Digital Discovery

ROYAL SOCIETY OF CHEMISTRY

PAPER

View Article Online
View Journal | View Issue

Cite this: *Digital Discovery*, 2025, **4**, 500

Exploring the expertise of large language models in materials science and metallurgical engineering†

Christophe Bajan ‡ and Guillaume Lambard ‡*

The integration of artificial intelligence into various domains is rapidly increasing, with Large Language Models (LLMs) becoming more prevalent in numerous applications. This work is included in an overall project which aims to train an LLM specifically in the field of materials science. To assess the impact of this specialized training, it is essential to establish the baseline performance of existing LLMs in materials science. In this study, we evaluated 15 different LLMs using the MaScQA question answering (Q&A) benchmark. This benchmark comprises questions from the Graduate Aptitude Test in Engineering (GATE), tailored to test models' capabilities in answering questions related to materials science and metallurgical engineering. Our results indicate that closed-source LLMs, such as Claude-3.5-Sonnet and GPT-4o, perform the best with an overall accuracy of ∼84%, while open-source models, such as Llama3-70b and Phi3-14b, top at ∼56% and ∼43%, respectively. These findings provide a baseline for the raw capabilities of LLMs on Q&A tasks applied to materials science, and emphasise the substantial improvement that could be brought to open-source models *via* prompt engineering and fine-tuning strategies. We anticipate that this work could push the adoption of LLMs as valuable assistants in materials science, demonstrating their utilities in this specialised domain and related sub-domains.

Received 2nd October 2024
Accepted 7th January 2025

DOI: 10.1039/d4dd00319e

rsc.li/digitaldiscovery

1 Introduction

Large Language Models (LLMs) represent a significant advancement in artificial intelligence (AI), demonstrating exceptional proficiency in natural language processing (NLP). These models are designed to generate human-like text based on the patterns extracted from large pre-training data. LLMs have shown notable progress in a range of NLP tasks, including text generation, translation, summarization, and question answering on various benchmarks.

However, LLMs' capabilities often degrade when addressing domain-specific requests, such as those in materials science.[1] This limitation arises because pre-training data typically come from diverse web sources, encompassing a wide range of domains. While this approach effectively compresses general knowledge into the LLM's parameters, it can lead to the merging of unrelated contexts during inference, potentially resulting in incorrect assertions.

To overcome this challenge and effectively utilize LLMs for domain-specific tasks, two primary strategies can be employed:

(i) Train a dedicated LLM from scratch with a smaller parameter count, specifically tailored to encapsulate the desired domain knowledge.

(ii) Fine-tune a pre-trained LLM to a specific domain.[2]

In this study, we adopt the second strategy, leveraging the instruction-following capabilities and general NLP proficiency of pre-existing models. Our final objective is to fine-tune an existing LLM and integrate it into a retrieval-augmented generation (RAG) system for materials science applications. To guide this future fine-tuning process and establish a baseline for evaluation, we first assess in the present study the capabilities of available LLMs in materials science. This evaluation aims to:

- Establish a comprehensive baseline performance on materials science tasks.
- Identify LLMs that balance high capabilities with modest parameter counts, crucial for efficient fine-tuning and deployment.
- Discover potential areas for improvement in the evaluation process itself.

1.1 LLMs in materials science

Recent years have witnessed significant advancements in leveraging LLMs for materials science and engineering. Domain-specific models and tools have emerged to address the challenges of applying NLP techniques to scientific research. Notable examples include:

Data-Driven Material Design Group, National Institute for Materials Science, Tsukuba, Japan. E-mail: BAJAN.Christophe@nims.go.jp; LAMBARD.Guillaume@nims.go.jp

† Electronic supplementary information (ESI) available. See DOI: https://doi.org/10.1039/d4dd00319e

‡ These authors contributed equally to this work.

• MatBERT:[3] a BERT-based model fine-tuned on materials science literature, enabling tasks such as information extraction and text classification.

• Mat2Vec:[4] provides word embeddings tailored for materials science, facilitating semantic analysis and knowledge representation.

• KGQA4MAT:[5] a knowledge-based system demonstrating the utility of knowledge graph question answering for structured scientific reasoning, particularly in applications like metal–organic frameworks.

• HoneyComb:[6] highlights the adaptability of LLMs to specialized agent-based systems that can assist in materials research workflows.

Furthermore, frameworks like SciQAG[7] have been developed to automatically generate question-answer (Q&A) pairs from scientific literature, addressing the need for domain-specific Q&A datasets. These efforts complement existing benchmarks such as ChemLLMBench[8] (for chemistry), MultiMedQA[9] (for medicine), and SciEval[10] (for STEM domains).

Despite these advancements, there remains a need for tailored benchmarks that specifically evaluate LLMs' understanding of materials science concepts. The MaScQA benchmark[1] addresses this gap by providing a curated dataset of 650 questions covering diverse sub-fields within materials science, including thermodynamics, atomic structure, mechanical behavior, and materials characterization. It allows for evaluating fundamental comprehension, conceptual reasoning, and numerical problem-solving—capabilities essential for real-world materials science tasks.

1.2 The MaScQA benchmark

While MaScQA is the most comprehensive benchmark tailored specifically to materials science and metallurgical engineering, alternative Q&A datasets focus on related scientific domains:

• SciQ:[11] a general science dataset with 13 679 questions across physics, chemistry, and biology, useful for evaluating broader scientific reasoning.

• ChemData700k and ChemBench4k:[12] benchmarks designed for chemistry competency, focusing on tasks related to chemical properties, reactions, and structures.

• MoleculeQA:[13] a dataset for molecular-level reasoning, particularly useful for tasks involving molecular properties and design.

These alternatives offer valuable insights but either lack the specificity of MaScQA or focus on narrower aspects of chemistry and molecular properties. MaScQA remains unique in its ability to test both conceptual understanding and numerical reasoning across diverse materials science sub-fields, making it the most suitable benchmark for this study.

Originally consisting of 650 questions derived from the Graduate Aptitude Test in Engineering (GATE), the MaScQA benchmark was refined by ourselves by manually removing 6 Q&A samples due to issues such as duplication or missing information (see Table 1 in the ESI† for details). This minor reduction does not significantly bias the evaluation outcomes.

The MaScQA benchmark is categorized by four types of questions:

• 283 Multiple Choice Questions (MCQs)
• 70 Matching Type Questions (MATCH)
• 67 Numerical Questions with Multiple Choices (MCQN)
• 224 Numerical Questions (NUM)

These question types test various aspects of materials science knowledge, from conceptual understanding to numerical problem-solving. The questions span 14 distinct sub-fields within materials science, as shown in Fig. 1.

We selected this benchmark due to its comprehensive coverage of various domains within materials science, the substantial number of questions with answers curated by hand by the MaScQA authors, and the diversity of question types that necessitate both broad knowledge and computational abilities. By establishing a baseline of LLM performance on the MaScQA benchmark, we can better understand their current limitations and potential areas for improvement in materials science applications.

1.3 LLM selection

The selection of LLMs for this study encompasses a variety of closed- and open-source models listed in Table 1. This diversity ensures a comprehensive evaluation across different architectures, accessibility, and fine-tunability.[14,15] The models were sourced from leading AI research organizations and companies, including Anthropic, OpenAI, Meta, Mistral AI, and Microsoft.

By evaluating models from these varied sources, we aim to capture a broad spectrum of performance characteristics, enabling a more thorough understanding of the current state of LLMs applied to materials science. This approach allows us to assess not only the raw performance of these models in answering materials science questions but also to capture the trade-off between their accessibility, affordability, and customization potential for further domain-specific fine-tuning.[16,17]

The choice of LLMs reflects models that were widely used and publicly available at the time of experimentation. Including both older and newer versions of the same models (*e.g.*, GPT-3.5-turbo and GPT-4) enables us to track progress and evaluate incremental improvements in reasoning and performance for domain-specific tasks. While newer models, such as Llama 3.1, were released after our experiments, the results presented here provide a valuable baseline for future comparisons. Notably, improvements observed for Llama 3.1:70b on benchmarks like MATH[18] suggest that further evaluation on MaScQA could yield insightful comparisons.

2 Methodology

2.1 LLM preparation

Our study diverges from the original work from Zaki *et al.*[1] on several key aspects. We expanded our evaluation to 15 different LLMs instead of only 3 (Llama2-70b, GPT-4, and GPT-3.5-turbo) to gain a broader understanding of LLM capabilities in materials science. Additionally, we chose not to include the chain-of-thought prompting method as preliminary results in ref. 1

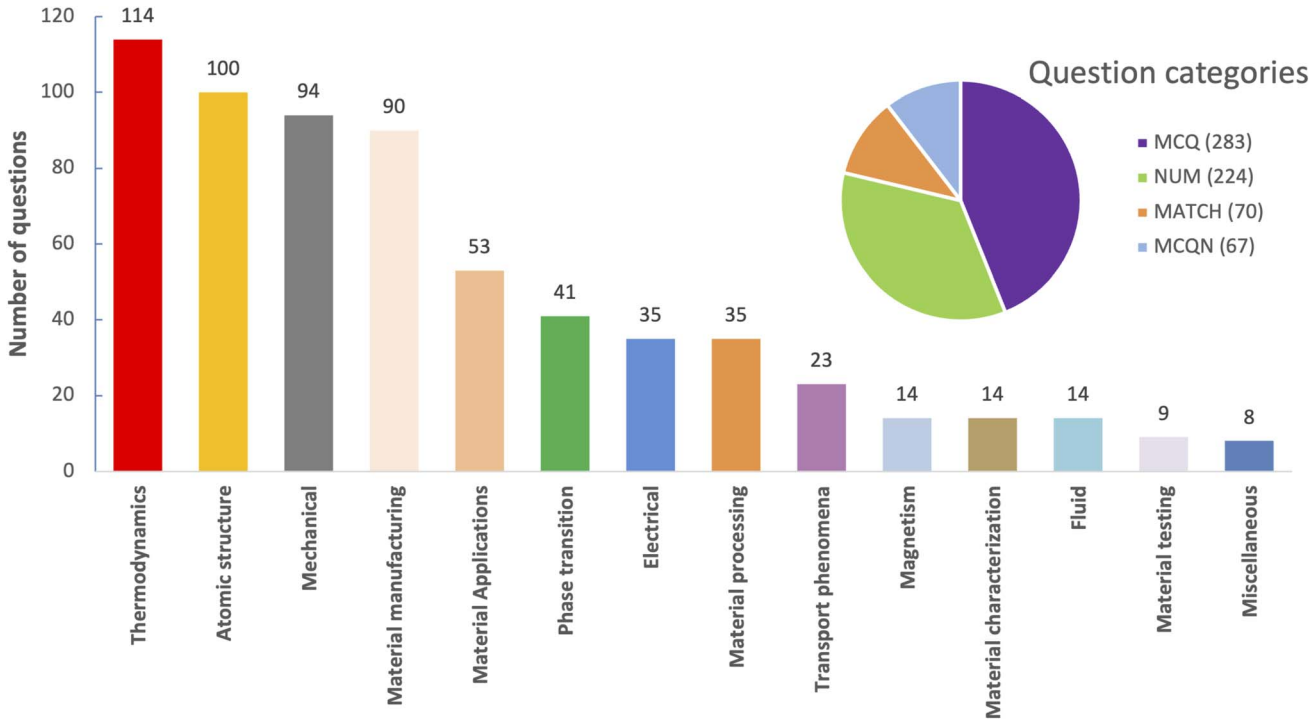

Fig. 1 Distribution of the number of questions per sub-field. On the top-right hand, the number of questions per type is also reported. Figure updated from Zaki *et al.*[1] after removal of 6 Q&A samples from the original MaScQA dataset.

indicated that it did not significantly influence the performance of LLMs in answering materials science related questions. Another important difference came from the temperature parameter that regulates the stochasticity of the LLM response. Zaki *et al.* used a temperature of 1 during LLM's evaluations which allows for more randomness in the model's responses. However, we opted to use a temperature of 0 to ensure maximum determinism and consistency in the answers. A temperature of 0 ensures that a model chooses the most probable answer and provides a fairer assessment of the models' knowledge integration and usage abilities. Indeed, with the shape of the posterior distribution of tokens for a given input sequence being unknown for every LLM, this would impose the proposal of two strategies for a fair evaluation: (i) fix the temperature as we did, or (ii) find the best temperature for each LLM. As the second strategy being costly and time-prohibitive, we opted for the first one such that the most probable output from each LLM is compared. To also ensure the reliability of our results, we submitted each question to the models three times to assess the repeatability of their answers. Indeed, even though a temperature of 0 was fixed to maximize determinism in answers, uncontrollable features leading to stochasticity still

Table 1 List of the LLMs and their characteristics selected for this study

Models	Developer	Open-source	Fine-tuning	Number of parameters
Claude-3-Haiku	Anthropic	✗	✗	—
Claude-3-Opus	Anthropic	✗	✗	—
Claude-3.5-Sonnet	Anthropic	✗	✗	—
GPT-3.5-turbo	OpenAI	✗	✓	—
GPT-4	OpenAI	✗	✓	—
GPT-4-turbo	OpenAI	✗	✗	—
GPT-4o	OpenAI	✗	✓	—
GPT-4o-mini	OpenAI	✗	✓	—
Llama2-7b	Meta	✓	✓	7B
Llama2-70b	Meta	✓	✓	70B
Llama3-8b	Meta	✓	✓	8B
Llama3-70b	Meta	✓	✓	70B
Mistral-7b	Mistral AI	✓	✓	7B
Phi3-3.8b	Microsoft	✓	✓	3.8B
Phi3-14b	Microsoft	✓	✓	14B

remain such as floating-point precision,[19] expert selection in mixture of experts (MoE) models like GPT-4 and Mixtral-8x7B,[20] multi-threaded operations, random number generator state differences between runs, *etc.*[21]

Finally, we maintained consistency with the original study by using the same assistant prompt preceding every question and instructing LLM's desired behaviour: "Solve the following question. Write the correct answer inside a list at the end". This approach allowed for direct comparison of our results to those of Zaki *et al.*[1]

We used the OpenAI, Anthropic and Ollama APIs to access the models.[22–24] The models used in this study are GPT-4-turbo, GPT-4o, GPT-4o-mini, GPT-4, GPT-3.5-turbo, Claude-3-Opus, Claude-3-Haiku, Claude-3.5-Sonnet, Llama2-7b, Llama2-70b, Llama3-8b, Llama3-70b, Mistral-7b, Phi3-3.8b and Phi3-14b. The tokenization process for all LLMs was handled automatically by the respective Python libraries, Ollama and OpenAI, which provide built-in tokenization as part of their APIs. No custom tokenization was applied in this study. Readers interested in the specifics of tokenization can refer to the official documentation of these libraries. The results were saved in*.txt files and are available on GitHub: **https://github.com/Lambard-ML-Team/LLM_comparison_4MS**.

The LLMs were tested on two different machines: a MacBook Pro M1 (2020, 8 GB RAM) and a GPU server (8× A100 40 GB PCIe NVIDIA GPUs). To assess the impact of hardware on performance only GPT-3.5-turbo, GPT-4, Llama2-7b, and Llama3-8b have been tested on both machines. For models such as GPT-3.5-turbo and GPT-4 which only rely on OpenAI's servers, the results remained consistent across both machines. However, for models like Llama2-7b and Llama3-8b, which run locally and are directly impacted by the host machine's specifications, performance variations were observed. Llama2-7b performed similarly on both machines, while Llama3-8b exhibited a 16% performance improvement on the GPU server. To ensure optimal testing conditions, we divided the models based on their computational requirements and on machines' availability. The distribution of models is as follows:

- MacBook Pro M1: GPT-4-turbo, GPT-4o, GPT-4, GPT-3.5-turbo, Claude-3-Opus, Claude-3-Haiku, Claude-3.5-Sonnet, Llama2-7b, and Llama3-8b.
- GPU server: GPT-4, GPT-4o-mini, GPT-3.5-turbo, Llama2-7b, Llama2-70b, Llama3-8b, Llama3-70b, Mistral-7b, Phi3-3.8b, and Phi3-14b.

This distribution ensures that local models benefit from the GPU server's superior computational resources, providing a more accurate assessment of LLMs' capabilities under optimal conditions. In the study conducted in ref. 1, the evaluation of the LLMs' responses was manually performed. However, our study involves a significantly larger amount of LLM responses to evaluate, 19 LLMs (15 unique models and 4 models assessed on both machines) across three iterations for each of the 644 questions, resulting in a total of ∼37 000 answers. Given the large scale of this dataset, manual evaluation would be impractical. Therefore, we applied a LLM-as-a-judge strategy[25] assisted by GPT-4o to handle this extensive volume efficiently and ensure accuracy. Fig. 2 summarises the entire pipeline for generating answers and evaluating them.

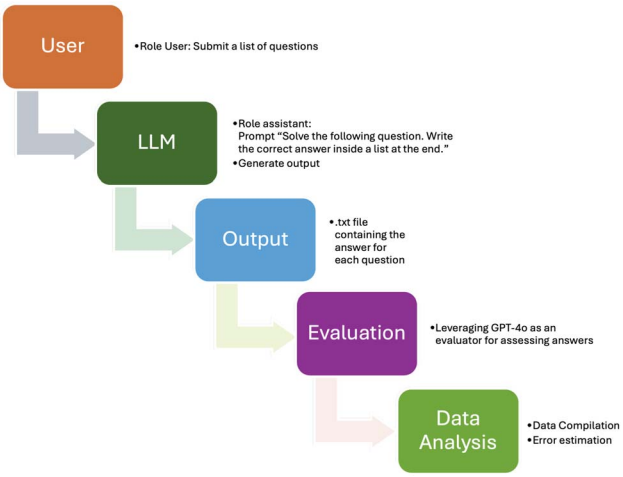

Fig. 2 Pipeline for generating and evaluating responses from LLMs to the MaScQA benchmark.

2.2 Autonomous answer analysis

To estimate the accuracy of GPT-4o to autonomously analyse LLM responses, we manually checked the results for four different LLMs. The manual analysis wasn't straightforward as certain models, mainly Llama2 and Llama3, provided ambiguous answers as shown in Fig. 3. Our approach for determining the correctness of these answers involved adopting the perspective of an examiner and evaluating whether the LLM's response matched the correct answer, focusing solely on the correctness of the selected option rather than the accompanying reasoning or explanatory text.

As shown in Fig. 3, there are several types of ambiguous answers from the Llama2-7b model. Fig. 3(a) illustrates a case where the reasoning and calculation are incorrect, but the correct letter is selected with an incorrect value association. Fig. 3(b) shows the model selecting the correct answer while providing contradictory reasoning. Fig. 3(c) demonstrates a situation where the reasoning and calculation are incorrect, yet the correct answer is chosen. Finally, Fig. 3(d) depicts the correct answer being selected despite incorrect reasoning and associated text.

In the case of MATCH, MCQ, and MCQN questions, responses are assessed solely based on the selected letter (A, B, C, or D) rather than the accompanying reasoning, calculations, or explanatory text. Consequently, for such questions, the answers depicted in Fig. 3 should be considered correct if they align with the expected answer's letter, regardless of any associated reasoning or textual explanations.

Finally, to validate GPT-4o's role as an evaluator, we performed a manual comparison of its judgments against human-assigned scores, as shown in Tables 2 and 3. This analysis demonstrates GPT-4o's accuracy as a judge while also identifying areas where discrepancies arise, particularly for questions requiring nuanced reasoning.

Question Number	QUESTION	Correct Answer	Llama2-7b Model response	GPT_4o_analysis
a) G-XEC-2012-18	['A plain 0.45 wt.% carbon steel is cooled slowly from 900°C to just below the eutectoid temperature (723°C) so that the following reaction occurs: γ (0.8 wt.% C) ↔ α (0.02 wt.% C) + Fe_3C (6.67 wt.% C). Find the volume % of pearlite for the steel just below 723°C for 0.45 wt.% carbon steel.\n', '(A) 44.9% (B) 55.1% (C) 40.9% (D) 59.1%']	B	To solve this problem, we need to use the Clausius-Clayton equation, which describes the transformation of austenite (γ) to pearlite (α) and carbides (Fe_3C) in a carbon steel: γ → α + Fe_3C We are given that the initial composition of the steel is 0.45 wt.% carbon, and we want to find the volume percentage of pearlite just below the eutectoid temperature (723°C). Using the Clausius-Clayton equation, we can write: α + Fe_3C = γ - C where C is the carbon content of the steel. In this case, C = 0.45 wt.%. Solving for α, we get: α = (γ - C) / (1 + (C / γ)) Substituting the given values, we get: α = (0.8 wt.% C - 0.45 wt.%) / (1 + (0.45 wt.% C / 0.8 wt.% C)) = 44.9% Therefore, the volume percentage of pearlite for the steel just below 723°C is: (B) 44.9%.	Incorrect
b) G-META-19-14	['Hydrogen bonds are stronger than\n', '(A) Ionic bonds (B) Metallic bonds (C) van der Waals bonds (D) Covalent bonds']	C	Hydrogen bonds are stronger than: (C) Van der Waals bonds Explanation: Hydrogen bonds are the weakest type of intermolecular force, with bond strengths ranging from 5-30 kJ/mol. In contrast, van der Waals bonds have a stronger bond strength, typically between 50-200 kJ/mol. Therefore, hydrogen bonds are weaker than van der Waals bonds. Answer: (C) Van der Waals bonds	Incorrect
c) G-META-18-34	['Given: Gas constant R = 8.314 J $mol^(-1)K^(-1)$, and Faraday's constant F = 96500 C $mol^(-1)$. What is the voltage required to electrolytically refine impure copper of activity $a_(Cu)$ = 0.9 (Raoultian standard state) to pure copper at 300 K\n', '(A) 1.36 mV (B) 2.72 mV (C) 0 mV (D) 5.44 mV']	A	To electrolytically refine impure copper of activity $a_(Cu)$ = 0.9 (Raoultian standard state) to pure copper at 300 K, the voltage required can be calculated using the Nernst equation: V = (E^2 - E1)^(1/2) / RT where V is the voltage required, E1 and E2 are the activities of the electrodes (in this case, pure copper), R is the gas constant, T is the temperature in Kelvin, and the subscript 1 and 2 refer to the electrode and the solution, respectively. Using the given values, we have: E1 = F / R * (a(Cu))^2 = 96500 C $mol^(-1)$ * (0.9)^2 = 87350 V E2 = RT / F = (8.314 J $mol^(-1)K^(-1)) * (300 K) = 2594.7 V Therefore, the voltage required to electrolytically refine impure copper of activity $a_(Cu)$ = 0.9 (Raoultian standard state) to pure copper at 300 K is: V = (87350 V - 2594.7 V)^(1/2) / R = 3670 V So the answer is (A) 1.36 mV.	Incorrect
d) G-META-14-11	['As mercury is cooled from room temperature to 3 K, its electrical behaviour changes from that of\n', '(A) conductor to insulator (B) semiconductor to insulator (C) conductor to superconductor (D) semiconductor to superconductor']	C	The correct answer is (C): As mercury is cooled from room temperature to 3 K, its electrical behavior changes from that of a conductor to an insulator. Explanation: Mercury is a good conductor at room temperature, but as it is cooled to lower temperatures, it becomes an insulator. This is because the electrons in mercury become Localized and cannot move freely, causing the material to lose its conductivity. At a temperature of 3 K (-270°C), mercury becomes an ideal insulator. Therefore, the correct answer is (C): As mercury is cooled from room temperature to 3 K, its electrical behavior changes from that of a conductor to an insulator.	Incorrect

Fig. 3 Example of ambiguous answers from the Llama2-7b model analysed by GPT-4o. (a) Wrong reasoning and calculation, selected the correct letter but associated the wrong value with it, (b) selected the correct answer but the reasoning says the opposite, (c) reasoning and calculation are incorrect but selected the correct answer, and (d) selected the correct answer but the reasoning and the text associated with the letter C are incorrect.

Table 2 Number of misclassifications (over 644 questions) and estimated accuracy of the evaluating model GPT-4o with the first approach

Models	Errors GPT-4o	Accuracy GPT-4o
Claude-3.5-Sonnet	10	98.4%
GPT-4-turbo	17	97.4%
Llama3-8b (MAC)	40	93.8%
Llama2-7b (GPU server)	48	92.5%
Overall accuracy	—	**95.5%**

2.2.1 First approach. Initially, we selected GPT-4o for this task, using a straightforward prompt: "Based on the question and the correct answer, You must tell if the other answer is correct or not by answering only with Correct or Incorrect" as shown in Fig. 4a and then submitted the question in the format: "The question is" + ⟨QUESTION⟩ + ", the correct answer is" + ⟨CORRECT ANSWER⟩ + "and the other answer is:" + ⟨MODEL ANSWER⟩. Consequently, the accuracy of GPT-4o in properly evaluating LLMs' answers was estimated to be an overall ∼95.5% which is a strong performance, as shown in Table 2. We

Table 3 Number of misclassifications (out of 644 questions) and the corresponding estimated accuracy of the evaluating model GPT-4o when applying the second approach. The table also includes a comparative analysis with GPT-4o-mini

Models	Errors GPT-4o	Accuracy GPT-4o	Errors GPT-4o-mini
Llama2-7b (GPU server)	15	97.7%	28
Llama3-8b (GPU server)	11	98.3%	—
Mistral-7b (GPU server)	16	97.5%	—
GPT-4 (GPU server)	11	98.3%	41
Overall accuracy	—	**97.9%**	**94.6%**

a) Based on the question and the correct answer, You must tell if the other answer is correct or not by answering only with Correct or Incorrect

b) You are an AI judge tasked with evaluating the correctness of a predicted answer to a given question. Your goal is to determine if the predicted answer is equivalent to the correct answer and if the reasoning behind the predicted answer is sound.

You will be provided with the following information:
<question>
{question}
</question>

<correct_answer>
{correct_answer}
</correct_answer>

<predicted_answer>
{predicted_answer}
</predicted_answer>

<question_type>
{question_type}
</question_type>

The question type will be one of the following: MCQ (Multiple Choice Question), MATCH (Match the Following), MCQN (Multiple Choice Numerical Question), or NUM (Numerical Question).

To evaluate the predicted answer, follow these steps:

1. Carefully read the question, correct answer, and predicted answer.

2. Based on the question type, use the following criteria:
- For MCQ: Check if the predicted answer matches the correct option(s). If multiple options are correct, ensure all are included in the predicted answer.
- For MATCH: Verify that the predicted answer contains the correct set of matched entities.
- For MCQN: Confirm that the predicted answer matches the correct numerical option and that the reasoning is sound.
- For NUM: Compare the predicted numerical answer to the correct answer, allowing for minor rounding differences as specified in the question.

3. Evaluate the reasoning behind the predicted answer, if provided. Ensure it is logically sound and relevant to the question.

4. In your internal thought process, consider the following:
- Is the predicted answer equivalent to the correct answer?
- Is the reasoning (if provided) correct and relevant?
- Are there any discrepancies or errors in the predicted answer?
- For numerical questions, is the answer within an acceptable range of the correct answer?

5. Based on your evaluation, prepare a brief explanation of your judgment. This explanation should be concise but clear, highlighting the key factors that influenced your decision.

6. Provide your final judgment and explanation in the following format:
<judgment>
Explanation: [Your brief explanation here]
Result: [CORRECT or INCORRECT]
</judgment>

Remember, a predicted answer should only be judged as correct if it is equal or very close to the correct answer AND the reasoning (if provided) is correct. Be thorough in your evaluation and clear in your explanation.

Fig. 4 Comparison of the prompt used for the evaluation of GPT-4o: (a) corresponds to the first approach with a straightforward prompt, while (b) corresponds to the second approach with a step-by-step protocol and detailed explanation required.

define here by "misclassification" the correct answers labelled as incorrect, and incorrect answers labelled as correct. However, we observed significant variation depending on the specific model being evaluated. Models with generally lower performance such as Llama2 and Llama3 were more susceptible to errors in the evaluation process. Notably, these models frequently had correct answers misclassified as incorrect more often than incorrect answers misclassified as correct. For instance, Llama2-7b initially demonstrated 85/644 correct answers; however, we observed that 48 correct answers were misclassified by GPT-4o. Despite maintaining an accuracy of ∼92.5%, this misclassification resulted in Llama2-7b having an increase to 133 correct answers in total, reflecting a difference of ∼56.5%.

2.2.2 Second approach. In an attempt to resolve the issue of misclassification with the first approach, we decided to update the prompt for the evaluation to a more sophisticated one. In this new approach, the questions were formatted differently and the prompt described the task that GPT-4o had to perform more precisely. Specifically, the prompt instructed the model to evaluate not only the accuracy of the predicted answer but also the validity of the reasoning behind it, if provided. The model was required to ensure that the predicted answer matched the correct option, contained the correct set of matched entities, or was numerically accurate within an acceptable range. Furthermore, the model was tasked with providing a clear and concise explanation of its judgment, focusing on the key factors that influenced its decision. This refined prompt, shown in Fig. 4b, enhanced the model's ability to interpret and evaluate answers more effectively, ultimately improving the accuracy and reliability of the evaluation process.

As shown in Table 3, the accuracy of the evaluation reached ∼97.9%, demonstrating greater stability across different models. Notably, Llama2-7b's misclassifications decreased from 48 in the initial approach to 15, and Llama3-8b's misclassifications dropped from 40 to 11. This significant decrease in misclassifications highlights the effectiveness of the revised evaluation prompt. However, if the revised prompt is applied to the GPT-4o-mini model as a judge, the results were less conclusive when compared to those of GPT-4o, with 28 misclassifications observed for Llama2-7b and 41 for GPT-4. Historically, the model GPT-4o-mini was made available to the public by OpenAI during the evaluation process of the LLMs' answers, and its more attractive price tag enabled us to try it out on the benchmark.

A key issue with GPT-4o-mini was its failure to recognize some correct answers when the evaluated LLM neglected to include the corresponding letter in its responses. This suggests that while the new prompt greatly enhances evaluation accuracy for higher-performing models, it may still be prone to errors with LLMs with lower reasoning capabilities or when critical elements, such as the letter designation in answers, are omitted. Future work could explore refining the prompt further to handle such cases more effectively or developing additional

layers of validation to ensure even greater accuracy and consistency across all model types.

2.3 Random baseline calculations

To assess the extent to which LLMs outperform chance-level guessing, we compute a random baseline for each of the question types in the MaScQA benchmark. Knowing that each of the MATCH, MCQ, and MCQN questions has four options, with one correct answer, we derive the mean, μ, and standard deviation, σ, of the expected number of correct answers from the properties of the binomial distribution, which models the number of successes (correct answers) in a fixed number of independent trials (questions), each with a fixed probability of success $p = 0.25$. Specifically,

$$\mu = n \times p,\ \sigma = \sqrt{n \times p \times (1-p)},$$

where n is the number of questions for a given category, and p is the probability of guessing correctly.

Therefore, and as reported in Table 4, we have:

- For MATCH questions (70 total):

$$\mu = 70 \times 0.25 \approx 17.5,\ \sigma = (70 \times 0.25 \times 0.75)^{0.5} \approx 3.6.$$

- For MCQ questions (283 total):

$$\mu = 283 \times 0.25 \approx 70.7,\ \sigma = (283 \times 0.25 \times 0.75)^{0.5} \approx 7.3.$$

- For MCQN questions (67 total):

$$\mu = 67 \times 0.25 \approx 16.7,\ \sigma = (67 \times 0.25 \times 0.75)^{0.5} \approx 3.5.$$

For NUM questions (224 total), a precise numerical reasoning is required, and the answers aren't multiple-choice. Thus, the probability of guessing correctly by chance is effectively close to zero. This stems from the nature of the problem: without predefined options, the likelihood of randomly selecting the correct answer in a continuous or large discrete range (*e.g.*, all real numbers or integers) is negligible. Consequently, we fix the mean baseline accuracy for NUM questions at 0% with equivalently 0% in standard deviation, acknowledging the unlikelihood of finding the correct answer randomly on a continuous range of real numbers.

Finally, the combined $\mu = 105.0$ and $\sigma \approx 8.9$ for the entire set of MATCH, MCQ, MCQN, and NUM questions are derived from the sum of the means and variances (σ^2) of each question category, respectively.

Thus, we can compare the performance of each LLM against this random baseline to highlight their ability for knowledge retrieval, logical reasoning, and numerical computation effectively.

3 Results

After establishing the accuracy of the methodology for the autonomous evaluation pipeline, the entire list of LLMs from Table 1 were evaluated on the MaScQA benchmark with the results presented in Tables 4 and 5. Table 4 summarizes the average correctness of each LLM across three iterations on the 644 benchmark questions. Additionally, to assess the impact of hardware on model performance, four LLMs (GPT-4, GPT-3.5-turbo, Llama2-7b and Llama3-8b) were tested on a MAC and a GPU server. This comparative evaluation offers valuable

Table 4 Number of correct answers achieved by 19 Large Language Models (LLMs) (representing 15 unique models) on the MaScQA[1] benchmark. Each model was evaluated through three submissions per question to ensure robustness and consistency of results. Some LLMs were tested on two different machines to assess potential variations in performance. The numbers in parentheses indicate the number of questions within each category. For comparison, we also incorporate a random baseline as computed in Section 2.3

Machine used	LLM	MATCH (70)	MCQ (283)	MCQN (67)	NUM (224)	Total correct answer (644)
Mac Pro M1	GPT-4-turbo	65.0 ± 1.0	236.8 ± 2.8	48.8 ± 2.7	141.2 ± 3.5	491.8 ± 4.5
	GPT-4o	67.9 ± 0.9	**260.1 ± 2.2**	50.7 ± 2.0	161.0 ± 5.9	539.7 ± 8.2
	GPT-4	60.4 ± 1.4	214.8 ± 2.4	34.4 ± 0.2	80.4 ± 6.9	390.1 ± 3.5
	GPT-3.5-turbo	25.1 ± 2.7	157.8 ± 2.2	29.1 ± 1.3	47.8 ± 3.7	259.8 ± 8.4
	Claude-3-Opus	68.7 ± 0.6	240.3 ± 0.6	49.2 ± 0.2	143.6 ± 3.8	501.8 ± 3.7
	Claude-3-Haiku	40.3 ± 0.6	205.1 ± 0.2	33.0 ± 0.3	77.0 ± 0.3	355.4 ± 0.5
	Claude-3.5-Sonnet	**69.0 ± 0.0**	248.8 ± 0.7	**55.1 ± 2.0**	**167.1 ± 0.2**	**540.0 ± 1.3**
	Llama2-7b	9.3 ± 2.4	99.2 ± 1.6	14.7 ± 4.8	5.8 ± 1.7	129.0 ± 4.9
	Llama3-8b	22.5 ± 0.7	132.9 ± 1.1	15.1 ± 0.8	18.2 ± 1.1	188.8 ± 1.2
GPU server	GPT-4	61.4 ± 0.5	212.4 ± 2.7	33.9 ± 1.7	85.7 ± 2.3	393.4 ± 3.6
	GPT-4o-mini	59.2 ± 0.4	226.9 ± 1.1	47.1 ± 0.9	120.8 ± 3.3	454.0 ± 4.6
	GPT-3.5-turbo	24.0 ± 3.6	158.3 ± 1.5	30.0 ± 3.2	49.9 ± 0.5	262.2 ± 0.9
	Llama2-7b	9.1 ± 3.7	98.9 ± 10.1	12.3 ± 2.8	5.0 ± 2.9	125.3 ± 10.4
	Llama2-70b	18.9 ± 3.6	129.3 ± 4.1	20.7 ± 3.0	11.8 ± 0.7	180.7 ± 8.7
	Llama3-8b	21.5 ± 4.2	153.8 ± 1.1	22.8 ± 4.1	21.1 ± 0.9	219.1 ± 5.1
	Llama3-70b	51.8 ± 0.9	199.2 ± 2.5	36.5 ± 2.0	73.0 ± 3.6	360.6 ± 1.9
	Mistral-7b	19.4 ± 2.9	129.2 ± 5.2	10.0 ± 2.9	14.4 ± 5.1	173.1 ± 6.8
	Phi3-3.8b	32.9 ± 1.4	146.8 ± 3.9	18.8 ± 1.0	36.8 ± 6.1	235.2 ± 9.6
	Phi3-14b	38.5 ± 3.5	170.5 ± 5.0	23.9 ± 3.6	43.0 ± 5.4	275.8 ± 7.4
Random baseline	—	17.5 ± 3.6	70.7 ± 7.3	16.7 ± 3.5	0.0 ± 0.0	105.0 ± 8.9

Table 5 Performance (accuracy (%)) for 15 different LLMs evaluated on the MaScQA[1] benchmark. Each LLM was assessed through three submissions for each question to ensure robustness and consistency of results. For comparison, we also incorporate a random baseline as computed in Section 2.3

Models	MATCH (%)	MCQ (%)	MCQN (%)	NUM (%)	Overall accuracy(%)
Claude-3-Haiku	57.6 ± 0.8	72.5 ± 0.1	49.3 ± 0.4	34.4 ± 0.1	55.2 ± 0.1
Claude-3-Opus	98.1 ± 0.8	84.9 ± 0.2	73.4 ± 0.3	64.1 ± 1.7	77.9 ± 0.6
Claude-3.5-Sonnet	**98.6 ± 0.0**	87.9 ± 0.2	**82.2 ± 3.0**	**74.6 ± 0.1**	**83.9 ± 0.2**
GPT-3.5-turbo	35.1 ± 4.2	55.9 ± 0.6	44.1 ± 3.3	21.8 ± 1.2	40.5 ± 0.9
GPT-4	87.0 ± 1.6	75.5 ± 0.9	51.0 ± 1.7	37.1 ± 2.4	60.8 ± 0.6
GPT-4-turbo	92.9 ± 1.4	83.7 ± 1.0	72.8 ± 4.1	63.0 ± 1.6	76.4 ± 0.7
GPT-4o	97.0 ± 1.2	**91.9 ± 0.8**	75.6 ± 3.0	71.9 ± 2.6	83.8 ± 1.3
GPT-4o-mini	84.6 ± 0.6	80.2 ± 0.4	70.3 ± 1.3	53.9 ± 1.5	70.5 ± 0.7
Llama2-7b	13.2 ± 4.0	35.0 ± 2.3	20.1 ± 5.6	2.4 ± 1.0	19.7 ± 1.2
Llama2-70b	27.0 ± 5.2	45.7 ± 1.4	30.8 ± 4.4	5.3 ± 0.3	28.1 ± 1.4
Llama3-8b	31.4 ± 3.9	50.6 ± 4.1	28.3 ± 7.4	8.8 ± 0.8	31.7 ± 2.6
Llama3-70b	74.0 ± 1.2	70.4 ± 0.9	54.5 ± 2.9	32.6 ± 1.6	56.0 ± 0.3
Mistral-7b	27.8 ± 4.1	45.7 ± 1.8	14.9 ± 4.3	6.4 ± 2.3	26.9 ± 1.0
Phi3-3.8b	47.0 ± 2.0	51.9 ± 1.4	28.1 ± 1.5	16.4 ± 2.7	36.5 ± 1.5
Phi3-14b	55.0 ± 5.0	60.2 ± 1.8	35.7 ± 5.3	19.2 ± 2.4	42.8 ± 1.1
Random baseline	25.0 ± 5.2	25.0 ± 2.6	25.0 ± 5.3	0.0 ± 0.0	16.3 ± 1.4

insights into how computational resources can influence the performance and accuracy of LLMs' responses. For GPT-4 and GPT-3.5-turbo, no performance differences were observed, as these models rely on the server infrastructure provided by OpenAI, thereby rendering the local hardware inconsequential. However, a notable performance increase of ∼16% was observed for Llama3-8b when run on the GPU server in comparison to MAC M1. Conversely, Llama2-7b showed no significant performance difference between the two machines, likely due to the MAC M1's sufficient capability to handle the model effectively.

This disparity in performance, particularly with Llama3-8b, can be attributed to the computational demands exceeding the MAC M1's capacity, whereas the GPU server, with superior hardware capabilities, could manage the workload without compromise. Additionally, when running Llama2-7b and Llama3-8b on the MAC M1, the system resources were fully utilized, leaving the machine unable to perform other tasks until completion. This was not the case on the GPU server, where system performance remained stable, underscoring the importance of hardware resources in managing complex models like Llama3-8b.

Fig. 5 illustrates that, in general, LLMs tend to demonstrate higher accuracy when responding to questions that provide a set of possible answers (MATCH, MCQ and MCQN). This phenomenon can be explained by the fact that, for the type of questions with multiple choices available, the model is required to select from a predefined list of options. Similar to a student guessing the correct answer, the model may choose the correct option even if the underlying reasoning or calculations are flawed. This tendency is further demonstrated in Fig. 3, where models exhibited correct selections despite incorrect reasoning.

An important aspect of our analysis is the evaluation of the LLMs on NUM, which present a unique challenge as they do not provide potential answers. This type of question requires models to rely solely on their internal knowledge, reasoning, and computational abilities. The results for NUM, as depicted in Table 5, offer a clear depiction of the LLMs' capabilities in these areas. Notably, the performance of the models on NUM questions reveals distinct groups. The difficulties observed in MaScQA's NUM and MCQN categories align with challenges reported in benchmarks such as MATH[18] and ChemBench4k.[12] These tasks often require multi-step computations, reasoning under constraints, and precision in numerical outputs—areas where current LLMs frequently fall short.

Models like Llama2-7b and Mistral-7b, which performed worse than random in MCQN, highlight a persistent issue of

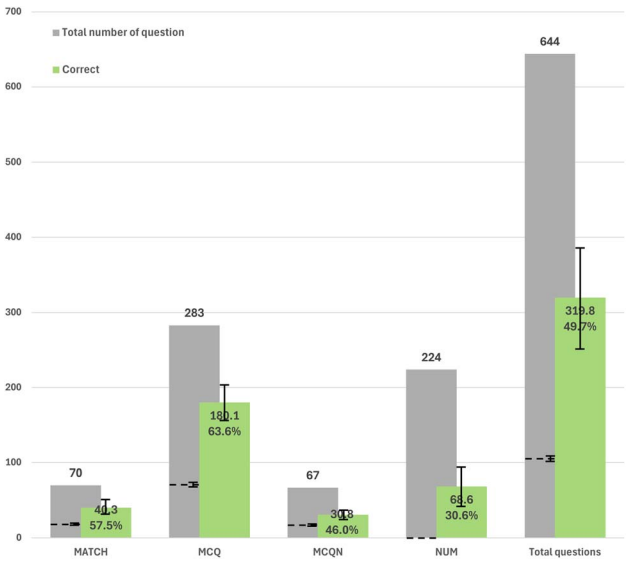

Fig. 5 Comparison of the number of average correct answers, including the 15 unique LLMs tested, to the total number of questions per category, *i.e.*, MATCH, MCQ, MCQN, and NUM, as well as for the whole set of questions. A random baseline per category is indicated as a dashed line.

shallow numerical reasoning and tokenization inefficiencies. Addressing these limitations may require targeted fine-tuning with domain-specific datasets or improved model architectures better suited for handling numerical reasoning tasks.

As shown in Tables 4, 5 and Fig. 5, most of the tested LLMs outperform in average the random baseline in all question categories, except for Llama2-7b in the MATCH and MCQN categories, as well as Mistral-7b in the MCQN category. For those two last LLMs, their results in the MCQN category seem to be hindered by their poor capability on numerical computations, as their performance on the MCQ category alone outperforms the random baseline. However, concerning the behavior of Llama2-7b in the MATCH category, it could imply that Llama2-7b follows systematic flawed reasoning patterns learned from its training data that aren't fitted to materials science and engineering. Additionally, the lack of domain-specific knowledge is hypothesized to also be a culprit. This emphasizes the need for domain-targeted fine-tuning or retraining to align LLMs with materials science tasks. Importantly, such behaviors underscore the value of rigorous benchmarking across diverse question types to identify and address weaknesses in model reasoning capabilities. Also, issues observed in MATCH and MCQ categories are not unique to MaScQA. Similar limitations have been identified in benchmarks like SciQ[11] and MoleculeQA.[13] For MATCH tasks, LLMs struggle to establish logical relationships between entities, often defaulting to heuristic-based reasoning. MCQ tasks, while simpler, can be impacted by pattern exploitation where models rely on superficial cues rather than true conceptual understanding.

These trends underscore the importance of prompt optimization and domain-specific fine-tuning to improve structured reasoning and conceptual alignment in materials science tasks. Future work could explore methods to guide models more effectively through MATCH-type reasoning frameworks and numerical computations.

Claude-3.5-Sonnet emerges as the top performer, closely followed by GPT-4o, both achieving an accuracy exceeding ~70%. This level of accuracy is considered acceptable given the complexity of the task. Claude-3-Opus and GPT-4-turbo closely follow with ~64–63%, both models demonstrating a large effectiveness at handling numerical computations by comparison to the average pool of LLMs topping at ~30.6% (see Fig. 5). Notably, the best studied open-source model, Llama3-70b, achieves results that are closely aligned with those of GPT-4 and Claude-3-Haiku with ~32.6%, underscoring its competitiveness with closed-source models.

Furthermore, the performance comparison between Phi3-3.8b, Phi3-14b, and GPT-3.5-turbo reveals minimal differences, suggesting that the parameter count may not be the sole determinant of a LLM's effectiveness. Interestingly, Phi3-3.8b outperforms several models with double its parameter count, including Llama3-8b, Mistral-7b, and Llama2-7b. The relatively poor performance of these larger models highlights the complexity of balancing model size with other factors such as architecture and training data quality, which can significantly impact overall performance.

The models utilized in the study by Zaki *et al.*[1] show comparable performance to those in our current study. Notably, Llama2-70b exhibited slightly improved performance in our evaluation, with an accuracy of 28.1 ± 1.4% compared to the 24.0% reported by Zaki *et al.* This difference could be attributed to the application of the chain-of-thought (CoT) technique on Llama2-70b in their study, as well as the systematic variation in computational resources and machines used.

In contrast, GPT-4 and GPT-3.5-turbo demonstrated consistent performance across both studies. Specifically, GPT-4 achieved an accuracy of 60.8 ± 0.6% in our work, closely aligning with the 61.38% reported by Zaki *et al.* Similarly, GPT-3.5-turbo performed at 40.5 ± 0.9%, which is consistent with the 38.31% observed in their study. These results suggest that the performance of these models is robust across different experimental setups and conditions. The slight variations in accuracy can likely be attributed to the difference in temperature settings used during evaluation.

The evaluation of the LLMs, shown in Table 5 and Fig. 6, demonstrates that Claude-3.5-Sonnet and GPT-4o are among the top performers, achieving overall accuracies of approximately 84% (see Fig. 1 in the ESI† for details concerning the LLMs' average accuracy on each category MATCH, MCQ, MCQN, and NUM). Claude-3.5-Sonnet emerges as the highest performer, with an overall accuracy of 83.9% with a high stability. Its exceptional performance across MATCH and NUM categories underscores its proficiency in pattern recognition and numerical reasoning, suggesting that it excels in tasks requiring both structured matching and complex calculations. GPT-4o closely follows with an overall accuracy of 83.8%. It demonstrates particular strength in the MCQ category, attaining the highest accuracy of 91.9%. This indicates that GPT-4o is highly effective at handling multiple-choice questions where options are provided. Additionally, GPT-4o's performance in NUM at 71.9% suggests a solid capability in numerical reasoning, although it slightly lags behind Claude-3.5-Sonnet in this area.

Claude-3-Opus and GPT-4-turbo also exhibit commendable performance, with overall accuracies of 77.9% and 76.4%, respectively. These models show a balanced capability across different question types, reflecting their robustness and versatility in handling diverse tasks. Their relatively high performance across MATCH and MCQ categories indicates that they are reliable choices for a range of question types, though they do not quite reach the top levels achieved by Claude-3.5-Sonnet and GPT-4o.

GPT-4 and GPT-4o-mini achieved overall accuracies of 60.8% and 70.5%, respectively. While GPT-4 had lower performance in the NUM category, it was relatively strong in MATCH and MCQ categories. Llama3-70b also falls into the mid-tier category with an overall accuracy of 56.0%. Although it did not outperform the leading models, it showed decent performance in MATCH and MCQ categories. This model's performance highlights its capability in handling structured questions, although it still lags behind the top performers. Llama2-7b, Llama2-70b, Llama3-8b, and Mistral-7b exhibited poor performance across all categories, with overall accuracies below 32%. These models

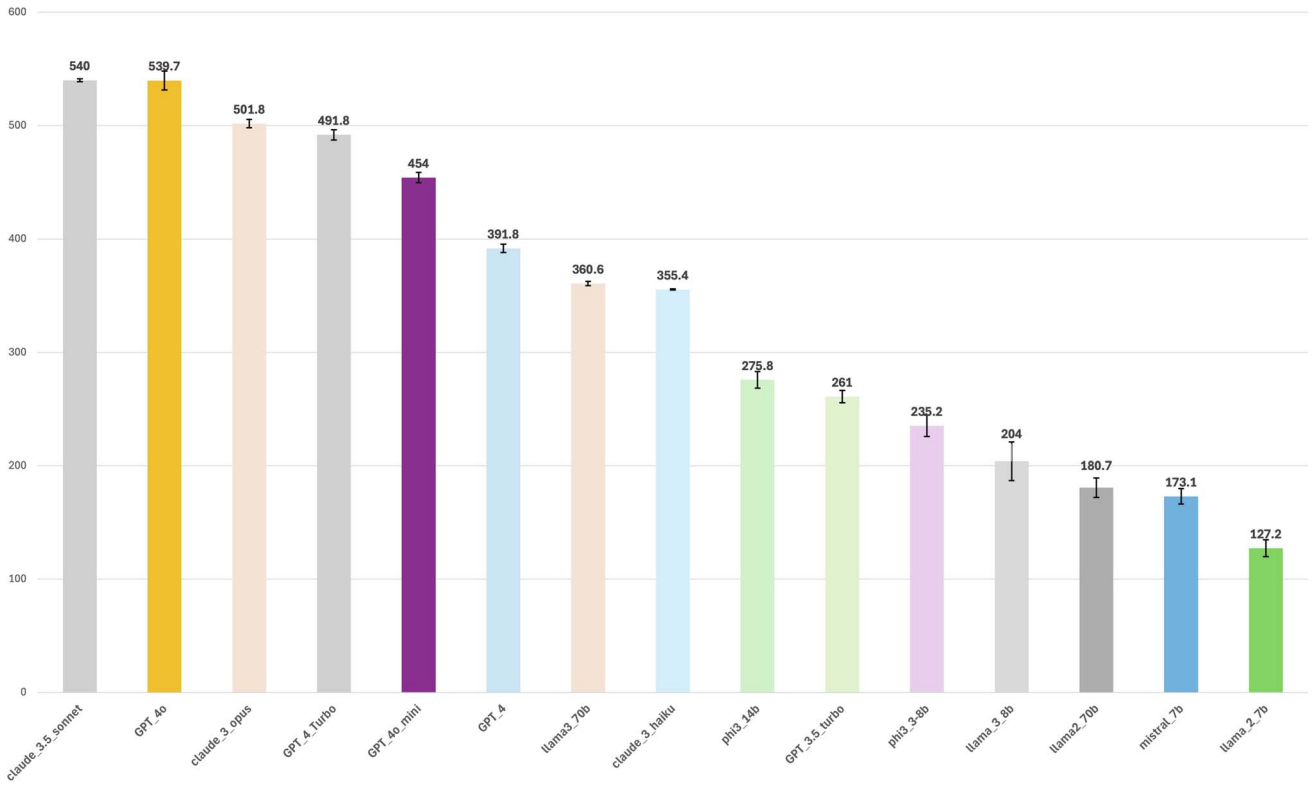

Fig. 6 Average overall performance for the studied 15 unique LLMs with their standard deviation obtained from three runs over the whole set of 644 MATCH, MCQ, MCQN, and NUM questions.

struggled particularly in the NUM category, where their accuracies were very low (ranging from 2.4% to 8.8%). This significant shortfall in numerical reasoning capabilities underscores major limitations in these models' ability to handle complex quantitative tasks, which might be due to their training data or architectural constraints. Also, several factors may explain the observed limitations of open-source models on numerical reasoning tasks:

- Training data limitations: open-source models are often trained on publicly available datasets, which may lack sufficient examples of numerical reasoning, particularly in scientific domains like materials science.
- Tokenization inefficiencies: numbers are tokenized as sequences rather than atomic units, leading to errors in operations involving precision or formatting.
- Smaller model capacity: models with fewer parameters have limited ability to perform complex, multi-step computations compared to their larger closed-source counterparts.
- Reasoning biases: open-source models prioritize fluency during pretraining, resulting in outputs that appear plausible but lack numerical accuracy.

Then, Phi3-3.8b and Phi3-14b performed better than the models explained before, with overall accuracies of 36.5% and 42.8%, respectively. Despite these improvements, their performance still fell short of the top-tier models, particularly in complex tasks such as MCQN and NUM. This suggests that while these models have some capabilities, they are not yet competitive with the leading models in handling more challenging question types.

Addressing these gaps requires a combination of strategies. For example, fine-tuning open-source models on curated datasets with extensive numerical tasks could significantly improve their reasoning capabilities. Additionally, advancements in tokenization strategies and enhanced pretraining methods could help smaller models better handle numerical precision, rounding, and formatting—critical elements for scientific applications like materials discovery.

Such targeted improvements are particularly relevant for tasks like calculating material properties or designing experiments, where numerical accuracy is essential. By bridging these gaps, open-source models can evolve into robust tools for domain-specific applications in materials science.

From the perspective of the categories of questions:

- MATCH: Claude-3.5-Sonnet achieved the highest accuracy (98.6%), closely followed by Claude-3-Opus (98.1%) and GPT-4o (97.0%). This suggests that these models are particularly adept at tasks requiring pattern recognition and matching. The high accuracy across these models suggests their robust capability in identifying and matching patterns effectively.
- MCQ: GPT-4o led in this category with a 91.9% accuracy, indicating its strength in handling multiple-choice questions with provided options, reflecting its ability to navigate through choices efficiently.
- MCQN: Claude-3.5-Sonnet achieved an accuracy of 82.2%, due to its capability to integrate numerical reasoning within the

context of multiple-choice questions. The model's strong performance in this category suggests that it can effectively handle questions that require both choice selection and numerical computation.

• NUM: the NUM category, which requires open-ended numerical answers without provided options, was the most challenging. Claude-3.5-Sonnet performed the best with a 74.6% accuracy, and its advanced numerical reasoning abilities suggests that it is particularly adept at generating accurate numerical responses when no options are provided.

The results in Fig. 6 highlight that while different models exhibit strengths in specific areas, Claude-3.5-Sonnet's performance across both pattern recognition and numerical reasoning tasks positions it as a particularly versatile model. The challenges observed in the NUM category across all models underscore the need for continued advancements in handling open-ended numerical reasoning tasks.

4 Discussion

The results of this study underscore the current superiority of closed-source models, such as GPT and Claude families of models, over their open-source counterparts like Llama, Mistral, and Phi3. Closed-source models consistently demonstrated higher accuracy across various question categories, indicating their advanced architecture, extensive training, and optimization for a broad range of tasks, including the fields of materials science and engineering. However, the potential of open-source models should not be overlooked. Despite their lower performance in this benchmark, open-source models offer opportunities for optimization through methods like prompt engineering and fine-tuning. Fine-tuning, in particular, is a powerful tool that allows these models to be adapted to specific tasks or datasets, potentially enhancing their performance in specialized domains such as in materials science and chemistry.

Overall, the inclusion of a random baseline for the MATCH, MCQ, MCQN, and NUM categories highlights the significant advantage provided by LLMs in answering materials science questions. For most of the tested LLMs, except Llama2-7b and Mistral-7b, they achieve accuracies demonstrating their ability to display reasoning, *i.e.*, a consistent arrangement of their fragments of memorized knowledge, and retrieve information far beyond chance-level guessing. Notably, the NUM category, which lacks predefined options, showcases the models' numerical reasoning capabilities—a critical skill for tasks such as calculating material properties or experimental parameters.

Phi3-3.8b stands out as a particularly promising candidate for such optimization. Despite having a relatively low number of parameters, it achieved an overall accuracy of 36.5%, which is commendable given its smaller scale. This suggests that with targeted fine-tuning and prompt optimization, Phi3-3.8b could potentially improve its performance significantly without demanding an expensive hardware load.

An interesting direction for future work could involve systematically fine-tuning Phi3-3.8b and other open-source models on domain-specific datasets, such as materials science or other technical fields. The MaScQA benchmark results directly inform the development of a RAG system tailored for materials science applications. Such a system will enable AI tools to assist researchers in tasks like synthesizing knowledge from massive literature corpora, proposing experimental designs, and predicting material properties with minimal human input.

For example, strong performance on NUM and MCQ questions demonstrates an LLM's capability to accurately calculate material parameters or resolve conceptual queries—skills essential for automating computational tasks or pre-experimental analyses. Fine-tuning open-source models like Phi3-3.8b using curated materials science datasets will ensure that these tools become domain-optimized, democratizing access to AI-powered solutions in materials research. Additionally, prompt engineering strategies could be explored to better leverage the model's existing capabilities, potentially boosting its performance in specific tasks. By carefully crafting prompts that guide the model's reasoning process, we can help it generate more accurate and contextually appropriate responses. This approach is particularly useful for numerical reasoning tasks, where precise wording can influence the model's output. These approaches not only aim to bridge the performance gap between open- and closed-source models but also promote the democratization of AI by enhancing the utility of models that are freely accessible to the community.

While closed-source models currently lead in performance, the flexibility and accessibility of open-source models present a valuable opportunity for ongoing research and development. By focusing on fine-tuning and prompt optimization, it is possible to enhance the performance of open-source models, making them viable alternatives for specialized applications and contributing to the advancement of open AI technologies for diverse domains, materials science included.

While GPT-4o provides a creative and scalable approach for automating performance evaluation, it is not without limitations. Discrepancies between GPT-4o's assessments and human-assigned scores highlight challenges such as potential biases in LLM judgments, inconsistencies in reasoning, and difficulties with questions requiring deeper conceptual understanding. For this reason, we have complemented GPT-4o-based evaluations with traditional accuracy metrics, ensuring that the results remain quantitatively robust and reliable. Future work could explore hybrid evaluation frameworks that combine automated LLM-based scoring with rigorous manual validation.

The discrepancy observed in evaluation errors for lower-performing models suggests that outputs from these models are more challenging for automated evaluators like GPT-4o to assess accurately. Also, several factors could contribute to the higher susceptibility of lower-performing models to evaluation errors:

• Ambiguity in outputs: lower-performing models often produce ambiguous or incomplete answers, which are inherently harder to evaluate. Outputs may include partially correct information or lack the precision required, particularly for numerical and structured tasks.

• Hallucinations and shallow reasoning: these models are more prone to hallucinations—confident but incorrect outputs—and rely on superficial reasoning, especially when confronted with multi-step or complex questions. Such outputs can mislead evaluators like GPT-4o.

• Tokenization and numerical precision issues: numerical reasoning tasks (*e.g.*, NUM) require strict handling of tokenization and precision. Lower-quality models frequently generate outputs with formatting errors or rounding inconsistencies, increasing evaluation discrepancies.

• Evaluator bias: automated evaluators like GPT-4o may exhibit biases toward linguistic fluency and coherence. Outputs from lower-performing models, which tend to lack these qualities, can be disproportionately misclassified.

These observations offer a preliminary explanation for the observed phenomenon. A more detailed investigation involving model-level diagnostics or deeper access to closed-source architectures would be required to fully analyze this behavior. Future work could focus on developing error analysis frameworks and improving evaluator calibration to better handle outputs from lower-performing models.

This study represents a critical first step in identifying the best-performing LLMs as candidates for fine-tuning and integration into a materials science RAG system. To further advance the applicability of LLMs in materials science, several directions for future work are identified:

• Fine-tuning open-source models: while models like Phi3-3.8b show promise, fine-tuning on curated, domain-specific datasets rich in materials science literature and numerical reasoning tasks will be essential for improving their capabilities.

• Exploring temperature effects: adjusting temperature settings could dynamically optimize model outputs for tasks requiring both creativity and precision, particularly in numerical and reasoning-heavy questions.

• Advanced error correction strategies: implementing techniques such as CoT prompting, in-context learning (ICL), and post-hoc validation methods will address hallucinations, ambiguity, and shallow reasoning in lower-performing models.

• Improved tokenization for numerical tasks: enhancing tokenization strategies to treat numerical inputs as atomic units rather than sequences will reduce errors in numerical reasoning and precision.

The end goal is to create an AI system capable of comprehensively reasoning over materials science knowledge, accelerating discoveries and reducing the time between hypothesis generation and experimental validation.

5 Conclusions

This study used the MaScQA benchmark, developed by Zaki *et al.*,[1] to assess the performance of 15 different LLMs across a diverse set of tasks. The MaScQA dataset is notable for its inclusion of questions from various sub-fields from materials science and engineering and its range of question types MATCH, MCQ, MCQN, and NUM, each of which evaluates different aspects of model capability, such as reasoning, pattern recognition, numerical computation, and decision-making. Among the models tested, two demonstrated exceptional performance: Claude-3.5-Sonnet and GPT-4o. Claude-3.5-Sonnet achieved an overall accuracy of 83.9 $\pm$ 0.2%, while GPT-4o closely followed with an accuracy of 83.8 $\pm$ 1.3%. These results highlight the advanced capabilities of these models in handling a wide array of tasks, particularly in domains requiring robust pattern recognition and complex numerical reasoning.

The variety of question types in the MaScQA benchmark allowed for a comprehensive evaluation of the LLMs, revealing not only the strengths of the top-performing models but also the specific areas where other models struggled. For instance, the NUM category, which involves open-ended numerical questions, proved to be particularly challenging for most models, underscoring the ongoing difficulties in developing LLMs with strong numerical computation abilities.

Overall, the findings from this study emphasize the potential of using benchmarks like MaScQA to push the boundaries of LLM capabilities for specific domains like materials science and engineering. The high performance of Claude-3.5-Sonnet and GPT-4o suggests that while state-of-the-art models continue to improve, there remains significant potential for further improvements, particularly for open-source models that can be fine-tuned and optimized for specific tasks. Future work in this area will focus on enhancing the capabilities of open-source models through targeted fine-tuning and prompt engineering, potentially narrowing the gap between open- and closed-source models and contributing to the broader development of accessible and high-performing AI systems for science.

Data availability

This study was carried out using publicly available data from MaScQA benchmark at **https://github.com/M3RG-IITD/MaScQA** related to the article: **https://pubs.rsc.org/en/content/articlelanding/2024/dd/d3dd00188a**. The code used for this study can be found in the repository LLM_comparison_4MS at **https://github.com/Lambard-ML-Team/LLM_comparison_4MS**. In this repository you can find the code used to submit the questions to the different models (**https://github.com/Lambard-ML-Team/LLM_comparison_4MS/tree/main/questions_to_models**) and the code used for the GPT-4o analysis of the response (**https://github.com/Lambard-ML-Team/LLM_comparison_4MS/tree/main/GPT_4o_Analysis**). Data for this paper, including the answer for each model and the result of the analysis by GPT-4o are also available at **https://github.com/Lambard-ML-Team/LLM_comparison_4MS**.

Author contributions

Christophe Bajan: conceptualization, methodology, software, data analysis, writing—original draft. Guillaume Lambard: conceptualization, methodology, software, validation, resources, supervision, funding acquisition, project administration, writing—final draft.

Conflicts of interest

There are no conflicts to declare.

Notes and references

1 M. Zaki, N. A. Krishnan, *et al.*, *Digital Discovery*, 2024, **3**, 313–327.
2 W. Lu, R. K. Luu and M. J. Buehler, *arXiv*, 2024, preprint, arXiv:2409.03444, pp. 1–56, DOI: **10.48550/arXiv.2409.03444**.
3 A. Trewartha, N. Walker, H. Huo, S. Lee, K. Cruse, J. Dagdelen, A. Dunn, K. A. Persson, G. Ceder and A. Jain, *Patterns*, 2022, **3**, 100488.
4 V. Tshitoyan, J. Dagdelen, L. Weston, A. Dunn, Z. Rong, O. Kononova, K. A. Persson, G. Ceder and A. Jain, *Nature*, 2019, **571**, 95–98.
5 Y. An, J. Greenberg, A. Kalinowski, X. Zhao, X. Hu, F. J. Uribe-Romo, K. Langlois, J. Furst and D. A. Gómez-Gualdrón, *arXiv*, 2024, preprint, arXiv:2309.11361, pp. 1–14, DOI: **10.48550/arXiv.2309.11361**.
6 H. Zhang, Y. Song, Z. Hou, S. Miret and B. Liu, *Findings of the Association for Computational Linguistics: EMNLP 2024*, Miami, Florida, USA, 2024, pp. 3369–3382.
7 Y. Wan, Y. Liu, A. Ajith, C. Grazian, B. Hoex, W. Zhang, C. Kit, T. Xie and I. Foster, *arXiv*, 2024, preprint, arXiv:2405.09939, pp. 1–22, DOI: **10.48550/arXiv.2405.09939**.
8 T. Guo, K. Guo, B. Nan, Z. Liang, Z. Guo, N. V. Chawla, O. Wiest and X. Zhang, *arXiv*, 2023, preprint, arXiv:2305.18365, pp. 1–27, DOI: **10.48550/arXiv.2305.18365**.
9 K. Singhal, S. Azizi, T. Tu, S. S. Mahdavi, J. Wei, H. W. Chung, N. Scales, A. Tanwani, H. Cole-Lewis, S. Pfohl, P. Payne, M. Seneviratne, P. Gamble, C. Kelly, A. Babiker, N. Schärli, A. Chowdhery, P. Mansfield, D. Demner-Fushman, B. Agüera y Arcas, D. Webster, G. S. Corrado, Y. Matias, K. Chou, J. Gottweis, N. Tomasev, Y. Liu, A. Rajkomar, J. Barral, C. Semturs, A. Karthikesalingam and V. Natarajan, *Nature*, 2023, **620**, 172–180.
10 L. Sun, Y. Han, Z. Zhao, D. Ma, Z. Shen, B. Chen, L. Chen and K. Yu, *arXiv*, 2023, preprint, arXiv:2308.13149, pp. 1–9, DOI: **10.48550/arXiv.2308.13149**.
11 J. Welbl, N. F. Liu and M. Gardner, *arXiv*, 2017, preprint, arXiv:1707.06209, pp. 1–13, DOI: **10.48550/arXiv.1707.06209**.
12 D. Zhang, W. Liu, Q. Tan, J. Chen, H. Yan, Y. Yan, J. Li, W. Huang, X. Yue, D. Zhou, S. Zhang, M. Su, H.-S. Zhong, Y. Li and W. Ouyang, *arXiv*, 2024, preprint, arXiv:2402.06852, pp. 1–26, DOI: **10.48550/arXiv.2402.06852**.
13 X. Lu, H. Cao, Z. Liu, S. Bai, L. Chen, Y. Yao, H.-T. Zheng and Y. Li, *arXiv*, 2024, preprint, arXiv:2403.08192, pp. 1–19, DOI: **10.48550/arXiv.2403.08192**.
14 Z.-Y. Chen, F.-K. Xie, M. Wan, Y. Yuan, M. Liu, Z.-G. Wang, S. Meng and Y.-G. Wang, *Chin. Phys. B*, 2023, **32**, 118104.
15 T. Xie, Y. Wan, W. Huang, Z. Yin, Y. Liu, S. Wang, Q. Linghu, C. Kit, C. Grazian, W. Zhang, I. Razzak and B. Hoex, *arXiv*, 2023, preprint, arXiv:2308.13565, pp. 1–19, DOI: **10.48550/arXiv.2308.13565**.
16 Y. Chiang, C.-H. Chou and J. Riebesell, *arXiv*, 2024, preprint, arXiv:2401.17244, pp. 1–32, DOI: **10.48550/arXiv.2401.17244**.
17 Y. Song, S. Miret and B. Liu, *arXiv*, 2023, preprint, arXiv:2305.08264, pp. 1–17, DOI: **10.48550/arXiv.2305.08264**.
18 D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song and J. Steinhardt, *arXiv*, 2021, preprint, arXiv:2103.03874, pp. 1–22, DOI: **10.48550/arXiv.2103.03874**.
19 M. Courbariaux, Y. Bengio and J.-P. David, *ICLR (Workshop)*, 2015.
20 A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. de las Casas, E. Bou Hanna, F. Bressand, G. Lengyel, G. Bour, G. Lample, L. Renard Lavaud, L. Saulnier, M.-A. Lachaux, P. Stock, S. Subramanian, S. Yang, S. Antoniak, T. Le Scao, T. Gervet, T. Lavril, T. Wang, T. Lacroix and W. El Sayed, *arXiv*, 2024, preprint, arXiv:2401.04088, pp. 1–13, DOI: **10.48550/arXiv.2401.04088**.
21 C. Brugger, S. Weithoffer, C. De Schryver, U. Wasenmüller and N. Wehn, *Adv. Radio Sci.*, 2014, **12**, 75–81.
22 OpenAI, OpenAI API, 2023, **https://openai.com/api/**, accessed: May-Aug 2024.
23 Anthropic, Anthropic API, 2023, **https://www.anthropic.com**, accessed: May-Aug 2024.
24 Ollama, Ollama API, 2023, **https://ollama.ai**, accessed: May-Aug 2024.
25 L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing, H. Zhang, J. E. Gonzalez and I. Stoica, *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.