RAZONAMIENTO EN GRANDES MODELOS DE LENGUAJE

Jorge A. Pichardo Cabrera, Franco Hernández Piloto, Hivan Cañizares Diaz, Alejandro García González, Ernesto Bárcena Trujillo, Carlos Mauricio Reyes Escudero

Universidad de la Habana, Facultad de Matemática y Computación

<thinking>
El usuario ha pedido que escriba un informe para un proyecto de la universidad.

Uh!, seguro le ha dado pereza escribirlo por sí mismo; debo usar un estilo formal para artículos académicos y mencionar toda la información relevante sobre los principales esfuerzos llevados a cabo para su desarrollo.

Comenzaré introduciendo los objetivos del proyecto y luego haré una revisión del *state-of-the-art*.

Al final, mencionaré los detalles sobre el dataset creado y los experimentos realizados.
</thinking>

Introducción

Este trabajo aborda el razonamiento en grandes modelos de lenguaje (LLMs) a través de diferentes perspectivas. Primero, se revisan las diversas técnicas existentes para potenciar las capacidades de razonamiento de los LLMs, junto con los datasets empleados para evaluar dichas capacidades en varios dominios.

Como contribución principal, se propone un nuevo dataset diseñado para evaluar el razonamiento en LLMs. Se presentan los fundamentos y características de este, así como experimentos para medir el desempeño de diversos LLMs en este nuevo entorno.

Finalmente, se prueba el uso de Many-shot con ejemplos obtenidos en pasos anteriores para mejorar las habilidades de razonamiento de los LLMs. La efectividad de esta técnica se evalúa en benchmarks establecidos, y se presentan y discuten los resultados comparativos obtenidos.

Trabajo Relacionado en Técnicas para Mejorar el Razonamiento en LLMs

La investigación actual en el área del razonamiento en grandes modelos de lenguaje (LLMs) ha explorado diversas técnicas para superar las limitaciones inherentes a estos modelos. Un enfoque fundamental es el *Chain-of-Thought* (CoT) prompting [Wei et al., 2023], que guía al modelo a generar una serie de pasos intermedios de razonamiento, de forma similar a como un humano explicaría su proceso de resolución de un problema, mejorando así su desempeño en tareas complejas. A diferencia del CoT estándar, que requiere ejemplos (few-shot learning), el *Zero-Shot CoT* [Kojima et al., 2023] induce este comportamiento sin ejemplos, simplemente añadiendo una instrucción como 'Pensemos paso a paso' al prompt. Otra técnica relacionada, pero con una aplicación diferente, es el uso de *scratchpads* [Nye et al., 2021]. Esta técnica no se centra en el razonamiento lingüístico paso a paso, sino en permitir que el LLM realice cálculos y manipulaciones intermedias de forma explícita, como si tuviera una 'hoja de borrador' para operaciones aritméticas o la ejecución de código, mejorando la precisión en tareas que requieren cómputo.

Variaciones de CoT incluyen el *Complexity-Based Prompting* [Fu et al., 2023], que selecciona ejemplos de razonamiento para el prompt en función de su complejidad, favoreciendo aquellos con más pasos inferenciales. Otros enfoques, como *Self-Ask* [Press et al., 2023], incorporan preguntas de seguimiento generadas por el propio modelo y la integración con motores de búsqueda para obtener información adicional. La auto-verificación [Weng et al., 2023] y la auto-consistencia [Wang et al., 2023] buscan mejorar la fiabilidad del razonamiento mediante la revisión y evaluación interna de las respuestas generadas, ya sea a través de un proceso de verificación 'hacia atrás' o muestreando múltiples rutas de razonamiento y seleccionando la más consistente.

Self-Debugging [Chen et al., 2023a] propone que el modelo trate de analizar el comportamiento del código verbalmente, mientras que técnicas como *Program of Thought* (PoT) [Chen et al., 2023b] y *Program aided Language-models* (PAL) [Gao et al., 2023] exploran la generación y ejecución de código para el razonamiento, delegando la computación a un intérprete externo. Métodos como REFINER [Paul et al., 2024], *Self-Improvement* [Huang et al., 2022], y *Self-Refine* [Madaan et al., 2023] se centran en el refinamiento iterativo de las respuestas mediante retroalimentación, ya sea generada por el propio modelo o por un modelo crítico, en un proceso similar a cómo un humano revisaría y editaría su propio trabajo. Estrategias como *Progressive Hint* [Zheng et al., 2024] y STaR [Zelikman et al., 2022] utilizan indicios y autoaprendizaje para mejorar el razonamiento, donde el modelo aprende de sus propios intentos, utilizando las respuestas correctas (o pistas sobre ellas) como guía.

Least-to-most (L2M) [Zhou et al., 2023] aborda la generalización a problemas más complejos descomponiéndolos en subproblemas más simples y resolviéndolos secuencialmente, utilizando las soluciones de los subproblemas anteriores como contexto para los siguientes. Enfoques más avanzados, como *Tree-of-Thoughts* (ToT) [Yao et al., 2023], *Graph-of-Thoughts* (GoT) [Besta et al., 2024], *Buffer of Thoughts* (BoT) [Yang et al., 2024], y *Chain of Continuous Thought* (COCONUT) [Hao et al., 2024] exploran estructuras de razonamiento más complejas, como árboles/grafos de pensamiento, espacios latentes continuos (para generar la cadena de razonamiento o plantillas de razonamiento como *BoT*), e incluso representaciones visuales como proponen en [Wu et al., 2024], permitiendo una búsqueda más exhaustiva y flexible en el espacio de posibles soluciones. Finalmente, el aprendizaje en contexto con múltiples ejemplos (*Many-shot In-Context Learning*) [Agarwal et al., 2024] ha demostrado ser una técnica poderosa, superando incluso, en algunos casos, al fine-tuning, por lo cual puede ser usado junto a *STaR* para una rápida prueba de la mejora por refuerzo positivo. Finalmente, mencionar el merito que supone el desarrollo de *DeepSeek-R1* [DeepSeek-AI et al., 2025], un modelo de lenguaje (LLM) de código abierto con notables capacidades de razonamiento, ilustrando el impacto de diferentes estrategias de entrenamiento, en particular el aprendizaje por refuerzo (RL) y el ajuste fino supervisado (SFT), usadas en conjunto para potenciar mutuamente las deficiencias que cada una supone.

Trabajo Relacionado en Datasets para Evaluar Razonamiento en LLMs

"When a measure becomes a target, it ceases to be a good measure"

Goodhart's maxim

La evaluación del razonamiento en Grandes Modelos de Lenguaje (LLMs) ha impulsado el desarrollo de una amplia variedad de *benchmarks*, cada uno con diferentes enfoques y niveles de dificultad. Algunos, como *CommonsenseQA* [Talmor et al., 2019], se centran en el razonamiento de sentido común, presentando preguntas que son triviales para los humanos pero desafiantes para los LLMs debido a su falta de conocimiento del mundo real y experiencia encarnada. Otros, como *DROP* [Dua et al., 2019], combinan la comprensión lectora con el razonamiento numérico y textual discreto, exigiendo que los modelos identifiquen y combinen información dispersa en un texto.

Un área importante de enfoque ha sido el razonamiento matemático. *GSM8K* [Cobbe et al., 2021] presenta problemas matemáticos verbales de nivel de primaria, mientras que *MathQA* [Amini et al., 2019] y, especialmente, *MATH* [Hendrycks et al., 2021a] elevan la dificultad a problemas de nivel de secundaria y competición, abarcando álgebra, geometría, probabilidad y cálculo. Estos *benchmarks* no solo evalúan la capacidad de cálculo, sino también la comprensión del lenguaje y la capacidad de seguir una cadena de razonamiento matemático.

El razonamiento lógico deductivo se evalúa a través de datasets como *RuleTaker* [Clark et al., 2020], que requiere la aplicación de reglas lógicas explícitas, y *EntailmentBank* [Dalvi et al., 2021], que exige la generación de cadenas de razonamiento que conecten premisas y conclusiones. Estos *benchmarks* ponen de manifiesto la dificultad de los LLMs para generalizar reglas lógicas y mantener la consistencia en inferencias complejas.

Para abordar una gama más amplia de conocimientos y habilidades, *MMLU* [Hendrycks et al., 2021b] evalúa el conocimiento general y la resolución de problemas en 57 temas diversos, desde matemáticas hasta ética, en configuraciones *zero-shot* y *few-shot*. *MMLU-Pro* [Wang et al., 2024] aumenta la dificultad de MMLU con preguntas más intensivas en razonamiento y más opciones de respuesta. *GPQA* [Rein et al., 2023] lleva la dificultad a un extremo, con preguntas de nivel de posgrado en biología, física y química, diseñadas para ser 'a prueba de Google'.

En el ámbito de la programación, *HumanEval* [Chen et al., 2021] evalúa la capacidad de los LLMs para generar código funcionalmente correcto a partir de descripciones en lenguaje natural, utilizando la métrica *pass@k* (puntuando si en k intentos acierta al menos una vez).

Un enfoque diferente, que busca evaluar la inteligencia general de una manera más similar a los tests psicométricos, es el *Abstraction and Reasoning Corpus* (ARC) [Chollet, 2019]. ARC presenta tareas abstractas que requieren inferir transformaciones a partir de pocos ejemplos, centrándose en la inteligencia fluida y la eficiencia en la adquisición de nuevas habilidades. A diferencia de muchos otros *benchmarks*, ARC controla estrictamente los *priors* de conocimiento, asumiendo solo los *priors* innatos de **Core Knowledge** [Spelke and Kinzler, 2007].

Otro enfoque es el presentado en *Easy Problems That LLMs Get Wrong* [Williams and Huckle, 2024], que se diferencia de ARC en que se centra en las limitaciones de los LLMs. El *benchmark* incluye preguntas sencillas en las que se espera un buen rendimiento de los modelos, pero en las que suelen cometer errores. Un claro ejemplo de esta sobreespecialización se encuentra en una variación del problema de Monty Hall:

> **Pregunta:** Suppose you're on a game show, and you're given the choice of three doors: Behind one door is a gold bar; behind the others, rotten vegetables. You pick a door, say No. 1, and the host asks you "Do you want to pick door No. 2 instead?" Is it to your advantage to switch your choice?
>
> **Respuesta de Claude 3 Opus:** "This is a classic probability problem known as the "Monty Hall problem." The answer is that it is always to your advantage to switch your choice. Here's why: ..."

Como se observa, el LLM responde incorrectamente, aplicando el razonamiento estándar del problema de Monty Hall clásico, a pesar de que la pregunta modificada elimina la acción clave del presentador. Esto demuestra una clara incapacidad para adaptar sus 'conocimientos previos' a una situación con condiciones diferentes, incluso cuando son mínimas. Además de la sobreespecialización, se identifican otros fallos comunes en los LLMs, como la falta de lógica o sentido común, dificultades con el razonamiento espacial, errores en cálculos matemáticos simples y una comprensión lingüística superficial.

En conjunto, estos *benchmarks* revelan que, si bien los LLMs han logrado avances notables, aún existen desafíos significativos en áreas que requieren razonamiento profundo, sentido común, generalización a partir de pocos ejemplos y la aplicación consistente de reglas lógicas y matemáticas. La diversidad de enfoques en la evaluación refleja la complejidad de medir la inteligencia artificial y la necesidad continua de desarrollar *benchmarks* que capturen las múltiples facetas del razonamiento.

Cuadro 1: *Clasificación de los Datasets por Tipo / Capacidades Evaluadas*

Tipo de Razonamiento	**Datasets**
Razonamiento Lógico y Deductivo	RuleTaker, EntailmentBank
Razonamiento Matemático	GSM8K, MathQA, MATH
Comprensión y Razonamiento sobre Texto	DROP
Conocimiento General y Sentido Común	CommonsenseQA, MMLU, MMLU-Pro
Conocimiento Experto Específico	GPQA
Generación de Código	HumanEval
Inteligencia General y Abstracción	ARC
Problemas sencillos que hacen sudar a los LLMs	Linguistic Benchmark

Nuestro Dataset

Para complementar la revisión teórica y proporcionar una base empírica para el análisis del razonamiento en LLMs, hemos desarrollado un nuevo dataset que abarca una variedad de preguntas extraídas de diferentes áreas de la Ciencia de la Computación. Este dataset incluye problemas que requieren la aplicación de lógica, algoritmos, teoría de números, combinatoria y teoría de grafos, entre otros, buscando así evaluar un espectro amplio de capacidades de razonamiento. A continuación, se detallan las categorías específicas incluidas en el dataset, junto con ejemplos representativos de cada una, con el objetivo de ilustrar la naturaleza y el nivel de dificultad de las tareas propuestas.

Lógica

Esta categoría se centra en problemas de razonamiento lógico, diseñados para evaluar la capacidad de deducción, inferencia y pensamiento crítico de los LLMs. Incluye una variedad de desafíos, como acertijos lógicos, problemas del tipo 'truhanes y caballeros' (donde se deben identificar quiénes mienten y quiénes dicen la verdad a partir de un conjunto de afirmaciones), y otros problemas que requieren la aplicación de principios lógicos para llegar a una conclusión correcta. La mayoría de las preguntas en esta sección (20 en total) se presentan en formato de selección múltiple, lo que facilita la evaluación automática de las respuestas y permite una comparación más directa del rendimiento entre diferentes modelos. Sin embargo, aunque el formato sea de selección múltiple, la resolución de los problemas subyacentes exige un razonamiento lógico riguroso, y no simplemente la identificación de patrones superficiales.

Debido al déficit de electricidad se realizan apagones programados. La Habana se divide en 4 bloques según su ubicación. A cada bloque le corresponden diferentes días de la semana en que tendrán afectaciones de electricidad. En la UH se desea saber a cuál bloque pertenece esta entidad y para ello cuentan con los planteamientos realizados por A, B y C que son habitantes de la Isla de los Truhanes y Caballeros:

Planteamiento 1 A: Si B y yo somos caballeros entonces la Universidad no pertenece al bloque 2
Planteamiento 2 B: Si todos somos caballeros entonces la Universidad es del bloque 3
Planteamiento 3 C: Si la Universidad no está ubicada en el bloque 4 entonces yo soy un caballero
Planteamiento 4 A: C es un truhán
Planteamiento 5 B: Entre C y yo hay al menos un truhán
Planteamiento 6 C: Todos somos truhanes

Además se conoce la siguiente información publicada por la Empresa Eléctrica de La Habana:
Planteamiento 7: Si la Universidad no pertenece al bloque 1 entonces le corresponde el bloque 2
Planteamiento 8: Si la Universidad no pertenece al bloque 3 entonces tampoco pertenece al 4

¿Qué es A, B y C y a cuál bloque pertenece la Universidad?

Figura 1: *Ejemplo de pregunta de Lógica*

Matemática Discreta

Esta categoría abarca una variedad de problemas extraídos de la matemática discreta, un campo fundamental de la Ciencia de la Computación. Las preguntas en esta sección están diseñadas para evaluar la capacidad de los LLMs para manejar conceptos y estructuras discretas, aplicar principios de conteo, y razonar sobre propiedades numéricas y combinatorias.

En particular, se incluyen problemas de las siguientes subáreas:

- **Teoría de Números:** Se presentan problemas relacionados con las propiedades de los números enteros, como divisibilidad, números primos, congruencias y ecuaciones diofánticas. La respuesta a estos problemas suele ser un valor entero.
- **Combinatoria:** Esta subárea se enfoca en problemas de conteo diversos. Las preguntas suelen requerir la derivación de una fórmula cerrada que exprese el número de configuraciones posibles que satisfacen ciertas condiciones, y luego la evaluación de esta fórmula para un valor numérico específico, resultando en una respuesta entera.
- **Teoría de Grafos:** Se incluyen problemas que involucran grafos (conjuntos de nodos y aristas). Estos problemas a menudo tienen un componente combinatorio, donde se busca contar el número de caminos, ciclos, árboles, subgrafos u otras estructuras que cumplen ciertas propiedades. También se pueden plantear preguntas sobre operaciones en grafos, como la búsqueda de caminos mínimos o la determinación de la conectividad. La salida de estos problemas suele ser un valor entero.

En total, hay 14 preguntas de Matemática Discreta en el conjunto de datos.

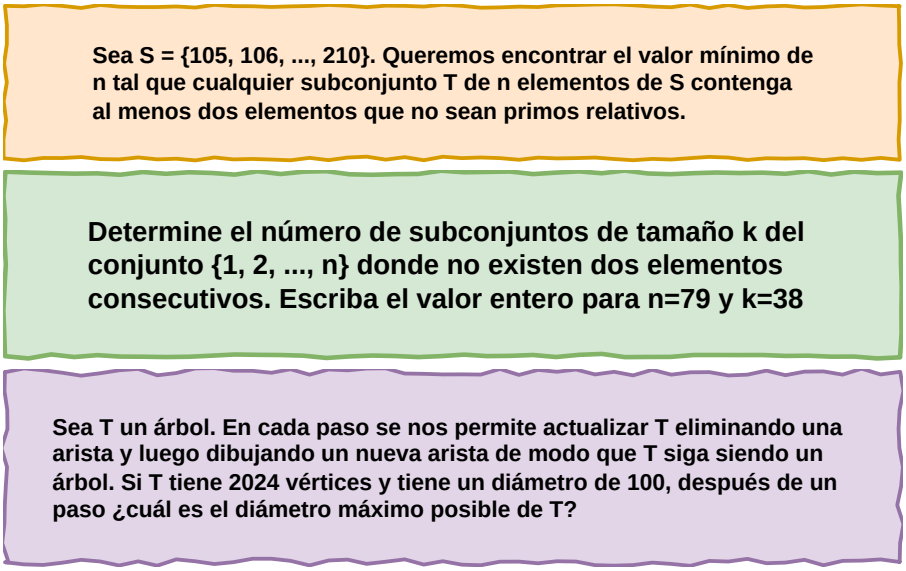

Figura 2: *Ejemplo de preguntas de Matemática Discreta*

Predicción de *Output* de Código

En esta sección, presentamos preguntas que requieren la ejecución mental de código en Python para determinar el output resultante. Las funciones pueden contener operaciones aparentemente aleatorias o con un propósito específico, evaluando la comprensión del modelo sobre la lógica de programación. Mientras que esta habilidad parece estar muy alejada de evaluar razonamiento, mide la consistencia en el seguimiento de la secuencia de pasos, lo cual es vital para crear mejores modelos razonadores. El dataset contiene 33 preguntas de predicción de output de código.

Categoría	Preguntas
Lógica	20
Matemática Discreta	14
Predicción de Código	33

Cuadro 2: *Resumen del número de preguntas por categoría.*

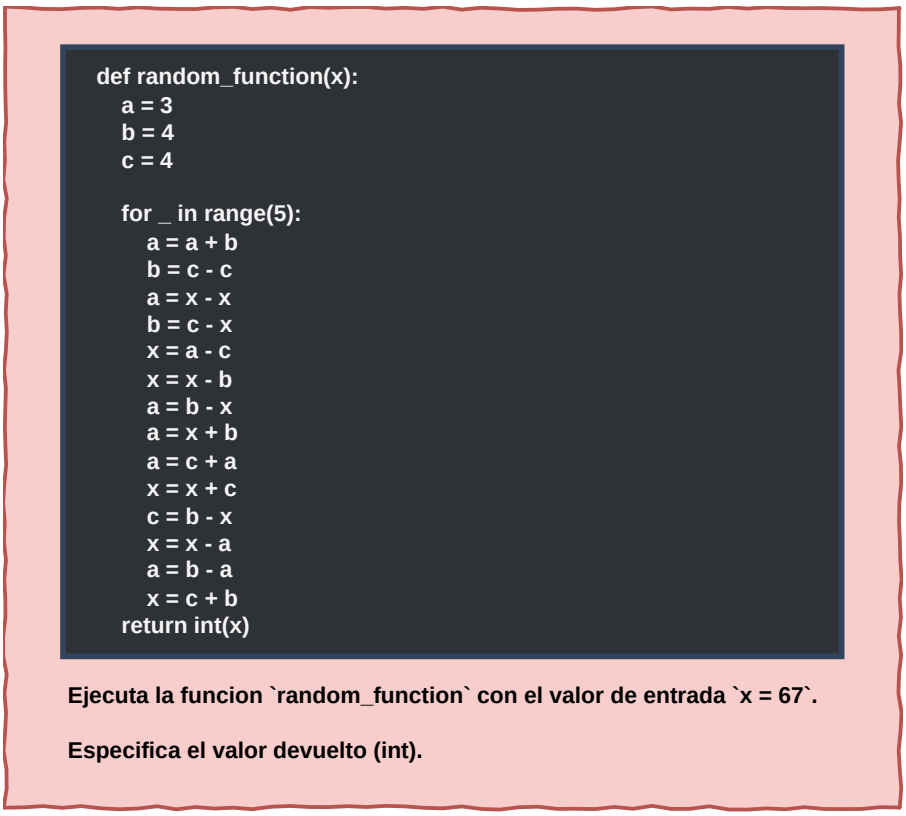

Figura 3: *Ejemplo de pregunta de Predicción de Output de Código*

Nuestros experimentos

Para analizar el valor y la dificultad del dataset propuesto, así como para establecer una línea base de comparación, se realizaron pruebas utilizando varios modelos de lenguaje. Estos modelos fueron evaluados en su configuración básica, y en algunos casos, se aplicaron técnicas de *prompting* como *scratchpad* y *Chain-of-Thought* (CoT) para investigar su impacto en el rendimiento. Los resultados de estas evaluaciones, que incluyen la precisión de los modelos en cada una de las categorías de preguntas (lógica, matemática discreta y predicción de código), se presentan a continuación.

Modelo	**Lógica (%)**	**Discreta (%)**	**Código (%)**
gemini-2.0-flash-exp	40,00	7,14	6,06
gemini-2.0-flash-exp w. scratchpad	65	35,71	12,12
gemini-2.0-flash-exp w. CoT	65,00	42,86	6,06
gemini-2.0-flash-thinking-exp-01-21	85,00	64,28	39,39
mistral-small-latest	30,00	14,29	0,00
mistral-small-latest *pass@3*	70,00	28,57	0,00
llama-v3p1-8b-instruct	35,00	0,00	0,00
llama-v3p1-8b-instruct *pass@3*	60,00	7,14	3,03

Cuadro 3: *Resultados de los Modelos en las Diferentes Categorías*

Para obtener una comprensión detallada del rendimiento de cada modelo en las distintas áreas de razonamiento evaluadas, se procedió a calcular estadísticas agregadas por modelo y categoría. Particularmente calculamos la Varianza Bootstrap Promedio (VBP) y la Deviacion Estándar Bootstrap Promedio (DEBP).

Modelo	VBP	DEBP
gemini-2.0-flash-exp	76,08	8,72
gemini-2.0-flash-thinking-exp-01-21	26,03	5,10
mistral-small-latest	34,00	5,83
llama-v3p1-8b-instruct	33,64	5,80

Cuadro 4: *Varianza Bootstrap Promedio y Deviacion Estándar Bootstrap Promedio en la categoría de Lógica.*

Modelo	VBP	DEBP
gemini-2.0-flash-exp	69,72	8,35
gemini-2.0-flash-thinking-exp-01-21	52,48	7,24
mistral-small-latest	53,01	7,28
llama-v3p1-8b-instruct	12,24	3,50

Cuadro 5: *Varianza Bootstrap Promedio y Deviacion Estándar Bootstrap Promedio en la categoría de Discreta.*

Modelo	VBP	DEBP
gemini-2.0-flash-exp	7,36	2,71
gemini-2.0-flash-thinking-exp-01-21	39,07	6,25
mistral-small-latest	0,0	0,0
llama-v3p1-8b-instruct	5,87	2,42

Cuadro 6: *Varianza Bootstrap Promedio y Deviacion Estándar Bootstrap Promedio en la categoría de Predicción de Código.*

Para finalizar comparamos con el baseline *random* en la categoria de Logica, que directamente podemos estimar que es de 25%. Los resultados aparecen en la siguiente tabla:

Modelo	Z-stat	p-val	Rechazo
gemini-2.0-flash-exp	2.77128	0.00279	1.0
gemini-2.0-flash-thinking-exp-01-21	7.62102	1.25836e-14	1.0
mistral-small-latest	4.84974	618110e-7	1.0
llama-v3p1-8b-instruct	-2.07846	0.98117	0.0

Cuadro 7: *Los resultados nos permiten rechazar la hipotesis nula y mostrar evidencia estadistica de que el modelo es mejor que un modelo aleatorio para todos los modelos menos para llama-v3p1-8b-instruct.*

Experimentos de Refuerzo Positivo

Para investigar el potencial del aprendizaje en contexto con múltiples ejemplos ('many-shot in-context learning'), y aprovechando la amplia ventana de contexto de modelos como Gemini (1M de tokens), se realizaron experimentos de refuerzo positivo utilizando un subconjunto de preguntas del dataset *Mathematics* de DeepMind Saxton et al. [2019]. Este dataset se compone de una variedad de problemas matemáticos de nivel escolar, abarcando álgebra, aritmética, cálculo, comparación, medición, teoría de números, y probabilidad. La diversidad de temas y la naturaleza estructurada de los problemas (pregunta y respuesta) lo hacen adecuado para evaluar la capacidad de razonamiento matemático de los LLMs.

El objetivo principal fue determinar si el rendimiento de un modelo en este tipo de tareas podía mejorarse proporcionándole ejemplos de razonamiento correctos dentro del contexto, sin necesidad de modificar los pesos del modelo (es decir, sin *fine-tuning*). Para ello, se evaluó el modelo *Gemini 2.0 Flash Lite* en tres escenarios diferentes:

1. **Línea Base (Sin Modificación):** El modelo se evaluó directamente en un conjunto de prueba del *dataset*, sin ningún tipo de *prompting* especial ni ejemplos en contexto. Esto proporciona una línea base de rendimiento del modelo 'en crudo'.

2. **Refuerzo Positivo con Ejemplos Correctos (Autogenerados):** Se seleccionó un conjunto de entrenamiento de preguntas del *dataset*. El modelo se utilizó para generar respuestas a estas preguntas de entrenamiento. Luego, para cada pregunta de prueba, se incluyó en el *prompt* un conjunto de k ejemplos de preguntas de entrenamiento con 'respuestas correctas' generadas por el propio modelo en la fase anterior. Es decir, el modelo se 'auto-reforzó' con sus propios ejemplos exitosos.

3. **Refuerzo Positivo con Ejemplos de un Modelo Superior:** Se utilizó el mismo conjunto de entrenamiento del escenario anterior. Sin embargo, en este caso, las respuestas a las preguntas de entrenamiento fueron generadas por un modelo superior, *Gemini 2.0 Flash Thinking*. Para cada pregunta de prueba se utilizaron como ejemplos en contexto k de estas respuestas correctas de un modelo de mayor capacidad, con la hipótesis de que esto podría proporcionar una mejor guía para el modelo.

Estos tres escenarios permiten evaluar no solo el rendimiento base del modelo, sino también el impacto del aprendizaje en contexto con ejemplos autogenerados y con ejemplos de un modelo superior. Se buscaba medir el limite superior del aprendizaje en contexto. Los resultados de estos experimentos, comparando la precisión del modelo en cada escenario, se presentan en la tabla siguiente. Se evaluaron previamente varios modelos para establecer una métrica de referencia.

Modelo	**accu.**
llama-v3p1-8b-instruct	29,00
llama-v3p1-8b-instruct *pass@3*	41,00
mistral-small-latest	65,00
mistral-small-latest *pass@3*	70,00
gemini-2.0-flash-lite-preview-02-05 *(caso 1)*	54,00
gemini-2.0-flash-lite-preview-02-05 *(caso 2)*	58,00
gemini-2.0-flash-lite-preview-02-05 *(caso 3)*	71,00
gemini-2.0-flash-thinking-exp-01-21	79,00

Cuadro 8: *Resultados del experimento de refuerzo*

Todas las pruebas se hicieron sobre un conjunto de *test* de 100 elementos, y los 'entrenamientos', sobre un conjunto *train* separado, de 560 elementos.

Referencias

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners, 2023. URL https://arxiv.org/abs/2205.11916.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. Show your work: Scratchpads for intermediate computation with language models, 2021. URL https://arxiv.org/abs/2112.00114.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. Complexity-based prompting for multi-step reasoning, 2023. URL https://arxiv.org/abs/2210.00720.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models, 2023. URL https://arxiv.org/abs/2210.03350.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. Large language models are better reasoners with self-verification, 2023. URL https://arxiv.org/abs/2212.09561.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2023. URL https://arxiv.org/abs/2203.11171.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug, 2023a. URL https://arxiv.org/abs/2304.05128.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks, 2023b. URL https://arxiv.org/abs/2211.12588.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models, 2023. URL https://arxiv.org/abs/2211.10435.

Debjit Paul, Mete Ismayilzada, Maxime Peyrard, Beatriz Borges, Antoine Bosselut, Robert West, and Boi Faltings. Refiner: Reasoning feedback on intermediate representations, 2024. URL https://arxiv.org/abs/2304.01904.

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve, 2022. URL https://arxiv.org/abs/2210.11610.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback, 2023. URL https://arxiv.org/abs/2303.17651.

Chuanyang Zheng, Zhengying Liu, Enze Xie, Zhenguo Li, and Yu Li. Progressive-hint prompting improves reasoning in large language models, 2024. URL https://arxiv.org/abs/2304.09797.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. Star: Bootstrapping reasoning with reasoning, 2022. URL https://arxiv.org/abs/2203.14465.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. Least-to-most prompting enables complex reasoning in large language models, 2023. URL https://arxiv.org/abs/2205.10625.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models, 2023. URL https://arxiv.org/abs/2305.10601.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of Thoughts: Solving Elaborate Problems with Large Language Models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):17682–17690, March 2024. ISSN 2374-3468, 2159-5399. doi: 10.1609/aaai.v38i16.29720.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E. Gonzalez, and Bin Cui. Buffer of thoughts: Thought-augmented reasoning with large language models, 2024. URL https://arxiv.org/abs/2406.04271.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space, 2024. URL https://arxiv.org/abs/2412.06769.

Wenshan Wu, Shaoguang Mao, Yadong Zhang, Yan Xia, Li Dong, Lei Cui, and Furu Wei. Mind's Eye of LLMs: Visualization-of-Thought Elicits Spatial Reasoning in Large Language Models. 2024.

Rishabh Agarwal, Avi Singh, Lei M. Zhang, Bernd Bohnet, Luis Rosias, Stephanie Chan, Biao Zhang, Ankesh Anand, Zaheer Abbas, Azade Nova, John D. Co-Reyes, Eric Chu, Feryal Behbahani, Aleksandra Faust, and Hugo Larochelle. Many-shot in-context learning, 2024. URL https://arxiv.org/abs/2404.11018.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng,

Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs, 2019. URL https://arxiv.org/abs/1903.00161.

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

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. Mathqa: Towards interpretable math word problem solving with operation-based formalisms, 2019. URL https://arxiv.org/abs/1905.13319.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset, 2021a. URL https://arxiv.org/abs/2103.03874.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. Transformers as soft reasoners over language. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3882–3890. International Joint Conferences on Artificial Intelligence Organization, 7 2020. doi: 10.24963/ijcai.2020/537. URL https://doi.org/10.24963/ijcai.2020/537. Main track.

Bhavana Dalvi, Peter Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark. Explaining answers with entailment trees. *EMNLP*, 2021.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021b. URL https://arxiv.org/abs/2009.03300.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi

Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark, 2024. URL https://arxiv.org/abs/2406.01574.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof qa benchmark, 2023. URL https://arxiv.org/abs/2311.12022.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

François Chollet. On the measure of intelligence, 2019. URL https://arxiv.org/abs/1911.01547.

Elizabeth S. Spelke and Katherine D. Kinzler. Core knowledge. *Developmental Science*, 10(1):89–96, January 2007. ISSN 1363-755X, 1467-7687. doi: 10.1111/j.1467-7687.2007.00569.x.

Sean Williams and James Huckle. Easy Problems That LLMs Get Wrong, May 2024.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. Analysing mathematical reasoning abilities of neural models, 2019. URL https://arxiv.org/abs/1904.01557.