Disentangling Text and Math in Word Problems: Evidence for the Bidimensional Structure of Large Language Models' Reasoning

Pedro Calais
UFMG
pcalais@dcc.ufmg.br

Gabriel Franco
Boston University
gvfranco@bu.edu

Zilu Tang
Boston University
zilutang@bu.edu

Themistoklis Nikas
Boston University
tnikas@bu.edu

Wagner Meira Jr.
UFMG
meira@dcc.ufmg.br

Evimaria Terzi
Boston University
evimaria@bu.edu

Mark Crovella
Boston University
crovella@bu.edu

Abstract

Do large language models (LLMs) process text and mathematics as a unified skill, or do these components rely on distinct underlying mechanisms? We investigate this question by disentangling the textual interpretation and mathematical solving steps in math word problems drawn from Brazil's largest college entrance exam (ENEM) and GSM8K, a popular grade school-level benchmark. Using the symbolic solver *SymPy*, we transform word problems into equivalent purely mathematical representations, isolating equation formulation from textual comprehension. Our extended benchmarks enable a structured analysis of LLM performance across these two dimensions. Through empirical evaluations, we find that small-scale LLMs struggle significantly more with text interpretation than with equation solving, with accuracy dropping by a factor of 2 to 7 when solving full word problems compared to their math-only counterparts. Exploratory factor analysis confirms a bidimensional structure in LLM reasoning, where models exhibit distinct proficiencies in textual and mathematical components, underscoring the need for targeted improvements in language comprehension. Through factor analysis, we provide insights into model selection, helping practitioners make informed choices based on computational costs and task requirements.

1 Introduction

Math word problems represent a crucial category of cognitive tasks evaluated in large language models, requiring decomposing complex textual descriptions into smaller components and applying systematic logical reasoning, typically over multiple steps (Strohmaier et al., 2022; Zhu et al., 2022). An example of a math word problem is the famous baseball bat and ball cost brain teaser (Leron and Hazzan, 2009): *A baseball bat and ball together cost one dollar and 10 cents. The bat costs one dollar more than the ball. How much does the ball cost?*. Arriving at the correct answer requires two steps: (1) interpreting the problem expressed in natural language, which involves extracting and relating variables to build the mathematical equation $x + (1 + x) = 1.10$, and (2) solving the equation using mathematical reasoning, where the linguistic details are no longer relevant.

Motivated by educational research showing that humans need a combination of cognitive skills to solve math word problems (Daroczy et al., 2015; Strohmaier et al., 2022), we pose two research questions: (1) How frequently do small-scale language models err in equation construction compared to equation solving? (2) Do different language models exhibit systematic differences in their ability to construct equations versus solve them? These questions are important for NLP because they help characterize the limitations of language models in real-world analytical problems, which come in textual, not math forms.

While correctly answering a math word problem suggests that a model possesses the necessary skills (though guessing remains a possibility), incorrect answers can stem from different sources: an error in interpreting the question and building the underlying mathematical equation, an error in solving the equation, or both. To decompose these sources of errors, we extend existing benchmarks containing word problems to include each problem in their math form, in which the textual interpretation and formulation of the underlying equation has already been conducted. In the baseball ball price problem, once the story-based question has been understood and the variables identified, it can be translated into a purely mathematical problem: *Solve x + (1 + x) = 1.10*. When a human or AI is presented with this version of the problem, it does not have to account for implicit details or linguistic ambiguities but must still demonstrate mathematical reasoning in solving a linear equation. Conversely, when faced with the original word problem, both humans and AIs can struggle to construct the correct equation. It is common for college-level students to misinterpret the problem and construct the incorrect equation $x + 1.00 = 1.10$, leading to the wrong answer of 0.10 instead of 0.05.

We use word problems sampled from two sources containing problems of different difficulty levels: (1) 40 questions extracted from the largest Brazilian college admission-level math exam (ENEM), which tests highly analytical capabilities (Almeida et al., 2023), and (2) an equal-sized sample from GSM8K (Cobbe et al., 2021), a popular benchmark containing grade-school level (and hence easier) word problems. To systematically convert word problems into their pure-math equivalents, we leverage Python's symbolic solver *SymPy* (Meurer et al., 2017). We use a state-of-the-art generative language model (gpt-4-turbo-2024-04-09) to generate the *SymPy* source code that represents the underlying mathematical structure of each problem. This symbolic representation serves as the basis for formulating the pure-math version of the question, removing any linguistic interpretation challenges. After constructing extended benchmarks based on ENEM and GSM8K and evaluating 35 models from different families in different versions and stages of fine-tuning and with zero-shot and two-shot chain-of-thought prompts, our findings are:

- For both grade-school and especially college entrance levels, language models struggle more with the textual interpretation step than with the mathematical step. In a typical question, errors in its textual version range from 2 to 7 times higher than in the pure-math form.

- The relative difficulty of questions in the math vs. word forms varies significantly in different benchmarks. This highlights how benchmarks that test the same type of broad class of problems ("math word problems") can actually evaluate different aspects of the models; some benchmarks may demand more math capabilities and others, textual interpretation.

- Exploratory factor analysis (Finch, 2013) unveils two latent factors that align strongly with text vs. math ability, indicating distinct skill profiles in small-scale language models: Some models excel at solving pure-math problems but struggle with word problems, while a smaller subset of models demonstrate strong performance in both areas. We show that factor analysis is a powerful tool for comparing models and prompt variations beyond simple accuracy metrics. By analyzing multiple variations of different versions of models from the same family, we can track their evolution over time and pinpoint whether a new version introduces a fundamentally new skill.

In addition to making the extended benchmarks publicly available to encourage further research, our work aligns with the growing trend of designing controlled benchmarks that test specific capabilities of language models by systematically varying key features of the problem space. This approach enables more nuanced analyses beyond accuracy (Burnell et al., 2023b) and increases the relevance and informative power of benchmarking efforts (v. Kistowski et al., 2015).

2 Related Work

In educational assessments, a word problem is a mathematical or logical problem presented as a narrative rather than a direct mathematical equation (Verschaffel et al., 2020). Math word problems require test-takers to identify explicit and implicit quantities in the text, model their relationships through mathematical operations, and apply appropriate mathematical techniques to solve them (Koncel-Kedziorski et al., 2016). These characteristics make word problems a compelling way to evaluate the reasoning and problem-solving capabilities of AI models, attracting the NLP community for some decades (Roy et al., 2015).

With the rise in popularity of large language models, benchmarks such as GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) have become popular choices for assessing model performance, as they provide structured problem sets with varying levels of complexity. To elucidate reasons that make language models fail, there is a growing trend in AI evaluation that designs benchmarks that systematically vary key features of problem formulations to enable a more nuanced analysis (Burnell et al., 2023b). Controlled benchmarks have explored different aspects, including the inclusion of distracting contextual information (Hosseini et al., 2024; Mirzadeh et al., 2024), variations in numerical representations (Shen et al., 2023), search space size (Lin et al., 2025), compositional tasks (Dziri et al., 2024) and asking the

LLM whether a given solution is correct (Zhang et al., 2024a). A critical aspect of those works – which we build upon – is that accuracy as an aggregate metric is not enough to understand reasoning and hence strategies that evaluate step-specific errors are crucial (Gaur and Saunshi, 2023; Zheng et al., 2024; Yan et al., 2024; Xia et al., 2024).

Our work introduces a new controlled dimension to LLM evaluation: comparing model performance on textual vs. symbolic representations of math word problems. Recent work has used LLMs to generate word problems (Xie et al., 2024; Akyürek et al., 2023; Christ et al., 2024); in this work we are interested in systematically converting existing word problems into their equivalent mathematical equations using Python's symbolic solver, *SymPy* (Meurer et al., 2017). This structured transformation allows for a controlled comparison of language model performance across different representations of the same underlying problem. While previous studies have used symbolic solvers to assist language models in solving word problems (He-Yueya et al., 2023; Kao et al., 2024), to the best of our knowledge we are the first to leverage symbolic solvers to construct a controlled benchmark that enables direct comparison between textual and symbolic problem representations in LLM evaluation.

Our work is also inspired by studies on the behavior and skills that humans need to solve math word problems (Daroczy et al., 2015; Strohmaier et al., 2022). The reasons that make word problems challenging for humans have been thoroughly investigated (Daroczy et al., 2015; Jaffe and Bolger, 2023), and it is accepted in the educational community that linguistic and mathematical factors are distinct, unique skills that contribute to student performance (Vessonen et al., 2024; Pongsakdi et al., 2020). Recent work has examined word problem characteristics impacting LLMs and found that they struggle more with longer questions and rare numerical tokens (Srivatsa and Kochmar, 2024). In our work, we deepen the analysis on how large language models solve word problems by applying *exploratory factor analysis* (Finch, 2013; Lorenzo-Seva and Ferrando, 2006), a dimensionality reduction technique widely used in the social sciences to uncover latent patterns in complex datasets. While at least one study has employed factor analysis to examine LLM response patterns—finding that reasoning, comprehension, and language modeling are distinct high-level capabilities in LLMs (Burnell et al., 2023a)—our study extends this approach by leveraging a fine-grained, question-level annotation of each problem (whether it is in the math or word form). This approach allows us to quantify latent cognitive dimensions that differentiate model performance across textual and symbolic formats, providing deeper insight into how LLMs engage with different types of problem representations.

3 Benchmarks

We have built two benchmarks following the same methodology but with a different source of word problems. The first source is GSM8K (Cobbe et al., 2021)[1], which contains grade school-level problems and has been used by many previous work evaluating language models (Liu et al., 2023). GSM8K is a source of *prototype word problems*, in which mathematical word problems follow a simple, linear syntax and are relatively short (Strohmaier et al., 2022; Mirzadeh et al., 2024).

To explore questions that involve more complex interactions between text interpretation and mathematical reasoning, we used the Math section of the Brazilian *Exame Nacional do Ensino Médio* (ENEM) as a source of *complex word problems*, characterized by presenting potentially redundant information in a syntax that does not merely mirror the mathematical task (Strohmaier, 2020). ENEM is the world's second-largest university entrance exam after China's Gaokao, taken annually by millions of Brazilian students (Silveira and Mauá, 2018), and Mathematics is consistently the most challenging section for both human test-takers and language models, with scores around 30% (Locatelli et al., 2024). According to the official documentation (Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira, 2025), the exam assesses geometric reasoning, magnitudes and measurement interpretation, data analysis from graphs and tables, and trend prediction through extrapolation and interpolation. A typical ENEM word problem is shown below:

> *The subway system in a municipality offers two types of tickets, differentiated by their colors: blue and red. These tickets are sold in booklets, each containing*

[1] Dataset released under the MIT License. As per `https://huggingface.co/datasets/openai/gsm8k`, the dataset was created to support the task of question answering on basic mathematical problems that require multi-step reasoning

> *nine tickets of the same color and with the same unit price. Two booklets of blue tickets and one booklet of red tickets are sold for R$ 32.40. It is known that the price of one blue ticket minus the price of one red ticket is equal to the price of one red ticket plus five cents. What is the price, in reais, of a booklet of red tickets?*

Transforming the word question in its pure math form. Starting from the observation that word problems require building a mathematical equation as a well-recognized subproblem (Daroczy et al., 2015), we leverage the symbolic solver *SymPy*[2] (Meurer et al., 2017) to generate mathematical unambiguous representations of word problems. This is how the subway ticket problem is modeled and solved using *SymPy*:

Listing 1: SymPy code that solves the subway ticket problem.

```
import sympy as sp

# Price of one blue and one red ticket
b = sp.Symbol(’b’)
r = sp.Symbol(’r’)

eq1 = 2 * 9 * b + 9 * r - 32.40
eq2 = b - r - (r + 0.05)

solution = sp.solve((eq1, eq2), (b, r))

price_red_booklet = 9 * solution[r]

solution, price_red_booklet
```

We then submit a new prompt to GPT-4 instructing it to convert the *SymPy* code into a question in the format "Solve... <equation>.," where all known variables are replaced with their actual values, and only the unknowns are retained. For the subway ticket ENEM word problem, the equivalent math formulation generated from the *SymPy* code above is _Solve 9*r from_ $18b + 9r = 32.40$ _and_ $b - r = r + 0.05$. A key advantage of using *SymPy* is its enforcement of a structured and precise format for mathematical expressions, eliminating unnecessary linguistic details and focusing solely on the mathematical formulation.

To construct the extended benchmark, we considered 40 ENEM math questions from exams spanning 2016 to 2024, using GPT-4 to generate their corresponding *SymPy* representations, yielding a total of 80 questions. We selected questions randomly to ensure diversity regarding difficulty and problem type, excluding those requiring images or tables to focus on language-based mathematical reasoning. The questions were translated into English using GPT-4 before the *SymPy* pipeline. To validate correctness, we executed the generated *SymPy* code and verified that it produced the expected answer. We manually reviewed and curated the *SymPy* representations to ensure accuracy and fidelity to the original problems. The same process has been conducted with GSM8K; the full prompts can be found in Appendix A, and examples of questions in their word and math forms can be found in Appendix B.

There is ongoing debate regarding the efficacy of multiple-choice versus open-ended questions for assessing reasoning abilities (Zhang et al., 2024b). We opted for open-ended responses to eliminate the confounding effect of random guessing in multiple-choice formats. This ensures cleaner signal integrity in our analysis, reducing noise and enabling more precise evaluation of model reasoning.

4 Accuracy in word and math forms

We evaluate 35 models, focusing on small language small language models to small-scale large language models (1.5B, 3B, 7B, 14B), across 9 families (Gemini, Qwen, Falcon, Phi, Mistral, Gemma, Llama, NuminaMath, distilled versions of DeepSeek-R1) and training strategies (pre-trained vs. fine-tuned). We included multiple versions of the same family and also math-specialized models (e.g., Qwen2.5-Math-7B) to track model evolution over time. The models were tested with 0-shot and 2-shot settings, using chain-of-thought prompt styles following prior work (Hosseini et al., 2024). The prompt details and full list of models is provided in Appendixes A and C.

Differences across benchmarks. Our first key investigation is how accuracy varies across question forms (word vs. math) for all models and for the two annotated benchmarks. In Figure 1, each datapoint represents a question (red is ENEM and blue, GSM8K), with its accuracy on the math and word forms plotted respectively on the x- and y-axes and averaged over all models and prompt combinations. As expected, the deviations from the $y = x$ line indicate that word problems are generally more challenging than their math counterparts. Some non-obvious patterns emerge:

1. College-admission level questions are typically harder than grade-school level questions

[2] Version 1.12.

both in terms of equation formulation and equation resolution. The median accuracy on the math version of college-admission problems (ENEM) is around 0.5, compared to 0.8 for grade-level problems. In the word forms, accuracy is around 0.1 for college-entrance and 0.5 for grade-level.

2. The clear separation of ENEM and GSM8K datapoints in the scatter plot, with a small overlap region, suggests that the nature of the questions differs between the two exams and, based on the accuracy on the word and math forms of a question, the exam it belongs to could be predicted. We look at the specific characteristics of questions in Section 5.

3. In the college-admission ENEM exam, accuracy varies significantly across the difficulty of the math form (from 0.3 to 0.8), but the accuracy considering the word forms is concentrated around 0.1 to 0.2.

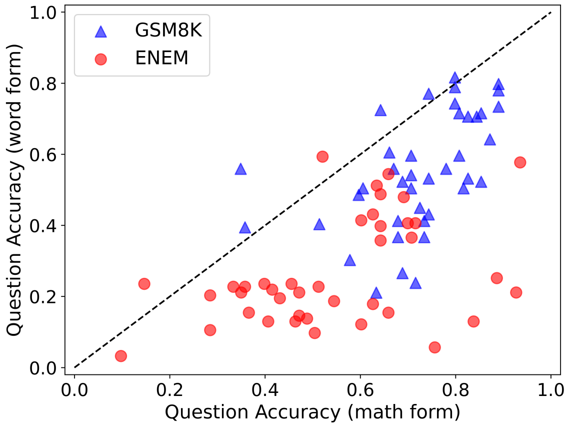

Figure 1: Differences in accuracy between math and word forms for all questions. For the models we experimented with, college-admission questions tend to be harder both in their word and math forms.

We can compute several descriptive statistics that summarize Figure 1 and highlights the differences between the two benchmarks. In Table 1, the lower Pearson and Spearman correlations in the ENEM exam means that the accuracy in math form is a poorer predictor of accuracy in the word form and vice-versa; the lower harmonic mean points to higher variability, possibly due to linguistic complexity or differences in problem structure. In GSM8K, the difficulty of the questions is more stable across formats and questions are closer to $y = x$, indicating that models generally maintain similar accuracy when switching from mathematical notation to word problems and performance is more balanced.

metric	**GSM8K**	**ENEM**
Avg. word accuracy	0.520	0.266
Avg. math accuracy	0.705	0.545
Avg. harmonic mean	0.586	0.338
Pearson correlation	0.515	0.426
Spearman correlation	0.573	0.380
Avg. distance to $y = x$	0.143	0.203

Table 1: Metrics comparing GSM8K and ENEM according to math vs. word accuracy averages and correlations.

Differences across models. Figure 2 focuses on the aggregate accuracy of individual models when presented with the different forms of the math word problems. At the extremes, the highest- and lowest-performing models on word problems show similar accuracy regardless of the form, and greater disparities are observed in models exhibiting average performance. The blue datapoints, representing models evaluated on GSM8K, tend to be closer to the $y = x$ line, indicating that the word-to-math translation step in GSM8K is easier when compared with ENEM. These findings, together with the per-question breakdown in Figure 1 and the metrics in Table 1, provide valuable insights for researchers and practitioners: depending on whether their focus is on language understanding or on pure mathematical reasoning, they can make more informed choices when selecting an appropriate benchmark that will demand from models more abilities on one skill or another, and also put special focus on specific questions that are more challenging either in their textual or mathematical components.

We also investigated the impact of the prompt choice in model performance in the word and math forms of the questions. In Figure 3, for ENEM, we see that chain-of-thought prompts improve math forms disproportionally more than the word forms; we see this pattern as evidence that the word forms demands a higher level of reasoning that chain-of-thought typically cannot reach.

Finally, to further investigate model performance, Figure 4 ranks models by accuracy on ENEM's word form of the exam. A clear pattern emerges: while accuracy in the word form exam declines sharply for most models, some maintain strong performance on math forms. Notably, the blue (math) and orange (word) bars are not strongly

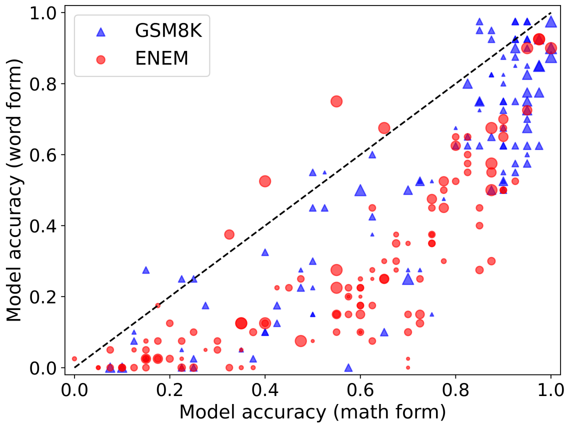

Figure 2: Differences in accuracy between math and word forms for all models and prompt combinations. Datapoint size is proportional to model size.

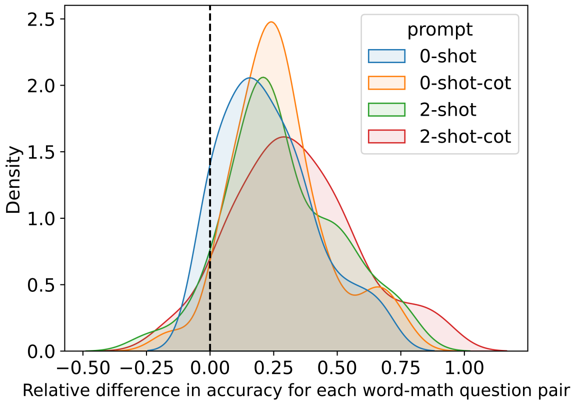

Figure 3: Density plot of the absolute difference in accuracy in math and word forms for ENEM. 2-shot-CoT disproportionately improves math questions.

correlated – some models achieve high math accuracy but struggle with word problems. This disparity suggests the presence of uncorrelated latent factors influencing language model performance on word problems, motivating the factor analysis we conduct in the next section.

5 Exploratory Factor Analysis

In pyschometrics, determining the dimensionality of an instrument is an important step towards assessing its validity and reliability. It informs whether the instrument measures one or multiple constructs, how its scores should be interpreted, and which models such as Item Response Theory apply (Gustafsson and Åberg-Bengtsson, 2010).

Unidimensional instruments (i.e., those measuring a single construct) tend to have high internal consistency because all items are strongly correlated with the same underlying factor. A popular metric is *Cronbach's Alpha* (α) (Cronbach, 1951), which measures the internal consistency of a set of items by calculating the average correlation between all pairs of items and considering how each item contributes to the overall variance of the test scores. For the ENEM and GSM8K we found α of 0.972 and 0.978, respectively, indicating very high internal consistency and minimal random error. The high consistency we found supports that the size of our benchmarks is enough for our study and is consistent with previous work that found that large benchmarks such as GSM8K, albeit containing thousands of items, can be reduced to a few dozen itens and keep a good degree of discrimination among LLMs (Polo et al., 2024).

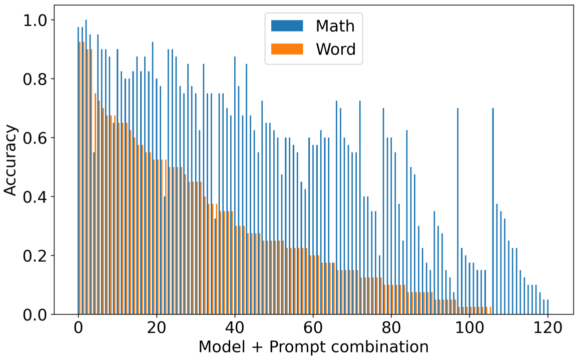

Figure 4: Accuracy by model, comparing performance on word and math questions on the ENEM exam. Models are sorted by accuracy on the word-based exam. While most models show a sharp decline in word-problem accuracy, some retain strong performance on math problems.

However, an instrument exhibiting high internal consistency is not necessarily unidimensional: multidimensional scales can still produce high α values if its items form strongly correlated subcomponents (Taber, 2018; Tavakol and Dennick, 2011). This observation, alongside the findings in Section 4, motivates conducting exploratory factor analysis – a well-established method for testing the dimensionality of an instrument (Tate, 2003). The goal is to identify distinct skills or dimensions that may underlie the observed variables. We employed the software *Factor* (Ferrando and Lorenzo-Seva, 2017), selecting unweighted least squares as the estimation method and using promax rotation. To determine the optimal number of factors that represent the data, we used parallel analysis, which compares the eigenvalues derived from the actual data with those from random datasets generated under equivalent conditions. Factors with eigenvalues exceeding those of the random data are considered significant, suggesting they account for meaningful variance (Timmerman and Lorenzo-Seva, 2011).

Our experiments revealed two statistically significant factors on both benchmarks. Details are provided in Table 2. Kaiser-Meyer-Olkin (KMO) is a measure of sampling adequacy of the data; values above 0.70 are considered suitable for factor analysis (Shrestha, 2021). Note that in both benchmarks, the KMO value is within an acceptable range, the factors explain about 40% of the variance in the data, and they are positively correlated.

metric	**ENEM**	**GSM8K**
Kaiser-Meyer-Olkin (KMO)	0.767	0.791
# of factors	2	2
Proportion var. explained	38.04%	43.48%
Correlation between factors	0.708	0.794

Table 2: Summary of factor analysis results.

For each benchmark, we calculate the proportion of word/math questions whose factor of highest loading is either factor F1 or factor F2 and also the average accuracy for the questions that are assigned to each (Table 3). Notably, in ENEM, there is a strong connection between factors and question types. While 85% of math questions align more closely with Factor F1, 80% of word questions are more strongly linked to Factor F2. Furthermore, the average accuracy on questions falling into F2 is roughly half of those linked to F1. Those numbers, in accordance to results in Section 4, suggest that language models possess distinct capabilities. We label F1 as the **equation resolution** skill, and F2 as the **text interpretation** skill. On GSM8K, all math-form questions load onto the first factor, whereas easier and harder word problems are split between the two factors. These results suggest that, in GSM8K, translating some word problems into their mathematical form is straightforward, while others require greater capacity to extract explicit and implicit quantities from the text. This kind of distinction can aid how practitioners inspect and debug their question-answering applications.

Factor scores. Do specific versions of a family of models possess different problem-solving skills? To answer that question, we resort to factor scores (DiStefano et al., 2019), which are numerical values representing an observation's position on the latent factors identified through factor analysis. These scores are derived from observed variable scores and factor loadings, providing a way to estimate the contribution of each factor to a specific case. In Table 4 we show factor scores for a selection of the model families doing the ENEM exam; the complete table can be consulted at Appendix C. Next, we highlight some nuances and judgments that can be made by examining factor scores, which aggregate accuracy metrics often overlook.

- **Gemini**: The evolution of factor scores for the Gemini family of models demonstrates that Gemini 2.0-Flash has greater generalization capability compared to earlier versions: while Gemini 1.5-Flash required more elaborately crafted prompts to solve pure math problems (F1), Gemini 2.0-Flash generalizes better by maintaining a high capacity for solving problems in their mathematical form, regardless of the prompting strategy. Additionally, there is a clear progression across versions in the ability to solve word problems (F2).

- **Qwen**: First, we see that moving from Qwen-2.5-7B-Instruct to Qwen-2.5-14B-Instruct does not improve the model's capacity for solving problems in their pure math forms (F1); it had already reached a plateau of around 12 in the 7B version. However, we observe a moderate increase in its ability to solve problems that require equation formulation (F2). Factor scores also highlight the impact of the instruct and math-specialized versions of Qwen: not only are they more consistent in solving pure math problems, but they have also developed a higher capacity for solving word problems, despite sharing the same size as Qwen-2.5-7B.

- **Distilled versions of DeepSeek-R1**: Regardless of the teacher model and the prompt, distilled DeepSeek-R1 models still need to improve in solving word problems, although they are already capable of solving pure math problems at the college-admission level.

Explaining factors. Recent work has looked at the characteristics that make word problems harder for language models (Srivatsa and Kochmar, 2024). We can extend this kind of analysis by investigating if there are characteristics that link a question to either F1 or F2. Table 5 shows how, on average, the features of the underlying equation associated with each problem vary according to the dominant factor and the question form, in the ENEM benchmark.

dominant factor	ENEM		GSM8K	
	F1 (equation resolution)	F2 (text interpretation)	F1	F2
math	0.85 (avg. acc. 0.56)	0.15 (0.25)	1.00 (0.76)	0.00
word	0.20 (0.40)	0.80 (0.21)	0.55 (0.70)	0.45 (0.39)

Table 3: Prevalence of question forms across dominant factors in each dataset. Average accuracy is reported. In ENEM, 85% of math questions load more heavily on F1, while 80% of word questions load more heavily on F2. In GSM8K, all math questions load more heavily on F1.

Notably, in word forms, additional tokens, terms, operators, and equation depth have a greater impact on the overall difficulty of the question. For example, while questions in their mathematical form are linked to F2 when they contain, on average, 13.67 terms, in word form, just 9.13 terms are sufficient.

6 Conclusions

Analytical problems require multiple capabilities, including understanding the problem statement and extracting and solving a mathematical model. To disentangle natural language understanding from pure mathematical reasoning, we used the symbolic solver *SymPy* to introduce extended versions of benchmarks (Brazilian ENEM and GSM8K) that present math word problems in both word-based forms and mathematical components only.

Through structured problem reformulation and factor analysis of 35 large language models, we found that while pure mathematical problem-solving skills are relatively common, only a subset of small-scale language models can effectively handle word problems requiring both interpretation and mathematical modeling. Our work contributes to the growing body of research focused on developing benchmarks that test specific model capabilities by systematically varying key features (Burnell et al., 2023b), and, as models become more multimodal and diverse in their reasoning capabilities, factor analysis can play a larger role as part of language model evaluation frameworks.

7 Reproducibility

The source code, the questions in their word and math forms, the prompts and the full output of factor analysis are available at http://www.dcc.ufmg.br/~pcalais/papers/facl-2025-data. The extended benchmarks are available for use in the same research contexts as their original versions.

8 Limitations

While our study provides evidence supporting a bidimensional structure of reasoning in small-scale large language models, some limitations should be noted.

First, our dataset construction relies on a symbolic solver to generate problem formulations that isolate mathematical reasoning from textual interpretation. Although this approach ensures a controlled separation of these components, complete disentanglement is difficult, as creating the underlying equation that represents the problem still requires some mathematical modeling (e.g., recognizing that the formula for the area of a circle applies in a given scenario).

Second, our analysis is limited to the GSM8K and Brazilian ENEM benchmarks, which contain grade-school and high-school level questions, respectively. To further explore the problem space, future work should consider advanced college-level mathematics, covering topics such as differential and integral calculus (Fan et al., 2024).

9 Ethical Considerations

In the context of solving math word problems, our findings may have broader implications. Large language models could be misused for purposes such as automating academic dishonesty or generating misleading solutions in academic or professional settings. There is also a risk that, due to the LLMs' difficulties with interpreting text, they may produce erroneous or misleading answers that could be misused, especially in critical areas like education or finance.

Acknowledgements

This work is partially supported by the Brazilian agencies CNPq, CAPES, and Fapemig, by the projects CIIA-Saude and IAIA-INCT on AI, and by NSF grants CNS-2312711 and CNS-2319369.

Table 4: Factor scores for top performing families of models on the ENEM dataset. Values greater than the median are highlighted.

model	prompt	F1	F2	acc.
gemini-2.0-flash-exp	0-shot	11.3	12.76	76
	0-shot-cot	11.24	12.26	74
	2-shot	11.3	12.76	76
	2-shot-cot	11.3	12.76	76
gemini-1.5-flash	0-shot	4.18	7.05	37
	0-shot-cot	6.69	8.33	52
	2-shot	9.05	9.8	53
	2-shot-cot	10.9	8.45	62
gemini-1.5-flash-8b	0-shot	3.24	0.41	19
	0-shot-cot	12.14	5.37	58
	2-shot	3.8	0.95	19
	2-shot-cot	11.52	5.38	55
gemini-1.0-pro	0-shot	4.96	1.18	21
	0-shot-cot	7.31	2.17	33
	2-shot	5.43	0.48	22
	2-shot-cot	8.06	1.65	31
Qwen2.5-14B-Instruct	0-shot	9.96	8.2	57
	0-shot-cot	11.78	9.58	67
	2-shot	12.12	8.82	64
	2-shot-cot	11.68	8.19	62
Qwen2.5-7B-Instruct	0-shot	12.46	7.54	63
	0-shot-cot	12.21	6.65	56
	2-shot	12.02	5.4	56
	2-shot-cot	12.58	5.42	58
Qwen2.5-Math-PRM-7B	0-shot	9.65	7.06	51
	0-shot-cot	8.99	9.07	53
	2-shot	10.55	6.9	59
	2-shot-cot	10.19	6.92	57
Qwen2.5-Math-7B-PRM800K	0-shot	9.95	5.24	48
	0-shot-cot	10.11	8.08	56
	2-shot	10.1	7.04	55
	2-shot-cot	10.12	7.7	58
Qwen2.5-7B	0-shot	2.17	0.64	13
	0-shot-cot	7.86	0.97	28
	2-shot	9.9	2.99	45
	2-shot-cot	10.18	2.62	44
Qwen2.5-Math-1.5B-Instruct	0-shot	1.96	0.74	13
	0-shot-cot	8.54	0.58	28
	2-shot	8.13	0.69	35
	2-shot-cot	7.1	0.05	32
Qwen2.5-3B-Instruct	0-shot	0.62	0.94	7
	0-shot-cot	2.32	1.06	14
	2-shot	9.89	2.91	45
	2-shot-cot	10.72	1.94	42
DeepSeek-R1-Distill-Qwen-14B	0-shot	9.05	3.28	49
	0-shot-cot	8.45	2.66	36
	2-shot	12.39	6.14	57
	2-shot-cot	11.17	5.21	52
DeepSeek-R1-Distill-Qwen-7B	0-shot	7.56	0.54	28
	0-shot-cot	8.34	2.4	34
	2-shot	10.85	5.83	52
	2-shot-cot	10.91	3.37	44
deepseek-math-7b-instruct	0-shot	8.49	0.83	33
	0-shot-cot	8.11	0.65	32
	2-shot	7.93	0.38	32
	2-shot-cot	7.93	0.38	31
DeepSeek-R1-Distill-Qwen-1.5B	0-shot	8.4	0.55	29
	0-shot-cot	6.58	1.64	28
	2-shot	9.74	1.23	31
	2-shot-cot	7.14	0.35	23

	Word		Math	
	F1	F2	F1	F2
# of Tokens	82.25	130.00	15.82	31.0
# of Terms	4.75	9.13	7.29	13.67
# of Unique Operators	2.50	3.13	2.82	4.00
Syntatic Tree Depth	3.25	4.62	4.18	5.33

Table 5: Average values for features of the underlying equations according to dominant factor of the questions in the ENEM dataset.

References

Afra Akyürek, Eric Pan, Garry Kuwanto, and Derry Wijaya. 2023. DUnE: Dataset for unified editing. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1847–1861, Singapore. Association for Computational Linguistics.

Thales Sales Almeida, Thiago Laitz, Giovana K Bonás, and Rodrigo Nogueira. 2023. Bluex: A benchmark based on brazilian leading universities entrance exams. In *Brazilian Conference on Intelligent Systems*, pages 337–347. Springer.

Ryan Burnell, Han Hao, Andrew R. A. Conway, and Jose Hernandez Orallo. 2023a. Revealing the structure of language model capabilities. *Preprint*, arXiv:2306.10062.

Ryan Burnell, Wout Schellaert, John Burden, Tomer D. Ullman, Fernando Martinez-Plumed, Joshua B. Tenenbaum, Danaja Rutar, Lucy G. Cheke, Jascha Sohl-Dickstein, Melanie Mitchell, Douwe Kiela, Murray Shanahan, Ellen M. Voorhees, Anthony G. Cohn, Joel Z. Leibo, and Jose Hernandez-Orallo. 2023b. Rethink reporting of evaluation results in ai. *Science*, 380(6641):136–138.

Bryan Christ, Jonathan Kropko, and Thomas Hartvigsen. 2024. Mathwell: Generating educational math word problems using teacher annotations. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11914–11938.

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

Lee J Cronbach. 1951. Coefficient alpha and the internal structure of tests. *psychometrika*, 16(3):297–334.

Gabriella Daroczy, Magdalena Wolska, Walt Detmar Meurers, and Hans-Christoph Nuerk. 2015. Word problems: A review of linguistic and numerical factors contributing to their difficulty. *Frontiers in psychology*, 6:348.

Christine DiStefano, Min Zhu, and Diana Mindrila. 2019. Understanding and using factor scores: Considerations for the applied researcher. *Practical assessment, research, and evaluation*, 14(1):20.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, et al. 2024. Faith and fate: Limits of transformers on compositionality. *Advances in Neural Information Processing Systems*, 36.

Jingxuan Fan, Sarah Martinson, Erik Y Wang, Kaylie Hausknecht, Jonah Brenner, Danxian Liu, Nianli Peng, Corey Wang, and Michael P Brenner. 2024. Hardmath: A benchmark dataset for challenging problems in applied mathematics. *arXiv preprint arXiv:2410.09988*.

Pere Joan Ferrando and Urbano Lorenzo-Seva. 2017. Program factor at 10: Origins, development and future directions. *Psicothema*, 29(2):236–240.

W. Holmes Finch. 2013. *Exploratory Factor Analysis*, pages 167–186. SensePublishers, Rotterdam.

Vedant Gaur and Nikunj Saunshi. 2023. Reasoning in large language models through symbolic math word problems. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5889–5903, Toronto, Canada. Association for Computational Linguistics.

Jan-Eric Gustafsson and Lisbeth Åberg-Bengtsson. 2010. Unidimensionality and interpretability of psychological instruments.

Joy He-Yueya, Gabriel Poesia, Rose E Wang, and Noah D Goodman. 2023. Solving math word problems by combining language models with symbolic solvers. *arXiv preprint arXiv:2304.09102*.

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*.

Arian Hosseini, Alessandro Sordoni, Daniel Toyama, Aaron Courville, and Rishabh Agarwal. 2024. Not all llm reasoners are created equal. *Preprint*, arXiv:2410.01748.

Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira. 2025. Microdados do Enem 2025.

Joshua Benjamin Jaffe and Donald Joseph Bolger. 2023. Cognitive processes, linguistic factors, and arithmetic word problem success: a review of behavioral studies. *Educational Psychology Review*, 35(4):105.

Kuei-Chun Kao, Ruochen Wang, and Cho-Jui Hsieh. 2024. Solving for x and beyond: Can large language models solve complex math problems with more-than-two unknowns? *arXiv preprint arXiv:2407.05134*.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. Mawps: A math word problem repository. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies*, pages 1152–1157.

Uri Leron and Orit Hazzan. 2009. Intuitive vs analytical thinking: Four perspectives. *Educational Studies in Mathematics*, 71:263–278.

Bill Yuchen Lin, Ronan Le Bras, Kyle Richardson, Ashish Sabharwal, Radha Poovendran, Peter Clark, and Yejin Choi. 2025. Zebralogic: On the scaling limits of llms for logical reasoning. *arXiv preprint arXiv:2502.01100*.

Bingbin Liu, Sebastien Bubeck, Ronen Eldan, Janardhan Kulkarni, Yuanzhi Li, Anh Nguyen, Rachel Ward, and Yi Zhang. 2023. Tinygsm: achieving> 80% on gsm8k with small language models. *arXiv preprint arXiv:2312.09241*.

Marcelo Sartori Locatelli, Matheus Prado Miranda, Igor Joaquim da Silva Costa, Matheus Torres Prates, Victor Thomé, Mateus Zaparoli Monteiro, Tomas Lacerda, Adriana Pagano, Eduardo Rios Neto, Wagner Meira Jr, et al. 2024. Examining the behavior of llm architectures within the framework of standardized national exams in brazil. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, volume 7, pages 879–890.

Urbano Lorenzo-Seva and Pere J Ferrando. 2006. Factor: A computer program to fit the exploratory factor analysis model. *Behavior research methods*, 38(1):88–91.

Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, Thilina Rathnayake, Sean Vig, Brian E. Granger, Richard P. Muller, Francesco Bonazzi, Harsh Gupta, Shivam Vats, Fredrik Johansson, Fabian Pedregosa, Matthew J. Curry, Andy R. Terrel, Štěpán Roučka, Ashutosh Saboo, Isuru Fernando, Sumith Kulal, Robert Cimrman, and Anthony Scopatz. 2017. Sympy: symbolic computing in python. *PeerJ Computer Science*, 3:e103.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2024. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv preprint arXiv:2410.05229*.

Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. 2024. tinybenchmarks: evaluating llms with fewer examples. *arXiv preprint arXiv:2402.14992*.

Nonmanut Pongsakdi, Anu Kajamies, Koen Veermans, Kalle Lertola, Marja Vauras, and Erno Lehtinen. 2020. What makes mathematical word problem solving challenging? exploring the roles of word problem characteristics, text comprehension, and arithmetic skills. *Zdm*, 52:33–44.

Subhro Roy, Tim Vieira, and Dan Roth. 2015. Reasoning about quantities in natural language. *Transactions of the Association for Computational Linguistics*, 3:1–13.

Ruoqi Shen, Sébastien Bubeck, Ronen Eldan, Yin Tat Lee, Yuanzhi Li, and Yi Zhang. 2023. Positional description matters for transformers arithmetic. *arXiv preprint arXiv:2311.14737*.

Noora Shrestha. 2021. Factor analysis as a tool for survey analysis. *American journal of Applied Mathematics and statistics*, 9(1):4–11.

Igor Cataneo Silveira and Denis Deratani Mauá. 2018. Advances in automatically solving the enem. In *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)*, pages 43–48. IEEE.

Kv Aditya Srivatsa and Ekaterina Kochmar. 2024. What makes math word problems challenging for LLMs? In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1138–1148, Mexico City, Mexico. Association for Computational Linguistics.

Anselm R Strohmaier, Frank Reinhold, Sarah Hofer, Michal Berkowitz, Birgit Vogel-Heuser, and Kristina Reiss. 2022. Different complex word problems require different combinations of cognitive skills. *Educational Studies in Mathematics*, 109(1):89–114.

Anselm Robert Strohmaier. 2020. *When reading meets mathematics: Using eye movements to analyze complex word problem solving*. Ph.D. thesis, Technische Universität München.

Keith S Taber. 2018. The use of cronbach's alpha when developing and reporting research instruments in science education. *Research in science education*, 48:1273–1296.

Richard Tate. 2003. A comparison of selected empirical methods for assessing the structure of responses to test items. *Applied Psychological Measurement*, 27(3):159–203.

Mohsen Tavakol and Reg Dennick. 2011. Making sense of cronbach's alpha. *International journal of medical education*, 2:53.

Marieke E Timmerman and Urbano Lorenzo-Seva. 2011. Dimensionality assessment of ordered polytomous items with parallel analysis. *Psychological methods*, 16(2):209.

Jóakim v. Kistowski, Jeremy A Arnold, Karl Huppler, Klaus-Dieter Lange, John L Henning, and Paul Cao. 2015. How to build a benchmark. In *Proceedings of the 6th ACM/SPEC international conference on performance engineering*, pages 333–336.

Lieven Verschaffel, Stanislaw Schukajlow, Jon Star, and Wim Van Dooren. 2020. Word problems in mathematics education: A survey. *Zdm*, 52:1–16.

T Vessonen, M Dahlberg, H Hellstrand, A Widlund, J Korhonen, P Aunio, and A Laine. 2024. Task characteristics associated with mathematical word problem-solving performance among elementary school-aged children: a systematic review and meta-analysis. *Educational Psychology Review*, 36(4):117.

Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. 2024. Evaluating mathematical reasoning beyond accuracy. *arXiv preprint arXiv:2404.05692*.

Roy Xie, Chengxuan Huang, Junlin Wang, and Bhuwan Dhingra. 2024. Adversarial math word problem generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 5075–5093.

Yibo Yan, Shen Wang, Jiahao Huo, Hang Li, Boyan Li, Jiamin Su, Xiong Gao, Yi-Fan Zhang, Tianlong Xu, Zhendong Chu, et al. 2024. Errorradar: Benchmarking complex mathematical reasoning of multimodal large language models via error detection. *arXiv preprint arXiv:2410.04509*.

Yidan Zhang, Mingfeng Xue, Dayiheng Liu, and Zhenan He. 2024a. Rationales for answers to simple math word problems confuse large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8853–8869, Bangkok, Thailand. Association for Computational Linguistics.

Ziyin Zhang, Lizhen Xu, Zhaokun Jiang, Hongkun Hao, and Rui Wang. 2024b. Multiple-choice questions are efficient and robust llm evaluators. *arXiv preprint arXiv:2405.11966*.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2024. Processbench: Identifying process errors in mathematical reasoning. *arXiv preprint arXiv:2412.06559*.

Xinyu Zhu, Junjie Wang, Lin Zhang, Yuxiang Zhang, Ruyi Gan, Jiaxing Zhang, and Yujiu Yang. 2022. Solving math word problems via cooperative reasoning induced language models. *arXiv preprint arXiv:2210.16257*.

A Prompts

In this section we show the prompts we have used to produce the results of the paper. Prompts used to create the math forms of the word questions were submitted against GPT-4.

A.1 Generating math forms of word problems

We leverage *SymPy* (Meurer et al., 2017) and two prompts to convert a word problem into its equivalent math form. First, we submit a prompt that generates the *SymPy* source code that solves the problem, as in the example:

SymPy code generation

Generate SymPy code that solves the following math word problem:

In the year a company launched its new smartphone model in the Brazilian market, it invested 45 million reais in the first semester in each of the five regions of the country, offering 30,000 devices for sale in each region. In the first semester, all the devices offered were sold, generating a total profit of 30 million reais. In the second semester, the company decided to make the same investment and offer the same quantities of devices per region. Due to the observed demand, the company assumed that all the devices of this model offered would be sold, and in addition, it planned to achieve a total profit 10% higher in the second semester than in the first. For the company to achieve the planned profit, what should the selling price, in reais, of a smartphone of this model be in the second semester of that year?

Output

```
import sympy as sp
from sympy import symbols, Eq, solve

# Define symbols
p1, p2 = symbols('p1␣p2')

# Equation for first semester
eq1 = Eq(150000 * p1 - 225000000, 30000000)

# Solve for p1 (price in the first semester)
p1_solution = solve(eq1, p1)[0]

# Equation for second semester
eq2 = Eq(150000 * p2 - 225000000, 33000000)

# Solve for p2 (price in the second semester)
p2_solution = solve(eq2, p2)[0]

p2_solution
```

The generated *SymPy* code is then used as an input for the next prompt:

Math form generation

I will give you a SymPy code and want you to produce a question in the following format: Solve <equation>, where <equation> is the equation contained in the SymPy code. Make sure you replace all known variables by their values first.

Sympy code:

```
import sympy as sp
from sympy import symbols, Eq, solve

# Define symbols
p1, p2 = symbols('p1␣p2')

# Equation for first semester
eq1 = Eq(150000 * p1 - 225000000, 30000000)

# Solve for p1 (price in the first semester)
p1_solution = solve(eq1, p1)[0]

# Equation for second semester
eq2 = Eq(150000 * p2 - 225000000, 33000000)

# Solve for p2 (price in the second semester)
p2_solution = solve(eq2, p2)[0]

p2_solution
```

Output

The question based on the given SymPy code would be:
Solve $150000 \times p_2 - 225000000 = 33000000$, where p_2 is the price in the second semester.

We manually validate that both the source code and the math form version of the word problem produces the correct answer, and make small adjustments to strip language details that are unnecessary to the question, like the explanation of the semantics of p_2 in the example, which is not required to solve the problem.

A.2 Question-solving prompts

We used the same prompt style as (Hosseini et al., 2024). Here as examples of 0-shot, 2-shot, 0-shot-CoT and 2-shot-CoT prompts. Note that the chain-of-thought prompts explicitly ask the model to think step by step.

0-shot prompt

Answer the following question. When you respond, respond only with the Solution of the final Problem. At the end of the Solution, when you give your final answer, write it in the form 'The final answer is ANSWER'.
Q: <question>
A:

2-shot prompt

I am going to give you a series of demonstrations of math Problems and Solutions.

When you respond, respond only with the Solution of the final Problem. At the end of the Solution, when you give your final answer, write it in the form ’The final answer is ANSWER’.

Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today?
A: There are 15 trees originally. Then there were 21 trees after some more were planted. So there must have been 21 - 15 = 6. The final answer is 6.

Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?
A: There are originally 3 cars. 2 more cars arrive. 3 + 2 = 5. The final answer is 5.

Q: <question>
A:

0-shot CoT prompt

Answer the following question. When you respond, respond only with the Solution of the final Problem. When you respond, respond only with the Solution of the final Problem, thinking step by step. At the end of the Solution, when you give your final answer, write it in the form ’The final answer is ANSWER’.
Q: <question>
A:

2-shot CoT prompt

I am going to give you a series of demonstrations of math Problems and Solutions. When you respond, respond only with the Solution of the final Problem, thinking step by step. At the end of the Solution, when you give your final answer, write it in the form ’The final answer is ANSWER’.

Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today?
A: There are 15 trees originally. Then there were 21 trees after some more were planted. So there must have been 21 - 15 = 6. The final answer is 6.

Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?
A: There are originally 3 cars. 2 more cars arrive. 3 + 2 = 5. The final answer is 5.

Q: <question>
A:

B Question examples

We show five examples of questions from ENEM and GSM8K in their word (original) and math versions. To give a sense of the varying levels of difficulty in the benchmarks, we sorted the questions according to their accuracies in the word form and show the questions from each of the five quintiles, starting with the easiest question. We also show the factor loadings associated to each question; notice that typically the word version of the question has a higher loading on F2, interpreted as the textual interpretation skill.

B.1 ENEM

word form: The goal is to buy lenses for glasses. The lenses should have thicknesses as close as possible to 3 mm. In a store’s stock, there are lenses with the following thicknesses: 3.10 mm, 3.021 mm, 2.96 mm, 2.099 mm, and 3.07 mm. If the lenses are purchased from this store, the chosen thickness, in millimeters, will be:

---

math form: Determine the closest value, in absolute terms, to 3: 3.10, 3.021, 2.96, 2.099, 3.07.

---

answer: 3.021

---

factor loadings (word): F1 0.333, F2 0.304
factor loadings (math): F1 0.292, F2 0.293

word form: In a school, the probability that a student understands and speaks English is 30%. Three students from this school, who are in the final phase of an exchange selection process, are waiting in a room to be called for an interview. However, instead of calling them one by one, the interviewer enters the room and asks an oral question in English, which can be answered by any of the students. The probability that the interviewer will be understood and have his question answered orally in English is

math form: Calculate $1 - (1 - 30/100)^3$.

answer: 0.657
factor loadings (word): F1 0.258, F2 0.372
factor loadings (math): F1 0.802, F2 -0.220

word form: The subway system in a municipality offers two types of tickets, differentiated by their colors: blue and red. These tickets are sold in booklets, each containing nine tickets of the same color and with the same unit price. Two booklets of blue tickets and one booklet of red tickets are sold for R$ 32.40. It is known that the price of one blue ticket minus the price of one red ticket is equal to the price of one red ticket plus five cents. What is the price, in reais, of a booklet of red tickets?

math form: Calculate $9 \times y : 2 \times 9 \times x + 9 \times y = 32.40$ and $x - y = y + 0.05$.

answer: 6.30

factor loadings (word): F1 -0.109, F2 0.797
factor loadings (math): F1 0.314, F2 0.441

word form: A building has floor numbers starting from the ground floor (T), continuing with the first, second, third, and so on, up to the top floor. A child entered the elevator and, touching the panel, followed a sequence of floors, stopping, opening, and closing the door at various floors. From where the child entered, the elevator went up seven floors, then down ten, down another thirteen, up nine, down four, and stopped at the fifth floor, completing the sequence. Consider that, in the path followed by the child, the elevator stopped once on the top floor of the building. Based on the given information, the top floor of the building is the

math form: Solve for T the equation $T + 7 - 10 - 13 + 9 - 4 = 5$ and then calculate $T + 7$.

answer: 23

factor loadings (word): F1 -0.272, F2 0.833
factor loadings (math): F1 0.155, F2 0.522

word form: According to data collected in the 2010 Census, for a population of 101.8 million Brazilians aged 10 or older who had some form of income in 2010, the average monthly income was R$ 1,202.00. The total monthly income of the poorest 10% accounted for only 1.1% of the total income of this population, while the total monthly income of the richest 10% accounted for 44.5% of the total. What was the difference, in reais, between the average monthly income of a Brazilian in the richest 10% and a Brazilian in the poorest 10%?

math form: Calculate $(101.8 \times 10^6 \times 1202 \times 0.445)/(101.8 \times 10^6 \times 0.1) - (101.8 \times 10^6 \times 1202 \times 0.011)/(101.8 \times 10^6 \times 0.1)$.

answer: 5216.68

factor loadings (word): F1 -0.339, F2 0.806
factor loadings (math): F1 0.015, F2 0.350

B.2 GSM8K

word form: A train travels 360 miles in 3 hours. At the same rate, how many additional hours would it take to travel an additional 240 miles?

math form: Solve $360 = 3 \times \text{rate}$, $240 = \text{rate} \times \text{time}$ for time, where rate $= 120$.

answer: 2

factor loadings (word): F1 0.877, F2 -0.333
factor loadings (math): F1 0.373, F2 0.112

word form: Darry is a roofer and has to climb ladders multiple times a day. He climbs his full ladder, which has 11 steps, 10 times today. He also climbs his smaller ladder, which has 6 steps, 7 times today. He did not climb any steps otherwise. In total, how many times has Darry climbed a step today?

math form: Solve $11 \times 10 + 6 \times 7$.

answer: 152

factor loadings (word): F1 0.667, F2 0.114
factor loadings (math): F1 0.417, F2 0.047

word form: Shannon bought 5 pints of frozen yogurt, two packs of chewing gum, and five trays of jumbo shrimp from The Food Place for a total of $55 If the price of a tray of jumbo shrimp is $5 and a pack of chewing gum costs half as much as a pint of frozen yogurt, what is the price of a pint of frozen yogurt?

math form: Solve $5 * \mathit{frozen\_yogurt\_price} + 2 \times (\mathit{frozen\_yogurt\_price}/2) + 5 \times 5 - 55 = 0$

answer: 5

factor loadings (word): F1 0.004, F2 0.732
factor loadings (math): F1 0.420, F2 0.356

word form: Bill gets paid $20 every hour he works up to a total of 40 hours, after which he gets paid double that amount per hour. How much does Bill get paid for a 50-hour workweek?

math form: Solve $20 \times 40 + (2 \times 20) \times (50 - 40)$.

answer: 1,200

factor loadings (word): F1 -0.080, F2 0.797
factor loadings (math): F1 0.727, F2 -0.231

word form: Steven is preparing a shipment of boxes to deliver to a customer for his chemical supply business. The products are very delicate and must be carefully packed, so partially filled boxes can't be shipped. Steven has three trucks that can be sent on each delivery. Each truck can carry a load of no more than 2,000 pounds of cargo. Some of the boxes weigh 10 pounds after being packed, and some of the boxes weigh 40 pounds when packed. Steven's customer has ordered equal quantities of both the lighter and heavier products. How many boxes of products can Steven ship to his customer in each delivery?

math form: Solve $10 \times x + 40 \times y = 6000$, $x = y$ and calculate $x + y$.

answer: 240

factor loadings (word): F1 -0.034, F2 0.525
factor loadings (math): F1 0.888, F2 -0.184

C Factor Scores

Models were run with default temperature in Google Colab Pro cloud computing service.

Table 6: Factor scores for top performing families of models in the ENEM dataset. Values greater than the median are highlighted.

model	prompt	F1	F2	acc.
gemini-2.0-flash-exp	0-shot	11.3	12.76	76
	0-shot-cot	11.24	12.26	74
	2-shot	11.3	12.76	76
	2-shot-cot	11.3	12.76	76
gemini-1.5-flash	0-shot	4.18	7.05	37
	0-shot-cot	6.69	8.33	52
	2-shot	9.05	9.8	53
	2-shot-cot	10.9	8.45	62
gemini-1.5-flash-8b	0-shot	3.24	0.41	19
	0-shot-cot	12.14	5.37	58
	2-shot	3.8	0.95	19
	2-shot-cot	11.52	5.38	55
gemini-1.0-pro	0-shot	4.96	1.18	21
	0-shot-cot	7.31	2.17	33
	2-shot	5.43	0.48	22
	2-shot-cot	8.06	1.65	31
Qwen2.5-14B-Instruct	0-shot	9.96	8.2	57
	0-shot-cot	11.78	9.58	67
	2-shot	12.12	8.82	64
	2-shot-cot	11.68	8.19	62
Qwen2.5-7B-Instruct	0-shot	12.46	7.54	63
	0-shot-cot	12.21	6.65	56
	2-shot	12.02	5.4	56
	2-shot-cot	12.58	5.42	58
Qwen2.5-Math-PRM-7B	0-shot	9.65	7.06	51
	0-shot-cot	8.99	9.07	53
	2-shot	10.55	6.9	59
	2-shot-cot	10.19	6.92	57
Qwen2.5-Math-7B-PRM800K	0-shot	9.95	5.24	48
	0-shot-cot	10.11	8.08	56
	2-shot	10.1	7.04	55
	2-shot-cot	10.12	7.7	58
Qwen2.5-7B	0-shot	2.17	0.64	13
	0-shot-cot	7.86	0.97	28
	2-shot	9.9	2.99	45
	2-shot-cot	10.18	2.62	44
Qwen2.5-Math-1.5B-Instruct	0-shot	1.96	0.74	13
	0-shot-cot	8.54	0.58	28
	2-shot	8.13	0.69	35
	2-shot-cot	7.1	0.05	32
Qwen2.5-3B-Instruct	0-shot	0.62	0.94	7
	0-shot-cot	2.32	1.06	14
	2-shot	9.89	2.91	45
	2-shot-cot	10.72	1.94	42
DeepSeek-R1-Distill-Qwen-14B	0-shot	9.05	3.28	49
	0-shot-cot	8.45	2.66	36
	2-shot	12.39	6.14	57
	2-shot-cot	11.17	5.21	52
DeepSeek-R1-Distill-Qwen-7B	0-shot	7.56	0.54	28
	0-shot-cot	8.34	2.4	34
	2-shot	10.85	5.83	52
	2-shot-cot	10.91	3.37	44
deepseek-math-7b-instruct	0-shot	8.49	0.83	33
	0-shot-cot	8.11	0.65	32
	2-shot	7.93	0.38	32
	2-shot-cot	7.93	0.38	31
DeepSeek-R1-Distill-Qwen-1.5B	0-shot	8.4	0.55	29
	0-shot-cot	6.58	1.64	28
	2-shot	9.74	1.23	31
	2-shot-cot	7.14	0.35	23
DeepSeek-R1-Distill-Llama-8B	0-shot	4.4	0.76	19

Table 7: Factor scores for bottom performing families of models in the ENEM dataset. Values greater than the median are highlighted.

model	prompt	F1	F2	acc.
DeepSeek-R1-Distill-Llama-8B	0-shot-cot	7.87	1.91	32
NuminaMath-7B-CoT	0-shot	8.06	2.68	31
	0-shot-cot	7.43	1.89	29
	2-shot	9.32	2.29	39
	2-shot-cot	9.33	2.2	39
NuminaMath-7B-TIR	0-shot	1.94	0.19	13
	0-shot-cot	5.79	-0.28	21
	2-shot	8.25	2.15	33
Falcon3-7B-Base	2-shot	6.5	0.74	31
Falcon3-7B-Instruct	0-shot	1.41	1.14	12
	0-shot-cot	1.7	0.74	14
	2-shot	9.59	2.78	43
	2-shot-cot	12.17	4.59	50
Falcon3-3B-Instruct	0-shot	0.44	0.43	9
	0-shot-cot	4.65	0.94	26
	2-shot	7.16	0.94	34
	2-shot-cot	8.83	1.19	38
Phi-3-mini-4k-instruct	0-shot	7.95	0.86	36
	0-shot-cot	9.59	1.48	43
	2-shot	10.39	3.96	42
	2-shot-cot	7.79	2.77	38
phi-4	0-shot	1.01	0.36	8
	0-shot-cot	0.7	0.17	7
	2-shot	2.79	4.5	28
	2-shot-cot	8.58	6.25	49
phi-2	0-shot	-0.28	0.61	1
	0-shot-cot	1.52	-0.17	2
	2-shot	2.77	-0.11	14
	2-shot-cot	3.52	-0.25	15
gemma-2-9b-it	0-shot	9.5	1.75	34
	0-shot-cot	11.42	5.09	47
	2-shot	8.47	3.95	41
	2-shot-cot	11.05	3.18	45
gemma-2-2b-it	0-shot	3.95	0.18	16
gemma-2-9b	0-shot	1.51	0.05	9
	0-shot-cot	2.2	0.01	10
gemma-2-2b	0-shot	0.32	-0.01	4
	0-shot-cot	0.32	-0.01	2
	2-shot	2.63	-0.35	9
	2-shot-cot	2.68	-0.21	10
Mistral-Nemo-Instruct-2407	0-shot	6.33	0.56	30
	0-shot-cot	6.61	1.13	28
	2-shot	8.45	0.76	35
	2-shot-cot	7.37	2.72	36
Mathstral-7b-v0.1	0-shot	4.44	2.33	29
	0-shot-cot	4.53	2.56	27
	2-shot	5.74	1.1	26
	2-shot-cot	6.04	1.44	28
Mistral-7B-Instruct-v0.2	0-shot	1.31	0.34	5
	0-shot-cot	1.1	0.53	8
	2-shot	1.54	0.24	10
	2-shot-cot	3.03	0.8	15
Mistral-7B-v0.3	0-shot	0.0	0.0	5
	0-shot-cot	1.62	-0.39	6
	2-shot	2.34	-0.25	14
Mistral-7B-Instruct-v0.1	0-shot	0.0	0.0	3
	0-shot-cot	0.99	0.31	8
	2-shot	0.32	-0.01	7
	2-shot-cot	0.62	0.29	7
Meta-Llama-3-8B	0-shot	0.32	-0.01	4
	0-shot-cot	1.07	-0.15	4
	2-shot	2.76	0.12	9

Table 8: Factor scores for top performing families of models in the GSM8K dataset. Values greater than the median are highlighted.

model	prompt	F1	F2	acc.
Qwen2.5-Math-7B-PRM800K	0-shot	15.14	8.09	77
	0-shot-cot	14.83	8.33	72
	2-shot	15.14	8.21	76
	2-shot-cot	15.71	8.08	77
Qwen2.5-Math-7B-Instruct	0-shot	14.26	8.34	76
	0-shot-cot	14.62	7.77	74
	2-shot	15.71	8.08	75
	2-shot-cot	14.8	8.0	73
Qwen2.5-Math-PRM-7B	0-shot	14.01	8.04	73
	0-shot-cot	14.38	8.15	72
	2-shot	14.83	8.41	75
	2-shot-cot	15.71	8.08	74
Qwen2.5-14B-Instruct	0-shot	15.05	6.55	71
	0-shot-cot	14.87	7.48	73
	2-shot	16.27	5.94	73
	2-shot-cot	15.71	8.08	76
Qwen2.5-14B	0-shot	15.69	4.82	69
	0-shot-cot	16.73	3.88	70
Qwen2.5-Math-1.5B-Instruct	0-shot	13.34	1.49	40
	0-shot-cot	13.15	4.94	51
	2-shot	14.41	6.17	69
	2-shot-cot	14.84	6.4	71
Qwen2.5-3B-Instruct	0-shot	8.44	2.2	26
	0-shot-cot	9.44	1.9	26
	2-shot	14.4	7.25	68
	2-shot-cot	14.71	7.61	68
Qwen2.5-7B	0-shot	9.12	6.45	38
	0-shot-cot	15.6	1.99	56
gemini-2.0-flash-exp	0-shot	15.67	8.38	79
	0-shot-cot	17.08	5.23	76
gemini-1.5-flash-8b	0-shot	13.06	1.89	38
	0-shot-cot	16.34	6.78	73
	2-shot	12.51	3.7	44
	2-shot-cot	15.88	6.59	75
DeepSeek-R1-Distill-Qwen-14B	0-shot	15.98	3.21	65
	0-shot-cot	16.27	4.34	67
	2-shot	16.09	4.88	70
	2-shot-cot	16.4	3.92	68
DeepSeek-R1-Distill-Qwen-7B	0-shot	13.8	3.43	59
	0-shot-cot	14.24	5.94	66
	2-shot	16.03	6.54	74
deepseek-math-7b-instruct	0-shot	14.82	2.63	61
	0-shot-cot	14.82	2.63	63
	2-shot	13.06	7.6	66
	2-shot-cot	14.06	7.25	67
DeepSeek-R1-Distill-Qwen-1.5B	0-shot	13.11	5.25	63
	0-shot-cot	11.25	7.1	59
	2-shot	7.81	5.08	43
	2-shot-cot	11.97	4.48	50
NuminaMath-7B-CoT	0-shot	15.36	4.13	66
	0-shot-cot	14.41	3.13	61
	2-shot	15.97	1.78	60
	2-shot-cot	16.9	1.42	62
NuminaMath-7B-TIR	0-shot	10.6	2.8	39
	0-shot-cot	9.86	5.39	42
	2-shot	13.92	1.01	57

Table 9: Factor scores for bottom performing families of models in the GSM8K dataset. Values greater than the median are highlighted.

model	prompt	F1	F2	acc.
NuminaMath-7B-TIR	2-shot-cot	15.98	3.2	64
gemma-2-9b-it	0-shot	17.01	2.26	63
	0-shot-cot	17.18	1.72	61
	2-shot	16.8	2.99	61
	2-shot-cot	17.58	3.39	64
gemma-2-2b-it	0-shot	12.62	-0.43	40
	0-shot-cot	12.61	-0.53	39
	2-shot	10.32	0.39	36
gemma-2-9b	0-shot	7.14	0.17	23
	0-shot-cot	9.16	0.22	30
gemma-2-2b	0-shot	2.97	0.27	11
Falcon3-3B-Instruct	0-shot	6.96	0.04	16
	0-shot-cot	15.71	1.22	50
	2-shot	14.22	4.59	62
	2-shot-cot	13.88	4.93	57
Falcon3-7B-Instruct	0-shot	4.68	0.17	11
	0-shot-cot	7.11	-0.74	19
	2-shot	15.02	5.81	72
	2-shot-cot	13.61	4.99	65
Falcon3-7B-Base	0-shot	6.09	1.29	20
	0-shot-cot	5.26	0.48	16
	2-shot	15.75	4.46	59
Phi-3-mini-4k-instruct	0-shot	13.43	4.77	58
	0-shot-cot	15.9	6.03	68
	2-shot	15.77	6.65	72
	2-shot-cot	15.77	6.65	71
phi-4	0-shot	0.66	-0.23	4
	0-shot-cot	0.66	-0.23	3
	2-shot	15.9	5.32	65
	2-shot-cot	16.36	3.77	64
phi-2	0-shot	1.87	0.11	7
	0-shot-cot	2.68	0.91	9
	2-shot	10.28	-0.28	30
	2-shot-cot	9.25	0.03	29
Mistral-Nemo-Instruct-2407	0-shot	14.05	-0.09	50
	0-shot-cot	15.33	0.96	48
	2-shot	16.57	1.94	57
	2-shot-cot	16.87	1.56	55
Mathstral-7b-v0.1	0-shot	14.58	2.18	49
	0-shot-cot	13.11	0.87	42
	2-shot	16.37	0.78	59
	2-shot-cot	15.92	1.39	58
Mistral-7B-Instruct-v0.2	0-shot	3.54	-0.09	8
	0-shot-cot	4.69	0.26	18
	2-shot	9.06	2.7	29
	2-shot-cot	10.12	0.71	32
Mistral-7B-Instruct-v0.1	0-shot	4.26	0.58	17
	0-shot-cot	6.08	0.68	20
	2-shot	6.31	-0.44	20
	2-shot-cot	6.95	-0.32	22
Meta-Llama-3-8B	0-shot	2.55	-0.1	10
	0-shot-cot	1.47	0.18	9
	2-shot	7.44	1.2	28
	2-shot-cot	8.97	-0.39	24

D GSM8K Plots

In this section we show the plots and tables for the GSM8K benchmark that were omitted in the main text due to space constraints.

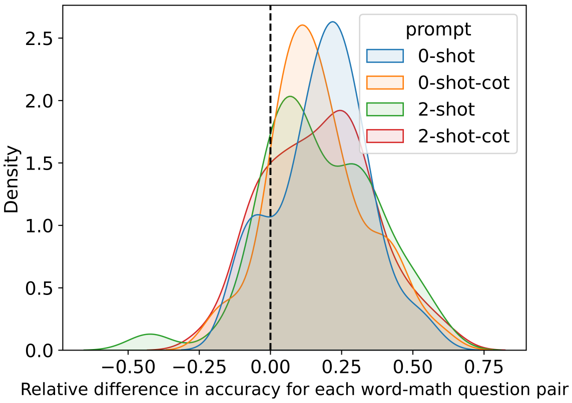

Figure 5: Density plot of the absolute difference in accuracy in word and math forms for GSM8K.

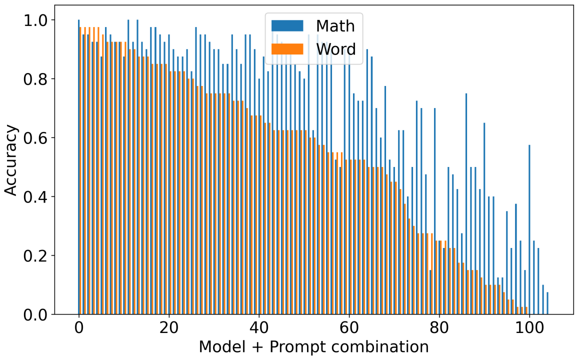

Figure 6: Accuracy by model, comparing performance on word and math questions on the GSM8K exam. Models are sorted by accuracy on the word-based exam. While most models show a sharp decline in word-problem accuracy, some retain strong performance on math problems.