KakuroBench: Arithmetic Constraint Satisfaction as a Benchmark for LLM Reasoning

Anonymous ACL submission

Abstract

Most existing benchmarks for evaluating reasoning in large language models (LLMs) focus on text-based, symbolic, or one-dimensional reasoning tasks, leaving fewer benchmarks for higher-dimensional structured reasoning. In this work, we introduce **KakuroBench**, a benchmark dataset designed to evaluate LLM performance on Kakuro puzzle solving. Kakuro puzzles are high-dimensional constraint-satisfaction problems that require arithmetic reasoning, yet they can be solved in a single generated response, enabling clean comparison across model families. We find that pretrained models cannot inherently solve these problems, making Kakuro a useful setting for studying the effects of supervised fine-tuning on structured reasoning. KakuroBench provides a new dataset of Kakuro puzzles and solutions, which we use to evaluate several open-weight LLMs and state-of-the-art API models under zero-shot, few-shot, and supervised fine-tuning conditions. Overall, our results establish Kakuro as a challenging and informative testbed for arithmetic reasoning, constraint satisfaction, and systematic generalization in LLMs.

1 Introduction

Over the past several years, large language models have achieved remarkable success across a wide range of natural language processing tasks, including text and code generation, question answering and complex instruction following (Brown et al., 2020; Touvron et al., 2023; Guo et al., 2025). However, despite these advances, modern LLMs still struggle with structured reasoning problems that require maintaining globally consistent solutions under strict rules and constraints (Valmeekam et al., 2022; Dziri et al., 2023). This limitation has led to growing interest in benchmark datasets based on logic puzzles and constraint satisfaction problems, since these tasks provide controlled environments for evaluating LLM reasoning (Chollet, 2019; Srivastava et al., 2023). Existing benchmarks explore a variety of structured reasoning settings, including Sudoku (Seely et al., 2025), crosswords (Leng et al., 2025), Minesweeper (Li et al., 2024), and broader textual reasoning tasks (Tsai et al., 2023). While these benchmarks test important aspects of logical consistency, they do not fully isolate the type of reasoning required for two-dimensional arithmetic constraint satisfaction. Crossword-style tasks depend heavily on language knowledge and memorization, while Sudoku constraints are purely symbolic and do not test arithmetic reasoning. Other recent reasoning environments often require chain-of-thought reasoning, iterative interaction, or external search strategies in order to achieve strong performance (Giannoulis et al., 2026). These factors can make it difficult to evaluate a model's direct ability to produce a globally consistent solution in a single response.

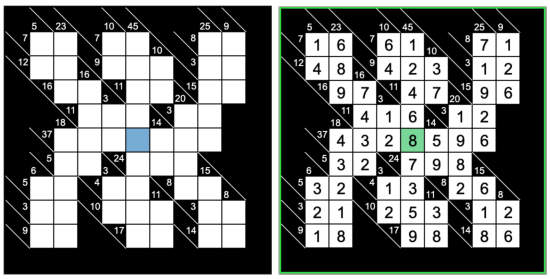

Figure 1: An example 11×11 Kakuro puzzle (left) and its corresponding solution (right). Kakuro puzzles require the player to fill empty cells with digits 1-9 such that each row/column sums to a given clue and contains no repeated digits.

To address these gaps in LLM reasoning evaluation, we introduce **KakuroBench**, a benchmark dataset based on the logic puzzle Kakuro. Kakuro combines arithmetic reasoning with combinatorial 2D constraint satisfaction, requiring the player to fill a grid such that every row and column sums to a given clue value, with no repeating digits. Solv-

ing Kakuro requires the joint satisfaction of several arithmetic constraints, with solution spaces that grow exponentially with grid size. This makes Kakuro a particularly challenging reasoning task and an ideal benchmark for evaluating structured arithmetic reasoning in large language models. Our contributions are as follows:

- We introduce **KakuroBench**, a new benchmark dataset for evaluating arithmetic reasoning and constraint satisfaction capabilities in large language models. We provide 60,000 puzzles with a grid size of 11x11 accompanied by their solutions.
- We evaluate several open-weight language models under zero-shot, few-shot, and supervised fine-tuning settings as well as several commercial language models. Our results show that the open-weight models achieve near-zero exact solution accuracy while fine-tuned and multi-step reasoning models can solve Kakuro puzzles with strong performance.

Overall, our work positions KakuroBench as a challenging and informative benchmark for measuring 2D arithmetic reasoning in language models. To the best of our knowledge, KakuroBench is the first large-scale benchmark specifically designed to evaluate LLMs on Kakuro puzzle solving. Our dataset and evaluation code will be made public upon acceptance.

2 Related Work

Arithmetic and Reasoning Benchmarks Recent research on LLMs has increasingly focused on structured reasoning tasks, 2D grid-based puzzles and constraint satisfaction problems. These tasks are especially challenging for autoregressive language models, since solving them often requires maintaining global consistency across many interacting constraints. A substantial body of work has sought to evaluate mathematical and logical reasoning in LLMs, including GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021), which have become standard benchmarks for arithmetic and algebraic reasoning. On the other hand, BIG-Bench (Srivastava et al., 2023) and BIG-Bench Hard (Suzgun et al., 2023) probe a broader range of cognitive tasks. More recent benchmarks such as MMLU (Hendrycks et al., 2020), target general knowledge, while ARC-AGI (Chollet, 2019) evaluates abstract reasoning through grid transformation problems. In general, ARC-style tasks have become common benchmarks, however they primarily focus on abstract visual pattern induction rather than arithmetic-constrained reasoning.

Games and Puzzles Games and puzzles have also attracted growing interest as evaluation environments due to several attractive qualities, such as their formal structure, objective correctness criteria, and controllable difficulty, which are not easily found in LLM evaluation benchmarks (Tsai et al., 2023). In this category, Sudoku and some of its variants have been extensively studied (Seely et al., 2025; Arias, 2026), but its constraints are purely symbolic, which means that valid solutions require only logical exclusion with no arithmetic computations. Another popular category includes crossword puzzle benchmarks (Leng et al., 2025), which test spatial and language reasoning but are entirely non-numerical. At the same time, Minesweeper (Li et al., 2024) and other spatial games have been introduced as benchmarks, but evaluation typically requires multi-turn interaction or explicit chain-of-thought reasoning, which complicates direct comparison across models.

2D Constraint Satisfaction More recently, researchers have explored the ability of LLMs to solve general combinatorial and constraint satisfaction problems including propositional satisfiability (Wei et al., 2025), graph coloring (Heyman and Zylberberg, 2025), and scheduling problems (Abgaryan et al., 2025) revealing that current models perform inconsistently on combinatorial tasks, often succeeding on small or highly structured instances while failing to scale to more complex problems. These findings suggest that many modern LLMs still struggle with maintaining consistent reasoning across large interacting constraint spaces and highlight the need for further evaluation and training benchmarks.

In contrast to prior puzzle-based reasoning benchmarks, Kakuro combines combinatorial constraint satisfaction with explicit arithmetic reasoning. This positions Kakuro as an important benchmark in this growing area of literature.

3 Dataset Construction and Overview

3.1 The Kakuro Puzzle

Kakuro is a logic puzzle played on a rectangular grid containing horizontal and vertical runs of

Model	Thinking Enabled	Baseline		Fine-tuned	
		Cell Acc.	Grid Acc.	Cell Acc.	Grid Acc.
Llama 3 8B Instruct	×	10.1 ± 0.1	0.0 ± 0.0	**91.0** ± **0.4**	**46.5** ± **0.6**
Deepseek Math 7B Instruct	×	10.8 ± 0.1	0.0 ± 0.0	87.4 ± 0.2	30.3 ± 0.6
Qwen 3 8B	×	4.7 ± 0.1	0.0 ± 0.0	90.7 ± 0.2	43.4 ± 0.7
Gemma 3 12B Instruct	×	3.7 ± 0.1	0.0 ± 0.0	90.1 ± 0.2	37.1 ± 0.7
Deepseek R1 14B	×	5.8 ± 0.1	0.0 ± 0.0	90.7 ± 0.2	41.9 ± 0.7
Qwen 3 14B	×	6.6 ± 0.1	0.0 ± 0.0	90.0 ± 0.2	39.6 ± 0.7
Mistral-3.1 24B Instruct	×	8.7 ± 0.1	0.0 ± 0.0	–	–
Claude Sonnet 4.5	×	13.6 ± 0.1	0.0 ± 0.0	–	–
GPT 5-nano	✓	74.0 ± 0.6	65.2 ± 0.7	–	–
GPT 5-mini	✓	**91.6** ± 0.4	**80.6** ± 0.6	–	–

Table 1: Zero-shot performance of baseline and fine-tuned language models. We report accuracy at two levels: cell accuracy, which measures the fraction of individual cells predicted correctly, and grid accuracy, which measures the fraction of complete grids solved with no errors. Reported ± values denote standard errors, calculated as $\text{SE} = \hat{\sigma}/\sqrt{n}$, where $\hat{\sigma}$ is the sample standard deviation of per-puzzle accuracy across the n evaluation examples.

white cells. Each run is annotated with a target sum. The objective is to fill each white cell with a number ranging from 1 to 9 such that every run sums exactly to the given clue value and no digit repeats. Figure 1 provides an example of a Kakuro grid. Kakuro is structurally similar to Sudoku and crossword puzzles, but with additional arithmetic constraints that prevent a purely symbolic solution strategy.

3.2 Dataset Overview and Creation

Our dataset consists of 60,000 11x11 Kakuro puzzles split into three subsets: $50,000$ training samples, $5,000$ validation samples and $5,000$ test samples. For each puzzle, we represent the grid as a list of rows with the following symbols:

- **"0"** : black cells that are empty and do not matter for puzzle solving.
- **"1"** : white cells that will later be filled by the models.
- **(clue_across, clue_down)**: clue cells that indicate the required sum for that row (across) and column (down). A value of 0 indicates there is no run in that direction.

For each puzzle we additionally provide the solution grid, with each **"1"** cell replaced by the correct answer. Each Kakuro board was generated using the pykakuro Python library (Thomson, 2010) using a unique seed.

4 Evaluations

4.1 Setup

All evaluations were conducted on the held-out test split of 5,000 puzzles. The models were provided with the Kakuro rules and an empty problem board that they were tasked to fill. The exact prompt used and other evaluation details are provided in Appendix A.1.

We report two primary metrics: **cell accuracy**, the fraction of individual white cells filled with the correct digit, and **grid accuracy**, the fraction of puzzles for which the entire solution is exactly correct. With the exception of the GPT 5 models, all models were evaluated without explicit multi-step thinking, though we note that some of the models (such as Deepseek R1) may have been trained with basic multi-step reasoning.

All evaluations were run on a single NVIDIA H200 GPU, while fine-tuning was performed using four NVIDIA H200 GPUs. More details about our evaluation setup can be found in Appendix A.2.

4.2 Open-Weight Model Benchmarking

In Table 1 we detail the baseline performance of various open-weight language models on the KakuroBench dataset. We consider six models at two different sizes, including a mix of base models, instruction-tuned models, and math-tuned models. Specifically, we consider Llama 3 Instruct 8B (Grattafiori et al., 2024), Deepseek Math Instruct 7B (Shao et al., 2024) and Deepseek R1 14B (Guo et al., 2025), Qwen 3 8B and 14B (Yang et al., 2025), and Gemma 3 12B (Team, 2025). More

specifically, Table 1 contains the 0-shot evaluation results for all base models, but we also provide an analysis of the few-shot evaluation setting in the Appendix A.4.

Across all models, cell accuracy remains uniformly low, with most models falling below 15%. Meanwhile, not a single model achieved a grid accuracy above 0%. These results demonstrate that current open-weight LLMs have little innate ability to solve Kakuro puzzles. Further, they suggest that Kakuro puzzles are not a strong signal in the pretraining datasets of these models, making KakuroBench a clean benchmark for evaluating reasoning ability on fine-tuned models.

4.3 API Model Benchmarking

Table 1 also shows the performance of Claude Sonnet 4.5 (Anthropic, 2025), GPT 5-mini (OpenAI, 2025), and GPT 5-nano (OpenAI, 2025a) under zero-shot prompting. These models were evaluated using the Anthropic (Anthropic, 2026) and OpenAI (OpenAI, 2025b) APIs respectively. Claude Sonnet 4.5 was evaluated with extended thinking disabled, while the GPT 5 variants can only be used with extended thinking.

Consistent with the open-weight models, Claude Sonnet 4.5 achieves a cell accuracy of only 13.6% and 0.0% grid accuracy. On the other hand, both GPT 5 models are pronounced outliers, achieving 91.6%, 74.0% cell accuracy and 80.6%, 65.2% grid accuracy respectively, far exceeding all other evaluated models. We attribute this to GPT 5's integrated chain-of- thought reasoning capabilities, which enable the model to systematically enumerate and verify arithmetic constraints rather than generate answers in a single unstructured pass.

4.4 Fine-Tuning

Finally, Table 1 details the performance of the 7-8B and 12-14B open-weight models after fine-tuning on the 50,000 sample training set. Each model was trained for three epochs using the HuggingFace trainer (HuggingFace, 2026). We used the AdamW optimizer with a learning rate of 2×10^{-5}. Additional details can be found in Appendix A.3.

After fine-tuning, all models exhibit dramatic improvements across both evaluation metrics. Cell accuracy increases to 87%–91%, while exact grid accuracy rises from 0% to 30.3%–46.5%, demonstrating that current LLMs can learn the underlying arithmetic and structural constraints of Kakuro when provided with sufficient task-specific training data.

4.5 Discussion

Across all evaluated architectures, baseline performance on KakuroBench remained consistently low, including for frontier commercial systems. It is only when models such as GPT 5-mini and -nano use multi-step reasoning, that performance increases significantly. This suggests that Kakuro was not a major signal in the pre-training corpora of the evaluated LLMs, and that their final fine-tuned performance is truly reflective of their ability to learn a new type of problem. Together, these findings suggest that KakuroBench is an informative benchmark for evaluating the extent to which language models can acquire and apply structured 2D arithmetic reasoning. To support our analysis, we also provide information about the models' failure modes in the Appendix A.5.

5 Conclusions

In this work, we introduced **KakuroBench**, a benchmark dataset for evaluating 2D arithmetic reasoning and constraint satisfaction in large language models. Our results show that baseline models generally struggle with Kakuro, while multi-step reasoning and supervised fine-tuning can produce strong performance. These findings suggest that language models are capable of solving structured 2D arithmetic problems, but that this ability depends strongly on the inference procedure and task-specific supervision. We hope KakuroBench provides a clean and useful testbed for future work on arithmetic reasoning, combinatorial generalization, and structured puzzle solving in language models.

6 Limitations

Although our experiments provide strong evidence for the difficulty and utility of KakuroBench, several limitations remain in the current evaluation setup.

First, our API-based experiments are limited to Claude Sonnet 4.5, GPT 5-mini, and GPT 5-nano, as evaluating a broader set of frontier commercial models was not feasible due to cost constraints. The strong performance of the GPT 5 models suggests that a more comprehensive evaluation of closed-source LLMs would be valuable future work, particularly as commercial models continue to evolve rapidly.

Another limitation of the current study is that our experiments focus on a single Kakuro grid size, rather than systematically evaluating performance across a broader range of sizes. At the same time, although KakuroBench includes difficulty scores, we do not perform an extensive scaling analysis on how reasoning performance changes as puzzle size and constraint density increase. Future work could investigate scaling behavior more systematically.

Additionally, our evaluation primarily considers direct-response generation and supervised fine-tuning, without exploring multi-step reasoning strategies. However, recent reasoning-focused models often rely on chain-of-thought prompting or iterative refinement to improve performance on complex tasks. In particular, during our evaluation of the GPT 5 systems, there was no way to prevent multi-step reasoning. As a result, we were unable to perform a clean ablation on explicit reasoning ability for these models. Since Kakuro naturally involves sequential constraint propagation and intermediate deduction steps, evaluating such reasoning paradigms could provide further insight into how language models perform structured arithmetic reasoning and whether explicit reasoning improves global solution consistency.

7 Ethical Considerations

This work introduces a benchmark dataset derived from systematically generated Kakuro puzzles and raises no significant ethical concerns.

References

Henrik Abgaryan, Tristan Cazenave, and Ararat Harutyunyan. 2025. Starjob: Dataset for llm-driven job shop scheduling. *arXiv preprint arXiv:2503.01877*.

Anthropic. 2025. Introducing claude sonnet 4.5. https://www.anthropic.com/news/claude-sonnet-4-5.

Anthropic. 2026. Claude api documentation. https://platform.claude.com/docs.

David Arias. 2026. Ninegrid: Sudoku reasoning benchmark for llms. Kaggle Benchmark.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

François Chollet. 2019. On the measure of intelligence. *arXiv preprint arXiv:1911.01547*.

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

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, and 1 others. 2023. Faith and fate: Limits of transformers on compositionality. *Advances in neural information processing systems*, 36:70293–70332.

Panagiotis Giannoulis, Yorgos Pantis, and Christos Tzamos. 2026. Teaching transformers to solve combinatorial problems through efficient trial & error. *Advances in Neural Information Processing Systems*, 38:133548–133580.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

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

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

Alex Heyman and Joel Zylberberg. 2025. Evaluating the systematic reasoning abilities of large language models through graph coloring. *arXiv preprint arXiv:2502.07087*.

HuggingFace. 2026. Trainer. https://huggingface.co/docs/transformers/en/trainer.

Jixuan Leng, Chengsong Huang, Langlin Huang, Bill Yuchen Lin, William W Cohen, Haohan Wang, and Jiaxin Huang. 2025. Crosswordbench: Evaluating the reasoning capabilities of llms and lvlms with controllable puzzle generation. *arXiv preprint arXiv:2504.00043*.

Yinghao Li, Haorui Wang, and Chao Zhang. 2024. Assessing logical puzzle solving in large language models: Insights from a minesweeper case study. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 59–81.

OpenAI. 2025. Gpt-5 mini model documentation. https://platform.openai.com/docs/models/gpt-5-mini.

OpenAI. 2025a. Gpt-5 nano. https://developers.openai.com/api/docs/models/gpt-5-nano. OpenAI API Documentation.

OpenAI. 2025b. Openai api platform. https://platform.openai.com/docs.

Jeffrey Seely, Yuki Imajuku, Tianyu Zhao, Edoardo Cetin, and Llion Jones. 2025. Sudoku-bench: Evaluating creative reasoning with sudoku variants. *arXiv preprint arXiv:2505.16135*.

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

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, and 1 others. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on machine learning research*.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and 1 others. 2023. 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.

Gemma Team. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.

Brandon Thomson. 2010. Kakuro tools for python - for solving or playing with kakuro (aka cross sum) puzzles. https://github.com/ianepperson/pykakuro.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Chen Feng Tsai, Xiaochen Zhou, Sierra S Liu, Jing Li, Mo Yu, and Hongyuan Mei. 2023. Can large language models play text games well? current state-of-the-art and open questions. *arXiv preprint arXiv:2304.02868*.

Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2022. Large language models still can't plan (a benchmark for llms on planning and reasoning about change). In *NeurIPS 2022 Foundation Models for Decision Making Workshop*.

Anjiang Wei, Yuheng Wu, Yingjia Wan, Tarun Suresh, Huanmi Tan, Zhanke Zhou, Sanmi Koyejo, Ke Wang, and Alex Aiken. 2025. Satbench: Benchmarking llms' logical reasoning via automated puzzle generation from sat formulas. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 33820–33837.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

A Appendix

A.1 Prompt Format

All models are prompted using the following system instruction:

> You are a Kakuro puzzle solver. Given a Kakuro grid, output the completed solution.
>
> Grid notation: 0 = black cell (r,d) = clue cell (r = right sum, d = down sum; 0 means no clue in that direction) 1 = empty cell to fill with a digit 1-9
>
> Rules: each run must sum to its clue, no digit repeats within a run.
>
> Output only the completed grid in the same format, replacing each 1 with the correct digit.

The user prompt then consists of the puzzle grid in row-wise format:

> Solve this Kakuro puzzle: [0,0,(0,16),(0,24),0,...]
> [0,(17,0),1,1,(0,29),...] ...

A.2 Evaluation Details

All open-weight models were evaluated using greedy decoding with sampling disabled. For each example, the model was given the formatted Kakuro puzzle and asked to output only the completed grid. We used the model-specific chat template provided by the corresponding HuggingFace tokenizer, with a generation prompt appended before decoding. Unless otherwise stated, models were evaluated with a maximum generation length of 512 new tokens and a batch size of 8. All evaluations were conducted on a single NVIDIA H200 GPU, and each evaluation took a maximum of 6 GPU hours.

Frontier models were evaluated differently depending whether the models were reasoning models or not. All frontier models were evaluated using batch APIs with zero-shot prompts. For reasoning models (GPT 5-mini and GPT 5-nano), we did not specify any token budget or temperature, in order to allow the models to produce results with all the required reasoning. For Claude Sonnet 4.5, and

Mistral-3.1 we set the token budget to 512 tokens. We also set temperature to 0 for Mistral-3.1. For Mistral-3.1 Small, we used a local API serving of the 24B parameter model with 8-bit GGuF quantization.

We evaluate models on the held-out test split. For each generated response, we parse the model output into a grid using the same grid format as the dataset. If the response cannot be parsed into a valid grid, it is counted as a parse failure. We report two accuracy metrics. Cell accuracy measures the fraction of fillable cells for which the model predicts the correct digit, while grid accuracy measures the fraction of puzzles for which every fillable cell is correct. Thus, grid accuracy is a stricter exact-match metric over the entire completed puzzle. Parse failures are assigned zero credit under both metrics.

A.3 Fine-tuning Details

We fine-tune open-weight causal language models using supervised next-token prediction on the KakuroBench training split. Each training example consists of the same instruction and puzzle format used at evaluation time, followed by the completed solution grid. The loss is applied only to the solution tokens: prompt tokens are masked so that the model is trained to generate the completed grid conditioned on the puzzle, rather than to reproduce the prompt.

All fine-tuning experiments use the HuggingFace Trainer. Unless otherwise stated, models are trained for three epochs with AdamW, a learning rate of 2×10^{-5}, per-device batch size 4, gradient accumulation over 4 steps, and bfloat16 precision. We use a maximum sequence length of 1024 tokens and enable gradient checkpointing to reduce memory usage. Training is performed with distributed data parallelism on four NVIDIA H200 GPUs. All fine-tunings required at most 40 GPU hours.

A.4 Few-Shot Evaluations

For few-shot evaluation, we prepend a small number of Kakuro examples and their solutions to the prompt before the test puzzle. We evaluate 0-shot, 1-shot, and 3-shot settings for the open-weight baseline models. Table 2 reports few-shot performance.

Overall, few-shot prompting does not substantially improve exact puzzle solving, though Llama 3 Instruct performed particularly well in the 3-shot setting. All models still achieve 0.0% grid accuracy across all few-shot settings. This suggests that simply providing a small number of input-output examples is insufficient for these models to infer and reliably execute the global arithmetic constraints required by Kakuro.

A.5 Failure Modes

We note distinct failure modes in different models in Table 3.

Based on these results, specifically for the frontier models used, we observe distinct sets of behaviors depending on the vendor, model and the type of inference. The most common parse failure occurred when models returned no output at all. In rarer cases, models began generating explanations of their reasoning process and exhausted the available output token budget before producing a final solution. Overall, the relatively low parse failure rates across most frontier models suggest that these systems are generally capable of understanding the task instructions, following the required output format, and processing the puzzles end-to-end.

Among the open-weight models, nearly all systems achieved very low parse failure rates, indicating that formatting and prompt interpretation were not a major source of error. The primary exception was Gemma 3, which exhibited an unusually high parse failure rate of 65.8% with zero-shot prompting, and Deepseek R1, which had a moderate parse failure rate of 37.5% with three-shot prompting. After an investigation into the model outputs, we found that Gemma was outputting poorly formatted boards, often excluding brackets and commas. Meanwhile, Deepseek R1 output formats that did not comply with the instructions and did not follow the few-shot examples, such as specifying its rows in plain text, rather than organized as a list.

A.6 Licenses of the Dataset and other artifacts

We release KakuroBench under the Creative Commons Attribution 4.0 International license (CC BY 4.0). This permissive license allows others to use, redistribute, and adapt the dataset, including for commercial purposes, provided that they give appropriate credit to the original authors. The dataset is synthetically generated and does not contain personal or user-generated data. Any accompanying code is released under the MIT License.

The dataset generation code builds on a modified version of the open-source `pykakuro` implementation, which is licensed under the GNU Affero General Public License v3.0 (AGPL-3.0). We acknowledge this dependency and release any modi-

Model	Cell Accuracy		Grid Accuracy	
	1-shot	3-shot	1-shot	3-shot
Llama 3 8B Instruct	11.7 ± 0.1	35.9 ± 0.2	0.0 ± 0.0	0.0 ± 0.0
Deepseek Math 7B Instruct	11.4 ± 0.1	12.5 ± 0.1	0.0 ± 0.0	0.0 ± 0.0
Qwen 3 8B	8.9 ± 0.1	3.4 ± 0.1	0.0 ± 0.0	0.0 ± 0.0
Gemma 3 12B Instruct	11.1 ± 0.0	11.4 ± 0.1	0.0 ± 0.0	0.0 ± 0.0
Deepseek R1 14B	7.1 ± 0.1	0.9 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
Qwen 3 14B	0.4 ± 0.0	0.7 ± 0.0	0.0 ± 0.0	0.0 ± 0.0

Table 2: Cell-level and grid-level accuracy for baseline non-finetuned language models evaluated under few-shot prompting settings. We report cell accuracy, which measures the fraction of individual cells predicted correctly, and grid accuracy, which measures the fraction of complete grids solved with no errors. Reported $\pm$ values denote standard errors. Models marked with * use multi-step reasoning.

Model	Parse Failures (%)		
	0-shot	1-shot	3-shot
Llama 3 8B Instruct	0.0	0.0	0.0
Deepseek Math 7B Instruct	2.6	0.0	0.0
Qwen 3 8B	1.6	0.1	0.1
Gemma 3 12B Instruct	65.8	0.0	0.0
Deepseek R1 14B	1.0	1.0	37.5
Qwen 3 14B	0.1	0.2	0.4
Mistral-3.1 Small	0.5	–	–
Claude Sonnet 4.5	7.4	–	–
GPT 5-nano*	16.6	–	–
GPT 5-mini*	4.5	–	–

Table 3: Parse failure rate for each evaluated model under different few-shot prompting settings. A parse failure occurs when the model response does not follow the specified Kakuro output format. Models marked with * use multi-step reasoning.

fied generator code derived from `pykakuro` under the same AGPL-3.0 license. Any additional evaluation or analysis code not derived from `pykakuro` is released separately under the MIT License.

A.7 AI Use Declaration

We have used LLMs to assist with the writing process following the AI assistance policy, where assistance was limited to grammar and tone editing.