the OpenMedCalc API system, designed the methodology of the study, performed data analysis, and wrote the first draft of the manuscript. D.R. contributed to OpenMedCalc exploratory analysis, revision experiments, provided critical review, and was a contributor to the manuscript. L.C. provided critical review, reviewed LLM responses for error classification, and was a contributor to the manuscript. All authors read and approved the final manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at <https://doi.org/10.1038/s41746-025-01475-8>.

Correspondence and requests for materials should be addressed to Alex J. Goodell.

Reprints and permissions information is available at <http://www.nature.com/reprints>

Publisher’s note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2025