Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

MineralBench: an evaluation benchmark of large language models for mineral resources

Jiahao Dan
China University of Geosciences (Wuhan)
Shengwen Li
China University of Geosciences (Wuhan)
Hong Yao
yaohong@cug.edu.cn
China University of Geosciences (Wuhan)
Yuan Cong
Institute of Mineral Resources, Chinese Academy of Geological Sciences
Bingyi Li
China University of Geosciences (Wuhan)
Hang Zhou
China University of Geosciences (Wuhan)

Research Article

Keywords: large language model, mineral resources domain, evaluation benchmark, multidimensional evaluation, domain adaptation

Posted Date: September 25th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10717678/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: Competing interest reported. The authors disclose that patent applications and software copyrights related to the broader research project have been filed or registered as part of the project requirements. No known commercial interests or financial benefits are associated with these intellectual property activities. The authors declare no other competing interests.

MineralBench: an evaluation benchmark of large language models for mineral resources

Danjiahao[1], Lishengwen[1], Yaohong[1*], Congyuan[2], Libingyi[1], Zhouhang[1]

[1]School of Computer Science, China University of Geosciences (Wuhan), Wuhan 430074, China

[2] Institute of Mineral Resources, Chinese Academy of Geological Sciences, Beijing 100037, China

[*]Corresponding author: Yaohong

E-mail: yaohong@cug.edu.cn

Acknowledgments

The authors acknowledge the support of the Deep Earth Probe and Mineral Resources Exploration National Science and Technology Major Project (Grant No. 2025ZD1008204).

Abstract: Large language models (LLMs) have demonstrated tremendous potential in various tasks related to mineral resources, but their performance varies significantly in these tasks. However, existing LLM evaluation studies in the mineral resources domain remain limited and do not provide a sufficiently comprehensive basis for model comparison and selection. To this end, this paper develops MineralBench, an evaluation benchmark for LLMs for mineral resources, which involves the systematic design and refinement of its evaluation tasks, data, and metrics. The benchmark features three tasks, namely Mineral Term Understanding, Mineral Property Identification, and Geological Entity Extraction, and defines four evaluation metrics, including Accuracy (Acc), Inter-round Consistency (IC), General Stability Score (GS), and Efficiency (Eff). Additionally, an automated evaluation tool that supports multi-model and multi-task evaluation is developed. Furthermore, this study conducts a unified evaluation of 13 LLMs accessed via online APIs or deployed in local environments. The experimental results show that the LLMs generally perform better on general tasks than on mineral resource tasks, and that LLMs exhibit substantial differences across multiple evaluation metrics, with considerable complementarity among them. Further domain adaptation experiments suggest that the effects of LoRA fine-tuning may be task-dependent, with performance improving on some tasks while decreasing on others. MineralBench provides data and methodological references for the evaluation and selection of LLMs in the mineral resources domain.

Keywords: large language model; mineral resources domain; evaluation benchmark; multidimensional evaluation; domain adaptation

1. Introduction

In the mineral resources domain, a vast amount of geological and mineral-related data has been accumulated over a long period from various sources. This data forms the basis for mineral resource exploration and analysis, as well as decision-making. In practice, extracting and applying this data relies heavily on manual processing, organization and analysis, and faces significant challenges such as low processing efficiency and high labor costs, particularly when dealing with large volumes of materials (Enkhsaikhan et al. 2021a; Lawley et al. 2023). In recent years, LLMs have demonstrated strong capabilities in natural language understanding and information extraction, providing a new technical approach for the automated processing of these materials (Xu et al. 2024). For example, some studies have applied LLMs to knowledge extraction and knowledge graph construction from geological reports (Ge et al. 2026), and explored domain-specific language models and specialized question-answering applications for geological and mineral surveys (Fu et al. 2025). LLMs demonstrate tremendous application potential and a wide range of applicable scenarios in the mineral resources domain.

Recent studies have revealed substantial differences in the task accuracy, output stability, and inference efficiency of LLMs across various tasks (Liang et al. 2023; Gjorgjevikj et al. 2025). Researchers have developed various evaluation benchmarks to identify differences among LLMs. For example, general and scientific evaluation benchmarks such as MMLU (Hendrycks et al. 2021a) and SciBench (Wang et al. 2024) cover a broad range of disciplinary knowledge and scientific problems; GeoBench (Deng et al. 2024), EarthSE (Xu et al. 2026), OceanBench (Bi et al. 2024) and ClimaQA (Manivannan et al. 2025) further focus on specialized evaluation in Earth science and its subdomains; MineBench (Yu et al. 2024) targets mineral exploration scenarios and mainly evaluates the ability of multimodal LLMs to understand and reason about remote-sensing information. Although existing evaluation benchmarks have been extended to Earth science and some mineral exploration scenarios, they cover a limited range of tasks and evaluation dimensions for mineral resource tasks. Consequently, these benchmarks remain insufficient to provide a solid basis for analyzing the performance of different LLMs and guiding model selection for various mineral resource tasks.

To address the above research gap, this paper develops MineralBench, an evaluation benchmark for evaluating LLMs in the mineral resources domain, which establishes a corresponding evaluation system comprising evaluation tasks, data, and evaluation metrics. The benchmark comprises three domain tasks, namely mineral term understanding, mineral property identification, and geological entity extraction. It introduces four evaluation metrics, namely Accuracy (Acc), Inter-round Consistency (IC), General Stability Score (GS), and Efficiency (Eff). It also develops a modular automated evaluation tool that supports multi-model and multi-task evaluation. In addition, this study conducts unified evaluation experiments on 13 LLMs accessed via online APIs or deployed in local environments, together with supplementary task-oriented domain adaptation experiments to explore possible performance changes after fine-tuning.

The main contributions of this paper are as follows:

(1) This paper develops MineralBench—an evaluation benchmark for LLMs in the mineral resources domain—which covers knowledge understanding and basic information processing tasks, establishes multidimensional metrics and an automated evaluation tool, and thus provides foundational data and methodological support for LLM selection in this field. (2) This study conducted a unified evaluation of 13 LLMs accessed via online APIs or deployed in local environments, together with task-oriented domain adaptation experiments. It analyzed differences in task performance across models and performance of LLM adaptation, providing references for capability analysis, evaluation, and model selection of LLMs in the mineral resources domain.

2. Related work

Existing studies on large language model evaluation cover general and domain-specific evaluation. This section reviews general and scientific capability evaluation, as well as Earth science evaluation related to the mineral resources domain.

2.1 Evaluation benchmarks in general domains

Existing studies have developed evaluation benchmarks for interdisciplinary knowledge, scientific problem solving, and capabilities in specific tasks. For example, MMLU (Hendrycks et al. 2021a) covers the humanities, social sciences, natural sciences, and multiple professional fields, mainly evaluating the breadth of models' knowledge and their problem-solving capabilities across different disciplines.

Compared with such comprehensive interdisciplinary evaluations, scientific capability evaluation benchmarks place greater emphasis on models' scientific problem-solving capabilities. For example, SciBench (Wang et al. 2024) evaluates models using complex problems in mathematics, physics, and chemistry, with a focus on scientific problem-solving capability; ARC-Challenge (Clark et al. 2018) and MATH (Hendrycks et al. 2021b) respectively focus on basic science question answering and mathematical problem solving, evaluating models' reasoning capabilities in specific tasks. In terms of evaluation scope, existing benchmarks show a trend of gradual refinement from comprehensive interdisciplinary capabilities to scientific problem solving and capabilities in specific tasks.

These studies demonstrate that different evaluation benchmarks typically involve specific tasks that target particular knowledge areas and capabilities. However, these evaluation tasks do not specifically focus on Earth science applications.

2.2 Evaluation benchmarks in the earth science domains

To fully evaluate the capabilities of LLMs in Earth science, some studies have developed several evaluation benchmarks for comprehensive Earth science. For example, GeoBench in K2 (Deng et al. 2024) covers knowledge in geology, geography, environmental science, and other areas, mainly

evaluating LLMs' ability to understand and apply geoscience knowledge. EarthSE (Xu et al. 2026) sets multiple types of tasks for comprehensive Earth science capabilities, further evaluating LLMs' ability to understand and analyze geoscience problems. Compared with such comprehensive Earth science evaluations, subdomain benchmarks focus more on model performance in specific disciplinary knowledge and tasks. For example, OceanBench (Bi et al. 2024) and ClimaQA (Manivannan et al. 2025) focus on ocean science and climate science, respectively, and evaluate LLMs' ability to understand and apply relevant specialized knowledge. In mineral-related fields, MineBench proposed by MineAgent (Yu et al. 2024) targets remote-sensing mineral exploration scenarios and mainly evaluates the ability of multimodal LLMs to understand visual information such as hyperspectral imagery. Overall, existing Earth science evaluations have covered comprehensive geoscience knowledge, subdomains such as ocean and climate science, and mineral exploration scenarios.

However, existing mineral-related evaluations have not yet sufficiently covered tasks in the mineral resources domain, and therefore remain insufficient to provide an adequate basis for comparing LLM performance and selecting LLMs for these tasks. To address this, this paper develops MineralBench, an evaluation benchmark of LLMs for mineral resources.

3. Method

3.1 Task design

MineralBench selected three tasks that adhere to the principles of prioritizing traceable sources, clearly defined reference answers and suitability for structured processing. These tasks also involve different information processing operations: mineral term understanding, mineral property identification and geological entity extraction. The input formats and target outputs for these tasks are defined as follows:

(1) The mineral term understanding task evaluates the model's ability to identify the corresponding term from a specialized concept description. The task is represented as:

$$D_i \longrightarrow T_i$$

where D_i denotes the definition of the i-th term, and T_i denotes the corresponding reference term. The model returns a predicted term, which is scored by matching it against the reference answer.

(2) The mineral property identification task evaluates the model's ability to provide the corresponding property value from a mineral name and a target property. The task is represented as:

$$(M_i, P_i) \longrightarrow V_i$$

where M_i, P_i and V_i denote the mineral name, target property, and corresponding reference property value in the i-th sample, respectively. The model returns the property value in the specified format to support result parsing and automated scoring.

(3) The geological entity extraction task evaluates the model's ability to identify entities of specified categories from unstructured geological text. The task is represented as:

$$(S_i, \mathcal{C}_i) \longmapsto \mathcal{E}_i = \left\{C_{ij}: \left[e_{ij1}, e_{ij2}, \ldots, e_{ijk_j}\right]\right\}_{j=1}^{m_i}$$

where S_i denotes the source text of the i-th sample,$\mathcal{C}_i = \{C_{i1}, C_{i2}, \ldots, C_{im_i}\}$ denotes the set of entity categories to be extracted, $\mathcal{E}_i$ denotes the reference answer consisting of the entity categories and their corresponding entity lists, and e_{ijl} denotes the l-th entity in category C_{ij}. The model output is organized according to a predefined structure to support entity-set parsing and automated scoring.

The three tasks aim to investigate the model's capabilities in term understanding, property identification, and entity extraction, reflecting performance differences among LLMs in the mineral resources domain.

3.2 Dataset construction

MineralBench uses the Mindat.org Glossary of Mineralogical Terms, the Mindat.org Mineral Species database (Ralph et al. 2025), and the open-source geological entity annotation dataset OzROCK (Enkhsaikhan et al. 2021b) as data sources for constructing the mineral term understanding, mineral property identification, and geological entity extraction, respectively. In addition, three general tasks, MATH (Hendrycks et al. 2021b), MMLU-Sci (Hendrycks et al. 2021a), and ARC-Challenge (Clark et al. 2018) are selected to evaluate mathematical problem solving, scientific knowledge understanding, and basic science reasoning, respectively.

MineralBench organizes all evaluation samples in the JSONL format. It stores each sample as an independent line, facilitating sample-by-sample loading. Although the specific fields of samples vary across tasks, each of them contains the sample identifier (ID), question type, task input, and reference answer, and additional information such as the data source, original record identifier, subject, or mineral name. The i-th sample can be abstractly represented as:

$$\mathcal{I}_i = (\mathrm{id}_i, \mathrm{type}_i, \mathcal{X}_i, \mathrm{answer}_i, \mathcal{A}_i)$$

where id_i denotes the sample identifier, type_i the question type,$\mathcal{X}_i$ the task input, and answer_i the reference answer, while $\mathcal{A}_i$ contains optional additional information. For multiple-choice samples $\mathcal{X}_i$ includes the question and options; for short-answer samples, it contains the question text; and for geological entity extraction samples, it contains the source text and target entity categories. Their reference answers are represented as option labels, short text, or structured JSON strings.

For the mineral resource tasks, evaluation samples are constructed from the original data following the above task categories. Mineral-Term uses term definitions as inputs and the corresponding terms as reference answers; Mineral-Property uses mineral names and target properties with valid values as inputs and the corresponding property values as reference answers; Geo-NER uses geological texts and target entity categories as inputs, with annotated entities organized by category into structured answers. The constructed samples are then mapped to a unified format for subsequent use by the evaluation program. After invalid, duplicate, and anomalous samples are removed, the final evaluation samples are randomly selected from the candidate samples of each task to reduce the influence of manual selection on sample composition. For the general tasks, fields such as the original questions, options, and answers are directly

mapped to unified format according to the corresponding question types, and samples are selected based on the characteristics of each task. ARC-Challenge uses random sampling; MATH uses stratified sampling by problem difficulty and type, with emphasis on Level 1–3 problems; MMLU-Sci selects questions related to physics and chemistry. These procedures retain the task content needed to assess knowledge and reasoning while controlling the overall evaluation scale.

Algorithm 1 details the construction and validation procedure from the original data to the final evaluation dataset. For the domain tasks, task samples are first constructed from the original data according to the task definitions; for the general tasks, the original data fields are mapped according to the corresponding question types. For each task, a complete set of evaluation samples $\mathcal{D}_t^{\text{full}}$ is first formed, from which the final evaluation dataset $\mathcal{D}_t$ is then selected.

Algorithm 1. MineralBench evaluation dataset construction and validation

Input: Task sets $\mathcal{T}_{\text{core}}$ and $\mathcal{T}_{\text{ref}}$, and raw records $\mathcal{R}_t$ for each task t.
Output: Evaluation dataset $\mathcal{D}_t$ for each task t.

for all $t \in \mathcal{T}_{\text{core}} \cup \mathcal{T}_{\text{ref}}$ **do**
  $\mathcal{D}_t^{\text{full}} \leftarrow \varnothing$
  if $t \in \mathcal{T}_{\text{core}}$ **then**
    for all $r \in \mathcal{R}_t$ **do**
      for all $s \in \text{Construct}(r, t)$ **do**
        if $\text{IsValid}(s)$ **then**
          $\mathcal{I} \leftarrow \text{MapToRecord}(s, t)$
          $\mathcal{D}_t^{\text{full}} \leftarrow \mathcal{D}_t^{\text{full}} \cup \{\mathcal{I}\}$
        end if
      end for
    end for
  else
    for all $r \in \mathcal{R}_t$ **do**
      $\mathcal{I} \leftarrow \text{MapToRecord}(r, t)$
      if $\text{IsValid}(\mathcal{I})$ **then**
        $\mathcal{D}_t^{\text{full}} \leftarrow \mathcal{D}_t^{\text{full}} \cup \{\mathcal{I}\}$
      end if
    end for
  end if
  $\mathcal{D}_t \leftarrow \text{Select}(\mathcal{D}_t^{\text{full}}, t)$
  $\mathcal{D}_t \leftarrow \text{Validate}(\mathcal{D}_t)$
  $\mathcal{D}_t \leftarrow \text{SpotCheck}(\mathcal{D}_t)$
  $\text{WriteJSONL}(\mathcal{D}_t)$
end for
return $\{\mathcal{D}_t \mid t \in \mathcal{T}_{\text{core}} \cup \mathcal{T}_{\text{ref}}\}$

During data validation, the completeness of required fields, the validity of reference answers and sample identifiers, and the formats of fields and text are checked. Obvious duplicate or anomalous samples are also removed or corrected through manual spot checks. The resulting evaluation dataset is organized in a unified JSONL format. MineralBench contains six evaluation tasks and 1,426 evaluation samples, including 450 samples from the three general tasks and 976 samples from the three domain tasks. The data sources, evaluation content, and final sample counts for each task are shown in Table 1.

Table 1. Statistics of the MineralBench evaluation dataset

Task Category	Dataset	Source	Task	Samples
General Reference Tasks	General Foundation	Math	Math Problem Solving	100
		MMLU-Sci	Science Knowledge	200
		ARC-Challenge	Science Reasoning	150
Core Domain Tasks	Mineral-Term	Mindat.org Glossary	Term Matching	200
	Mineral-Property	Mindat.org Mineral	Property Query	576
	Geo-NER	OzROCK	Entity Extraction	200

Table 2 presents typical inputs and reference answers for the three domain tasks. Mineral-Term and Mineral-Property use short-text reference answers, while Geo-NER uses structured entity sets organized by the specified entity categories.

Table 2. Examples of MineralBench tasks in the mineral resources domain

Dataset	Question	Answer
Mineral-Term	Commonly known as pyrite disease, is the oxidation of iron sulfide and some other non-iron sulfide minerals	pyrite decay
Mineral-Property	For the mineral Actinolite, what is its element_list?	"O, Si, Mg, Ca, Fe, H"
Geo-NER	Text：The ranger river gold nls indee project is located further to the south within the mallina formation Entity：MINERAL, STRAT	{"MINERAL":["gold"], "STRAT":["mallina formation"]}

3.3 Definition of evaluation metrics

Multidimensional evaluation provides a broader view of LLM performance (Liang et al. 2023). According to the evaluation objectives of MineralBench, four metrics are derived, including Accuracy (Acc), Inter-round Consistency (IC), General Stability Score (GS), and Efficiency (Eff), which evaluate models in terms of task correctness, stability of repeated outputs, performance across tasks, and efficiency in actual use, respectively.

(1) Accuracy (Acc) measures the correctness of model outputs across evaluation tasks. Here, task-specific sample-level scoring methods are applied to fit diverse answer formats. Specifically, for multiple-choice questions and short-answer tasks, model outputs are normalized and matched against the reference answers, yielding a sample-level score of 0 or 1. For entity extraction tasks, model outputs are converted into entity sets, and the F1 score based on matches between predicted and annotated entities is

used as the sample-level score.

For the q-th question, after R repeated evaluations, accuracy Acc_q is defined as:

$$\mathrm{Acc}_q = \frac{1}{R}\sum_{r=1}^{R} s_r$$

where R denotes the number of repeated evaluations, and s_r denotes the model's score for the q-th question in the r-th evaluation, with $s_r \in [0,1]$. For multiple-choice and short-answer tasks, s_r takes a value of 0 or 1, while for entity extraction tasks, it takes the sample-level F1 score.

After averaging the score over multiple rounds for each sample, the task-level accuracy score Acc_t for task t is defined as:

$$\mathrm{Acc}_t = \left(\frac{1}{N_t}\sum_{q=1}^{N_t} \mathrm{r}Acc_q\right) \times 100$$

where N_t denotes the number of questions in the t-th task. This calculation yields task-level accuracy scores on a 0–100 scale, facilitating comparison of model performance across evaluation tasks.

(2) Inter-round Consistency (IC) measures the stability of model outputs across repeated evaluations under the same conditions. For the same sample, identical or similar outputs across repeated evaluations indicate higher inter-round consistency, whereas substantial variation indicates lower consistency.

For the q-th sample in a task, let $z_{q,r}$ denote the normalized LLM output in the r-th evaluation round, and R the number of repeated evaluations. The sample-level inter-round consistency is a weighted combination of the Exact Consistency Rate (ECR) and the Structural Consistency Rate (SCR), which are defined as:

$$\mathrm{ECR}_q = \frac{\max_z c\,(z)}{R}$$

$$\mathrm{SCR}_q = \frac{2}{R(R-1)}\sum_{1\le i<j\le R} \mathrm{sim}\left(z_{q,i}, z_{q,j}\right)$$

To quantify the lexical overlap between repeated outputs, $\mathrm{sim}(\cdot)$ is computed using the Jaccard similarity coefficient (Kosub 2019):

$$\mathrm{sim}\left(z_{q,i}, z_{q,j}\right) = \frac{\left|W\left(z_{q,i}\right) \cap W\left(z_{q,j}\right)\right|}{\left|W\left(z_{q,i}\right) \cup W\left(z_{q,j}\right)\right|}$$

where $\mathrm{W}(z)$ denotes the token set of a normalized output. The resulting similarity ranges from 0 to 1 and reflects token-level lexical overlap rather than semantic similarity.

$$\mathrm{IC}_q = \alpha_{\mathrm{IC}}\mathrm{ECR}_q + (1-\alpha_{\mathrm{IC}})\mathrm{SCR}_q$$

where $\mathrm{c(z)}$ denotes the number of times output z appears across repeated evaluations of sample q, and α_{IC} denotes the weighting parameter between the Exact Consistency Rate and the Structural Consistency Rate.

The Inter-round Consistency of a model on task t is defined as the average consistency score over all samples in that task:

$$\mathrm{IC}_t = \left(\frac{1}{N_t}\sum_{q=1}^{N_t} \mathrm{IC}_q\right) \times 100$$

where N_t denotes the number of samples in the t-th task. A higher IC_t indicates that the model produces more stable outputs when repeatedly performing the same task under the same conditions. Inter-round Consistency reflects output stability; however, it does not imply that the output is correct.

(3) General Stability Score (GS) measures the overall performance of a model across multiple evaluation tasks and the performance variation across tasks. Results from a single task can only reflect the model's capability on that specific task and cannot fully represent its overall performance across multiple tasks. Therefore, the General Stability Score is calculated from the model's task-level accuracy scores, considering both the average performance across tasks and the performance variation among tasks.

Let T denote the number of evaluation tasks, and let $\mathrm{Acc_t}$ denote the model's accuracy score on the t-th task. The model's mean performance across tasks is defined as:

$$\mu_m = \frac{1}{T}\sum_{t=1}^{T} \mathrm{Acc}_t$$

Performance variation across tasks is defined as:

$$\sigma_m = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(\mathrm{Acc}_t - \mu_m)^2}$$

where μ_m denotes the mean accuracy of model m across all evaluation tasks, and σ_m denotes its performance variation across tasks. On this basis, the General Stability Score of the model is defined as:

$$\mathrm{GS}_m = \mu_m - \alpha_{\mathrm{GS}}\sigma_m$$

where α_{GS} denotes the penalty coefficient for performance variation across tasks. A higher GS_m indicates higher average performance across multiple tasks with relatively smaller performance variation.

Because GS_m contains information on both average task performance and performance variation among tasks, an Auxiliary Stability Metric (ST) is additionally defined to reflect only cross-task performance variation. This avoids double counting with the accuracy metric in the auxiliary composite evaluation:

$$\mathrm{ST}_m = 100 - \beta_{\mathrm{ST}}\sigma_m$$

where β_{ST} denotes the stability penalty parameter. A higher ST_m indicates smaller performance variation across tasks. This metric is used only for calculating the Auxiliary Composite Score and does not replace the General Stability Score GS_m.

(4) Efficiency (Eff) measures the speed at which a model processes evaluation requests in actual access or deployment environments. During the evaluation, the number of successfully completed requests and the corresponding processing time are recorded, and throughput (requests per second, req/s) is used to quantify request-processing efficiency.

The Efficiency of the model on the t-th task is defined as:

$$\mathrm{Eff}_t = \frac{K_t}{\tau_t}$$

where $\mathrm{K_t}$ denotes the number of requests successfully completed by the model in the t-th task, and τ_t

denotes the cumulative time required to complete these requests.

The overall Efficiency of the model across all tasks is defined as:

$$\mathrm{Eff}_{\mathrm{m}} = \frac{\sum_{t=1}^{T} K_t}{\sum_{t=1}^{T} \tau_t}$$

where T denotes the number of evaluation tasks. This metric reflects the end-to-end request processing efficiency of the model under the current test environment. Its value is influenced not only by model size and the inference process, but also by operating conditions such as hardware configuration, inference framework, network transmission, and service response time.

(5) Based on the four evaluation dimensions described above, the task-level metrics are first aggregated at the model level to summarize model performance across tasks and evaluation dimensions. For a task-level evaluation metric X, the model-level weighted result is defined as:

$$X_{\mathrm{m}} = \frac{\sum_{t=1}^{T} w_t X_t}{\sum_{t=1}^{T} w_t}, \qquad X \in \{\mathrm{Acc}, \mathrm{IC}\}$$

where X_t denotes the task-level evaluation result of the model on the t -th task, and w_t denotes the weight of the t-th task. The specific weights for each task are given in the experimental settings.

A logarithmic ratio normalization based on the highest efficiency is applied to map Efficiency to a unified 0–100 scale and reduce the influence of extreme efficiency differences on the normalization results:

$$\mathrm{Eff}_{\mathrm{m}}^{\mathrm{norm}} = \frac{100}{1 + \log_{10}\left(\frac{\max_j(\mathrm{Eff}_{\mathrm{j}})}{\mathrm{Eff}_{\mathrm{m}}}\right)}$$

where $\max_j(\mathrm{Eff}_{\mathrm{j}})$ denotes the highest overall Efficiency among all evaluated models. The normalized Efficiency $\mathrm{Eff}_{\mathrm{m}}^{\mathrm{norm}}$ is expressed on a 0–100 scale, with the most efficient model receiving a normalized score of 100. The logarithmic transformation compresses large differences in raw throughput, thereby limiting the influence of extreme Efficiency values on the Auxiliary Composite Score. However, it does not eliminate the effects of access and deployment environments; therefore, the normalized Efficiency should be interpreted as a relative measure under the current experimental conditions rather than a hardware-independent measure of model efficiency.

An Auxiliary Composite Score is further constructed to summarize model performance across multiple evaluation dimensions. The score is calculated as a weighted combination of model-level Accuracy, model-level Inter-round Consistency, the Auxiliary Stability Metric, and normalized Efficiency. It provides a summary of overall performance under a given evaluation emphasis but is not intended as an absolute measure of overall model capability. The Auxiliary Composite Score is defined as:

$$\mathrm{Score}_{\mathrm{m}} = \frac{\lambda_{\mathrm{Acc}}\,\mathrm{Acc}_{\mathrm{m}} + \lambda_{\mathrm{IC}}\,\mathrm{IC}_{\mathrm{m}} + \lambda_{\mathrm{ST}}\,\mathrm{ST}_{\mathrm{m}} + \lambda_{\mathrm{Eff}}\,\mathrm{Eff}_{\mathrm{m}}^{\mathrm{norm}}}{\lambda_{\mathrm{Acc}} + \lambda_{\mathrm{IC}} + \lambda_{\mathrm{ST}} + \lambda_{\mathrm{Eff}}}$$

where $\lambda_{\mathrm{Acc}}, \lambda_{\mathrm{IC}}, \lambda_{\mathrm{ST}}$ and $\lambda_{\mathrm{Eff}f}$ denote the weights of model-level Accuracy, model-level Inter-round Consistency, the Auxiliary Stability Metric, and normalized Efficiency, respectively, in the Auxiliary Composite Score. The specific settings are given in Section 4.2.

(6) To estimate the empirical performance upper bound of the current candidate model pool under ideal question-level selection, this paper introduces Oracle analysis. Considering the variation across repeated evaluations, the scores from multiple evaluation rounds are first averaged for the same model on the same sample, and the model with the highest average score for that sample is then selected from the candidate model pool. For multiple-choice and short-answer tasks, the single-round score is 0 or 1; for the geological entity extraction task, it is the sample-level entity-set F1.

For the t-th task, $\mathrm{Oracle}_{\mathrm{t}}^{(\mathcal{P})}$ is defined as:

$$\mathrm{Oracle}_{\mathrm{t}}^{(\mathcal{P})} = \left[\frac{1}{N_t}\sum_{i=1}^{N_t}\max_{m\in\mathcal{P}}\left(\frac{1}{R}\sum_{r=1}^{R}s_{m,i,r}\right)\right]\times 100$$

where N_t denotes the number of samples in the t-th task, R denotes the number of repeated evaluations, and $s_{m,i,r}$ denotes the sample-level score of model m on sample i in the r-th evaluation. $\mathcal{P}$ denotes the candidate model pool, whose specific composition is given in the experimental section.

On this basis, the overall Oracle score across tasks is calculated using the same task weights as those used to aggregate Accuracy:

$$\mathrm{Oracle}^{(\mathcal{P})} = \frac{\sum_{t=1}^{T} w_t\ \mathrm{Oracle}_{\mathrm{t}}^{(\mathcal{P})}}{\sum_{t=1}^{T} w_t}$$

where $\mathrm{Oracle}^{(\mathcal{P})}$ denotes the overall Oracle score of candidate model pool $\mathcal{P}$ across all tasks,$\mathrm{Oracle}_{\mathrm{t}}^{(\mathcal{P})}$ denotes the task-level Oracle score of candidate model pool $\mathcal{P}$ on the t-th task, and w_t denotes the weight of the t-th task. Oracle reflects the empirical performance upper bound of the candidate model pool under post hoc ideal question-level selection based on the current evaluation results.

3.4 Automated evaluation framework

The framework integrates the evaluation tasks, datasets, and methods described above into an automated process. As shown in Fig.1, the framework consists of three modules: Setup, Executor, and Evaluator, which are responsible for evaluation resource preparation, model inference, and result computation, respectively. The Setup module centrally manages the evaluation configuration, evaluation datasets, prompt templates, and model pool. The evaluation data follow the JSONL format defined in Section 3.2; the prompt templates specify the input organization, answer requirements, and output formats for different tasks; and the model pool is used to manage the candidate models participating in the evaluation and their invocation configurations.

The Executor module is responsible for Prompt Construction, Model Inference, and Output Collection. For different tasks, the tool combines the System Prompt, Task Instruction, and the task input field $\mathcal{X}_i$ of the current sample $\mathcal{I}_i$ using the corresponding template to construct the complete Model Input. For example, for multiple-choice questions, the question and candidate options are filled into the corresponding fields of the template, while the answer format and other response requirements are specified consistently in the prompt template. After the model input is constructed, the tool performs inference via online APIs or local deployment with vLLM (Kwon et al. 2023), depending on the model

access or deployment mode, and saves the model outputs and run logs separately. For multi-round evaluation, the inference results from each round are recorded independently.

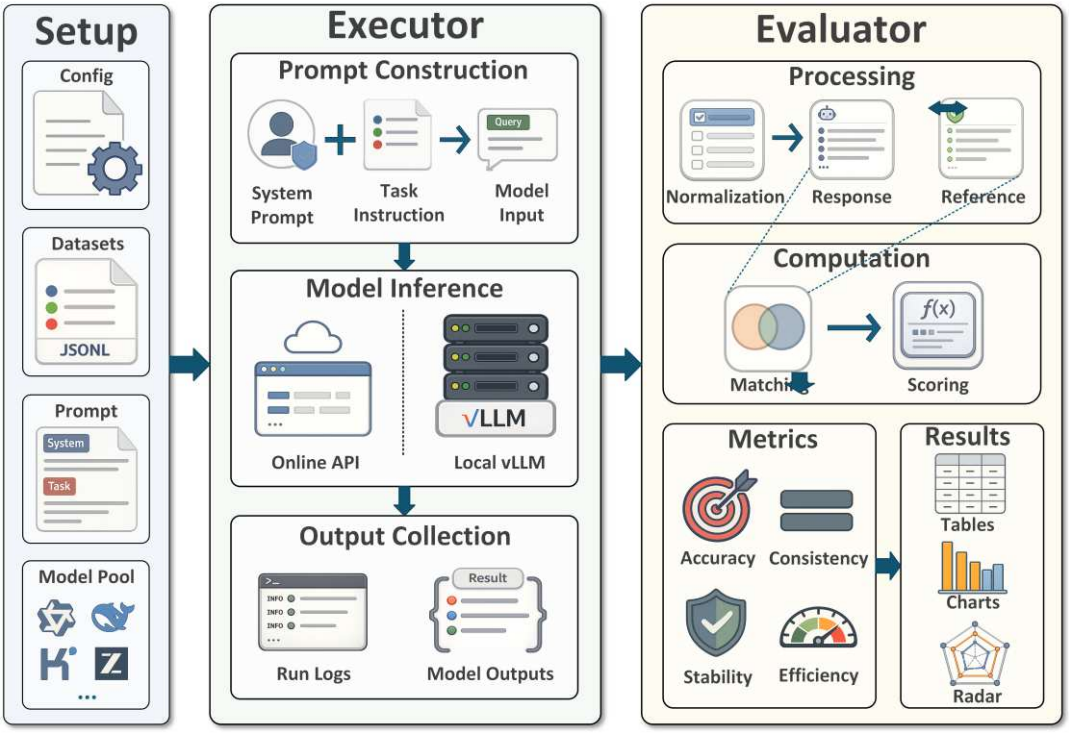


Fig.1 Evaluation framework

The Evaluator module performs output processing, score computation, metric calculation, and result generation in sequence. During output processing, the Evaluator applies bounded surface normalization to remove formatting variations that do not alter the intended answer, such as character-form differences, outer formatting markers, predefined answer prefixes, and redundant whitespace. The normalization process standardizes only answer representations and does not search explanatory text for answers or perform semantic correction, inference, or completion. The normalized outputs are then processed using predefined task-specific parsers and matched against the corresponding reference answers according to the scoring methods defined in Section 3.3. Outputs that cannot be validly parsed under these rules are scored as incorrect. The same parsing and scoring rules are applied consistently across all models and evaluation rounds. Based on the sample-level scores and inference records, Accuracy, Inter-round Consistency, General Stability Score, and Efficiency are further calculated, and the evaluation results are presented in tables, charts, and radar plots.

4. Experiments

This section covers the models selected in the experiments and the experimental settings, the overall results across six tasks, model differences across four evaluation dimensions—Accuracy, Inter-round Consistency, General Stability Score, and Efficiency—and the performance changes of LLMs before and

after fine-tuning.

4.1 LLM selection

As listed in Table 3, the selected 13 LLMs are sourced from four mainstream open-source LLM families—Qwen, DeepSeek, GLM, and Kimi—to assess model performance and MineralBench's ability to distinguish among models with different capabilities. They span multiple parameter scales to compare performance across different model sizes. The evaluation includes both online models and local models to reflect differences in computing resources and inference environments in practical applications. For clarity, LLMs accessed via online APIs are hereafter referred to as online models, while LLMs deployed in local environments are referred to as local models. Specifically, the online models include eight larger LLMs, whereas the local models include five smaller AWQ-quantized LLMs (Lin et al. 2024).

The online models are accessed through the Alibaba Cloud BaiLian platform (Alibaba Cloud n.d.), while the local models use publicly available weights from the ModelScope community (ModelScope n.d.).

Table 3. List of selected LLMs

Type	Model	Total Parameters
Online	Qwen3-max (Yang et al. 2025)	>1T
	Qwen3.5-35B-A3B	35B
	GLM-5 (GLM-5 Team et al. 2026)	744B
	GLM-4.5-Air (GLM-4.5 Team et al. 2025)	106B
	DeepSeek V4 Pro	1.6T
	DeepSeek V4 Flash	284B
	DeepSeek V3.2 (DeepSeek-AI et al. 2025b)	671B
	Kimi K2 Instruct (Kimi Team et al. 2025)	1T
Local	Qwen3-32B-AWQ (Yang et al. 2025)	32.8B
	Qwen3.5-27B-AWQ	27B
	Qwen3-14B-AWQ (Yang et al. 2025)	14.8B
	DeepSeek-R1-32B-AWQ (DeepSeek-AI et al. 2025a)	32B
	DeepSeek-R1-14B-AWQ (DeepSeek-AI et al. 2025a)	14B

4.2 Experimental settings

Table 4. Inference settings

Category	Parameter	Setting
Environment	GPU	NVIDIA A100-SXM4-80GB
	Inference backend	vLLM
Context	Maximum context length	4096 tokens

Decoding	Sampling strategy	Stochastic sampling
	Temperature	0.7
	Top-p	0.95
	Top-k	50
	Repetition penalty	1.1
Generation	Maximum generation length	2048 tokens
Failure handling	Maximum retries	2

All locally deployed models run in the same hardware and software environment to reduce variation caused by differences in computing and inference conditions. The experimental platform uses an NVIDIA A100-SXM4-80GB GPU and vLLM for inference. The maximum context length is 4096 tokens for all locally deployed LLMs, covering the input and output requirements of the evaluation tasks. Local inference uses stochastic sampling with a temperature of 0.7, top-p of 0.95, top-k of 50, a repetition penalty of 1.1, and a maximum generation length of 2048 tokens. Table 4 summarizes the local inference settings. The online models are accessed via APIs, for which model-specific inference configurations are not fully controllable or uniformly disclosed. Therefore, the evaluation does not assume identical inference configurations between online and locally deployed models. Instead, both groups use the same evaluation data, task order, prompt templates, output requirements, and scoring rules. Cross-group comparisons thus reflect model performance under their respective access and deployment conditions rather than under fully identical inference environments.

Table 5. Task weight settings

Task	General Reference Tasks			Core Domain Tasks		
Dataset	ARC-Challenge	Math	MMLU-Sci	Mineral-Term	Mineral-Property	Geo-NER
Symbol	w_1	w_2	w_3	w_4	w_5	w_6
Weight	1	1	1	2	2	2

Table 6. Composite score metric weight settings

Metric	Acc_m	IC_m	ST_m	Eff_m
Symbol	$\lambda_{\mathrm{A}cc}$	λ_{IC}	λ_{ST}	$\lambda_{\mathrm{E}ff}$
Weight	2	1	1	1

The experiments follow the procedure described in Section 3.4, with each LLM completing all evaluation tasks sequentially. Scoring follows the methods defined in Section 3.3. The number of repeated evaluations is set to $\mathrm{R} = 3$,balancing the reliability of Inter-round Consistency estimation and evaluation cost. The Inter-round Consistency parameter $\alpha_{\mathrm{IC}} = 0.5$ balances the Exact Consistency Rate and the Structural Consistency Rate. The General Stability Score uses $\alpha_{\mathrm{GS}} = 0.5$, while the Auxiliary Stability Metric uses $\beta_{\mathrm{ST}} = 2.0$ to control the penalty for performance variation across tasks. Geological entity extraction uses question-level aggregation, response time uses seconds, and Efficiency is calculated from global throughput across all tasks, while normalized Efficiency follows the logarithmic

ratio normalization defined in Section 3.3. The task weights for Accuracy and Inter-round Consistency are listed in Table 5. General tasks have a weight of 1, while domain tasks have a weight of 2 to emphasize performance on domain tasks. The metric weights for the composite score are listed in Table 6. Weighted Accuracy has a weight of 2, making task correctness the primary component of the composite evaluation; Inter-round Consistency, Auxiliary Stability Metric, and Efficiency each have a weight of 1, supplementing the evaluation with output repeatability, cross-task performance stability, and operating performance. In the Oracle analysis, $\mathcal{P} \in \{\text{Local}, \text{Online}, \text{All Models}\}$.

4.3 Overall results

Table 7 reports the Accuracy scores of the 13 LLMs across the six tasks, their Weighted Accuracy scores, and the Oracle results across different candidate LLM pools. Fig.2 shows the capability distribution of the evaluated LLMs across the six tasks. Table 8 further reports the model-level evaluation metrics and the Auxiliary Composite Score, while Fig.3 shows LLM performance across the four core dimensions of Accuracy, Inter-round Consistency, General Stability Score, and Efficiency. This section analyzes the overall results from three perspectives: task-level Accuracy distributions, the empirical performance upper bound estimated by Oracle, and multidimensional model performance. These analyses characterize differences in LLM capabilities, complementarity among candidate LLMs, and the contributions of different evaluation metrics.

Table 7. Task Accuracy results

Type	Model/Pool	ARC-Challenge	MMLU-Sci	Math	Geo-NER	Mineral-Property	Mineral-Term	Weighted-Acc
Online	Qwen3-max	97.78	80.17	70.67	**69.64**	65.34	24.67	63.10
	Qwen3.5-35B-A3B	96.00	76.00	49.00	17.45	64.12	17.00	46.46
	GLM-5	**98.22**	79.83	64.00	58.92	74.82	30.33	63.35
	GLM-4.5-Air	92.00	63.50	58.33	42.41	56.08	18.67	49.79
	DeepSeek V4 Pro	96.45	**87.17**	**76.33**	66.59	**80.55**	**31.00**	**68.47**
	DeepSeek V4 Flash	97.55	81.67	66.67	69.06	75.00	22.17	64.26
	DeepSeek V3.2	97.56	67.33	76.00	67.23	74.77	19.67	62.69
	Kimi K2 Instruct	96.00	72.00	67.00	48.99	73.21	20.33	57.78
Local	Qwen3-32B-AWQ	92.22	70.50	53.67	36.23	28.01	5.17	39.46
	Qwen3.5-27B-AWQ	93.33	64.33	15.00	34.65	20.83	3.33	32.25
	Qwen3-14B-AWQ	47.11	40.83	58.33	16.08	25.06	5.17	26.54
	DeepSeek-R1-32B-AWQ	81.78	63.33	75.67	25.36	40.04	11.17	41.54
	DeepSeek-R1-14B-AWQ	84.44	67.83	63.00	17.70	31.54	13.33	37.82
Oracle	Local	99.33	86.50	87.67	54.86	59.55	18.67	59.96
	Online	100.00	93.67	97.67	81.91	89.93	45.50	80.66
	All Models	100.00	94.50	97.67	82.57	90.10	46.83	81.24

Table 7 shows that different LLMs exhibit performance differences across the six tasks. The evaluated LLMs generally achieve relatively high scores on ARC-Challenge, while their performance on the Mineral Term Understanding task remains relatively low. The Geological Entity Extraction and Mineral Property Identification tasks also reveal substantial performance differences among LLMs.

DeepSeek V4 Pro achieves the highest Weighted Accuracy score of 68.47, but no single LLM achieves the highest score across all individual tasks. For example, GLM-5 reaches 98.22 on ARC-Challenge, while Qwen3-max reaches 69.64 on the Geological Entity Extraction task. Fig.2 further visualizes the task capability profiles of different LLMs. In particular, across the three core domain tasks, the LLMs show clear differences in their strengths and weaknesses. These results indicate that LLM strengths are task-dependent and that the three core domain tasks provide additional information for distinguishing among models beyond the general tasks.

The Oracle analysis compares three candidate model pools—Local, Online, and All Models—to estimate their performance upper bounds under ideal question-level selection and examine question-level complementarity among models. Their Oracle scores are 59.96, 80.66, and 81.24, respectively, each exceeding the best single-model score in the corresponding pool. The All Models Oracle score exceeds the best single-model score by 12.77 points, indicating that different models perform best on partially different samples and therefore exhibit complementary capabilities. However, it exceeds the Online Oracle score by only 0.58 points, suggesting that the online model pool covers most questions handled well by the smaller local models and that MineralBench contains relatively few cases of such performance reversals, where smaller models outperform all larger online candidates. Oracle represents an empirical performance upper bound under post hoc ideal question-level selection rather than the performance achievable by an actual routing system. The results suggest that task- or question-aware model selection and routing may better exploit complementary model capabilities, although their practical benefits require validation through independent routing experiments.

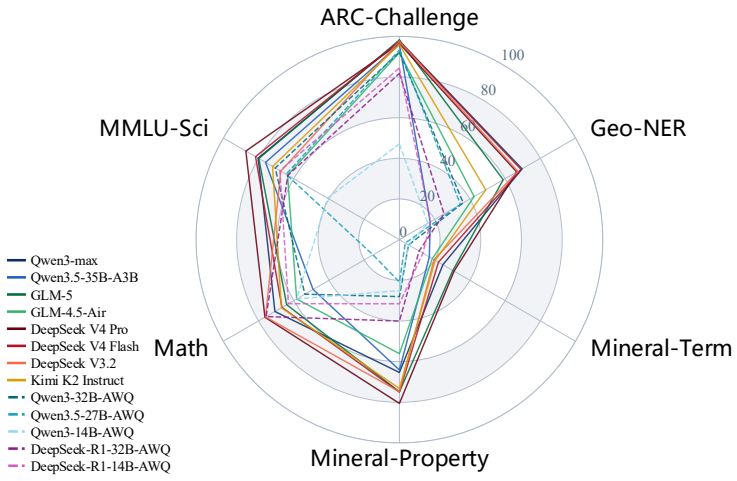


Fig.2 Accuracy distribution across the six tasks

Table 8 shows the model rankings vary across evaluation metrics. DeepSeek V4 Pro achieves the highest Weighted Accuracy and General Stability Score, at 68.47 and 62.55, respectively; DeepSeek V4 Flash achieves the highest Inter-round Consistency and Auxiliary Composite Score, at 93.05 and 71.97, respectively; and GLM-4.5-Air ranks first in Efficiency among the evaluated models. Fig.3 further shows distinct model strengths across the four core evaluation dimensions, with no single model leading in all four. These results indicate that Accuracy, Inter-round Consistency, General Stability Score, and Efficiency capture complementary aspects of model performance, making single-metric evaluation

insufficient for overall comparison. This finding is consistent with the multi-metric approach adopted in previous multidimensional LLM evaluation (Liang et al. 2023). Among these metrics, Efficiency requires particular caution in cross-model comparison because it mainly reflects end-to-end processing performance under the current access or deployment conditions rather than intrinsic inference efficiency, especially when online and local models operate under different conditions. Logarithmic normalization reduces the impact of large throughput differences on the Auxiliary Composite Score but cannot eliminate environment-related effects. Therefore, raw Efficiency is more comparable within the same deployment group, while the Auxiliary Composite Score serves only as a unified reference under the current experimental conditions. Model comparison and selection should therefore consider target tasks, output stability, and deployment efficiency rather than relying on Accuracy or the Auxiliary Composite Score alone.

Table 8. Overall evaluation results

Type	Model	Acc_m	IC_m	GS_m	ST_m	Eff_m^{norm}	Score
Online	Qwen3-max	63.10	89.65	56.99	55.8	60.94	66.52
	Qwen3.5-35B-A3B	46.46	83.90	38.71	41.82	66.31	56.99
	GLM-5	63.35	83.85	<u>57.24</u>	**58.23**	54.52	64.66
	GLM-4.5-Air	49.79	91.53	44.11	55.78	**100.00**	69.38
	DeepSeek V4 Pro	**68.47**	<u>92.21</u>	**62.55**	<u>58.17</u>	62.17	<u>69.90</u>
	DeepSeek V4 Flash	<u>64.26</u>	**93.05**	57.12	53.74	<u>84.57</u>	**71.97**
	DeepSeek V3.2	62.69	87.50	55.33	52.98	54.45	64.06
	Kimi K2 Instruct	57.78	86.74	51.18	53.04	62.62	63.59
Local	Qwen3-32B-AWQ	39.46	45.37	33.39	43.04	42.02	41.87
	Qwen3.5-27B-AWQ	32.25	40.46	23.04	37.87	38.53	36.27
	Qwen3-14B-AWQ	26.54	33.91	22.91	63.29	42.49	38.55
	DeepSeek-R1-32B-AWQ	41.54	56.73	36.55	47.97	61.23	49.80
	DeepSeek-R1-14B-AWQ	37.82	53.64	32.89	46.33	55.49	46.22

Under the current model pool, task scope, and experimental settings, MineralBench reveals differences in task performance, question-level complementarity, and multidimensional model characteristics across the evaluated LLMs. These results indicate that MineralBench provides discriminative and interpretable information for task-specific model comparison and selection, with potential value for model routing. These conclusions remain limited by the tested model scope, task coverage, and sample size. The Oracle results remain empirical performance upper bounds, while Efficiency depends partly on access methods and deployment environments. Overall, the results in this section support the applicability of MineralBench within the current experimental scope. Further studies are still needed to examine its stability and generalizability across broader task and model sets, different experimental settings, and actual routing scenarios.

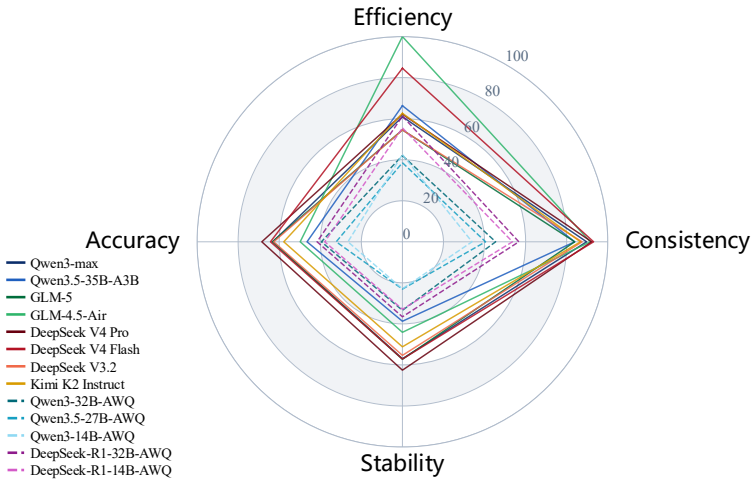


Fig.3 Results across four evaluation dimensions

4.4 Task accuracy comparison

Task Accuracy directly reflects model correctness on individual tasks and serves as a basic metric for comparing model performance across tasks. Fig.4 presents a heatmap of the Accuracy scores across the six tasks to examine cross-task differences in LLM performance, showing both task-level score distributions and model performance profiles.

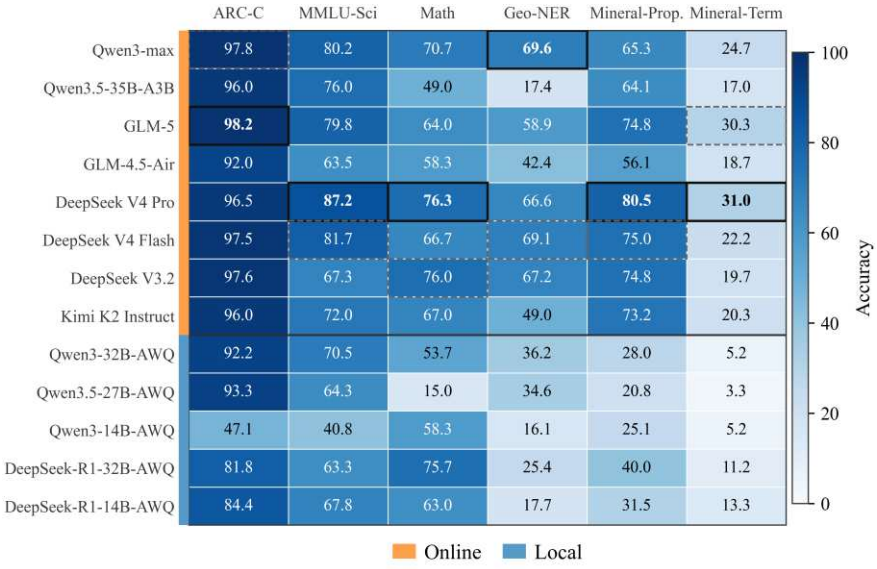


Fig.4 Heatmap of task Accuracy

At the task level, Fig.4 shows that the LLMs generally achieve higher scores on the three general tasks, with ARC-Challenge scores showing relatively small variation. In contrast, the three core domain tasks show more pronounced differences among models. Within these domain tasks, some models achieve relatively high scores on Geo-NER and Mineral-Property, but substantial gaps remain across models. Mineral-Term has the lowest overall scores, with the highest score reaching only 31.0. High performance on general tasks does not consistently extend to the domain tasks, indicating that results on

general tasks alone cannot fully represent LLM performance in the mineral resources domain.

At the model level, Fig.4 shows that no single model achieves the highest score across all tasks. DeepSeek V4 Pro achieves the highest scores on MMLU-Sci, Math, Mineral-Property, and Mineral-Term; Qwen3-max performs best on Geo-NER; and GLM-5 achieves the highest score on ARC-Challenge. The online models generally obtain higher scores on most tasks. Some local models remain competitive on general tasks such as Math, but their performance on the core domain tasks is relatively limited. Together, these model-level results indicate that models have different strengths in scientific knowledge, reasoning, and domain information processing, and that a single aggregate score cannot fully capture their task-specific strengths and weaknesses. Similar performance differences across tasks have also been observed in previous large-scale multi-task evaluation studies (Srivastava et al. 2023).

Overall, these results suggest that practical model comparison and selection should consider task-specific performance rather than relying solely on overall Accuracy or general-task results.

4.5 Inter-round consistency comparison



Fig.5 Heatmap of Inter-round Consistency

Inter-round Consistency characterizes the stability across repeated runs of the same task under the same evaluation settings. Fig.5 presents a heatmap of the task-level Inter-round Consistency scores across the six tasks to compare cross-task consistency differences among the 13 models.

Overall, most online models maintain high consistency across tasks, while local models show greater variation. On ARC-Challenge, consistency scores show relatively little variation, whereas Geo-NER, Mineral-Property, and Mineral-Term show larger differences across models. The online models generally maintain higher consistency on the three core domain tasks, while local models show lower consistency on Mineral-Property and Mineral-Term, indicating clear differences in output stability across repeated evaluations of domain tasks.

Comparison with the Accuracy results in Fig.4 shows that Inter-round Consistency and task

correctness do not always vary together. For example, Qwen3.5-35B-A3B achieves an Inter-round Consistency score of 93.2 on Mineral-Term, but its Accuracy is only 17.0, indicating that the model may repeatedly produce similar yet incorrect outputs. Its Inter-round Consistency score on Geo-NER drops to 48.9, reflecting greater variation across repeated outputs. In contrast, DeepSeek V4 Flash maintains high Accuracy and Inter-round Consistency on Geo-NER and Mineral-Property. Among locally deployed models, the DeepSeek-R1 series generally shows higher Inter-round Consistency than the local Qwen series on Math, Mineral-Property, and Mineral-Term, although its consistency still varies across tasks.

These results show that Inter-round Consistency and task Accuracy do not consistently vary together, and that higher consistency does not necessarily correspond to higher Accuracy. Taken together, these results suggest that Inter-round Consistency and task Accuracy need to be considered together: Accuracy reflects correctness, while consistency reflects repeatability across repeated runs of the same task. The two metrics help distinguish correct and consistent outputs, incorrect but consistent outputs, and unstable outputs.

4.6 Analysis of weighted results and general stability

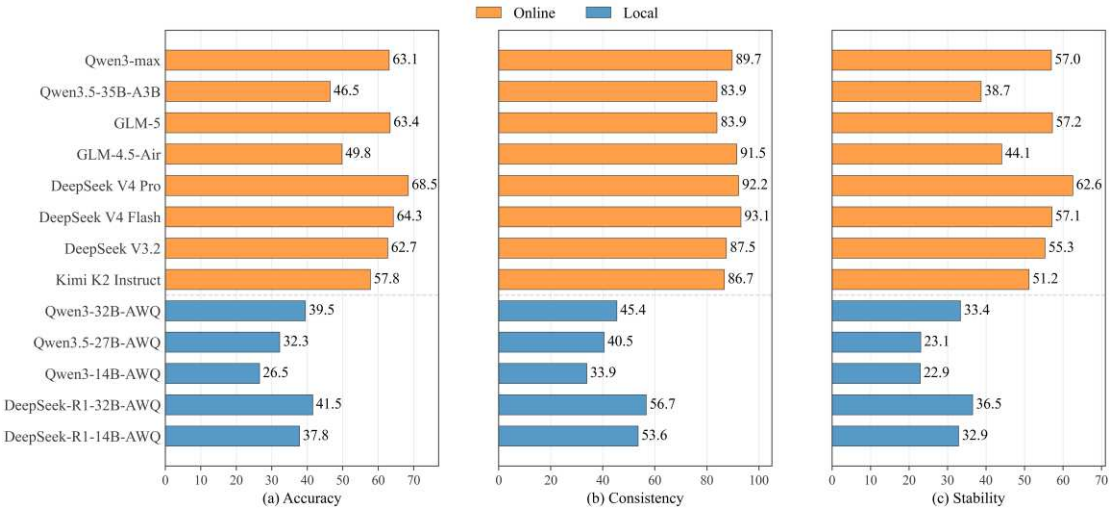


Fig.6 Comprehensive evaluation statistics

Weighted Accuracy reflects overall performance across the six tasks, but an average score may mask task-specific variation. Fig.6 compares the Weighted Accuracy, Inter-round Consistency, and General Stability Score to examine how evenly each model performs across tasks. The analysis focuses on General Stability Score, with Weighted Accuracy and Inter-round Consistency providing reference points.

As shown in Fig.6 (a) and Fig.6 (c), General Stability Score follows a broadly similar distribution to Weighted Accuracy because it combines mean task Accuracy with a penalty for cross-task variation. However, the model rankings are not identical. DeepSeek V4 Pro achieves the highest Weighted Accuracy and General Stability Score, at 68.5 and 62.6, respectively, indicating both strong overall Accuracy and relatively balanced cross-task performance. The other online models show similar Weighted Accuracy scores but more varied General Stability Scores after cross-task variation is

considered. Local models generally have lower General Stability Scores, with Qwen3.5-27B-AWQ and Qwen3-14B-AWQ scoring 23.1 and 22.9, respectively, reflecting larger cross-task variation. These results show that Weighted Accuracy summarizes average task performance, while General Stability Score reveals cross-task imbalance and task-specific weaknesses that the average may mask.

Fig.6 (b) shows performance consistency across repeated evaluations. Inter-round Consistency and General Stability Score do not consistently vary together. For example, GLM-4.5-Air achieves an Inter-round Consistency score of 91.5, while its Weighted Accuracy and General Stability Score are 49.8 and 44.1, respectively. This indicates that high output consistency does not necessarily imply high or balanced Accuracy across tasks. In comparison, DeepSeek V4 Pro and DeepSeek V4 Flash maintain high scores across all three metrics.

These results indicate that General Stability Score captures cross-task balance and provides information complementary to Weighted Accuracy and Inter-round Consistency. Overall, model selection for multi-task scenarios should consider both Weighted Accuracy and General Stability Score to avoid overlooking task-specific weaknesses hidden by average performance. Inter-round Consistency provides a complementary measure of repeatability across evaluations.

4.7 Efficiency comparison

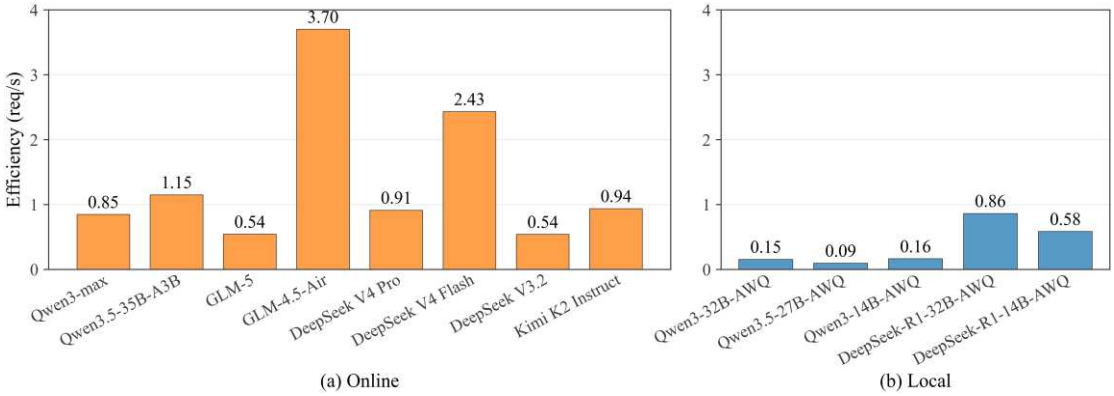


Fig.7 Comparison of model Efficiency

Efficiency reflects request-processing performance under the current evaluation environment and provides a separate consideration for model deployment and application. Fig.7 presents the overall throughput of online and local models.

Among the online models, GLM-4.5-Air achieves the highest throughput at 3.70 req/s, followed by DeepSeek V4 Flash at 2.43 req/s, while all other models remain below 1.20 req/s. Combined with the preceding Accuracy results, it can be seen that Efficiency and task Accuracy do not consistently vary together. For example, DeepSeek V4 Pro achieves the highest Weighted Accuracy, but its throughput is only 0.91 req/s; GLM-4.5-Air leads in throughput without showing a comparable advantage in Accuracy. This indicates that task performance and request-processing efficiency capture distinct aspects of model performance.

Among local models, DeepSeek-R1-32B-AWQ and DeepSeek-R1-14B-AWQ achieve throughput

values of 0.86 req/s and 0.58 req/s, respectively, both higher than those of the three local Qwen models. The current results show no simple monotonic relationship between parameter count and throughput, suggesting that actual operating efficiency may also be affected by factors such as model architecture, quantization method, and inference implementation.

Online models run in external API service environments, whereas local models use a shared hardware and inference framework. Fig.7 mainly reflects end-to-end processing efficiency under the current access or deployment conditions rather than directly comparable intrinsic model efficiency across online and local groups. In practical model selection, task Accuracy and Efficiency should be considered together. For applications that prioritize response speed and concurrent processing, throughput deserves greater attention, whereas tasks that prioritize output correctness should not rely on Efficiency alone for model selection.

4.8 Domain adaptation experiments

The main experimental results show substantial room for improvement on some domain tasks. The domain adaptation experiment applies LoRA fine-tuning (Hu et al. 2022) to DeepSeek-R1-14B to examine changes in task performance and output behavior and whether MineralBench captures them. Because the 4-bit AWQ version used in the main experiments is not directly compatible with the current fine-tuning framework, the corresponding original FP16 model serves as the LoRA fine-tuning base, producing FP16-FT. FP16 and FP16-FT form the primary fine-tuning comparison, while AWQ is retained only as a quantized reference to the main experiments.

During LoRA fine-tuning, the base-model weights remain frozen and only the LoRA adapter parameters are updated. The fine-tuning data consist of external samples aligned with the domain tasks and remain fully separated from the formal MineralBench evaluation set. After fine-tuning, FP16-FT follows the same evaluation procedure as the main experiments and is compared with the original FP16 model.

Fine-tuning effects are evaluated using Accuracy and Output-format Compliance Rate (Compliance). Accuracy follows the scoring procedure used in the main experiments: model answers are first normalized and parsed and then compared with the reference answers to calculate task-level correctness. The normalization tolerates common parseable outputs, such as converting “Answer: A” to “A” or extracting “1.02” from “\boxed{1.02}”. Compliance, in contrast, strictly evaluates whether the response follows the output form specified in each task prompt; a response is counted as valid only when it satisfies the required format, regardless of whether its answer content can otherwise be parsed correctly. Compliance is calculated as:

$$Compliance = \frac{N_{\text{valid}}}{N_{\text{total}}} \times 100$$

where N_{valid} denotes the number of format-compliant responses, and N_{total} denotes the total number of evaluated responses. The same task-specific compliance rules are applied across all model versions.

The three model versions and their GPU memory usage are listed in Table 9.

Table 9. Model versions and GPU memory usage

Model	Description	GPU Memory (GB)
FP16	Original Version	27.59
AWQ	4-bit Quantized	9.35
FP16-FT	LoRA Fine-tuning	27.59

Fig.8(a) shows task-level Accuracy across the three model versions. Compared with FP16, FP16-FT improves the Geo-NER Accuracy from 3.3 to 55.6, the largest gain across the six tasks. In contrast, Math Accuracy decreases from 51.3 to 17.3, while the other tasks also decline to varying degrees. These results show that the Accuracy gains from LoRA fine-tuning are concentrated on Geo-NER rather than consistently distributed across tasks, suggesting that the adaptation effect may be task-dependent.

Fig.8(b) shows the Output-format Compliance Rate across the three model versions. In contrast to the Accuracy results, FP16-FT achieves higher Compliance than FP16 across all six tasks. In particular, Math Compliance increases from 9.3 to 92.7, while Mineral-Term shows only a limited improvement. Considering both metrics, changes in Compliance and Accuracy do not always align: both improve on Geo-NER, whereas on Math, Compliance increases substantially while Accuracy decreases markedly. These results indicate that domain adaptation may help improve the LLM's ability to follow specified output formats, but such improvement does not necessarily translate into a corresponding increase in task Accuracy.

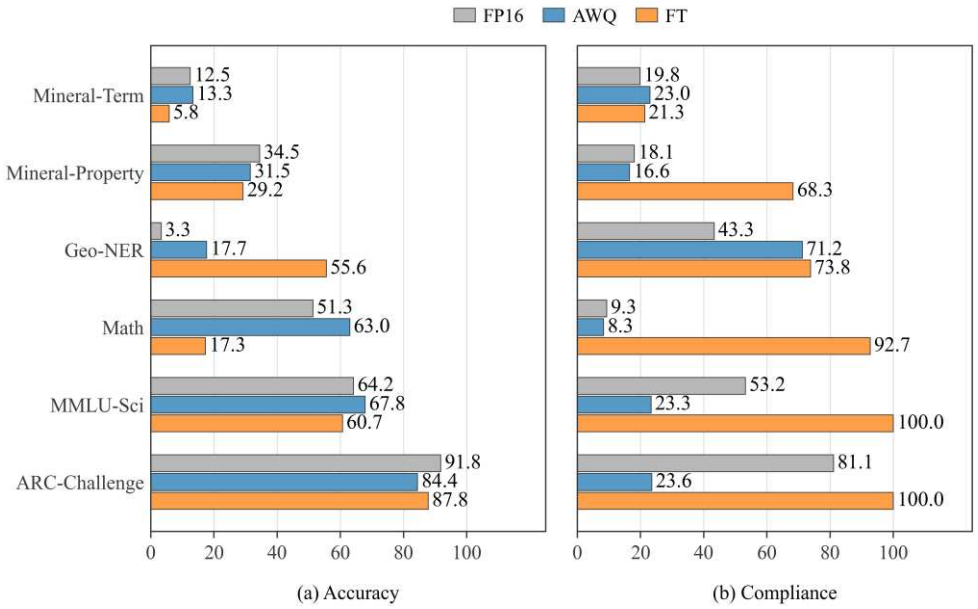


Fig.8 Domain adaptation experimental results

Because quantization may alter the model output distribution, Accuracy differences between AWQ and FP16 do not follow a consistent direction. Accordingly, the effect of fine-tuning is evaluated primarily through the FP16–FP16-FT comparison rather than the difference between AWQ and FP16. Overall, the domain adaptation experiment shows a substantial Accuracy gain on Geo-NER and generally higher output-format compliance, while Accuracy declines on several other tasks. These results suggest that the effects of LoRA fine-tuning may be task-dependent. Evaluating fine-tuning effects requires

considering not only gains on the target task but also capability retention across other tasks.

5. Conclusion

This paper develops MineralBench, an evaluation benchmark for LLMs in the mineral resources domain, and conducts a unified evaluation of 13 LLMs across four dimensions: Accuracy, Inter-round Consistency, General Stability Score, and Efficiency. The experimental results show that the LLMs generally perform better on general tasks than on mineral resource tasks, and that LLMs exhibit substantial differences across multiple evaluation metrics, with considerable complementarity among them. Further domain adaptation experiments suggest that the effects of LoRA fine-tuning may be task-dependent, with performance improving on some tasks while decreasing on others.

One limitation of this study is that MineralBench currently has limited coverage of knowledge and task types in the mineral resources domain. Expanding the range of domain evaluation data and task types is a worthwhile endeavor for accurately characterizing the performance of LLMs in this field. In addition, the experiments cover a relatively limited range of LLMs. The experimental conclusions mainly reflect the overall characteristics of the selected LLMs, and their scope of applicability is limited.

Appendix

Appendix A. MineralBench Prompt Templates

To ensure consistency in the evaluation process, this paper uses fixed prompt templates for different task types. The templates draw on prompt designs from existing studies on LLMs in the geological domain and zero-shot named entity recognition (Zamai et al. 2025; Ma et al. 2026), and are adapted according to the task formats and output requirements of MineralBench. The three main types of prompts used in the evaluation are presented below. In the templates, underlined placeholders enclosed in braces, such as "{question}" and "{options}", denote dynamically populated fields corresponding to components of the task input $\mathcal{X}_i$ defined in Section 3.2.

Single Choice Prompt

System Prompt:

```
You are a knowledgeable assistant.
```

User Prompt:

```
Please give the correct answer to the following multiple - choice question:
Question: {question}
Options:
{options}
Requirements:
```

```
    1. Output only one option label (A, B, C, D, ...), with no extra words.
    2. Do not provide explanation or reasoning.
    3. Do not fabricate an answer.
Please output the answer label directly (A, B, C, D, ...):
```

Listing A-1. Single choice prompt

```
System Prompt:
You are a domain expert.
User Prompt:
Please provide a detailed, accurate, and academically rigorous answer with professional depth for the following short-answer question.
Strictly prohibit fabricating dates or numerical values. Ensure the answer is accurate and does not omit any key information.
Question:
{question}
Requirements:
    1. Output only the core answer or term. Keep it concise without additional explanations.
    2. Use rigorous academic phrasing and professional terminology.
    3. Avoid phrases such as 'according to research' or 'literature indicates'; state facts and conclusions directly.
```

Listing A-2. Short answer prompt

```
System Prompt:
You are a domain expert in geology and mineral exploration.
User Prompt:
Instruction: Given the label set of entities, please recognize all the entities in the text. The answer format should be "{type1: [entity1, entity2], type2: [entity3]}". Below is an example.
Label Set: [MINERAL]
Text: [Quartz and feldspar are common]
Answer: [{"MINERAL":["Quartz","Feldspar"]}]
Label Set: [{entity type}]
Text:{{text}}
Answer:[]
```

Listing A-3. Named entity recognition prompt

The above prompt templates are stored in structured configuration files and are automatically invoked by the evaluation program according to the task type. During evaluation, the program fills the sample fields into the corresponding templates to generate the model input, after which the model outputs enter a unified process of normalization, parsing, and scoring.

Statements and Declarations

Funding

This work was supported by the Deep Earth Probe and Mineral Resources Exploration National Science and Technology Major Project (Grant No. 2025ZD1008204). The funding source had no role in the design of the study, data analysis, interpretation of the results, or preparation of the manuscript.

Competing Interests

The authors disclose that patent applications and software copyrights related to the broader research project have been filed or registered as part of the project requirements. These intellectual property activities are not known to involve any current commercial interests or direct financial benefits related to the present study. The authors declare no other competing interests.

Author Contributions

Danjiahao conducted the main experiments, developed the evaluation software, curated the data, performed the formal analysis and visualization, contributed to the methodology, and drafted the original manuscript. Lishengwen and Yaohong supervised the study, contributed to methodological discussions, and reviewed and edited the manuscript. Congyuan contributed to the initial conceptualization of the evaluation framework, project coordination, and methodological discussions. Libingyi and Zhouhang assisted with the experiments and contributed to manuscript review and editing.

Data Availability

The data used in this study include publicly available benchmark datasets and mineral-related data derived from the sources described in the manuscript. Releasable portions of the MineralBench evaluation dataset and the associated source code are planned to be made publicly available after further organization and documentation. Additional information regarding the data and code may be obtained from the corresponding author upon reasonable request.

References

Alibaba Cloud Alibaba Cloud Model Studio API Access Documentation. In: Alibaba Cloud Doc. https://help.aliyun.com/zh/model-studio/qwen-api-via-openai-chat-completions. Accessed 10 Aug 2026

Bi Z, Zhang N, Xue Y, et al (2024) OceanGPT: A large language model for ocean science tasks. In: Ku L-W, Martins A, Srikumar V (eds) Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Bangkok, Thailand, pp 3357–3372

Clark P, Cowhey I, Etzioni O, et al (2018) Think you have solved question answering? Try ARC, the AI2

reasoning challenge

DeepSeek-AI, Guo D, Yang D, et al (2025a) DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. Nature 645:633–638. https://doi.org/10.1038/s41586-025-09422-z

DeepSeek-AI, Liu A, Mei A, et al (2025b) DeepSeek-V3.2: Pushing the frontier of open large language models

Deng C, Zhang T, He Z, et al (2024) K2: A foundation language model for geoscience knowledge understanding and utilization. In: Proceedings of the 17th ACM International Conference on Web Search and Data Mining. Association for Computing Machinery, New York, NY, USA, pp 161–170

Enkhsaikhan M, Holden E-J, Duuring P, Liu W (2021a) Understanding ore-forming conditions using machine reading of text. Ore Geol Rev 135:104200. https://doi.org/10.1016/j.oregeorev.2021.104200

Enkhsaikhan M, Liu W, Holden E-J, Duuring P (2021b) Auto-labelling entities in low-resource text: A geological case study. Knowl Inf Syst 63:695–715. https://doi.org/10.1007/s10115-020-01532-6

Fu Y, Wang M, Wang C, et al (2025) GeoMinLM: A large language model in geology and mineral survey in yunnan province. Ore Geol Rev 182:106638. https://doi.org/10.1016/j.oregeorev.2025.106638

Ge Q, Li P, Li J, et al (2026) Leveraging large language models for automated knowledge extraction from geological reports. J Rock Mech Geotech Eng S1674775526001216. https://doi.org/10.1016/j.jrmge.2025.12.038

Gjorgjevikj A, Nikolikj A, Koroušić Seljak B, Eftimov T (2025) User-defined trade-offs in LLM benchmarking: Balancing accuracy, scale, and sustainability. Knowl-Based Syst 330:114405. https://doi.org/10.1016/j.knosys.2025.114405

GLM-4.5 Team, Zeng A, Lv X, et al (2025) GLM-4.5: Agentic, reasoning, and coding (ARC) foundation models

GLM-5 Team, Zeng A, Lv X, et al (2026) GLM-5: From vibe coding to agentic engineering

Hendrycks D, Burns C, Basart S, et al (2021a) Measuring massive multitask language understanding. In: International Conference on Learning Representations

Hendrycks D, Burns C, Kadavath S, et al (2021b) Measuring mathematical problem solving with the MATH dataset. Proc Neural Inf Process Syst Track Datasets Benchmarks 1:

Hu EJ, Shen Y, Wallis P, et al (2022) LoRA: Low-rank adaptation of large language models. In: International Conference on Learning Representations

Kimi Team, Bai Y, Bao Y, et al (2025) Kimi K2: Open agentic intelligence

Kosub S (2019) A note on the triangle inequality for the jaccard distance. Pattern Recognit Lett 120:36–38. https://doi.org/10.1016/j.patrec.2018.12.007

Kwon W, Li Z, Zhuang S, et al (2023) Efficient memory management for large language model serving with PagedAttention. In: Proceedings of the 29th Symposium on Operating Systems Principles. Association for Computing Machinery, New York, NY, USA, pp 611–626

Lawley CJM, Gadd MG, Parsa M, et al (2023) Applications of natural language processing to geoscience text data and prospectivity modeling. Nat Resour Res 32:1503–1527.

https://doi.org/10.1007/s11053-023-10216-1

Liang P, Bommasani R, Lee T, et al (2023) Holistic evaluation of language models. Trans Mach Learn Res

Lin J, Tang J, Tang H, et al (2024) AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In: Proceedings of Machine Learning and Systems. pp 87–100

Ma J, Zhou Y, Su L, et al (2026) A domain-specific language model for engineering-scale geological reasoning and mineral exploration in the qin-hang belt. Eng Appl Artif Intell 167:113651. https://doi.org/10.1016/j.engappai.2025.113651

Manivannan VV, Jafari Y, Eranky S, et al (2025) ClimaQA: An automated evaluation framework for climate question answering models. In: International Conference on Learning Representations. pp 25357–25382

ModelScope ModelScope. In: ModelScope. https://modelscope.cn/models. Accessed 10 Aug 2026

Ralph J, Von Bargen D, Martynov P, et al (2025) Mindat.org: The open access mineralogy database to accelerate data-intensive geoscience research. Am Mineral 110:833–844. https://doi.org/10.2138/am-2024-9486

Srivastava A, Rastogi A, Rao A, et al (2023) Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. Trans Mach Learn Res

Wang X, Hu Z, Lu P, et al (2024) SciBench: Evaluating college-level scientific problem-solving abilities of large language models. In: Proceedings of the 41st International Conference on Machine Learning. PMLR, pp 50622–50649

Xu D, Chen W, Peng W, et al (2024) Large language models for generative information extraction: A survey. Front Comput Sci 18:186357. https://doi.org/10.1007/s11704-024-40555-y

Xu W, Zhao X, Zhou Y, et al (2026) EarthSE: A benchmark evaluating earth scientific exploration capability for large language models. In: International Conference on Learning Representations

Yang A, Li A, Yang B, et al (2025) Qwen3 technical report

Yu B, Shen T, Na H, et al (2024) MineAgent: Towards remote-sensing mineral exploration with multimodal large language models

Zamai A, Zugarini A, Rigutini L, et al (2025) Show less, instruct more: Enriching prompts with definitions and guidelines for zero-shot NER. In: 2025 International Joint Conference on Neural Networks (IJCNN). pp 1–8