LiveBench: A Challenging, Contamination-Free LLM Benchmark

Colin White* [1], Samuel Dooley*[1], Manley Roberts*[1], Arka Pal*[1], Benjamin Feuer[2], Siddhartha Jain[3], Ravid Shwartz-Ziv[2], Neel Jain[4], Khalid Saifullah[4], Siddartha Naidu[1], Chinmay Hegde[2], Yann LeCun[2], Tom Goldstein[4], Willie Neiswanger[5], Micah Goldblum[2]
[1] Abacus.AI, [2] NYU, [3] Nvidia, [4] UMD, [5] USC

Abstract

Test set contamination, wherein test data from a benchmark ends up in a newer model's training set, is a well-documented obstacle for fair LLM evaluation and can quickly render benchmarks obsolete. To mitigate this, many recent benchmarks crowdsource new prompts and evaluations from human or LLM judges; however, these can introduce significant biases, and break down when scoring hard questions. In this work, we introduce a new benchmark for LLMs designed to be immune to both test set contamination and the pitfalls of LLM judging and human crowdsourcing. We release `LiveBench`, the first benchmark that (1) contains frequently-updated questions from recent information sources, (2) scores answers automatically according to objective ground-truth values, and (3) contains a wide variety of challenging tasks, spanning math, coding, reasoning, language, instruction following, and data analysis. To achieve this, `LiveBench` contains questions that are based on recently-released math competitions, arXiv papers, news articles, and datasets, and it contains harder, contamination-free versions of tasks from previous benchmarks such as Big-Bench Hard, AMPS, bAbI, and IFEval. We evaluate many prominent closed-source models, as well as dozens of open-source models ranging from 0.5B to 110B in size. LiveBench is difficult, with top models achieving below 60% accuracy. We release all questions, code, and model answers. Questions will be added and updated on a monthly basis, and we will release new tasks and harder versions of tasks over time so that LiveBench can distinguish between the capabilities of LLMs as they improve in the future. We welcome community engagement and collaboration for expanding the benchmark tasks and models.

1 Introduction

In recent years, as large language models (LLMs) have risen in prominence, it has become increasingly clear that traditional machine learning benchmark frameworks are no longer sufficient to evaluate new models. Benchmarks are typically published on the internet, and most modern LLMs include large swaths of the internet in their training data. If the LLM has seen the questions of a benchmark during training, its performance on that benchmark will be artificially inflated [14, 15, 20, 39], hence making many LLM benchmarks unreliable. Recent evidence of test set contamination includes the observation that LLMs' performance on Codeforces plummet after the training cutoff date of the LLM [24, 39], and before the cutoff date, performance is highly correlated with the number of times

*Correspondence to: `colin@abacus.ai`, `samuel@abacus.ai`, `goldblum@nyu.edu`.

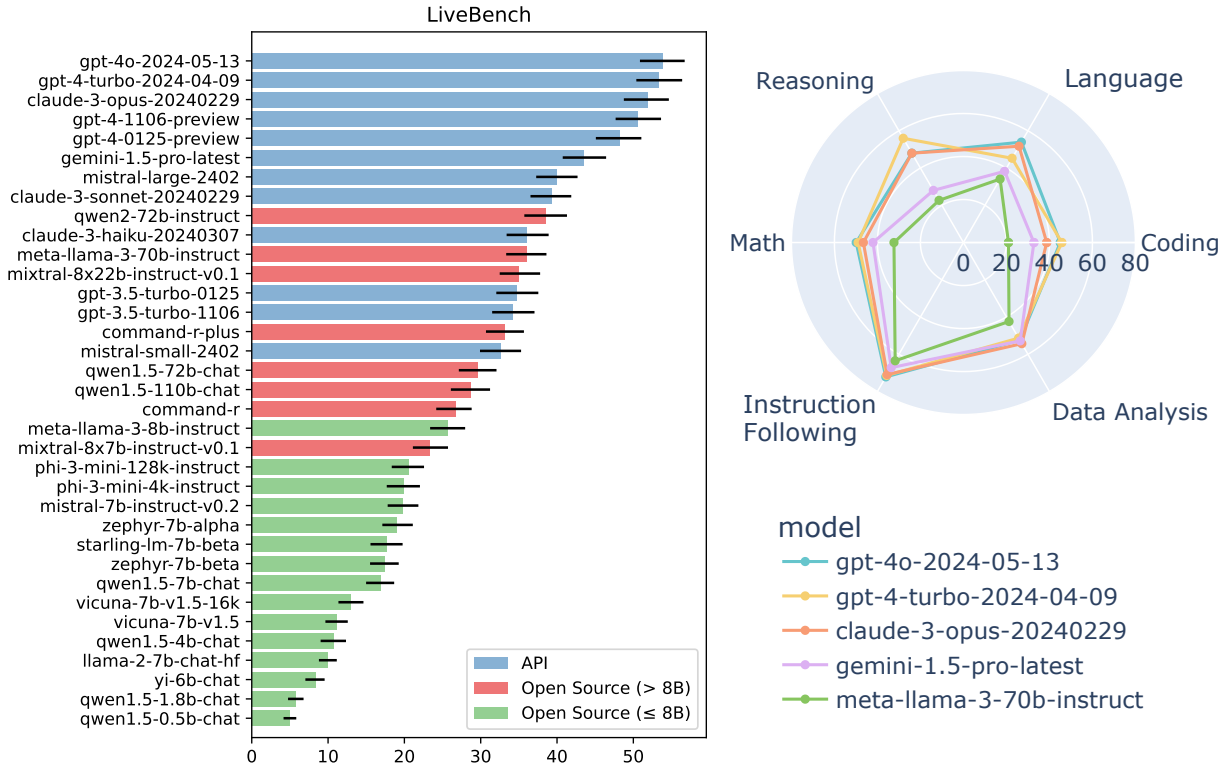

Figure 1: Results on `LiveBench` for all models, showing 95% bootstrap confidence intervals (left). A radar plot for select models across `LiveBench`'s six categories demonstrating the that ordering of top models varies between each category (right).

the problem appears on GitHub [39]. Similarly, a recent hand-crafted variant of the established math dataset, GSM8K, shows evidence that several models have overfit to this benchmark [12, 52].

To lessen dataset contamination, benchmarks using LLM or human prompting and judging have become increasingly popular [10, 24, 30, 53]. However, using these techniques comes with significant downsides. While LLM judges have multiple advantages, such as their speed and ability to evaluate open-ended questions, they are prone to making mistakes and can have several biases. For example, we will show in Section 3 that for challenging reasoning and math problems, the pass/fail judgments from `GPT-4-Turbo` **have an error rate of up to 46%**. Furthermore, LLMs often favor their own answers over other LLMs, and LLMs favor more verbose answers [16, 30, 31]. Additionally, using humans to provide evaluations of LLMs can inject biases such as formatting of the output, and the tone and formality of the writing [10]. Using humans to generate questions also presents limitations. Human participants might not ask diverse questions, may favor certain topics that do not probe a model's general capabilities, or may construct their prompts poorly [53].

In this work, we introduce a framework for benchmarking LLMs designed to be immune to both test set contamination and the pitfalls of LLM judging and human crowdsourcing. We use this framework to create `LiveBench`, the first benchmark with these three desiderata: *(1)* `LiveBench` contains frequently-updated questions based on recent information sources; *(2)* `LiveBench` is scored automatically according to the objective ground-truth without the use of an LLM judge; and *(3)*

LiveBench questions are drawn from a diverse set of six categories. We ensure *(2)* by only including questions that have an objectively correct answer in LiveBench. LiveBench questions are *difficult*; no current model achieves higher than 60% accuracy. Questions will be added and updated on a monthly basis, and we will release new tasks and harder versions of tasks over time so that LiveBench can distinguish between the capabilities of LLMs as they improve in the future.

Overview of tasks. LiveBench currently consists of 18 tasks across 6 categories: math, coding, reasoning, language, instruction following, and data analysis. Each task falls into one of two types: *(1)* tasks which use an information source for their questions, e.g., data analysis questions based on recent Kaggle datasets, or fixing typos in recent arXiv abstracts; and *(2)* tasks which are more challenging or diverse versions of existing benchmark tasks, e.g., from Big-Bench Hard [43], IFEval [54], bAbI [48], or AMPS [22]. The categories and tasks included in LiveBench are:

- **Math:** questions from high school math competitions from the past 12 months (AMC12, AIME, USAMO, IMO, SMC), as well as harder versions of AMPS questions [23]
- **Coding:** code generation questions from Leetcode and AtCoder (via LiveCodeBench [24]), as well as a novel code completion task
- **Reasoning:** a harder version of Web of Lies from Big-Bench Hard [43], a harder version of PathFinding from bAbI [48], and Zebra Puzzles (e.g., [25])
- **Language Comprehension:** three tasks: Connections word puzzles, a typo-fixing task, and a movie synopsis unscrambling task from recent movies on IMDb and Wikipedia
- **Instruction Following:** four tasks to paraphrase, simplify, summarize, or generate stories about recent new articles from The Guardian [21], subject to one or more instructions such as word limits or incorporating specific elements in the response
- **Data Analysis:** three tasks using recent datasets from Kaggle and Socrata: table reformatting (among JSON, JSONL, Markdown, CSV, TSV, and HTML), predicting which columns can be used to join two tables, and predicting the correct type annotation of a data column

We evaluate dozens of models, including proprietary models as well as open-source models with sizes ranging from 0.5B to 8x22B. We release all questions, code, and model answers, and we welcome community engagement and collaboration. Our codebase is available at https://github.com/livebench/livebench, and our leaderboard is available at https://livebench.ai.

2 LiveBench Description

In this section, we introduce LiveBench. It currently has six categories: math, coding, reasoning, data analysis, instruction following, and language comprehension. Categories are diverse with two to four tasks per problem. Each task either includes recent information sources (such as very recent news articles, movie synopses, or datasets) or is a more challenging, more diverse version of an existing benchmark task.

Each task is designed to span a range of difficulty, from easy to very challenging, while loosely aiming for a 30-70% success rate on the top models. Prompts are tailored for each category and task but typically include the following: zero-shot chain of thought [27, 47], asking the model to make its best guess if it does not know the answer, and asking the LLM to output its final answer in a way that is easy to parse, such as in **double asterisks**. In the following sections, we give an overview description of each task from each category.

2.1 Math Category

Evaluating the mathematical abilities of LLMs has been one of the cornerstones of recent research in LLMs, featuring prominently in many releases and reports [6, 7, 35, 38]. Our benchmark includes math questions of three types: questions from recent high school math competitions, fill-in-the-blank questions from recent proof-based USAMO and IMO problems, and questions from our new, harder version of the AMPS dataset [23].

Our first two math tasks, `Competitions` and `Proof Competition`, use expert human-designed math problems that offer a wide variety in terms of problem type and solution technique. First, we include questions from AMC12 2023, SMC 2023, and AIME 2024 in `Competitions` and from USAMO 2023 and IMO 2023 in `Proof Competitions`. These are challenging and prestigious competitions for high school students in the USA (AMC, AIME, USAMO), in the UK (SMC), and internationally (IMO). The competitions test mathematical problem solving with arithmetic, algebra, counting, geometry, number theory, probability, and other secondary school math topics [18].

Finally, we release synthetically generated math questions in the `AMPS_Hard` task. This task is inspired by the math question generation used to create the MATH and AMPS datasets [23]. We generate harder questions by drawing random primitives, using a larger and more challenging distribution than AMPS across the 10 hardest tasks within AMPS.

2.2 Coding Category

The coding ability of LLMs is one of the most widely studied and sought-after skills for LLMs [24, 29, 34]. We include two coding tasks in `LiveBench`: a modified version of the code generation task from LiveCodeBench (LCB) [24], and a novel code completion task combining LCB problems with partial solutions collected from GitHub sources.

In the `LCB Generation` task, we assess a model's ability to parse a competition coding question statement and write a correct answer. We include 50 questions from LiveCodeBench [24] which has several tasks to assess the coding capabilities of large language models.

The `Completion` task specifically focuses on the ability of models to complete a partially correct solution—assessing whether a model can parse the question, identify the function of the existing code, and determine how to complete it. We use LeetCode medium and hard problems from LiveCodeBench's [24] April 2024 release, combined with matching solutions from https://github.com/kamyu104/LeetCode-Solutions, omitting the last 15% of each solution and asking the LLM to complete the solution.

2.3 Reasoning Category

The reasoning ability of large language models is another highly-benchmarked and analyzed skill of LLMs [43, 47, 49]. In `LiveBench`, we include three reasoning tasks: our harder versions of tasks from Big-Bench Hard [43] and bAbI [48], and Zebra puzzles.

The `Web of Lies v2` task is an advancement of the similarly named task included in Big-Bench [5] and Big-Bench Hard [43]. The task is to evaluate the truth value of a random Boolean function expressed as a natural-language word problem. Already by October 2022, LLMs achieved near 100% on this task, and furthermore, there are concerns that Big-Bench tasks leaked into the training data of LLMs such as GPT-4, despite using canary strings [35]. For `LiveBench`, we create a new, significantly harder version by including additional deductive components and red herrings.

Next, we include a harder version of the `PathFinding` task from bAbI [48] that we call 'House Traversal'. The original task consists of sentences of the form, 'The bedroom is West of the kitchen. The kitchen is South of the garden', asking the model where a room is in relation to another room. We make this task significantly harder by adding a distinct person in each room and asking the LLM who a person would see if they traverse in a few directions.

The final reasoning task we include is `Zebra Puzzles`, a well-known reasoning task [25] that tests the ability of the model to follow a set of statements that set up constraints, and then logically deduce the requested information. We build on an existing repository for procedural generation of Zebra puzzles [36]; the repository allows for randomizing the number of people, the number of attributes, and the set of constraint statements provided. Below, we provide an example question from the `Zebra Puzzles` task.

An example question from the `Zebra Puzzle` task.

There are 3 people standing in a line numbered 1 through 3 in a left to right order.
Each person has a set of attributes: Food, Nationality, Hobby.
The attributes have the following possible values:
- Food: nectarine, garlic, cucumber
- Nationality: chinese, japanese, thai
- Hobby: magic-tricks, filmmaking, puzzles

and exactly one person in the line has a given value for an attribute.
Given the following premises about the line of people:
- the person that likes garlic is on the far left
- the person who is thai is somewhere to the right of the person who likes magic-tricks
- the person who is chinese is somewhere between the person that likes cucumber and the person who likes puzzles

Answer the following question: What is the hobby of the person who is thai? Return your answer as a single word, in the following format: **X**, where X is the answer.

2.4 Data Analysis Category

While LLMs' abilities in the previous three categories have been widely studied, we also include a category that is significantly less studied but is a practical application of LLMs that requires more attention: data analysis tasks. We include three tasks in which the LLM assists in data analysis or data science: column type annotation, table join prediction, and table reformatting. Each question makes use of a recent dataset from Kaggle or Socrata.

The first task is to predict the type of a column of a data table. To create a question for the column table annotation task (`CTA`), we randomly sample a table and randomly sample a column from that table. We use the actual column name of that column as the ground truth and then retrieve some column samples from that column. We provide the name of all the columns from that table and ask the LLM to select the true column name from those options.

Data analysts often also require a table to be reformatted from one type to another, e.g., json to CSV or XML to TSV. We emulate that task in `TableReformat` by providing a table in one format and asking the LLM to reformat it into the target format.

Finally, another common application of LLMs in data analysis is to perform table joins. In the `TableJoin` task, each question prompts an LLM to decide which columns can be used to join two different CSV tables.

2.5 Instruction Following Category

An important ability of an LLM is its capability to follow instructions. To this end, we include instruction-following questions in our benchmark, inspired by IFEval [54], which is an instruction-following evaluation for LLMs containing verifiable instructions such as "write more than 300 words" or "Finish your response with this exact phrase: {end_phrase}." While IFEval used a list of 25 verifiable instructions, we use a subset of 16 that excludes instructions that do not reflect real-world use-cases. See Appendix Table 4. Furthermore, in contrast to IFEval, which presents only the task and instructions with a simple prompt like "write a travel blog about Japan", we provide the models with an article from The Guardian [21], asking the models to adhere to multiple randomly-drawn instructions while asking the model to complete one of four tasks related to the article: `Paraphrase`, `Simplify`, `Story Generation`, and `Summarize`. We score tasks purely by their adherence to the instructions.

2.6 Language Comprehension Category

Finally, we include multiple language comprehension tasks. These tasks assess the language model's ability to reason about language itself by, (1) completing word puzzles, (2) fixing misspellings but leaving other stylistic changes in place, and (3) reordering scrambled plots of unknown movies.

First, we include the `Connections` category. Connections is a word puzzle popularized by the New York Times (although similar ideas have existed previously). In this task, we present questions of varying levels of difficulty with 8, 12, and 16-word varieties. The objective of the game is to sort the words into sets of four words, such that each set has a 'connection' between them, e.g., types of fruits, homophones, or words that come after the word 'fire'. Due to the variety of possible connection types, this task is challenging for LLMs, as shown by prior work that tested the task on the GPT family of models [44].

Next, we include the `Typos` task. The idea behind this task is inspired by the common use case for LLMs in which a user asks the LLM to identify typos and misspellings in some written text but to leave other aspects of the text unchanged. It is common for the LLM to impose its own writing style onto that of the input text, such as switching from British to US spellings or adding the serial comma, which may not be desirable. We create the questions for this task from recent ArXiv abstracts, which we ensure originally have no typos, by programmatically injecting common human typos into the text. Below is an example question from the `Typos` task.

An example question from the `Typos` task.

Please output this exact text, with no changes at all except for fixing the misspellings. Please leave all other stylistic decisions like commas and US vs British spellings as in the original text.

We inctroduce a Bayesian estimation approach forther passive localization of an accoustic source in shallow water using a single mobile receiver. The proposed probablistic focalization method estimates the timne-varying source location inther presense of measurement-origin uncertainty. In particular, probabilistic data assocation is performed to match tiome-differences-of-arival (TDOA) observations extracted from the acoustic signal to TDOA predicitons provded by the statistical modle. The performence of our approach is evaluated useing rela acoustic data recorded by a single mobile reciever.

Table 1: **LiveBench results across the 15 top-performing models.** We display in this table the highest-performing models on `LiveBench`, outputting the results on each main category, as well as each model's overall performance. See Table 3 for the results on all 34 models.

Model	LiveBench Score	Coding	Data Analysis	Instruction Following	Language	Math	Reasoning
gpt-4o-2024-05-13	**53.6**	45.1	52.4	**72.2**	**53.9**	**49.9**	48.0
gpt-4-turbo-2024-04-09	53.1	**45.7**	51.3	71.4	45.3	49.0	**56.0**
claude-3-opus-20240229	51.7	38.7	**54.3**	70.9	51.7	46.5	48.0
gpt-4-1106-preview	50.4	43.1	51.3	69.4	48.4	47.6	42.7
gpt-4-0125-preview	47.9	42.7	54.1	63.9	43.6	42.7	40.7
gemini-1.5-pro-latest	43.5	32.8	52.8	67.2	38.3	42.1	28.0
claude-3-sonnet-20240229	39.1	23.9	44.6	65.0	38.1	29.6	33.3
qwen2-72b-instruct	38.5	31.8	26.2	68.3	29.2	43.4	32.0
mistral-large-2402	37.8	16.3	42.6	68.2	28.7	32.2	38.7
claude-3-haiku-20240307	36.1	24.5	41.5	64.0	30.1	25.7	30.7
meta-llama-3-70b-instruct	36.0	20.9	42.4	63.5	34.1	32.3	22.7
mixtral-8x22b-instruct-v0.1	34.8	31.8	30.3	63.2	26.5	26.9	30.0
gpt-3.5-turbo-0125	34.8	29.2	41.2	60.5	24.2	25.5	28.0
gpt-3.5-turbo-1106	34.2	26.8	41.7	51.5	28.6	27.8	28.7
command-r-plus	32.3	15.0	24.6	71.5	23.9	24.9	34.0

Finally, we include the `Plot Unscrambling` task, which takes the plot synopses of recently-released movies from IMDb or Wikipedia. We randomly shuffle the synopses sentences and then ask the LLM to simply reorder the sentences into the original plot. We find that this task is very challenging for LLMs, as it measures their abilities to reason through plausible sequences of events.

3 Experiments

In this section, first we describe our experimental setup and present full results for 34 LLMs on all 18 tasks of `LiveBench`. Next, we give an empirical comparison of `LiveBench` to existing prominent LLM benchmarks, and finally, we present ablation studies.

Experimental setup. Our experiments include 34 LLMs total, with a mix of top proprietary models, large open-source models, and small open-source models. In particular, for proprietary models, we include six GPT models: `gpt-4o-2024-05-13`, `gpt-4-turbo-2024-04-09`, `gpt-4-1106-preview`, `gpt-4-0125-preview`, `gpt-3.5-turbo-1106`, `gpt-3.5-turbo-0125` [6, 35]; three Anthropic models: `claude-3-opus-20240229`, `claude-3-sonnet-20240229`, `claude-3-haiku-20240307` [2]; two Mistral models: `mistral-large-2402`, `mistral-small -2402` [26]; and `gemini-1.5-pro-latest` (the API version) [38].

For large open-source models, we include `command-r`, `command-r-plus` [13], `meta-llama-3-70b-instruct` [33], `mixtral-8x22b-instruct-v0.1`, `mixtral-8x7b -instruct-v0.1` [26], `qwen1.5-110b-chat`, and `qwen1.5-72b-chat` [3].

For small open-source models, we include `llama-2-7b-chat-hf` [45], `llama-3-8b-instruct` [33], `mistral-7b-instruct-v0.2` [26], `phi-3-mini-128k-instruct`, `phi-3-mini-4k-instruct` [1], `qwen1.5-0.5b-chat`, `qwen1.5-1.8b-chat`, `qwen1.5-4b-chat`, `qwen1.5-7b-chat` [3],

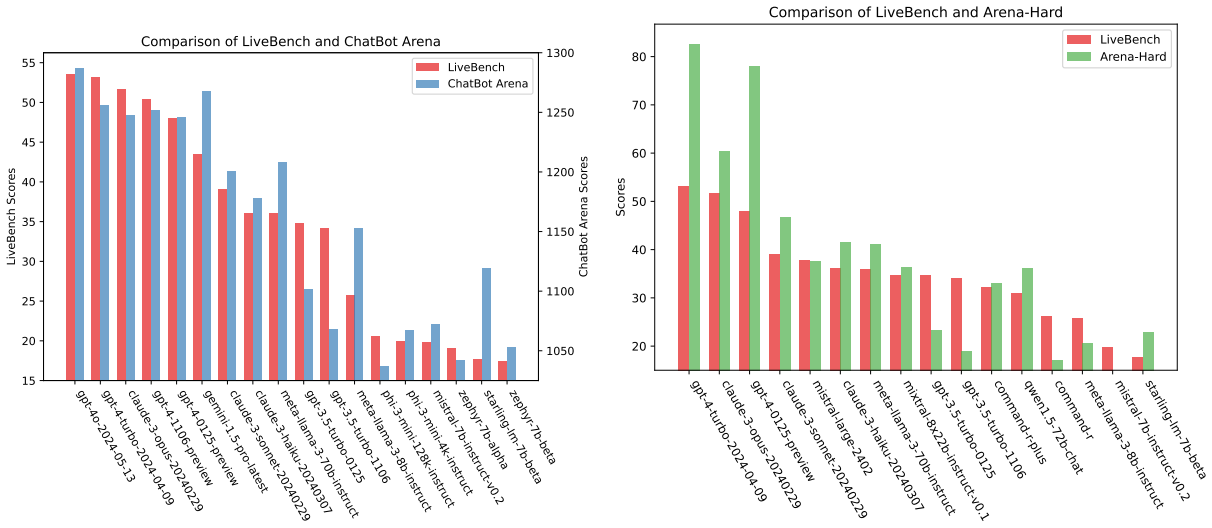

Figure 2: **Comparison of LiveBench to other LLM benchmarks.** We compare `LiveBench` to ChatBot Arena (left) and Arena-Hard (right). We see that while there are generally similar trends, some models are noticeably stronger on one benchmark vs. the other. For example, both GPT-4 models are substantially better on Arena-Hard, likely due to the known bias from using `gpt-4` itself as the LLM judge [30].

`starling-lm-7b-beta` [55], `vicuna-7b-v1.5`, `vicuna-7b-v1.5-16k` [9], `yi-6b-chat` [50], `zephyr-7b-alpha`, and `zephyr-7b-beta` [46].

For all models and tasks, we perform single-turn evaluation with temperature 0. All models run with their respective templates from FastChat [53]. We run all open-source models with `bfloat16`. For each question, a model receives a score from 0 to 1. For each model, we compute the score on each task as the average of all questions, we compute the score on each of the six categories as the average of all their tasks, and we compute the final `LiveBench` score as the average of all six categories.

3.1 Discussion of Results

We compare all 34 models on `LiveBench` according to the experimental setup described above; see Table 1 and Table 3. We find that `gpt-4o-2024-05-13` performs the best overall, with `gpt-4-turbo-2024-04-09` and `claude-3-opus-20240229` not far behind. The best-performing open-source model is `qwen2-72b-instruct`, and `meta-llama-3-8b-instruct` is the best-performing open-source model that is 8B or smaller.

We find that `gpt-4-turbo-2024-04-09` is the highest-performing LLM in the reasoning category, 8% above the next-best model. Furthermore, the GPT-4 models, particularly `gpt-4-turbo-2024-04-09`, are the highest-performing models in coding, which is in line with recent existing results [24, 41]. `gpt-4-turbo-2024-04-09` performs best on average over the 'quantitative reasoning tasks': coding, data analysis, math, and reasoning.

Table 2: **LLM judges cannot accurately evaluate challenging math and reasoning questions.** Error rate of LLM-as-a-judge scoring on challenging math (AMC, AIME, SMC) and reasoning (Zebra puzzles) tasks. On all tasks, the error rate is surprisingly high, showing that LLMs are not reliable judges for these tasks.

Model	Judge	AMC12 2024	AIME 2024	SMC 2023	Zebra Puzzles
GPT-4-Turbo	GPT-4-Turbo	0.380	0.214	0.353	0.420
Claude-3-Opus	GPT-4-Turbo	0.388	0.103	0.294	0.460

3.2 Comparison to Other LLM Benchmarks

Next, we compare `LiveBench` to two prominent benchmarks, ChatBot Arena [10] and Arena-Hard [30]. In Figure 2, we show a bar plot comparison among models that are common to both benchmarks, and in Figure 3, we compare the performance of these models to a best-fit line. We also compute the correlation coefficient of model scores among the benchmarks: `LiveBench` has a 0.92 and 0.90 correlation with ChatBot Arena and Arena-Hard, respectively.

Based on the plots and the correlation coefficients, we see that there are generally similar trends to `LiveBench`, yet some models are noticeably stronger on one benchmark vs. the other. For example, `gpt-4-0125-preview` and `gpt-4-turbo-2024-04-09` perform substantially better on Arena-Hard compared to `LiveBench` – likely due to the known bias from using `gpt-4` itself as the LLM judge [30]. We hypothesize that the strong performance of some models such as `gemini-1.5-pro-latest` and `starling-lm-7b-beta` on ChatBot Arena compared to `LiveBench` may be due to having an output style that is preferred by humans. These observations emphasize the benefit of using ground-truth judging, which is immune to biases based on the style of the output.

3.3 Comparison between Ground-Truth and LLM-Judging

In this section, we run an ablation study to compare the result of ground-truth judging with LLM judging, by taking three math sub-tasks and one reasoning task and scoring them by either matching with the ground-truth answer or by asking an LLM judge to score the answer as either correct or incorrect. We use a judge prompt based on the MT-Bench judge prompt (see Appendix A.1 for details), and we use `gpt-4-turbo-2024-04-09` as the judge. We judge the model outputs of both `gpt-4-turbo-2024-04-09` and `claude-3-opus-20240229` in Table 2. We find that the error rate for all tasks is far above a reasonable value, indicating that LLM judges are not appropriate for challenging math and logic tasks. Interestingly, the lowest error rates are on AIME 2024, which is also the task with the lowest overall success rate according to ground-truth judgment.

4 Related Work

We describe the most prominent LLM benchmarks and the ones that are most related to our work. For a comprehensive survey, see [8]. The Huggingface Open LLM Leaderboard [4, 19] is a widely-used benchmark suite that consists of six static datasets: ARC [11], GSM8K [12], HellaSwag [51], MMLU [22], TruthfulQA [32], and Winogrande [40]. While this has been incredibly useful in tracking the performance of LLMs, its static nature has left it prone to test set contamination by models.

LLMs-as-a-judge. AlpacaEval [16, 17, 31], MT-Bench [10], and Arena-Hard [30] are benchmarks that employ LLM judges on a fixed set of questions. Using an LLM-as-a-judge is fast and relatively cheap. Furthermore, this strategy has the flexibility of being able to evaluate open-ended questions, instruction-following questions, and chatbots. However, LLM judging also has downsides. First, LLMs have biases towards their own answers [30]. In addition to favoring their own answers [30], GPT-4 judges have a noticeable difference in terms of variance and favorability of other models compared to Claude judges. Additionally, LLMs make errors. As one concrete example, question 2 in Arena-Hard asks a model to write a C++ program to compute whether a given string can be converted to 'abc' by swapping two letters. GPT-4 incorrectly judges `gpt-4-0314`'s solution as incorrect [30].

Humans-as-a-judge. ChatBot Arena [10, 53] leverages human prompting and feedback on a large scale. Users ask questions and receive outputs of two randomly selected models and have to pick which output they prefer. This preference feedback is aggregated into Elo scores for the different models. While human evaluation is great for capturing the preferences of a crowd, using a human-as-a-judge has many disadvantages. First, human-judging can be quite labor-intensive, especially for certain tasks included in `LiveBench` such as complex math, coding, or long-context reasoning problems. Whenever humans are involved in annotation (of which judging is a sub-case), design choices or factors can cause high error rates [28], and even in well-designed human-annotation setups, high variability from human to human leads to unpredictable outcomes [37].

Other benchmarks Perhaps the most-related benchmark to ours is LiveCodeBench [24], which also regularly releases new questions and makes use of ground-truth judging. However, it is limited to only coding tasks. Concurrent work, the SEAL Benchmark [41], uses private questions with expert human scorers, however, the benchmark currently only contains the following categories: Math, Coding, Instruction Following, and Spanish. In [42], the authors modify the original MATH dataset [23] by changing numbers in the problem setup. They find drastic declines in model performance for all LLMs including the frontier ones. However, while such work can evaluate LLMs on data that is not in the pretraining set, the data still ends up being highly similar to the kind of data likely seen in the pretraining set. In addition, the hardness of the benchmark remains the same over time.

5 Conclusions, Limitations, and Future Work

In this work, we introduced LiveBench, an LLM benchmark designed to mitigate both test set contamination and the pitfalls of LLM judging and human crowdsourcing. LiveBench is the first benchmark that (1) contains frequently updated questions from new information sources, in which questions become harder over time, (2) scores answers automatically according to objective ground-truth values, without the use of LLM judges, and (3) contains a wide variety of challenging tasks, spanning math, coding, reasoning, language, instruction following, and data analysis. LiveBench contains questions that are based on recently released math competitions, arXiv papers, and datasets, and it contains harder, 'contamination-proof' versions of previously released benchmarks. We released all questions, code, and model answers, and questions will be added and updated on a monthly basis. We welcome community collaboration for expanding the benchmark tasks and models.

Limitations and Future Work. While we attempted to make `LiveBench` as diverse as possible, there are still additions from which it would benefit. For example, we hope to add non-English language tasks in the future. Furthermore, while ground truth scoring is beneficial in many ways, it still cannot be used for certain use cases, such as 'write an email to my boss', or 'write a travel guide to Hawaii' in which it is hard to define a ground truth. Finally, while we attempted to make all tasks and categories fair for all models, there are still biases due to certain LLM families favoring certain prompt types. We plan to update the prompts (at the start and end of each question) in the future, as new prompt strategies are developed. Similarly, we plan to update the `LiveBench` leaderboard as new LLMs are released.

References

[1] Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

[2] AI Anthropic. The claude 3 model family: Opus, sonnet, haiku. *Claude-3 Model Card*, 2024.

[3] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[4] Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. Open llm leaderboard. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard, 2023.

[5] BIG bench authors. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*, 2023.

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[7] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

[8] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45, 2024.

[9] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6, 2023.

[10] Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, et al. Chatbot arena: An open platform for evaluating llms by human preference. *arXiv preprint arXiv:2403.04132*, 2024.

[11] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018.

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

[13] Cohere. Command r: Retrieval-augmented generation at production scale. https://txt.cohere.com/command-r, March 2024.

[14] Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan. Investigating data contamination in modern benchmarks for large language models. *arXiv preprint arXiv:2311.09783*, 2023.

[15] Yihong Dong, Xue Jiang, Huanyu Liu, Zhi Jin, and Ge Li. Generalization or memorization: Data contamination and trustworthy evaluation for large language models. *arXiv preprint arXiv:2402.15938*, 2024.

[16] Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024.

[17] Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacafarm: A simulation framework for methods that learn from human feedback, 2023.

[18] J Douglas Faires and David Wells. *The Contest Problem Book VIII: American Mathematics Competitions (AMC 10) 2000–2007*, volume 19. American Mathematical Society, 2022.

[19] Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, September 2021.

[20] Shahriar Golchin and Mihai Surdeanu. Time travel in llms: Tracing data contamination in large language models. *arXiv preprint arXiv:2308.08493*, 2023.

[21] Guardian Media Group. The guardian. https://www.theguardian.com/, 1821. Accessed: 2024-01-20.

[22] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

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

[24] Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

[25] S Jeremy. Einstein's riddle: Riddles, paradoxes, and conundrums to stretch your mind. *Bloomsbury USA*, pages 10–11, 2009.

[26] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[27] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[28] Matthew Lease. On quality control and machine learning in crowdsourcing. In *Workshops at the twenty-fifth AAAI conference on artificial intelligence*, 2011.

[29] Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, et al. Starcoder: may the source be with you! *arXiv preprint arXiv:2305.06161*, 2023.

[30] Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. From live data to high-quality benchmarks: The arena-hard pipeline, April 2024.

[31] Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval, 2023.

[32] Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2022.

[33] Meta. Introducing meta llama 3: The most capable openly available llm to date. https://ai.meta.com/blog/meta-llama-3/, April 2024. Accessed: June 4, 2024.

[34] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015.

[35] OpenAI. Gpt-4 technical report. *Technical Report*, 2023.

[36] quint t. Puzzle generator and puzzle solver. https://github.com/quint-t/Puzzle-Generator-and-Solver, 2023.

[37] Hannah Rashkin, Vitaly Nikolaev, Matthew Lamm, Lora Aroyo, Michael Collins, Dipanjan Das, Slav Petrov, Gaurav Singh Tomar, Iulia Turc, and David Reitter. Measuring attribution in natural language generation models. *Computational Linguistics*, 49(4):777–840, 2023.

[38] Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[39] Manley Roberts, Himanshu Thakur, Christine Herlihy, Colin White, and Samuel Dooley. To the cutoff... and beyond? a longitudinal perspective on llm data contamination. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

[40] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8732–8740, Apr. 2020.

[41] Scale AI. Seal leaderboards. https://scale.com/leaderboard, May 2024.

[42] Saurabh Srivastava, Anto PV, Shashank Menon, Ajay Sukumar, Alan Philipose, Stevin Prince, Sooraj Thomas, et al. Functional benchmarks for robust evaluation of reasoning performance, and the reasoning gap. *arXiv preprint arXiv:2402.19450*, 2024.

[43] Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, 2023.

[44] Graham Todd, Tim Merino, Sam Earle, and Julian Togelius. Missed connections: Lateral thinking puzzles for large language models, 2024.

[45] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[46] Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. Zephyr: Direct distillation of lm alignment, 2023.

[47] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[48] Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart Van Merriënboer, Armand Joulin, and Tomas Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. In *4th International Conference on Learning Representations, ICLR 2016*, 2016.

[49] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 36, 2024.

[50] Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, et al. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*, 2024.

[51] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics.

[52] Hugh Zhang, Jeff Da, Dean Lee, Vaughn Robinson, Catherine Wu, Will Song, Tiffany Zhao, Pranav Raja, Dylan Slack, Qin Lyu, et al. A careful examination of large language model performance on grade school arithmetic. *arXiv preprint arXiv:2405.00332*, 2024.

[53] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36, 2024.

[54] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

[55] Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, and Jiantao Jiao. Starling-7b: Improving llm helpfulness & harmlessness with rlaif, November 2023.

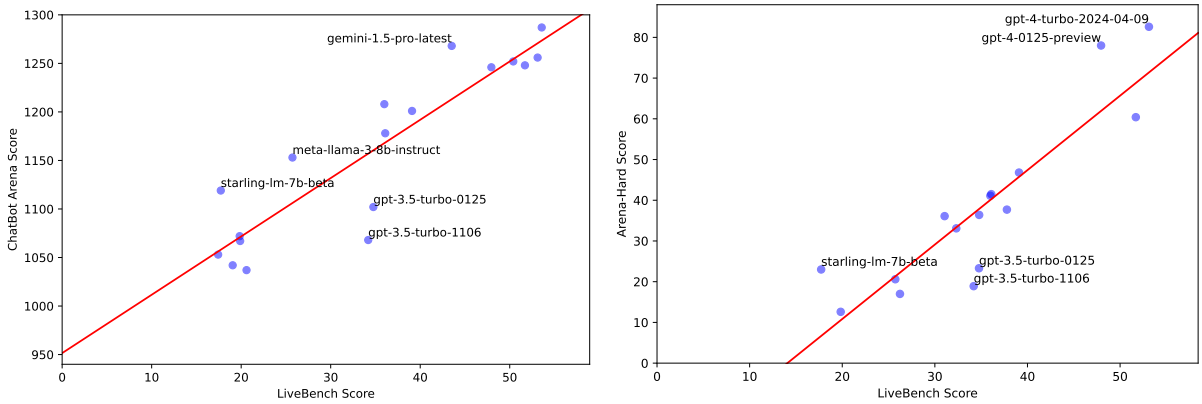

Figure 3: **The performance of models on different benchmarks, compared to a best-fit line.** We compare the different in relative performance of LLMs on `LiveBench` vs. ChatBot Arena, and `LiveBench` vs. Arena-Hard. We see that while many models are near the best-fit lines, a few are notable outliers, providing evidence that their output style may be noticeably better or worse than their ability to answer questions.

A Additional Details about LiveBench Experiments

In this section, we detail further descriptions about the `LiveBench` benchmark itself and our experiments. For example, we include further depictions of the comparisons of `LiveBench` to ChatBot Arena and Arena-Hard in Figure 3. We display the full results table for `LiveBench` in Table 3. We display the list of all verifiable instructions in Table 4. Other details are below.

A.1 Details from Ablation Studies

In this section, we give more details from Section 3.

Recall that in Section 3.3, we ran an ablation study by taking three math sub-tasks and one reasoning task, and scoring them by either matching with the ground truth answer, or by asking an LLM judge to score the answer as either correct or incorrect. We used a judge prompt similar to MT-Bench, which we duplicate below. Furthermore, to complement Table 2, we give the model performance scores for ground-truth and LLM judging for the respective models and tasks, in Table 5.

> [Instruction] Please act as an impartial judge and evaluate the quality of the response provided by an AI assistant to the user question displayed below. Your evaluation should consider correctness alone. Identify and correct any mistakes. Be as objective as possible. After providing your explanation, you must rate the response on a scale of 1 to 10 by strictly following this format: "[[rating]]", for example: "Rating: [[5]]". [Question] question [The Start of Assistant's Answer] answer [The End of Assistant's Answer]

Table 3: **LiveBench Results across 34 models.** We run 14 proprietary and 20 open-source models on `LiveBench`, outputting the results on each main category, as well as each model's overall performance.

Model	LiveBench Score	Coding	Data Analysis	Instruction Following	Language	Math	Reasoning
gpt-4o-2024-05-13	**53.6**	45.1	52.4	**72.2**	**53.9**	**49.9**	48.0
gpt-4-turbo-2024-04-09	53.1	**45.7**	51.3	71.4	45.3	49.0	**56.0**
claude-3-opus-20240229	51.7	38.7	**54.3**	70.9	51.7	46.5	48.0
gpt-4-1106-preview	50.4	43.1	51.3	69.4	48.4	47.6	42.7
gpt-4-0125-preview	47.9	42.7	54.1	63.9	43.6	42.7	40.7
gemini-1.5-pro-latest	43.5	32.8	52.8	67.2	38.3	42.1	28.0
claude-3-sonnet-20240229	39.1	23.9	44.6	65.0	38.1	29.6	33.3
qwen2-72b-instruct	38.5	31.8	26.2	68.3	29.2	43.4	32.0
mistral-large-2402	37.8	16.3	42.6	68.2	28.7	32.2	38.7
claude-3-haiku-20240307	36.1	24.5	41.5	64.0	30.1	25.7	30.7
meta-llama-3-70b-instruct	36.0	20.9	42.4	63.5	34.1	32.3	22.7
mixtral-8x22b-instruct-v0.1	34.8	31.8	30.3	63.2	26.5	26.9	30.0
gpt-3.5-turbo-0125	34.8	29.2	41.2	60.5	24.2	25.5	28.0
gpt-3.5-turbo-1106	34.2	26.8	41.7	51.5	28.6	27.8	28.7
command-r-plus	32.3	15.0	24.6	71.5	23.9	24.9	34.0
mistral-small-2402	31.4	16.3	31.9	63.9	22.1	26.8	27.3
qwen1.5-72b-chat	31.1	22.9	33.0	58.2	11.4	26.8	34.0
qwen1.5-110b-chat	28.8	22.2	31.5	55.3	13.2	25.6	25.0
command-r	26.2	12.3	31.7	57.2	14.6	16.9	24.7
meta-llama-3-8b-instruct	25.7	18.3	23.3	57.1	18.7	17.6	19.3
mixtral-8x7b-instruct-v0.1	23.2	10.0	28.1	44.8	13.8	19.0	23.3
phi-3-mini-128k-instruct	20.6	11.6	8.7	49.6	6.8	21.5	25.3
phi-3-mini-4k-instruct	19.9	14.9	14.7	40.1	7.1	19.9	22.7
mistral-7b-instruct-v0.2	19.8	11.6	14.6	51.6	9.1	16.0	16.0
zephyr-7b-alpha	19.1	11.3	17.4	52.8	7.2	9.6	16.0
qwen1.5-7b-chat	16.9	6.6	16.2	44.1	6.2	12.9	15.3
vicuna-7b-v1.5-16k	13.0	1.3	9.3	42.1	7.9	6.6	10.7
vicuna-7b-v1.5	11.2	1.0	2.7	41.8	8.7	4.3	8.7
qwen1.5-4b-chat	10.7	4.0	9.1	27.7	5.8	6.7	10.7
llama-2-7b-chat-hf	10.0	0.0	0.0	44.9	6.9	4.8	3.3
yi-6b-chat	8.3	1.3	4.4	27.2	4.7	7.1	5.3
qwen1.5-1.8b-chat	5.8	0.0	3.3	22.9	3.2	2.1	3.3
qwen1.5-0.5b-chat	5.0	0.0	0.0	21.3	2.9	3.4	2.7

Table 4: The list of 25 instructions used in [54], and the 16 that are both 'real-world' and automatically verifiable, which we used in `LiveBench`. Descriptions are from [54].

Instruction Group	Instruction	Description	In IFEval	In LiveBench
Keywords	Include Keywords	Include keywords {keyword1}, {keyword2} in your response	✓	✓
Keywords	Keyword Frequency	In your response, the word word should appear {N} times.	✓	
Keywords	Forbidden Words	Do not include keywords {forbidden words} in the response.	✓	✓
Keywords	Letter Frequency	In your response, the letter {letter} should appear {N} times.	✓	
Language	Response Language	Your ENTIRE response should be in {language}, no other language is allowed.	✓	
Length Constraints	Number Paragraphs	Your response should contain {N} paragraphs. You separate paragraphs using the markdown divider: * * *	✓	✓
Length Constraints	Number Words	Answer with at least / around / at most {N} words.	✓	✓
Length Constraints	Number Sentences	Answer with at least / around / at most {N} sentences.	✓	✓
Length Constraints	Number Paragraphs + First Word in i-th Paragraph	There should be {N} paragraphs. Paragraphs and only paragraphs are separated with each other by two line breaks. The {i}-th paragraph must start with word {first_word}.	✓	✓
Detectable Content	Postscript	At the end of your response, please explicitly add a postscript starting with {postscript marker}	✓	✓
Detectable Content	Number Placeholder	The response must contain at least {N} placeholders represented by square brackets, such as [address].	✓	
Detectable Format	Number Bullets	Your answer must contain exactly {N} bullet points. Use the markdown bullet points such as: * This is a point.	✓	✓
Detectable Format	Title	Your answer must contain a title, wrapped in double angular brackets, such as <<poem of joy>>.	✓	✓
Detectable Format	Choose From	Answer with one of the following options: {options}	✓	
Detectable Format	Minimum Number Highlighted Section	Highlight at least {N} sections in your answer with markdown, i.e. *highlighted section*	✓	
Detectable Format	Multiple Sections	Your response must have {N} sections. Mark the beginning of each section with {section_splitter} X.	✓	✓
Detectable Format	JSON Format	Entire output should be wrapped in JSON format.	✓	✓
Combination	Repeat Prompt	First, repeat the request without change, then give your answer (do not say anything before repeating the request; the request you need to repeat does not include this sentence)	✓	✓
Combination	Two Responses	Give two different responses. Responses and only responses should be separated by 6 asterisk symbols: ******.	✓	✓
Change Cases	All Uppercase	Your entire response should be in English, capital letters only.	✓	
Change Cases	All Lowercase	Your entire response should be in English, and in all lowercase letters. No capital letters are allowed.	✓	
Change Cases	Frequency of All-capital Words	In your response, words with all capital letters should appear at least / around / at most {N} times.	✓	
Start with / End with	End Checker	Finish your response with this exact phrase {end_phrase}. No other words should follow this phrase.	✓	✓
Start with / End with	Quotation	Wrap your entire response with double quotation marks.	✓	✓
Punctuation	No Commas	In your entire response, refrain from the use of any commas.	✓	

Table 5: Model Performance on math and reasoning tasks with both ground-truth (GT) or LLM judging (LLM-Jdg.)

	AMC12 2024		AIME 2024		SMC 2023		Zebra Puzzles	
	GT	LLM-Jdg.	GT	LLM-Jdg.	GT	LLM-Jdg.	GT	LLM-Jdg.
GPT-4-Turbo	54	64.000	13.793	35.714	70.588	58.824	38	68
Claude-3-Opus	56	42.857	6.897	17.241	58.824	52.941	34	52