EngiBench: A Benchmark for Evaluating Large Language Models on Engineering Problem Solving

Xiyuan Zhou[1]*, Xinlei Wang[2,3]*, Yirui He[4,5], Yang Wu[4], Ruixi Zou[4], Yuheng Cheng[4], Yulu Xie[6], Wenxuan Liu[1], Huan Zhao[7], Yan Xu[1]†, Jinjin Gu[3]†, Junhua Zhao[4,8]†

[1]Nanyang Technological University, [2]The University of Sydney,
[3]INSAIT, Sofia University "St. Kliment Ohridski",
[4]The Chinese University of Hong Kong, Shenzhen, [5]Shenzhen Loop Area Institute,
[6]The University of Hong Kong, [7]Hong Kong Polytechnic University, [8]AIRS
xiyuan002@e.ntu.edu.sg, {xinlei.wang,jinjin.gu}@insait.ai
xuyan@ntu.edu.sg, zhaojunhua@cuhk.edu.cn

Abstract

Large language models (LLMs) have shown strong performance on mathematical reasoning under well-defined conditions. However, real-world engineering problems involve uncertainty, context, and open-ended settings that extend beyond symbolic computation. Existing benchmarks largely focus on well-defined or abstract reasoning and therefore fail to capture these complexities. We introduce EngiBench, a hierarchical benchmark designed to evaluate LLMs on solving engineering problems. It spans three levels of increasing difficulty (foundational knowledge retrieval, contextual reasoning, and open-ended modeling) and covers diverse engineering subfields. To facilitate a deeper understanding of model performance, we systematically rewrite each problem into three controlled variants (perturbed, knowledge-enhanced, and math abstraction), enabling us to separately evaluate the model's robustness, domain-specific knowledge, and mathematical reasoning abilities. Experimental results show clear performance stratification across difficulty levels: model accuracy declines with task complexity, degrades under minor perturbations, and remains substantially below human performance on high-level engineering tasks. These findings reveal that current LLMs still lack the high-level reasoning needed for real-world engineering, highlighting the need for future models with deeper and more reliable problem-solving capabilities. Our source code and data are available at https://github.com/AI4Engi/EngiBench.

1 Introduction

Large language models (LLMs) have demonstrated promising capabilities in a range of mathematical reasoning tasks, from foundational skills such as basic computation and structured problem-solving (Cobbe et al., 2021), multi-step reasoning (Shao et al., 2024; Wei et al., 2022), to more complex applications like mathematical modeling (Guo et al., 2025) and the generation or verification of mathematical proofs (Yang et al., 2023; Lin et al., 2025; Ren et al., 2025). However, just using mathematical reasoning is not enough for real-world applications. In practice, many applications arise not in abstract mathematical settings but in engineering contexts, where problems are grounded in physical systems and must handle uncertainty and real-world constraints. These characteristics require not only mathematical computation, but also broader capabilities to understand engineering contexts and solve complex engineering problems.

Engineering problems differ fundamentally from mathematical problems (Hendrycks et al., 2021). Rather than seeking single closed-form answers, engineering requires finding feasible solutions that balance objectives under real-world constraints (Dym et al., 2005; Zhou et al., 2026). For example, designing a drone system (Table 1) involves identifying operational requirements and managing trade-offs among range, payload, and energy limits. As shown in Figure 1, solving such problems requires more than recalling formulas or executing isolated calculations; it involves a sequence of interconnected cognitive steps, from understanding context and selecting appropriate assumptions to navigating trade-offs and addressing uncertainties. We refer to this broader set of competencies as the engineering problem-solving capability, consisting of four dimensions: *information extraction, domain-specific reasoning, multi-objective decision-making, and uncertainty handling*.

Despite the broader requirements of real-world

*Equal contribution.
†Corresponding authors.

Findings of the Association for Computational Linguistics: ACL 2026, pages 36308–36334
July 2-7, 2026

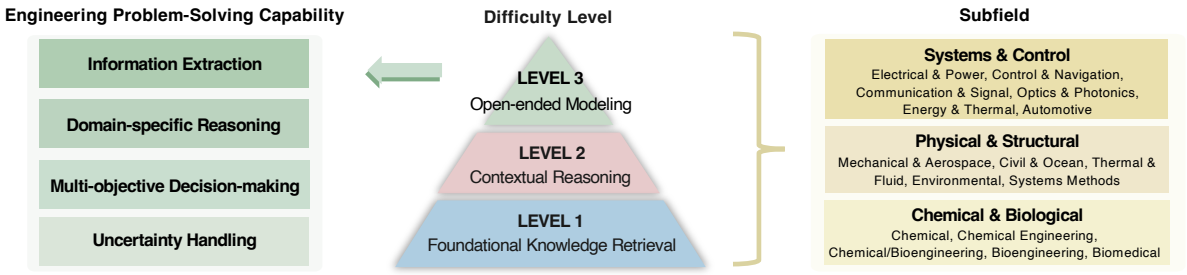


Figure 1: Task taxonomy of EngiBench organized by difficulty, capability, and subfield. Problems are grouped into three difficulty levels, with Level 3 specifically designed to evaluate engineering problem-solving capabilities. All tasks are additionally categorised into three major engineering subfields.

engineering tasks, most existing benchmarks focus narrowly on well-defined mathematical problems. Benchmarks such as GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), and Omni-MATH (Gao et al., 2025) primarily assess symbolic reasoning, calculation, and formal problem-solving under clean and well-defined conditions. Although some include basic engineering questions, they fail to capture the deeper reasoning required for real-world problem solving (Hendrycks et al., 2021; Wang et al., 2024c; Albalak et al., 2025; Du et al., 2025). A further limitation is that many benchmarks rely on public datasets without systematic rewriting, increasing the risk of pretraining overlap and inflated scores (Deng et al., 2024; Huang et al., 2025; Sainz et al., 2023). For example, GSM1k re-creates GSM8k-style questions to reduce overlap and observes performance drops of up to 8% (Zhang et al., 2024). Without such safeguards, evaluations may reflect memorization rather than genuine reasoning, providing limited insight into an LLM's ability to address realistic engineering tasks.

In this work, we introduce EngiBench, an evaluation framework designed not only to assess LLMs' engineering problem-solving capabilities but also to diagnose where and why these capabilities fail. The benchmark spans multiple engineering subfields and structures tasks into three difficulty levels that reflect the progression from foundational knowledge retrieval to contextual reasoning and open-ended modeling. To support fine-grained diagnosis, each problem is provided in three controlled variants that separate robustness, domain knowledge, and mathematical reasoning. Evaluation centers on four capability dimensions essential to engineering problem solving: *information extraction, domain-specific reasoning, multi-objective decision-making, and uncertainty handling*. For open-ended tasks, we further adopt rubric-based evaluation using expert-designed scoring criteria to ensure consistent and reliable assessment. Together, these components create a diverse, high-quality, and contamination-aware benchmark for evaluating LLMs' engineering problem-solving capabilities.

Experiment results show clear stratification across difficulty levels, with higher-level tasks highlighting distinct capability gaps. The perturbed variant leads to performance drops, even in strong models, revealing that prior evaluations may overestimate true generalization. Most importantly, current LLMs perform poorly on Level 3 tasks involving open-ended, high-level engineering reasoning and remain far below human experts. These findings suggest that today's LLMs are still far from reliably addressing real-world engineering problems, leaving substantial room for future improvement.

Our contributions can be summarized as follows: (1) We are among the first to systematically evaluate LLMs on real-world engineering problems; (2) We design a hierarchical benchmark with three difficulty levels and multiple problem variants, enabling fine-grained analysis of model reasoning capabilities and limitations; (3) Unlike prior benchmarks, our benchmark systematically evaluates LLM performance on open-ended engineering tasks; (4) We evaluate a broad set of mainstream LLMs, providing insights that can aid future model development and enhance engineering capabilities.

2 Related Works

LLMs for Engineering Problems. LLMs integrate broad domain knowledge with strong multi-step reasoning, making them promising tools for complex problem solving. Engineering problems, however, require modeling real-world systems and reasoning under practical constraints. Despite the growing use of LLMs in engineering appli-

Level	Definition	Example
Mathematics	Mathematical tasks are typically well-posed and self-contained, with complete information and clearly defined solution spaces.	A machine produces 45 parts per minute. If it operates continuously for 2 hours, how many parts will it produce in total? 👉 *This task requires only basic multiplication and does not involve any domain knowledge. It represents a typical closed-form numerical computation problem.*
💡 **Upgrading Condition:** Incorporating domain-specific engineering knowledge		
Engineering Level 1: Foundational Knowledge Retrieval	Apply basic engineering concepts or formulas to structured problems via single-step computation.	A drone operates at a constant power of 200W for 30 minutes. Calculate the total energy consumption in joules. 👉 *This task requires applying the basic physical formula $E = P \times t$, with unit conversion from minutes to seconds. It tests the model's ability to retrieve and apply foundational engineering knowledge in a single-step calculation.*
💡 **Upgrading Condition:** Multi-step reasoning and contextual integration		
Engineering Level 2: Contextual Reasoning	Perform multi-step reasoning under well-defined constraints by integrating conditions and domain knowledge.	A drone needs to fly 6 km. The first half is uphill, increasing power usage by 20%, while the second half is flat at 180W. The drone flies at 30 km/h and uses a battery rated at 8000mAh, 11.1V. Can the battery support the trip? 👉 *This task requires multi-step reasoning: estimate flight time, adjust power consumption, and compare with battery capacity.*
💡 **Upgrading Condition:** Solving open-ended, under-specified problems		
Engineering Level 3: Open-ended Modeling	Solve open-ended, real-world problems through information extraction, trade-off reasoning, and uncertainty handling.	Design a drone system for urban delivery that balances multiple factors, including flight range, payload capacity, and cost control. Propose a feasible solution and justify your design decisions. 👉 *This is an open-ended problem with incomplete constraints and potentially conflicting objectives, requiring information extraction, trade-off analysis, and robustness under uncertainty.*
🔍 Information Extraction	Identify and extract relevant information from complex or redundant problem descriptions.	Identify critical variables—such as payload weight, wind speed, flight duration, and battery margin—from complex or verbose task descriptions.
📖 Domain-specific Reasoning	Apply specialized engineering principles and structured knowledge to guide logical inference and solution formulation.	Apply specialized engineering knowledge—such as flight mechanics and battery discharge principles—to formulate models and perform technical analysis.
⚖️ Multi-objective Decision-making	Make justified trade-offs between competing in the absence of a single optimal solution.	Justify trade-offs among competing objectives like range, cost, safety, and operational efficiency when no single optimal solution exists.
🌫️ Uncertainty Handling	Ensure solution robustness by reasoning under incomplete, variable, or ambiguous real-world conditions.	Account for unpredictable factors such as weather, task variation, and battery aging, and design robust strategies (e.g., adding 20% battery reserve) to ensure reliable performance.

Table 1: Hierarchical difficulty from mathematics to real-world engineering. This illustrates three levels of increasing complexity. Examples show the progression from closed-form math problems to open-ended engineering scenarios.

cations, their true engineering problem-solving capability remains unclear due to the limitations of existing benchmarks (Wang et al., 2024b; Ma et al., 2024; Tang et al., 2024; Cheng et al., 2025). General-purpose benchmarks, including MMLU (Hendrycks et al., 2021), MMLU-Pro (Wang et al., 2024c), BIG-Math (Albalak et al., 2025), and SuperGPQA (Du et al., 2025), contain only limited engineering content. Most questions emphasize factual recall in multiple-choice form, failing to capture core engineering reasoning. Several domain-specific engineering benchmarks have been proposed, including EEE-Bench (Li et al., 2024), ElecBench (Zhou et al., 2024), FEABench (Mudur et al., 2025), TransportBench (Syed et al., 2024), and JEEBench (Arora et al., 2023). However, they largely focus on single disciplines and well-defined tasks, limiting their ability to evaluate open-ended and cross-disciplinary engineering reasoning. To address this gap, we introduce a multi-level engineering benchmark spanning multiple subfields and incorporating both closed-form and open-ended tasks, enabling a comprehensive evaluation of engineering capabilities.

LLM for Mathematical Problems. A closely related area that has been extensively studied is mathematics. Because solving mathematical problems demands strong logical ability, multi-step reasoning, and symbolic manipulation, it has become a primary proving ground for evaluating LLMs. Early benchmarks focus on elementary problems (Cobbe et al., 2021; Hendrycks et al., 2021; Patel et al., 2021; Amini et al., 2019) and higher-level symbolic reasoning (Hendrycks et al., 2021; Albalak et al., 2025). Recent efforts like MiniF2F (Zheng et al., 2022), UniMath (Liang et al., 2023), Omni-MATH (Gao et al., 2024), and MathVista (Lu et al., 2024) expand to theorem proving and multimodal tasks. MATH-Vision (Wang et al., 2024a) improves coverage by introducing diverse topics and difficulty levels from real competitions, and SMART-840 (Cherian et al., 2024) benchmarks model performance against human children across grades. While these benchmarks provide rigorous evaluations of mathematical competence, they do not capture engineering-specific reasoning such as modeling, decision-making under constraints, or domain-based assumptions. Our work builds on their methodological insights but shifts the focus toward real-world engineering tasks.

Evaluation Challenges. Evaluating the capability of LLMs to solve engineering problems is challenging due to the inherent complexity involved. Current evaluation methods for LLMs fall into four main categories: reference-based, task-oriented, preference-based, and rubric-based. The first two are effective for problems with clear ground truths or executable outputs – e.g., MathVista (Lu et al., 2024), CHAMP (Mao et al., 2024) (reference-based), and EEE-Bench (Li et al., 2024), FEABench (task-oriented) (Mudur et al., 2025). However, the core capabilities of the engineering field we are discussing cannot be effectively evaluated by such closed-form problems. For open-ended tasks, preference-based methods such as MT-Bench-101 (Bai et al., 2024) use pairwise comparisons, but are often biased by model-specific generation patterns, limiting objectivity and real-world applicability. Rubric-based evaluations aim to improve transparency by scoring along multi-

ple criteria, with general-purpose frameworks like Prometheus (Kim et al., 2024) focusing on abilities such as context retention and rephrasing.

3 Methodology

3.1 Engineering Problem-Solving Capability

Engineering problems require context-aware solutions under real-world constraints, distinguishing them from mathematical problems that typically operate in well-defined, closed-form settings (Dym et al., 2005; Hendrycks et al., 2021). In this work, engineering problems are defined as tasks that apply scientific principles to the modeling and analysis of systems under such constraints. Beyond abstraction and logical rigor, engineering problem solving involves a sequence of interconnected cognitive steps, from interpreting problem context to making decisions under constraints and uncertainty. We refer to this as engineering problem-solving ability, which comprises four key dimensions: information extraction, domain-specific reasoning, multi-objective decision-making, and uncertainty handling. These dimensions align with established paradigms in engineering modeling, including information filtering, constraint-based and multi-objective formulation, and robustness analysis, and can be interpreted as a reasoning-level abstraction of classical engineering design processes (Beitz et al., 1996), spanning stages such as task clarification, conceptual design, embodiment design, and detail design (see Figure 1 and Table 1).

Information Extraction. The capability to identify and organize key variables, constraints, and objectives from complex or noisy descriptions. It reflects the model's capacity to filter irrelevant information and transform unstructured text into structured representations for downstream reasoning.

Domain-specific Reasoning. The capability to apply engineering knowledge, including physical principles, empirical rules, and domain conventions, to interpret scenarios and choose appropriate solution strategies. It involves recognizing valid approximations, implicit assumptions, and methods used in engineering practice.

Multi-objective Decision-making. The capability to balance competing objectives such as cost, performance, and safety when no single optimal solution exists. This dimension assesses whether a model can justify trade-offs under constraints, a defining feature of engineering problem solving.

Uncertainty Handling. The capability to reason under incomplete, noisy, or dynamic information. It includes anticipating uncertainties, incorporating safety margins or fallback strategies, and generating solutions that remain robust despite ambiguity. This capability is essential for making reliable engineering decisions in real-world settings.

3.2 Problem Hierarchical Difficulty Design

Engineering problem solving involves multiple distinct capabilities, making it difficult to assess through a single task or a one-dimensional hierarchy. A clear taxonomy is therefore essential for identifying where models succeed or fail. To provide such structure, EngiBench organizes engineering tasks into three complementary levels: foundational knowledge retrieval, contextual reasoning, and open-ended modeling, each reflecting different cognitive demands. Rather than simply aggregating tasks, this framework organizes evaluation by reasoning complexity, forming a hierarchy consistent with Bloom's Taxonomy (Krathwohl, 2002).

Level 1. Tasks are well-defined and self-contained, typically requiring only single-step application of fundamental engineering formulas. They emphasize factual recall, accurate computation, and minimal contextual reasoning. This level assesses whether a model has a stable engineering knowledge base and can reliably retrieve and apply it to straightforward problems.

Level 2. Tasks require multi-step reasoning under contextual constraints such as units, physical limits, and coupled variables. Although these problems are well-defined and have unique solutions, models must interpret structured descriptions and integrate domain knowledge across steps to generate correct answers. Compared with Level 1, simple recall is insufficient; models need to handle structured complexity to generate correct solutions.

Level 3. Tasks reflect open-ended engineering challenges with uncertainty, incomplete information, and conflicting objectives. They require the full engineering problem-solving capability. Unlike Level 1 and Level 2, problems do not have a single correct answer, and evaluation focuses on how well models demonstrate robust and adaptive reasoning under open-ended conditions.

3.3 Dataset Construction

Data Sources. We collect data from three primary sources: problems selected from existing public benchmarks, university educational materials, and modeling competitions. These problems reflect

the intended hierarchy of difficulty described above and address the lack of open-ended engineering modeling problems with expert-defined evaluation criteria in existing datasets.

Construction Process. Levels 1 and 2 contain structured engineering problems with standard answers, drawn from general-domain benchmarks such as SuperGPQA (Du et al., 2025), MMLU (Hendrycks et al., 2021), MATH (Hendrycks et al., 2021), GSM8k (Cobbe et al., 2021), Orca-Math (Mitra et al., 2024), HARP (Yue et al., 2024), Omni-MATH (Gao et al., 2025), Big-Math (Albalak et al., 2025), and selected university resources. Although these datasets are broad in scope, all problems used in EngiBench were passed through an engineering relevance filtering procedure (Appendix D.1) to retain only questions that align with engineering knowledge. All selected problems were further standardized and validated.

Level 3 introduces the first systematic collection of open-ended engineering tasks, comprising 43 problems from major modeling competitions. Each problem includes official scoring rubrics and reference solutions provided by top-ranking competition winners. All task rewrites and scoring rubrics were finalized by domain experts, with LLMs providing auxiliary support during intermediate steps, ensuring clarity, rigor, and reliable assessment (see Appendix D.2 and Appendix F.2).

Problem Annotation and Quality Control. Level 3 problems and their scoring rubrics were expert-reviewed by 20 PhD students and engineering professionals, with LLMs used only as auxiliary tools. From nearly 1,000 competition questions, we retained only those with official rubrics and performed extensive text-based reformulation of formulas, tables, and diagrams. The released scoring scripts implement these expert-defined rubrics for reproducible downstream evaluation. All annotation and quality control in this section are limited to problem and rubric construction (see Appendix F.2.1), while model response scoring is described in Section 4.1.

Coverage and Classification. EngiBench spans three subfields: Systems & Control (939 problems), Physical & Structural (354 problems), and Chemical & Biological (467 problems). This categorization reflects differences in problem focus, underlying domain knowledge, and the reasoning processes required to solve them.

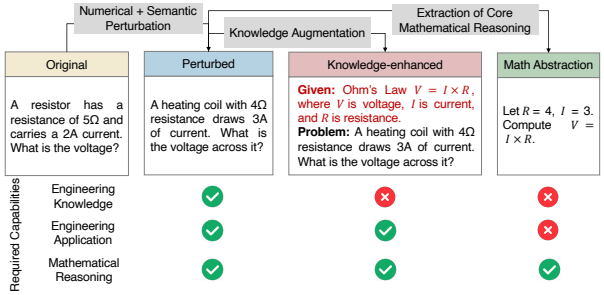


Figure 2: We create variants of the original problem to test different reasoning skills. *Perturbed* changes context and numbers to assess robustness. *Knowledge-enhanced* adds domain knowledge to focus on reasoning. *Math Abstraction* isolates engineering knowledge to test math ability. Each version targets specific capabilities.

3.4 Controlled Problem Variants

Evaluating LLMs on engineering tasks requires more than measuring overall accuracy. A correct answer may arise from data memorization rather than reasoning (Huang et al., 2025; Zhang et al., 2024; Mirzadeh et al., 2025; Srivastava et al., 2024; Gulati et al., 2024), while an incorrect answer may reflect missing domain knowledge, weak mathematical skills, or failures in interpreting engineering constraints. Without separating these factors, accuracy alone provides limited diagnostic value.

To enable deeper analysis, each problem is rewritten into three controlled variants derived from the original form (Figure 2). (1) The *perturbed variant* introduces numerical and semantic changes to assess robustness and reduce possible overlap with pretraining data. (2) The *knowledge-enhanced variant* adds essential domain information such as formulas, constants, and key definitions so that errors caused by missing knowledge can be distinguished from reasoning failures. (3) The *math abstraction variant* removes contextual and domain-specific elements while preserving the underlying mathematical structure, allowing us to isolate mathematical reasoning and quantify how much engineering context affects performance.

These controlled variants provide a structured way to distinguish why a model succeeds or fails, giving a capability-oriented evaluation of engineering problem solving. For Level 1 and Level 2, all three variants are constructed systematically from the original problem. For Level 3, the open-ended nature and inherent complexity make knowledge-enhanced and math abstraction variants impractical, so only the perturbed variant is included.

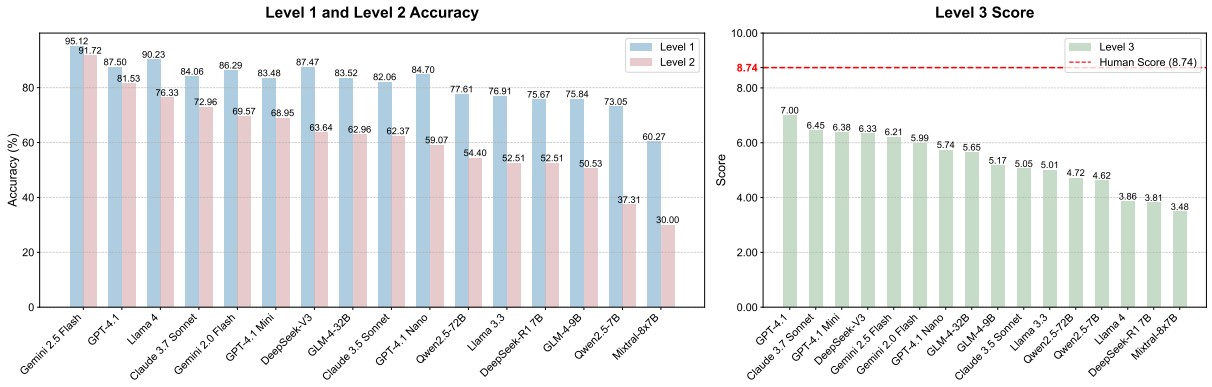


Figure 3: Overview of model performance across engineering reasoning tasks. The left subfigure shows model accuracy on Level 1 and Level 2 tasks, while the right subfigure presents scores on Level 3 open-ended tasks, with the human expert score indicated by the red line.

4 Experiments

4.1 Experiment Setup

Evaluated LLMs. As the first batch, 16 LLMs were evaluated under the zero-shot setting, covering a representative range of model types. Specifically, we include: (1) closed-source models such as GPT-4.1, GPT-4.1 Mini, and GPT-4.1 Nano from OpenAI (Achiam et al., 2023); Claude 3.7 Sonnet and Claude 3.5 Sonnet from Anthropic (Anthropic, 2024b,a); and Gemini 2.5 Flash and Gemini 2.0 Flash from Google DeepMind (Team et al., 2023, 2024); (2) open-source models, including GLM-4-32B and GLM-4-9B from THUDM (GLM et al., 2024), Qwen2.5-72B and Qwen2.5-7B from Alibaba (Yang et al., 2024), Llama 4 Maverick (referred to as Llama 4) and Llama 3.3-70B (referred to as Llama 3.3) from Meta (Grattafiori et al., 2024), and DeepSeek-V3-671B (referred to as DeepSeek-V3) and DeepSeek-R1-Distill-Qwen-7B (referred to as DeepSeek-R1 7B) from DeepSeek (Guo et al., 2025), Mixtral-8x7B-Instruct-v0.1 (referred to as Mixtral 8x7B) from Mistral AI (Jiang et al., 2024). This selection spans diverse model sizes, training paradigms, and accessibility levels. We ensured consistent formatting and output parsing across all models.

Evaluation protocols. Level 1 and Level 2 consist of well-defined problems with unique solutions and are evaluated using binary scoring. Evaluation consistency is verified through multi-model cross-checking and random human spot checks. Further details are provided in Appendix F.1. Level 3 tasks are open-ended and are evaluated using a rubric-based framework derived from official criteria and refined by domain experts. Scoring is performed by LLMs following the same rubrics, and all Level 3 scores are subsequently reviewed and calibrated by human annotators following the same criteria. Further details are provided in Appendices F.2.2 and G.1.

Also, we introduce human scores for Level 3 tasks for comparison with LLMs' performance. We obtain human scores from two sources: award-winning competition submissions (original version) and manual solutions by top-performing students for the perturbed variant. All human and LLM responses are evaluated using the same rubric to ensure consistency and fairness.

4.2 Results

4.2.1 Overall

Model stratification and design validation. Model performance exhibits a clear downward trend from Level 1 to Level 3, demonstrating the effectiveness of our hierarchical difficulty design. As shown in Figure 3, most models achieve high accuracy on Level 1, perform moderately on Level 2, and show a clear performance decline on Level 3. This trend indicates that our hierarchical framework successfully separates problems by cognitive difficulty, with each level reflecting distinct capability thresholds. The results validate that a multi-level design is necessary to capture the full range of engineering problem-solving capabilities.

Evaluating high-level engineering reasoning. Level 3 is designed to assess high-level engineering reasoning that goes beyond formulaic computation. Unlike Level 1 and Level 2, which focus on structured problem solving, Level 3 features open-ended and underspecified tasks that better reflect real-world engineering challenges. The sharp performance drop at this level reveals the current

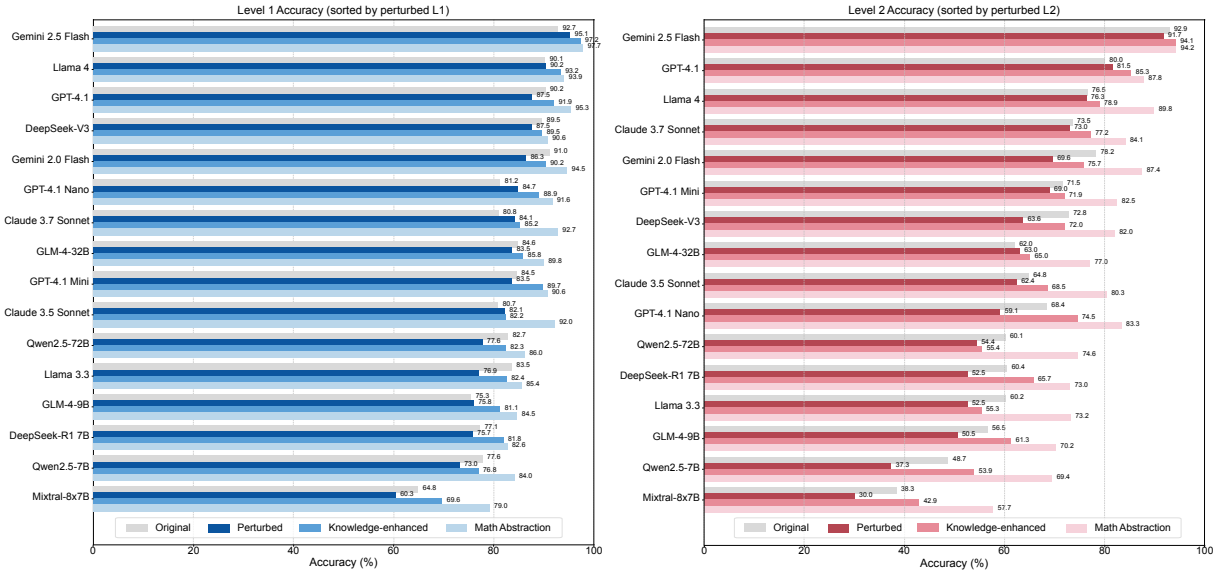


Figure 4: Accuracy of LLMs on Level 1 (left) and Level 2 (right) across the Original, Perturbed, Knowledge-enhanced, and Math Abstraction variants. Drops under the Perturbed variant reflect sensitivity to input changes, while gains on the latter two indicate that models benefit from added knowledge or simplified formulations.

limitations of LLMs in handling such complex scenarios. Besides, the gap between LLMs and human experts at Level 3 also reveals a key deficiency in high-level engineering capabilities. All evaluated models score well below the human expert, who achieves an average of 8.74, indicating that current LLMs are still far from reliably handling complex engineering problems. This underscores the need for further research to bridge this gap.

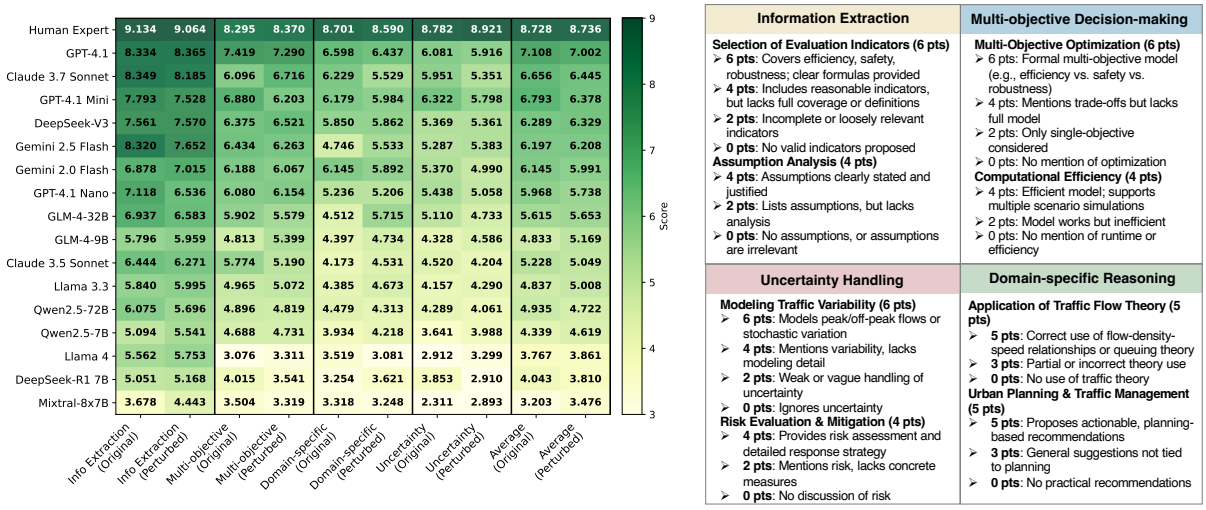
Smaller-scale LLMs struggle with complex tasks. While all LLMs show room for improvement on complex, open-ended engineering tasks, smaller-scale LLMs exhibit significantly greater limitations. As task complexity increases, performance disparities widen. At Level 1, most models still cluster within 70–90%. But at Level 2, leading models such as GPT-4.1 and Gemini 2.5 Flash achieve accuracies above 80%, whereas DeepSeek-R1 7B reaches only about 52% and other lightweight models often fall below 40%. This divergence is most evident at Level 3, where state-of-the-art models approach scores of 7.0, while lightweight models remain under 4.0. These results show that EngiBench is not saturated and continues to distinguish models across scales.

Robustness and contamination risk. Some LLMs may achieve high scores not through internal reasoning, but due to overlap with pretraining data. To reveal this, we use a perturbed variant that applies minor contextual and numerical changes but keep the core structure unchanged. As shown in Figure 4, model performance remains relatively stable on Level 1 but drops sharply on Level 2. For example, on Level 2, accuracy decreases by 9.3% for GPT-4.1 Nano, 11.4% for Qwen2.5-7B, and 8.3% for Mixtral-8x7B. These declines suggest a stronger reliance on surface-level pattern matching, rather than robust reasoning, highlighting the role of perturbation-based evaluation in diagnosing overestimated capabilities.

4.2.2 Performance for Level 1 & Level 2 Tasks

Knowledge Enhancement Improves Accuracy. Adding explicit domain knowledge consistently improves accuracy across all levels, especially for weaker models. As shown in Figure 4, models perform better on knowledge-enhanced variants than on perturbed inputs. These gains suggest two main failure sources: lacking essential domain knowledge or failing to apply it correctly during reasoning. Providing explicit knowledge therefore offers a clear diagnostic signal that helps distinguish knowledge deficits from reasoning errors, which is a key capability for engineering evaluation.

Math Abstraction Further Improves Performance. LLMs perform even better when engineering problems are rewritten into purely mathematical form, removing contextual details. As shown in Figure 4, most models achieve their highest accuracy under this variant, especially smaller models that struggle with contextual interpretation. This pattern suggests that the main challenge in engineering tasks is not computation, but the earlier step of translating natural-language descrip-



(a) Level 3 Model Evaluation.

Information Extraction

Selection of Evaluation Indicators (6 pts)
- **6 pts**: Covers efficiency, safety, robustness; clear formulas provided
- **4 pts**: Includes reasonable indicators, but lacks full coverage or definitions
- **2 pts**: Incomplete or loosely relevant indicators
- **0 pts**: No valid indicators proposed

Assumption Analysis (4 pts)
- **4 pts**: Assumptions clearly stated and justified
- **2 pts**: Lists assumptions, but lacks analysis
- **0 pts**: No assumptions, or assumptions are irrelevant

Multi-objective Decision-making

Multi-Objective Optimization (6 pts)
- 6 pts: Formal multi-objective model (e.g., efficiency vs. safety vs. robustness)
- 4 pts: Mentions trade-offs but lacks full model
- 2 pts: Only single-objective considered
- 0 pts: No mention of optimization

Computational Efficiency (4 pts)
- 4 pts: Efficient model; supports multiple scenario simulations
- 2 pts: Model works but inefficient
- 0 pts: No mention of runtime or efficiency

Uncertainty Handling

Modeling Traffic Variability (6 pts)
- **6 pts**: Models peak/off-peak flows or stochastic variation
- **4 pts**: Mentions variability, lacks modeling detail
- **2 pts**: Weak or vague handling of uncertainty
- **0 pts**: Ignores uncertainty

Risk Evaluation & Mitigation (4 pts)
- **4 pts**: Provides risk assessment and detailed response strategy
- **2 pts**: Mentions risk, lacks concrete measures
- **0 pts**: No discussion of risk

Domain-specific Reasoning

Application of Traffic Flow Theory (5 pts)
- **5 pts**: Correct use of flow-density-speed relationships or queuing theory
- **3 pts**: Partial or incorrect theory use
- **0 pts**: No use of traffic theory

Urban Planning & Traffic Management (5 pts)
- **5 pts**: Proposes actionable, planning-based recommendations
- **3 pts**: General suggestions not tied to planning
- **0 pts**: No practical recommendations

(b) Scoring rubric example.

Figure 5: Level 3 Model Evaluation and Scoring Rubric. This figure summarizes Level 3 evaluation results and scoring standards. Subfigure (a) reports average model scores across four capabilities under both original and perturbed inputs. Subfigure (b) shows an example rubric outlining scoring criteria across capability dimensions.

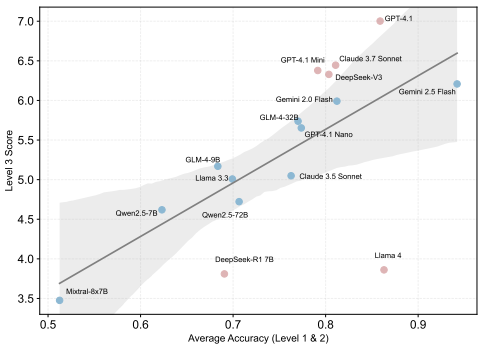


Figure 6: Correlation between structured tasks (Level 1&2) and open-ended tasks (Level 3).

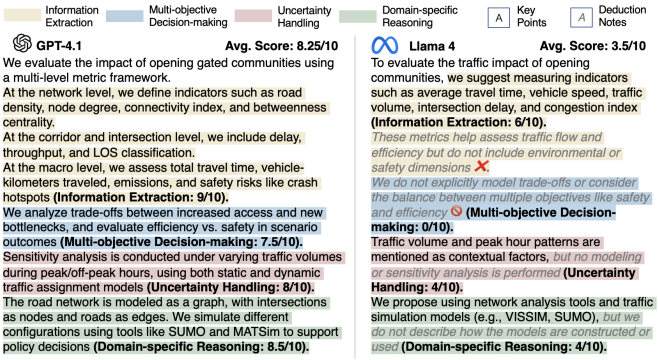


Figure 7: Case study showing why Llama 4 received low Level 3 scores.

tions into well-structured mathematical formulations. This underscores the importance of evaluating the reasoning steps that precede formula application, as these upstream processes are not captured by traditional math benchmarks.

Smaller Models Are More Sensitive to Input Variants. Smaller-scale LLMs exhibit much larger performance fluctuations across input versions, indicating limited generalization and unstable reasoning. As shown in Figure 4, in Level 2, Qwen2.5-7B drops by 11.4% under the perturbed variant, yet gains 16.6% with added domain knowledge and another 15.5% under math abstraction. In contrast, Gemini 2.5 Flash remains highly stable: its accuracy decreases by only 1.2% under the perturbed version and increases by 2.4% and 2.5% under the knowledge-enhanced and math abstraction variants, respectively. This comparison shows that smaller models are more sensitive to input formulation and tend to rely on surface patterns rather than consistent, context-aware reasoning.

4.2.3 Performance for Level 3 Tasks

Dimension-wise and model-wise performance. As shown in Figure 5a, human experts lead across all four dimensions with a balanced capability profile. In contrast, LLMs show uneven performance across the four dimensions. They handle redundant information extraction relatively well and perform moderately on multi-objective decision-making but struggle with domain-specific reasoning and uncertainty handling. This pattern indicates that their abilities are imbalanced, with clear deficiencies in key engineering-oriented skills. Results also demonstrate that model performance correlates with scale and accessibility. Larger, closed-source models like GPT-4.1 and Claude 3.7 Sonnet, consis-

tently achieve average scores above 6. In contrast, smaller open-source models (e.g., Mixtral-8x7B) average below 4, with common omissions in aspects such as trade-off reasoning and uncertainty consideration.

Correlation analysis. To quantify this trend, Figure 6 illustrates the relationship between model performance on structured tasks (Levels 1 & 2) and open-ended tasks (Level 3). Overall, we observe a clear positive correlation: *models that achieve higher accuracy on structured tasks tend to also perform well on open-ended tasks*, suggesting a general consistency across task types. At the same time, some models deviate from this general trend. GPT-4.1, Claude 3.7 Sonnet, and DeepSeek-V3 show notably stronger performance on Level 3 than their results on Levels 1 and 2 would suggest, indicating more advanced reasoning and modeling abilities than what structured tasks alone reveal.

In contrast, models like Llama 4 perform pretty well on structured tasks but falter on open-ended ones, revealing weak high-level reasoning. Figure 7 illustrates this gap: Llama 4 scores 0 in multi-objective decision-making due to missing trade-off analysis, while GPT-4.1 provides a structured evaluation and scores 7.5. A similar shortfall also appears in uncertainty handling. These examples show that Llama 4 can recall facts but struggles to apply them in complex, judgment-based scenarios.

5 Conclusion

We introduce **EngiBench**, a benchmark for evaluating LLMs on engineering problem solving across increasing levels of complexity. Our results show that while current models perform well on foundational knowledge retrieval, their performance declines significantly in multi-step contextual reasoning tasks, due to both domain knowledge gaps and limited mathematical reasoning. On open-ended modeling tasks, even the strongest models fall short of human-level performance, revealing persistent limitations in high-level reasoning, trade-off analysis, and uncertainty handling. These findings underscore the need for LLMs to move beyond pattern matching and toward deeper reasoning capabilities for real-world engineering applications.

Limitations

While EngiBench provides the first systematic evaluation of LLMs on real-world engineering problems, covering multi-level tasks, variant-based reasoning diagnostics, and open-ended modeling, several limitations remain that we plan to address in future work.

Multimodal Support. Many real-world engineering problems involve visual elements such as diagrams, schematics, or structured tables. The current version of EngiBench does not include multimodal tasks, as most existing LLMs still lack stable and consistent multimodal input capabilities. To avoid confounding engineering reasoning performance with visual processing variability and to ensure fair and comparable evaluation across models, we restrict all inputs to text-only formats.

Long-Context Support. Some engineering tasks involve long problem descriptions or extensive tabular data that exceed the input length limits of current LLMs. To avoid unfair model truncation effects and ensure uniform evaluation settings, such problems are excluded from this version.

Human-in-the-loop Construction. Building the dataset involves substantial human effort, including problem collection, answer generation, and variant validation. This ensures data quality and alignment with engineering standards, but also reflects the significant manual effort behind the benchmark.

Acknowledgements

This work was supported in part by the Ministry of Education and Science of Bulgaria (support for INSAIT, part of the Bulgarian National Roadmap for Research Infrastructure), the Shenzhen Institute of Artificial Intelligence and Robotics for Society (AIRS), the Shenzhen Key Laboratory of Crowd Intelligence Empowered Low-Carbon Energy Network (No. ZDSYS20220606100601002), the National Natural Science Foundation of China (No. 72331009), and the Ministry of Education (MOE), Republic of Singapore, under Grant AcRF Tier 1 (RG59/22).

We would like to express our sincere gratitude to Mo Chen, Zixuan Cui, Rui Jin, Kaicheng Li, Zhuoqi Li, Jili Tu, Bihua Wen, and Jiaxiang Xie for their valuable contributions to the construction, annotation, and validation of the EngiBench.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Alon Albalak, Duy Phung, Nathan Lile, Rafael Rafailov, Kanishk Gandhi, Louis Castricato, Anikait Singh, Chase Blagden, Violet Xiang, Dakota Mahan, and 1 others. 2025. Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models. *arXiv preprint arXiv:2502.17387*.

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. *arXiv preprint arXiv:1905.13319*.

Anthropic. 2024a. Claude-3 family: Opus, sonnet, haiku. Available at: https://assets.anthropic.com/m/61e7d27f8c8f5919/original/Claude-3-Model-Card.pdf.

Anthropic. 2024b. Claude-3.5 sonnet. Available at: https://www.anthropic.com/news/claude-3-5-sonnet.

Daman Arora, Himanshu Singh, and 1 others. 2023. Have llms advanced enough? a challenging problem solving benchmark for large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7527–7543.

Ge Bai, Jie Liu, Xingyuan Bu, Yancheng He, Jiaheng Liu, Zhanhui Zhou, Zhuoran Lin, Wenbo Su, Tiezheng Ge, Bo Zheng, and 1 others. 2024. Mt-bench-101: A fine-grained benchmark for evaluating large language models in multi-turn dialogues. *arXiv preprint arXiv:2402.14762*.

W Beitz, G Pahl, and K Grote. 1996. Engineering design: a systematic approach. *Mrs Bulletin*, 71(30):3.

Yuheng Cheng, Huan Zhao, Xiyuan Zhou, Junhua Zhao, Yuji Cao, Chao Yang, and Xinlei Cai. 2025. A large language model for advanced power dispatch. *Scientific Reports*, 15(1):8925.

Anoop Cherian, Kuan-Chuan Peng, Suhas Lohit, Joanna Matthiesen, Kevin Smith, and Josh Tenenbaum. 2024. Evaluating large vision-and-language models on children's mathematical olympiads. *Advances in Neural Information Processing Systems*, 37:15779–15800.

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

Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan. 2024. Investigating data contamination in modern benchmarks for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8706–8719, Mexico City, Mexico. Association for Computational Linguistics.

Xinrun Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, King Zhu, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, and 1 others. 2025. Supergpqa: Scaling llm evaluation across 285 graduate disciplines. *arXiv preprint arXiv:2502.14739*.

Clive L Dym, Alice M Agogino, Ozgur Eris, Daniel D Frey, and Larry J Leifer. 2005. Engineering design thinking, teaching, and learning. *Journal of engineering education*, 94(1):103–120.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, Zhengyang Tang, Benyou Wang, Daoguang Zan, Shanghaoran Quan, Ge Zhang, Lei Sha, Yichang Zhang, Xuancheng Ren, Tianyu Liu, and Baobao Chang. 2025. Omni-MATH: A universal olympiad level mathematic benchmark for large language models. In *The Thirteenth International Conference on Learning Representations*.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, and 1 others. 2024. Omni-math: A universal olympiad level mathematic benchmark for large language models. *arXiv preprint arXiv:2410.07985*.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, and 1 others. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*.

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

Aryan Gulati, Brando Miranda, Eric Chen, Emily Xia, Kai Fronsdal, Bruno de Moraes Dumont, and Sanmi Koyejo. 2024. Putnam-axiom: A functional and static benchmark for measuring higher level mathematical reasoning. In *The 4th Workshop on Mathematical Reasoning and AI at NeurIPS'24*.

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

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

Kaixuan Huang, Jiacheng Guo, Zihao Li, Xiang Ji, Jiawei Ge, Wenzhe Li, Yingqing Guo, Tianle Cai, Hui Yuan, Runzhe Wang, and 1 others. 2025. Math-perturb: Benchmarking llms' math reasoning abilities against hard perturbations. *arXiv preprint arXiv:2502.06453*.

Yiming Huang, Zhenghao Lin, Xiao Liu, Yeyun Gong, Shuai Lu, Fangyu Lei, Yaobo Liang, Yelong Shen, Chen Lin, Nan Duan, and 1 others. 2023. Competition-level problems are effective llm evaluators. *arXiv preprint arXiv:2312.02143*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, and 1 others. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. 2024. Prometheus: Inducing fine-grained evaluation capability in language models. In *The Twelfth International Conference on Learning Representations*.

David R Krathwohl. 2002. A revision of bloom's taxonomy: An overview. *Theory into practice*, 41(4):212–218.

Ming Li, Jike Zhong, Tianle Chen, Yuxiang Lai, and Konstantinos Psounis. 2024. Eee-bench: A comprehensive multimodal electrical and electronics engineering benchmark. *arXiv preprint arXiv:2411.01492*.

Zhenwen Liang, Tianyu Yang, Jipeng Zhang, and Xiangliang Zhang. 2023. Unimath: A foundational and multimodal mathematical reasoner. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7126–7133.

Yong Lin, Shange Tang, Bohan Lyu, Jiayun Wu, Hongzhou Lin, Kaiyu Yang, Jia Li, Mengzhou Xia, Danqi Chen, Sanjeev Arora, and 1 others. 2025. Goedel-prover: A frontier model for open-source automated theorem proving. *arXiv preprint arXiv:2502.07640*.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. 2024. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *The Twelfth International Conference on Learning Representations*.

Pingchuan Ma, Tsun-Hsuan Wang, Minghao Guo, Zhiqing Sun, Joshua B Tenenbaum, Daniela Rus, Chuang Gan, and Wojciech Matusik. 2024. Llm and simulation as bilevel optimizers: A new paradigm to advance physical scientific discovery. *arXiv preprint arXiv:2405.09783*.

Yujun Mao, Yoon Kim, and Yilun Zhou. 2024. CHAMP: A competition-level dataset for fine-grained analyses of LLMs' mathematical reasoning capabilities. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13256–13274, Bangkok, Thailand. Association for Computational Linguistics.

Seyed Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2025. GSM-symbolic: Understanding the limitations of mathematical reasoning in large language models. In *The Thirteenth International Conference on Learning Representations*.

Arindam Mitra, Hamed Khanpour, Corby Rosset, and Ahmed Awadallah. 2024. Orca-math: Unlocking the potential of slms in grade school math. *arXiv preprint arXiv:2402.14830*.

Nayantara Mudur, Hao Cui, Subhashini Venugopalan, Paul Raccuglia, Michael P Brenner, and Peter Norgaard. 2025. Feabench: Evaluating language models on multiphysics reasoning ability. *arXiv preprint arXiv:2504.06260*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094.

ZZ Ren, Zhihong Shao, Junxiao Song, Huajian Xin, Haocheng Wang, Wanjia Zhao, Liyue Zhang, Zhe Fu, Qihao Zhu, Dejian Yang, and 1 others. 2025. Deepseek-prover-v2: Advancing formal mathematical reasoning via reinforcement learning for subgoal decomposition. *arXiv preprint arXiv:2504.21801*.

Oscar Sainz, Jon Ander Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. 2023. NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

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

Saurabh Srivastava, Anto PV, Shashank Menon, Ajay Sukumar, Alan Philipose, Stevin Prince, Sooraj Thomas, and 1 others. 2024. Functional benchmarks for robust evaluation of reasoning performance, and the reasoning gap. *arXiv preprint arXiv:2402.19450*.

Usman Syed, Ethan Light, Xingang Guo, Huan Zhang, Lianhui Qin, Yanfeng Ouyang, and Bin Hu. 2024. Benchmarking the capabilities of large language models in transportation system engineering: Accuracy, consistency, and reasoning behaviors. *arXiv preprint arXiv:2408.08302*.

Zhengyang Tang, Chenyu Huang, Xin Zheng, Shixi Hu, Zizhuo Wang, Dongdong Ge, and Benyou Wang. 2024. Orlm: Training large language models for optimization modeling. *arXiv preprint arXiv:2405.17743*.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, and 1 others. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, and 1 others. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. 2024a. Measuring multimodal mathematical reasoning with MATH-vision dataset. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Xinlei Wang, Maike Feng, Jing Qiu, Jinjin Gu, and Junhua Zhao. 2024b. From news to forecast: Integrating event analysis in llm-based time series forecasting with reflection. *Advances in Neural Information Processing Systems*, 37:58118–58153.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024c. MMLU-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Kaiyu Yang, Aidan Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan J Prenger, and Animashree Anandkumar. 2023. Leandojo: Theorem proving with retrieval-augmented language models. *Advances in Neural Information Processing Systems*, 36:21573–21612.

Albert S Yue, Lovish Madaan, Ted Moskovitz, DJ Strouse, and Aaditya K Singh. 2024. Harp: A challenging human-annotated math reasoning benchmark. *arXiv preprint arXiv:2412.08819*.

Hugh Zhang, Jeff Da, Dean Lee, Vaughn Robinson, Catherine Wu, William Song, Tiffany Zhao, Pranav Raja, Charlotte Zhuang, Dylan Slack, and 1 others. 2024. A careful examination of large language model performance on grade school arithmetic. *Advances in Neural Information Processing Systems*, 37:46819–46836.

Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. 2022. minif2f: a cross-system benchmark for formal olympiad-level mathematics. In *International Conference on Learning Representations*.

Xiyuan Zhou, Yan Xu, Junhua Zhao, and Rui Zhang. 2026. Large language model applications in power systems: A comprehensive review and outlook. *Journal of Modern Power Systems and Clean Energy*.

Xiyuan Zhou, Huan Zhao, Yuheng Cheng, Yuji Cao, Gaoqi Liang, Guolong Liu, Wenxuan Liu, Yan Xu, and Junhua Zhao. 2024. Elecbench: a power dispatch evaluation benchmark for large language models. *arXiv preprint arXiv:2407.05365*.

Appendix

Contents

A The Use of Large Language Models

In this work, LLMs were used in three ways: (1) grammar checking and language polishing during paper writing, (2) generating controlled problem variants in the benchmark construction process, and (3) serving as both the models under evaluation and auxiliary judges for rubric-based scoring.

B Ethical Considerations

This work introduces a benchmark for evaluating large language models on engineering tasks. The problems are derived from publicly available benchmarks, academic competitions, and educational materials. For open-ended tasks, human participants voluntarily contributed reference solutions and evaluation scores using publicly available rubric criteria, and personal information was collected only for inclusion in the acknowledgment section with explicit consent. The dataset does not contain sensitive data or enable harmful applications. EngiBench is designed as an evaluation framework for systematically analyzing and comparing model behaviors across diverse engineering task settings. The goal of EngiBench is to promote rigorous, transparent, and fair evaluation of language models in engineering contexts, and we affirm adherence to the ACL Code of Ethics, including principles of fairness, transparency, and research integrity.

C Future Works

While EngiBench establishes a strong foundation for evaluating LLMs on engineering problem-solving, several avenues remain for further development and expansion:

Scalability Across Engineering Domains. EngiBench currently covers three core engineering subfields—Systems & Control, Physical & Structural, and Chemical & Biological—which together span a wide range of disciplines such as Mechanical, Electrical, and Chemical/Biological Engineering. The benchmark framework is designed to be broadly applicable and adaptable across domains. In future work, we plan to expand the dataset by incorporating problems from additional engineering disciplines to further enhance data volume and subject diversity.

Multimodal Evaluation Extensions. Future versions of EngiBench will introduce a dedicated multimodal subset to evaluate models on tasks involving vision-language reasoning. This will enable systematic assessment of model performance in scenarios that demand visual interpretation alongside textual understanding.

Support for Long-Context Reasoning. We plan to extend the benchmark to include long-context engineering tasks by leveraging models with expanded context windows or hierarchical processing capabilities. This will allow for evaluation of more complex, information-rich tasks currently excluded due to input length limitations.

D Dataset Construction

D.1 Level 1 & Level 2 Extraction Process

To construct a high-quality and diverse dataset for Level 1 and Level 2, we systematically extract relevant tasks from a range of established public bench-

marks, including MMLU (Hendrycks et al., 2021), MATH (Hendrycks et al., 2021), GSM8k (Cobbe et al., 2021), Orca-Math (Mitra et al., 2024), HARP (Yue et al., 2024), Omni-MATH (Gao et al., 2025), Big-MATH (Albalak et al., 2025), and competition datasets such as cn_k12, Olympiads, AOPS forum, and AMC-AIME (Huang et al., 2023). In addition to these public sources, we also incorporate university-level engineering educational materials, including assignments, examinations, and instructor-provided teaching content, to further increase task diversity and real-world relevance.

To transform mathematical and logic-oriented problems into engineering-relevant evaluation tasks, we design a structured data processing pipeline that combines LLM-based analysis with human verification to ensure engineering relevance and classification accuracy. This pipeline ensures that all included problems align with real-world engineering semantics and reasoning demands, forming the basis for Level 1 and Level 2 in EngiBench.

The processing pipeline consists of the following steps:

1. **Engineering Relevance Filtering:** Each problem is evaluated for its applicability to engineering scenarios. Problems lacking domain relevance are excluded to maintain the technical integrity of the benchmark. The prompt used to determine whether a problem pertains to engineering is as follows:

```
"""Determine if ORIGINAL problem can be solved with ONLY mathematical knowledge (NO engineering background):
- False if requires any domain-specific knowledge
- True if solvable through pure mathematical calculations"""
```

2. **Discipline and Subfield Classification:** Relevant problems are first assigned to a specific engineering discipline (e.g., Electrical, Civil, Mechanical), and then grouped into one of EngiBench's three high-level analytical subfields: Systems & Control, Physical & Structural, or Chemical & Biological. The prompt used for assigning a problem to a specific engineering discipline is as follows:

```
"""If yes, which engineering category? (Chemical/Bioengineering/Geotechnical/Energy/Nuclear/Aerospace/Automotive/Biomedical/Civil/Control/Electrical/Industrial/Mechanical/Ocean/Environmental/Other) (Please try to avoid Other)
If not an engineering problem, return "N/A"."""
```

3. **Difficulty Level Assignment:** Based on the complexity of the required reasoning process, tasks are categorized into Level 1 or Level 2. Level 1 includes basic knowledge recall and single-step computation, while Level 2 involves multi-step inference, contextual understanding, and integration of structured constraints. The prompt used for classifying the difficulty level of a problem is as follows:

```
"""Difficulty level? (Level 1/Level 2) (Please try to avoid unknown):
- Level 1: The problem can be solved by a direct retrieval of information or by directly substituting values into a known formulai.e., the shortest possible solution path. No chaining of intermediate steps is required. (Example: Using Ohm's Law, V = IR, to directly compute voltage when given current and resistance.)
- Level 2: The problem requires multi-step reasoningmeaning that it involves chaining together several logical deductions, intermediate calculations, or systematic strategies beyond a single direct formula application. (Example: Analyzing a circuit to compute total resistance by first calculating individual branch resistances and then combining them.)"""
```

D.2 Level 3 Data Collection and Processing

To construct the Level 3 dataset in **EngiBench**, we focus on real-world, open-ended engineering tasks sourced from major mathematical modeling competitions. Specifically, we collect problems from publicly accessible archives of contests such as the China Undergraduate Mathematical Contest in Modeling (CUMCM), the Mathematical Contest in Modeling / Interdisciplinary Contest in Modeling (MCM/ICM), and the Asia and Pacific Mathematical Contest in Modeling (APMCM), covering the years 2010 to 2024.

To ensure domain relevance and evaluation consistency, we apply strict filtering criteria. We retain

only problems with clear engineering context and official scoring rubrics, and exclude those that depend heavily on complex diagrams or large external tables requiring multimodal input.

We standardize the selected problems using a structured pipeline that combines LLM-based processing with human oversight. This ensures language clarity, formatting consistency, and reduced risk of data contamination. The pipeline includes the following steps:

1. **Language Normalization:** Non-English problems are translated into fluent English using machine translation, while preserving the original engineering semantics.

2. **Expression Rewriting:** To minimize potential overlap with pretraining data, each problem is paraphrased by the LLM using diverse sentence structures and reasoning styles. While surface expressions are significantly altered, the core logic, numerical values, and solution paths remain unchanged. This step produces the *perturbed version* of each task, which is used to evaluate model robustness to superficial input variations.

3. **Multimodal Simplification:** For problems containing simple figures or tables, we extract and describe the essential information using plain text or LaTeX-formatted representations to support uniform text-based evaluation.

LLM Prompt Template: The following instruction prompt is used to guide the LLM in modifying each problem:

```
"""Assuming you are a question expert, please translate this question into English. And while ensuring that the meaning of the question remains unchanged ( preserving all logic, values, and the type of reasoning required), change the way the question is expressed by rewriting it in a way that is radically different from your regular logical structure, simulating the randomness of manual rewriting by human experts, and using as many sentence variations as possible. If there is a table, please convert it into a table form using LaTeX. For simple pictures, please describe them directly. The question is required to be converted into is in str format."""

```

To ensure the technical rigor and domain consistency of the Level 3 dataset, the entire generation and transformation process was closely supervised and iteratively revised by doctoral-level professionals with extensive expertise in engineering and mathematical modeling. These experts reviewed both the selection of source problems and the outputs produced by the language model, verifying that each task preserved the original problem's intent, accurately reflected real-world engineering reasoning, and met the standards expected in academic and professional modeling contexts.

The details of how the original contest scoring standards were mapped into EngiBench's formal scoring rubrics are described in the later subsection (see Section F.2).

D.3 Version Variant Generation

To assess model robustness and isolate specific reasoning limitations, we generate three structured variants for each Level 1 and Level 2 problem: *Perturbed*, *Knowledge-Enhanced*, and *Math Abstraction*. These variants are created through LLM prompting, with manually verified outputs to ensure alignment with the original problem logic and correctness. Below, we describe the purpose and generation criteria for each variant, accompanied by illustrative prompts.

- **Perturbed Variant.** This variant alters the surface form of the original problem—either through numerical or linguistic changes—while preserving its core logic and computational requirements. The purpose is to test whether model performance stems from true reasoning ability or superficial pattern matching. A rewriting suitability code (0–3) guides the type of modification to apply. The prompt used to generate the perturbed version and related content is as follows:

```
"""
1. Rewriting Suitability: Determine the type (0-3):
   - 0: Non-rewritable (use only when necessary)
   - 1: Modify expressions only
   - 2: Modify numerical values only
   - 3: Modify both expressions and numerical values
   // Note: All rewrites must maintain the original problem logic, engineering context, and reasoning/computational requirements

```

```
2. Rewritten Problem: Rewrite the problem according to the type of rewriting suitability above. Make the answer as difficult as possible while ensuring that the answer is correct. (Please rewrite the problem in a way that is radically different from your regular logical structure by: (1) avoiding common reasoning patterns in your model, (2) simulating human expert manual rewriting randomness, and (3) using maximum sentence variation.)
   - If 0, return original problem unchanged
   - If 1, modify expressions only
   - If 2, modify numerical values only
   - If 3, modify both expressions and values

3. Rewritten Solution Process: Provide step-by-step explanation including all reasoning, calculations and logic. Clearly state if answer can be obtained directly through formula substitution (shortest solution path without intermediate steps).

4. Rewritten Answer: Provide correct answer for rewritten problem (only types 2/3 may change)"""
```

- **Knowledge-enhanced Variant.** In this version, relevant domain knowledge—such as formulas, constants, and conversions—is explicitly provided before the original question. This allows us to evaluate whether performance deficits are due to missing knowledge or failures in application. The question itself is unchanged to isolate the impact of added context. The prompt used to generate the knowledge-enhanced version is as follows:

```
"""Knowledge-Enhanced Version:
WARNING: Make sure the final numerical answer to the converted mathematical problem is exactly the same as the original problem.

Given:
- List all relevant formulas or principles (e.g., Ohm's Law: V = I * R)
- Include physical constants with values if they are involved (e.g., g = 9.8 m/s^2)
- Specify unit conversions if applicable (e.g., 1 kWh = 3.6 * 10^6 J)
- State any assumptions or ideal conditions if necessary (e.g., assume no heat loss)

Problem:
Repeat the original question exactly as stated

Example:
Original: "Calculate voltage across 5 Ohm resistor with 2 A current"
Enhanced:
"Given:
- Ohm's Law: V = I * R
- Problem: Calculate voltage across 5 Ohm resistor with 2 A current"
"""

```

- **Math Abstraction Variant.** This version reformulates the original engineering problem into a purely mathematical format by removing all domain-specific context. Variables and operations are explicitly defined to preserve the exact calculation logic. This allows us to isolate whether reasoning failure arises from contextual understanding or mathematical ability. The prompt used to generate the math abstraction version is as follows:

```
"""Rewrite the given problem into a purely mathematical version by:

a. Remove all domain-specific context (e.g., chemistry, physics, economics).
b. Keep only numbers, variables, and math operations.
c. If domain-specific knowledge is required (e.g., reaction ratio, atomic mass), extract only the final numerical ratio or constant and include it directly.
d. Maintain the exact calculation logic and final answer.
e. Use structured symbolic language in a compact form:
- Introduce variables explicitly (e.g., "Let x = 2 and y = 3.")
- Define the calculation clearly (e.g., "Total z = min(x, y) * 2.")
- End with "Find the result."

WARNING: Make sure the final numerical answer to the converted mathematical problem is exactly the same as the original problem.

Examples:

Original: "In the reaction: Cl2 + H2 -> 2HCl, 1 mole of Cl2 reacts with 2 moles of H2. How many moles of HCl can be formed?"
```

```
converted_problem: "Let x = 1 and y = 2. They react in the ratio x : y : z = 1 : 1 : 2. Total product z = min(x, y) * 2. Find the result."

Original: "A 2m wide platform sinks 0.01m under 60kg. Estimate its length assuming water density = 1000 kg/m^3."
converted_problem: "Let x = 60 / (2 * 0.01 * 1000). Find the result." """

```

E Dataset URLs, License, and Hosting Plan

EngiBench is released for research and evaluation purposes only. All third-party artifacts are used in accordance with their original licenses. The released benchmark does not redistribute restricted original content, and commercial use of the benchmark is not permitted.

E.1 Dataset Instance Metadata

For the EngiBench dataset, each instance corresponds to an engineering task and is stored in a structured format. Instances are categorized according to task difficulty (Level 1, 2, or 3) and are constructed with multiple versions to enable fine-grained evaluation of different capabilities. The metadata fields for each level are described below:

Level 1 and Level 2 Each row in the Level 1 & 2 dataset corresponds to a closed-form or structured engineering problem, and includes the following fields:

- **problem** – Original natural language problem statement.
- **answer** – Ground truth answer to the original problem.
- **subfield** – Engineering subfield to which the problem belongs (e.g., Systems & Control).
- **category** – Topic-specific classification within the subfield (e.g., Thermodynamics).
- **difficulty** – Either `Level 1` (Foundational Knowledge Retrieval) or `Level 2` (Contextual Reasoning).
- **converted_problem** – Abstract mathematical formulation of the problem.
- **converted_problem_llm_answer** – LLM-generated response to the converted problem.
- **knowledge_enhanced_problem** – Problem reformulated with explicit formulas and domain definitions.
- **rewritten_problem** – Semantically or numerically perturbed variant of the original problem.
- **rewritten_answer** – Answer to the rewritten problem.
- **rewritten_converted_problem** – Mathematical abstraction of the rewritten problem.
- **rewritten_converted_problem_llm_answer** – LLM response to the rewritten converted problem.
- **rewritten_knowledge_enhanced_problem** – Knowledge-enhanced version of the rewritten problem.

Level 3 Each Level 3 instance represents an open-ended modeling task and includes both the problem prompt and a rubric-based evaluation across multiple capability dimensions:

- **question** – English translation of the open-ended modeling task.
- **question_modified** – Semantically perturbed variant of the task.
- **source_detail** – Source of the modeling task (e.g., MCM, coursework).
- **official_scoring_standard** – English translation of rubric criteria.
- **subfield** – Engineering subfield of the task.
- **category** – Domain or topic under which the task is categorized.
- **information_extraction_score** – Score for identifying relevant variables and constraints.
- **multi_objective_decision_score** – Score for resolving trade-offs across objectives.
- **uncertainty_handling_score** – Score for reasoning under ambiguity or variable inputs.
- **domain_specific_reasoning_score** – Score for applying engineering-specific logic and formulas.

F Evaluation Details

F.1 Level 1 & Level 2 Evaluation Details

Level 1 and Level 2 tasks consist of well-defined problems with clearly defined and unique solutions. We therefore adopt a *binary scoring* scheme, where each model-generated answer is compared against a reference answer and marked as either correct (1) or incorrect (0). Overall performance is reported in terms of accuracy.

Evaluation is conducted through an automated comparison procedure. To handle diverse numerical formats, units, and equivalent expressions, we design a standardized evaluation prompt, which is independently executed by GPT-4.1 and Gemini 2.5 Flash. For cases where the two evaluators produce inconsistent judgments, manual verification is performed to determine the final decision. The evaluator determines whether a generated answer matches the reference answer based on mathematical correctness, unit validity, and logical consistency. For numerical questions, a tolerance of $\pm 2\%$ is allowed to account for rounding effects in multi-step calculations. The evaluator is instructed to output only a Boolean decision ("True" or "False") to ensure consistent and reproducible scoring.

To verify evaluation consistency and reliability, we perform multi-model cross-checking and human spot checks. Specifically, all Level 1–2 responses are independently evaluated by GPT-4.1 and Gemini 2.5 Flash, and their results are compared. In addition, we randomly sample 300 problems for manual verification. On this subset, GPT-4.1 achieves an evaluation accuracy of 98.67%, while Gemini 2.5 Flash achieves 98.33%. These results demonstrate consistent evaluation behavior and show that the automated procedure closely aligns with deterministic answer matching for Level 1 and Level 2 tasks.

```
"""Please analyze these two answers carefully:
Generated Answer: {generated_answer}
Standard Answer: {correct_answer}

Follow these rules for comparison:
1. For calculation-focused problems:
   - If the numerical values match, consider it correct even if units are missing
   - Focus on the mathematical reasoning and final numerical result
   - Check if the core calculation steps are correct
   - For complex calculations, allow 2% tolerance in the final numerical result

2. For conceptual or unit-specific problems:
   - Units and their consistency must be considered
   - The complete answer including units is required

3. Consider the answer correct if:
   - The mathematical reasoning is sound
   - The final numerical value matches (within 2% tolerance for complex calculations)
   - For calculation-focused problems, matching units are not mandatory

Reply only with "True" or "False". """
```

F.2 Level 3 Evaluation Details

F.2.1 Rubric Construction

To enable systematic evaluation of open-ended modeling tasks, we construct structured scoring rubrics from official contest-provided scoring standards.

For each problem, the official scoring description is paired with the problem statement and provided to an LLM using a carefully designed instruction prompt to generate an initial rubric draft aligned with the four target engineering capabilities. The LLM serves only as an auxiliary tool for structuring and organizing the rubric.

Each rubric is then independently reviewed and cross-checked by two reviewers with engineering backgrounds. In cases of disagreement, final decisions are adjudicated by experts who have won national or international first prizes in engineering modeling competitions, ensuring technical rigor and accuracy.

The prompt used for rubric generation is provided below.

```
"""Assume you are an expert in problem design and grading, with deep familiarity with mathematical modeling. Please help me design an evaluation rubric for assessing large language models' engineering capabilities. Specifically, I will provide a problem and its scoring criteria, and you will tell me which of the following capabilities are assessed by this rubric: redundant_information_filtering_score, multi_objective_tradeoff_score, uncertainty_handling_score, and deep_knowledge_integration_score. In particular, please identify how each capability is assessed through specific aspects of the problem or rubric.
```

```

For each capability that is covered, provide a scoring rubric in the following format:

Problem [(Problem ID)]:
redundant_information_filtering_score: (1)(2)...
multi_objective_tradeoff_score: (1)(2)...
uncertainty_handling_score: (1)(2)...
deep_knowledge_integration_score: (1)(2)...

Notes: Each capability has a total possible score of 10 points. In other words, the total score for each listed capability should sum to 10 points. Capabilities that are not covered in this problem receive 0 points. The rubric should further specify, under each capability, the different score levels (e.g., 1 point, 2 points, 3 points, etc.) and the corresponding specific behaviors or response characteristics associated with each level.

Please read the problem and rubric carefully and provide a capability-based evaluation rubric for how this problem assesses the output of large language models."""
```

F.2.2 Rubric-based Scoring and Human Calibration

The finalized rubrics are applied to evaluate model-generated responses for Level 3 tasks. We implement an automated LLM-based scoring pipeline that assesses solution quality along multiple capability dimensions defined by the rubrics. Specifically, scores are independently produced by GPT-4.1 and Gemini 2.5 Flash, and the final score is obtained by averaging the two to reduce evaluator-specific variability.

To ensure the reliability of the reported results, LLM-generated scores are reviewed and calibrated by annotators with engineering backgrounds. The main results in this paper report calibrated scores. We note that fully automated LLM-based scoring already provides a strong and practical reference, as further supported by the consistency and validity analysis in Appendix G.1.

The prompt used to evaluate the generated answer against the rubric is as follows:

```
f"""
    You are a professional modeling competition judge with extensive experience in evaluating mathematical and engineering models.
    Please conduct a rigorous evaluation of the following answer based on the provided criteria.

    Answer to evaluate:
    {answer}

    Evaluation Criteria:
    {score_criteria}

    Please evaluate strictly according to the criteria and provide your assessment in the following JSON format:
    {{
        "score": <score between 0-10, can use decimal points for precision>,
        "reason": "Detailed evaluation breakdown:\n
                1. [Specific criterion] - [sub-score] points: [justification]\n
                2. [Specific criterion] - [sub-score] points: [justification]\n
                3. [Specific criterion] - [sub-score] points: [justification]\n
                Final score: [total] points"
    }}

    Note:
    - Break down your scoring into specific components
    - Provide clear justification for each sub-score
    - Be objective and consistent in your evaluation
    - Consider both the technical accuracy and the methodology
    """
```

F.3 Level 3 Scoring Examples

As results shown in section 4.2.3, the answers of LLMs to open-ended tasks show significant differences in four dimensions of information extraction, multi-objective decision making, uncertainty handling and domain-specific reasoning. Figure 7 preliminarily presents two scoring segments, 3 points and 8 points, for the evaluation of models' answers. To demonstrate the response performance of different segments more clearly and intuitively, we provide the following examples with more Level 3 scoring details:

1. **Full Mark (Avg. Score: 9.475):** The problem requires optimizing Hu sheep farm pen utilization under stochastic conditions (conception rates, gestation periods, litter sizes) while adhering to strict capacity constraints and cohabitation rules. The solution must minimize

expected losses from idle pens (1 unit/day) or shortages (3 units/day) through dynamic scheduling and statistical validation.

- **Information Extraction (10/10):**
  Exclusion of Deterministic Assumptions (5/5): Section 1 (System Overview) clarifies all critical parameters modeled as random variables (e.g., "$X_c \sim$ Binomial$(N_m, 0.85)$: Number of successful conceptions; $G \sim U[147, 150]$: Gestation days; $L_s \sim$ Poisson$(\lambda = 2.2)$: Liveborn lambs per ewe, with 3% mortality ($L_a = L_s \cdot 0.97$); $L_d \sim U[35, 45]$: Lactation days"). Section 3A (Scenario Generation) replaces fixed values with dynamic sampling (e.g., "For each scenario, sample: - Which ewes conceive (Bernoulli, 85%) - Their gestation (G) - Number of lambs (L_s), apply mortality - Lactation length (L_d)"). Section 6B (Robust Planning) makes flexible scheduling responsive to stochastic outcomes (e.g., "Adjust mating/rest period within allowed windows to shift animal flows.").
  Identification of Valid Uncertainty Parameters (5/5): Section 1 clarifies explicit distributions for all uncertainties (e.g., "$X_c \sim$ Binomial$(N_m, 0.85)$... $G \sim U[147, 150]$... $L_s \sim$ Poisson(2.2)... $L_d \sim U[35, 45]$"). Section 3A ensures consistent application in scenario generation (e.g., "Sample conception (Bernoulli), gestation (G), litter size (L_s), lactation (L_d)."). Section 5 (Loss Function) offers loss calculation integrating stochastic inputs (e.g., "$\mathbb{E}_{scenario}\left[\sum_t [I_t + 3S_t]\right]$").
- **Multi-objective Decision making (9.2/10):**
  Minimized Expected Loss & Output Maximization (4.5/5): Section 5 (Loss Function) contains rigorous mathematical formulation balancing idle (1 unit) vs. shortage (3 unit) costs (e.g., "Objective: $\min \mathbb{E}_{scenario}\left[\sum_t [I_t + 3S_t]\right]$ $I_t =$ Idle pens, $S_t =$ Shortages"). Section 7B (Robust Planning) includes statistical validation of tradeoffs (e.g., "Monte Carlo over Scenarios: Simulate losses across all scenarios for each candidate policy.") Section 8 (Results Table) applies quantitative comparison of policies.
  Lactation Flexibility & Fattening Tradeoffs (4.7/5): Section 1 (System Overview) makes explicit dynamic linkage between lactation and fattening (e.g., "$L_d \sim U[35, 45]$: Lactation days $\rightarrow F_d = 210 + 2 \cdot (40 - L_d)$: Fattening days"). Section 6B (Robust Planning) considers operational use of flexibility to smooth demand (e.g., "Adjust rest periods to align cohorts, minimizing 'loner pens'."). Section 3A (Scenario Generation) has stochastic integration of tradeoff (e.g., "Sample lactation length (L_d), impact on fattening (F_d).").
- **Uncertainty Handling (9.2/10):**
  Stochastic Process Models (4/4): Section 1 (System Overview) specifies explicit distributions for all stochastic parameters (e.g., "$X_c \sim$ Binomial$(N_m, 0.85)$, $G \sim U[147, 150]$, $L_s \sim$ Poisson(2.2), $L_d \sim U[35, 45]$"). Section 3A (Scenario Generation) implements full Monte Carlo (e.g., "Generate 1000 scenarios... sample conception (Bernoulli), gestation (G), litter size (L_s), lactation (L_d)."). Section 7B (Robust Planning) includes statistical validation of stochastic outcomes (e.g., "For each candidate policy, simulate losses across all scenarios.").
  Dynamic Adjustment Strategies (2.7/3): Section 1 (Fattening Calculation) establishes mechanistic linkage of lactation-fattening tradeoff (e.g., "$F_d = 210 + 2 \cdot (40 - L_d)$: Fattening days adjusted by lactation."). Section 6B (Robust Planning) makes adaptive scheduling but lacks two-way feedback (e.g., "Adjust rest periods to align cohorts... weekly rolling re-optimization.").
  Contingency Sets (2.5/3): Section 2 (Cohabitation Rules) contains hard-coded tolerance for uncertainty (e.g., "Group into largest feasible penfuls within 7-day windows."). Section 8 (Statistical Assessment) analyzes multi-scenario sensitivity (e.g., "Tabulate average loss, shortage probability, and max pen use.").
- **Domain-specific Reasoning (9.5/10):**
  Integration of Empirical Rules (4/4): Section 2 (Cohabitation Rules) adds

hard-codes industry constraints into algorithms (e.g., "7-day tolerance window for nursing ewes, lambs, and resting ewes... Group into largest feasible penfuls (14 fattening lambs/pen, 6 nursing ewes/pen)."). Section 1 (System Overview) uses embeds empirical flexibility ranges as distributions (e.g., "$L_d \sim U[35, 45]$: Lactation days... $R \sim U[18, 22]$: Adjustable rest period.") Section 6B (Robust Planning) operationalizes flexible rest rules (e.g., "Extend rest periods to align cohorts if pens would otherwise idle.").

Expected Loss Functions (3/3): Section 5 (Loss Function) has rigorous probabilistic loss aggregation (e.g., "$\min \mathbb{E}_{scenario}\left[\sum_t [I_t + 3S_t]\right]$ $I_t = \max(P_{avail} - P_{req}(t), 0)$, $S_t = \max(P_{req}(t) - P_{avail}, 0)$."). Section 8 (Results Table) quantifies loss distribution across scenarios. Section 3B (State Evolution) links stochastic occupancy to loss calculation (e.g., "For each day t: Compute $P_{req}(t)$ from sampled cohorts.").

Stochastic Optimization Algorithms (2.5/3): Section 7B (Robust Planning) applies sample average approximation (SAA) method (e.g., "Monte Carlo simulation over 1000 scenarios to evaluate policies."). Section 6A (Rolling Horizon) uses heuristic dynamic programming (e.g., "Re-optimize mating batches weekly to maximize cohabitation.").

2. **5 points (Avg. Score: 5.375):** The problem involves modeling a team coordination exercise ("Unity Drum") where 8 members control a drum's tilt by pulling ropes to bounce a ball. Key tasks include: 1. Calculating the drum's tilt angle at t=0.1s based on force/timing inputs (Table 1), accounting for initial 11cm displacement. 2. Ensuring physics-based accuracy in torque, angular acceleration, and geometric relationships.

   - **Information Extraction (7.5/10):**

     Error Source Analysis (5/6): Explicit Recognition: Timing errors-"Some members may apply force slightly before others" (Algorithm section); strength variation-"Members likely have different strengths" (Considerations). Partial Implementation: Timing logic in code (if timing$[i] \leq 0.1$) is noted but lacks vector-time coupling; force scaling (effective_force $= \frac{\text{force(member\_id}-1)}{10}$) is arbitrary.

     Physical Model Simplification (2.5/4): Justified Simplifications: "Ignores damping for short-duration calculation" (Considerations); Drum as uniform cylinder ($I = 0.5 \cdot \text{drum\_mass} \cdot r^2$). Over-Simplifications: Fixed torque angle ($\sin\left(\frac{\pi}{2}\right)$) ignores vector geometry; rope tautness assumption ("If the drum tilts too far, ropes could slack") not modeled.

   - **Multi-objective Decision making (6.5/10):**

     Tilt Angle and Force Relationship (4.5/6): Physics Foundation: Correctly derives torque ($\tau = r \cdot F \cdot \sin(\theta)$), inertia ($I = 0.5 \cdot m \cdot r^2$), and angular kinematics ($\theta = \theta_0 + \frac{1}{2}\alpha t^2$); maps rope geometry (angle_radians $= (\text{member\_id} - 1) \cdot \left(\frac{2\pi}{8}\right)$). Implementation Gaps: Timing logic (if timing$[i] \leq 0.1$) is crude; forces are binary (on/off) rather than time-interpolated; no optimization for tilt minimization (e.g., predictive control or force balancing).

     Computational Efficiency (2/4): Basic Looping-iterates over 8 members with O(1) operations per member (e.g., torque $=$ drum_radius $\cdot$ force $\cdot \sin\left(\frac{\pi}{2}\right)$). No Advanced Techniques-lacks vectorization, memoization, or scalability for larger teams.

   - **Uncertainty Handling (2/10):**

     Error Propagation Analysis (2/4): Acknowledgment Only: Mentions "members likely have different strengths and reaction times" (Considerations); suggests "extended to simulate more realistic distributions" but provides no math or implementation. No Quantification: Lacks sensitivity analysis or error bounds on tilt angle.

     Numerical Simulation Estimation (0/4): No Monte Carlo: Code calculates tilt for fixed inputs only (force_data); no randomization of force/timing or statistical

output (mean/variance).

Methodological Clarity (N/A): Physics steps are clear but irrelevant to uncertainty scoring.

- **Domain-specific Reasoning(5.5/10):**

  3D Mechanics Modeling (2.5/6): 2D Limitation: Explicitly states "our coordinate system will be planar (X and Y only)" (Key Equations); torque calculation ($\tau = r \cdot F \cdot \sin(\theta)$) ignores out-of-plane forces. Partial Physics: Correctly models drum as cylinder ($I = 0.5 \cdot m \cdot r^2$) but lacks 3D rotation dynamics.

  Model-Based Optimization Strategy (3/4): Suggestions Without Implementation: Proposes "damping term proportional to angular velocity" (Considerations); mentions "member variation" but no adaptive control (e.g., PID for tilt correction).

3. **1 point (Avg. Score: 1.25):** The problem involves coordinating multiple meteorological units (each with 1 primary and 2 secondary stations) to ensure reliable hourly weather data collection and full data sharing under strict communication constraints. Key challenges include managing transmission reliability (80% for secondaries, 100% for primaries), message capacity limits, and achieving 97% success probability within 8 minutes for primary data exchange. The goal is to determine the maximum number of units (Nmax), design transmission schemes, and compute performance metrics.

   - **Information Extraction (2/10):** High-Probability Constraint Processing (0/5): Failure to Address Probabilistic Guarantee: The answer calculates secondary transmission success as "expected number of reports received... is $4 \times 0.8 = 3.2$" (Step 4) but never models retransmissions or redundancy to achieve 97% success. The assumption of direct success ignores the problem's explicit probability requirement. Missing Critical Logic: No discussion of how to compensate for the 20% failure rate (e.g., retrying failed transmissions, acknowledgments, or error correction).

     Time Window Isolation (2/5): Interleaved Logs Without Justification: The primary and secondary transmission logs (Tables 1 2) are interleaved in the solution ("Round 1: Primary 1→2; Round 1: Secondary 1→1a"), but no protocol ensures collision avoidance (e.g., TDMA, priority scheduling). Unverified Simultaneity Assumption: The answer states "Simultaneous reception allowed during transmission" (Step 1) but doesn't prove this suffices for concurrent primary/secondary transmissions under the 8-minute constraint.

   - **Multi-objective Decision making (2/10):**

     3D Parameter Optimization (0/6): Single-Parameter Focus: The answer only optimizes for **N_max** ("$N(N-1)/28 \rightarrow N_{\text{max}} = 4$", Step 2) but ignores joint optimization of capability (no analysis of 158-character message limits or segment splitting efficiency), reliability (no adjustment for secondary station 80% success rate such as no retransmission strategy) and time (assumes 8 minutes suffice without validating secondary transmission overhead). Missed Pareto Frontier: Fails to explore tradeoffs (e.g., "Could N=5 work if secondary transmissions are reduced?").

     Resource Allocation Strategy (2/4): Equal Bandwidth Only: Primary stations follow a round-robin schedule ("1→2, 1→3, 1→4, 2→3, ...", Table 1), and secondaries transmit uniformly ("1→1a, 1→1b, 2→2a, ...", Table 2). No Prioritization: Critical objectives (e.g., ensuring 97% success) aren't prioritized in scheduling.

   - **Uncertainty Handling (0/10):**

     High-Order Probability Events (0/6): No Threshold Calculation: The answer states secondary stations have an "80% transmission/reception success rate" (Step 1) but never computes the probability of achieving 97% success (e.g., via binomial distribution for multiple retries). Misleading Metric: The "mean secondary reports received per primary station (3.2)" (Step 4) is irrelevant to the

cumulative success probability requirement.

Asymmetric Loss (0/4): No Cost Analysis: The solution ignores idle time cost (unused transmission slots due to failures) and rental loss (penalties for delayed data delivery implied by "critical rescue operations").

- **Domain-specific Reasoning (1/10):** Mixed-Integer Programming (0/5): No Optimization Model: The answer derives $N_{\max} = 4$ via a simple inequality ("$\frac{N(N-1)}{2} \leq 8$", Step 2) but lacks an objective function (e.g., "maximize N while meeting time/reliability constraints"), and omits integer constraints (N must be discrete) or linear relaxation techniques. Ad-Hoc Calculation: No use of MINLP (Mixed-Integer Nonlinear Programming) to jointly optimize N, transmission scheduling, and reliability. Fault-Tolerant Protocol Design (1/5): Basic Segmentation: Mentions "reports can split into two 50-character segments" (Step 1) but no dual verification (never states if segments are sent redundantly to different primaries) and no formal protocol (assumes secondary stations report to all primaries without fault recovery like checksums, ACKs).

G Additional Analysis

G.1 Level 3 Scoring Consistency Analysis

To further examine the reliability of our evaluation protocol, we compare three scoring variants for Level 3 tasks: human-calibrated scoring (reported as the main results), fully automated LLM-only scoring, and fully manual human scoring (human-only).

Table 2 summarizes the average scores under these three scoring settings for both original and perturbed tasks. Across models and task settings, LLM-only scores are generally close to human-calibrated scores, with differences typically within a small margin. This indicates that the proposed rubric enables reliable automated evaluation, while human calibration mainly serves to correct a limited number of edge cases and ensure maximum rigor in the reported results.

Regarding the validation of scoring consistency, the relative ordering of models remains largely consistent across the three scoring variants. Notably, the human-only scoring serves as a ground truth baseline, confirming that the trends observed in automated and calibrated scoring are robust.

These results support the practical use of fully automated scoring for large-scale benchmarking, while human calibration provides additional assurance when reporting final evaluation results.

G.2 Level 1 Analysis

Minor perturbations cause performance drops, revealing shallow generalization. Figure 8 (left) presents model accuracy on Level 1 tasks across four input variants: Original, Perturbed, Knowledge-enhanced, and Math Abstraction. When problems are perturbed through minor changes in wording or numerical values, average model accuracy drops from 82.9% to 81.5%. Notably, Llama 3.3 and Qwen2.5-72B decline by 6.6% and 5.1%, respectively. This indicates that some models exhibit limited robustness and often rely on memorized phrasing or surface patterns rather than generalizable reasoning.

Explicit knowledge prompts mitigate reasoning failures in weaker models. When explicit domain knowledge—such as formulas, constants, or unit conversions—is added to the input, accuracy improves to 85.5% on average. Weaker models benefit the most: GPT-4.1 Mini gains 6.2% and Mixtral-8x7B improves by 9.3%. This pattern suggests that many errors are not caused by a complete lack of knowledge, but rather by the inability to retrieve and apply relevant concepts without targeted prompting. Explicitly embedding domain knowledge thus serves as an effective intervention for enhancing reasoning activation.

Removing contextual language highlights semantic limitations. Performance further increases to 89.4% when problems are rewritten into abstract mathematical form, removing all contextual language. For example, Qwen2.5-7B and Mixtral-8x7B improve by 10.9% and 18.8%, respectively. This reveals that most Level 1 failures are not due to weak computational ability, but rather arise during semantic interpretation and variable binding. Once language ambiguity is removed, models can more reliably execute the required calculations, underscoring a gap between symbolic proficiency and contextual understanding.

Table 2: Level 3 average scores under three scoring variants. Human-calibrated scores are reported as the main results; LLM-only scores are produced by the automated scoring pipeline; Human-only scores are reference scores from official solutions and expert grading.

Model	Human-calibrated		LLM-only		Human-only	
	Original	Perturbed	Original	Perturbed	Original	Perturbed
Human Expert	8.728	8.736	8.697	8.702	8.735	8.729
GPT-4.1	7.108	7.002	7.053	6.972	7.208	7.043
Claude 3.7 Sonnet	6.656	6.445	6.713	6.619	6.970	6.526
GPT-4.1 Mini	6.793	6.378	6.581	6.334	6.705	6.558
DeepSeek-V3	6.289	6.329	6.358	6.264	6.396	6.386
Gemini 2.5 Flash	6.197	6.208	6.002	6.145	6.063	6.185
Gemini 2.0 Flash	6.145	5.991	5.989	5.902	6.167	6.035
GPT-4.1 Nano	5.968	5.738	5.764	5.673	6.074	5.882
GLM-4-32B	5.615	5.653	5.860	5.761	5.760	5.694
GLM-4-9B	4.833	5.169	5.079	5.227	4.822	5.168
Claude 3.5 Sonnet	5.228	5.049	5.317	5.254	5.187	5.106
Llama 3.3	4.837	5.008	4.937	4.804	4.939	5.055
Qwen2.5-72B	4.935	4.722	4.836	4.665	5.007	4.831
Qwen2.5-7B	4.339	4.619	4.580	4.591	4.362	4.669
Llama 4	3.767	3.861	3.808	3.926	3.943	3.892
DeepSeek-R1 7B	4.043	3.810	3.775	3.648	4.105	3.989
Mixtral-8×7B	3.203	3.476	3.110	3.279	3.372	3.577

G.3 Level 2 Analysis

Level 2 tasks emphasize multi-step reasoning under structured constraints, making them more sensitive to input variability. As shown in Figure 8 (right), the average model accuracy declines from 66.6% on the Original version to 61.6% on the Perturbed variant. This 5.0% drop indicates that even minor changes to semantic phrasing or numerical values can significantly disrupt reasoning chains. For instance, GPT-4.1 Nano drops by 9.3% and Qwen2.5-7B by 11.4%, revealing their limited robustness when facing contextual and structural perturbations in problem inputs.

Incorporating explicit domain knowledge helps reduce ambiguity and recover performance. With knowledge-enhanced inputs, the average accuracy rises to 68.6%, a 7.0% improvement over the perturbed baseline. Larger gains are observed for models such as GPT-4.1 Nano (+15.4%) and Qwen2.5-7B (+16.6%), suggesting that knowledge prompts assist in constraint interpretation and formula selection. However, some models such as DeepSeek-V3 show minimal improvement, implying that knowledge access alone may not compensate for limitations in multi-step reasoning capabilities.

Symbolic abstraction of Level 2 tasks into pure mathematical form results in the largest performance gains. The average accuracy increases to 79.2%, with many models gaining over 15%. This trend is especially prominent for weaker models like Qwen2.5-7B (from 37.3% to 69.4%) and Mixtral-8x7B (from 30.0% to 57.7%). These improvements confirm that many model failures stem not from computational weakness, but from difficulties parsing, organizing, and executing the reasoning steps embedded in natural language problem statements. This underscores the importance of assessing upstream cognitive processes that precede symbolic computation—dimensions often underexamined in traditional mathematical benchmarks.

G.4 Level 3 Analysis

Figure 9 presents the performance of various models across four key capabilities: Redundant Information, Multi-Objective Decision, Domain Knowledge, and Uncertainty Handling. The results are

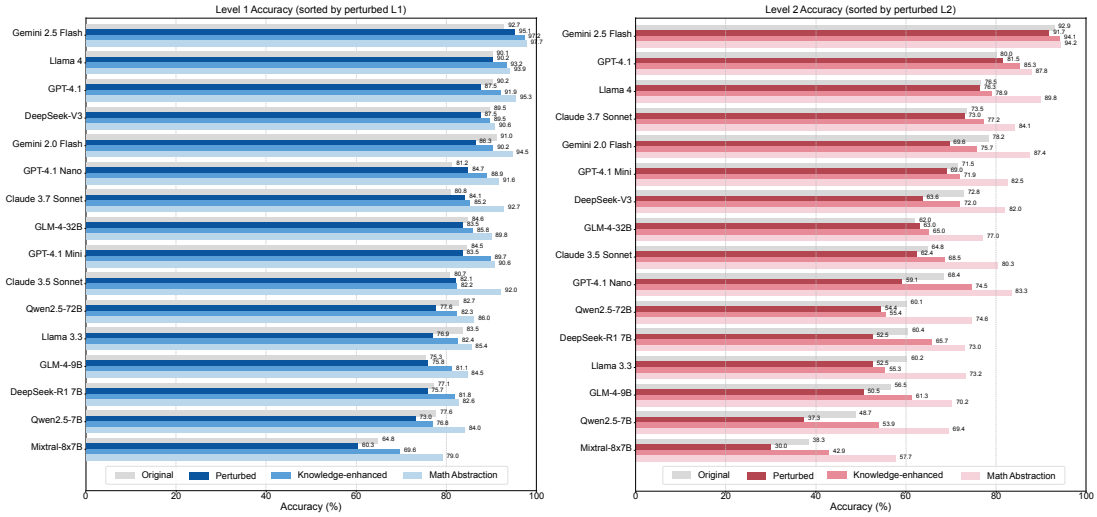


Figure 8: Accuracy of LLMs on Level 1 (left) and Level 2 (right) tasks across four variants: Original, Perturbed, Knowledge-enhanced, and Math Abstraction. Drops in the Perturbed version indicate sensitivity to input changes, while gains in the latter two show that current LLMs require external knowledge or reformulation to improve accuracy—highlighting their lack of these abilities.

further separated into *original* and *perturbed* problem formulations. Overall, human experts substantially outperform all models across all dimensions, with average scores of 8.73 (original) and 8.74 (perturbed). In contrast, LLMs demonstrate significantly lower scores, revealing a persistent gap between current LLMs' capabilities and human-level reasoning. The average model scores before and after rewriting are 5.372 and 5.341, respectively—a marginal difference of only 0.58%. This indicates that most models possess a reasonable degree of generalization, and the benchmark shows no signs of data contamination across reformulated prompts, preserving task consistency.

Based on the overall average scores, we categorize model performance into three tiers:

Tier 1 (Average Score > 6.5) This tier includes GPT-4.1, Claude 3.7 Sonnet, and GPT-4.1 Mini. These models demonstrate strong performance across all four evaluated capabilities. In particular, their scores in Information Extraction and Multi-Objective Decision often exceed 7, approaching human expert levels. Their performance in Domain Knowledge and Uncertainty Handling also remains consistently above 6, indicating robust reasoning capabilities and broad task adaptability.

Tier 2 (Average Score $\approx$ 5.5–6.5) This tier consists of DeepSeek-V3, Gemini 2.5 Flash, Gemini 2.0 Flash, GPT-4.1 Nano, and GLM-4-32B. These models achieve reasonable performance in Information Extraction and Multi-Objective Decision, but exhibit noticeable weaknesses in Domain Knowledge and Uncertainty Handling, where scores commonly fall below 6. Some models approach the 5-point threshold in these dimensions, reflecting limitations in complex reasoning and knowledge integration.

Tier 3 (Average Score < 5.5) This tier includes GLM-4-9B, Claude 3.5 Sonnet, Llama 3.3, Qwen2.5-72B, Qwen2.5-7B, Llama4, DeepSeek-R1 7B, and Mixtral-8x7B. These models consistently underperform across all four capabilities, typically scoring between 3 and 5. Their weakest areas are Domain Knowledge and Uncertainty Handling, where some models fall below 4. These results indicate substantial deficiencies in background reasoning and generalization to ambiguous or underspecified tasks.

G.5 Subfield Performance Analysis

Figures 10 and 11 present an overview of model accuracy across engineering subfields and problem variants for Level 1 and Level 2, respectively.

Model performance varies substantially across engineering subfields. Chemical and biological engineering demonstrates the strongest robustness, with large models maintaining accuracies above 85%, while structural and physical engineering achieves 70–80% and systems and control engineering performs the worst, with large models dropping to 60–70% and small models often below 40%. These results suggest that robustness to contextual perturbations is closely tied to the task characteristics: chemical and biological problems rely more on formulaic knowledge and are less sensitive to input variations, whereas systems and control problems involve more complex reasoning chains and are more vulnerable to perturbations.

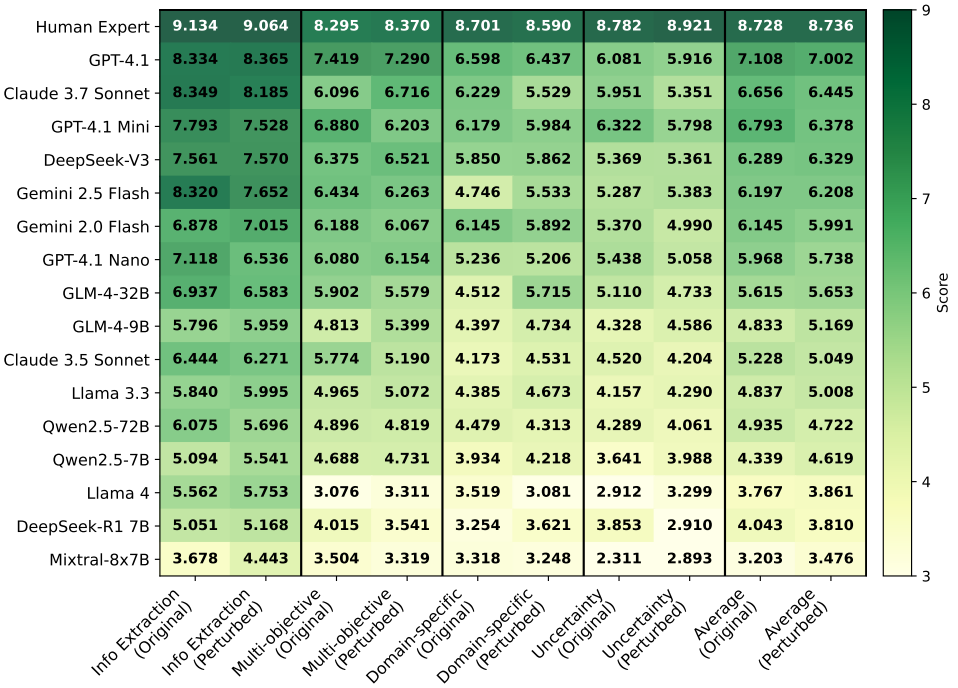


Figure 9: Level 3 Model Evaluation. The figure presents average model performance on Level 3 tasks across four capability dimensions, including information extraction, domain-specific reasoning, multi-objective decision-making, and uncertainty handling, under both original and perturbed problem formulations.

Problem variants reveal subfield-specific differences in knowledge use, reasoning, and robustness, showing that these abilities differ significantly between engineering domains. The knowledge-enhanced variant substantially improves performance in chemical and biological engineering, moderately benefits structural and physical engineering, and shows limited gains in systems and control engineering, suggesting the latter's inability to effectively leverage explicit knowledge. Similarly, the math abstraction variant, which isolates mathematical reasoning by removing context, favors chemical and biological engineering, followed by structural and physical engineering, while systems and control engineering remains the weakest. These patterns indicate that the ability to utilize injected knowledge and maintain mathematical reasoning varies considerably across subfields.

The robustness and capability differences across subfields become even more evident under higher task complexity in Level 2. Compared to Level 1, Level 2 shows larger performance drops under perturbed inputs, highlighting more severe robustness issues. The positive effects of knowledge-enhanced and math abstraction variants remain concentrated in chemical and biological engineering, with only marginal improvements in structural and physical engineering and negligible gains in systems and control engineering. This indicates that in more complex reasoning and contextual integration tasks, current large language models struggle even more to handle input perturbations, exploit external knowledge effectively, and maintain consistent reasoning, further widening the capability gap across subfields.

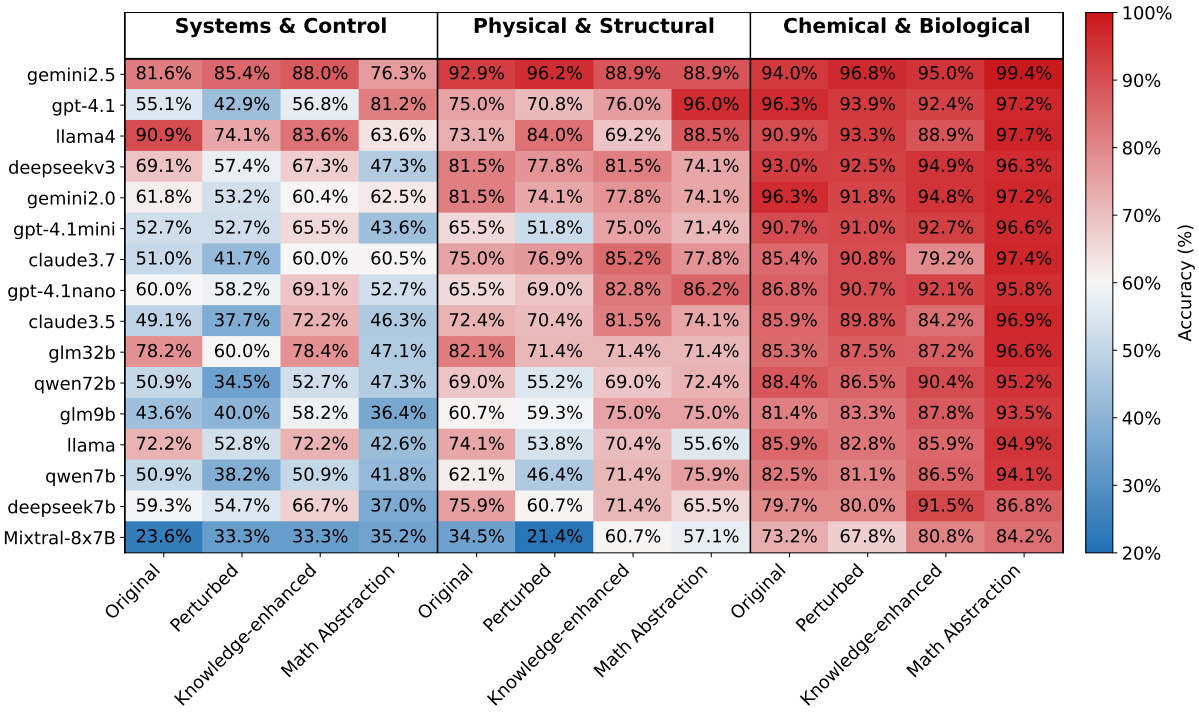


Figure 10: Accuracy across engineering subfields and problem variants in Level 1.

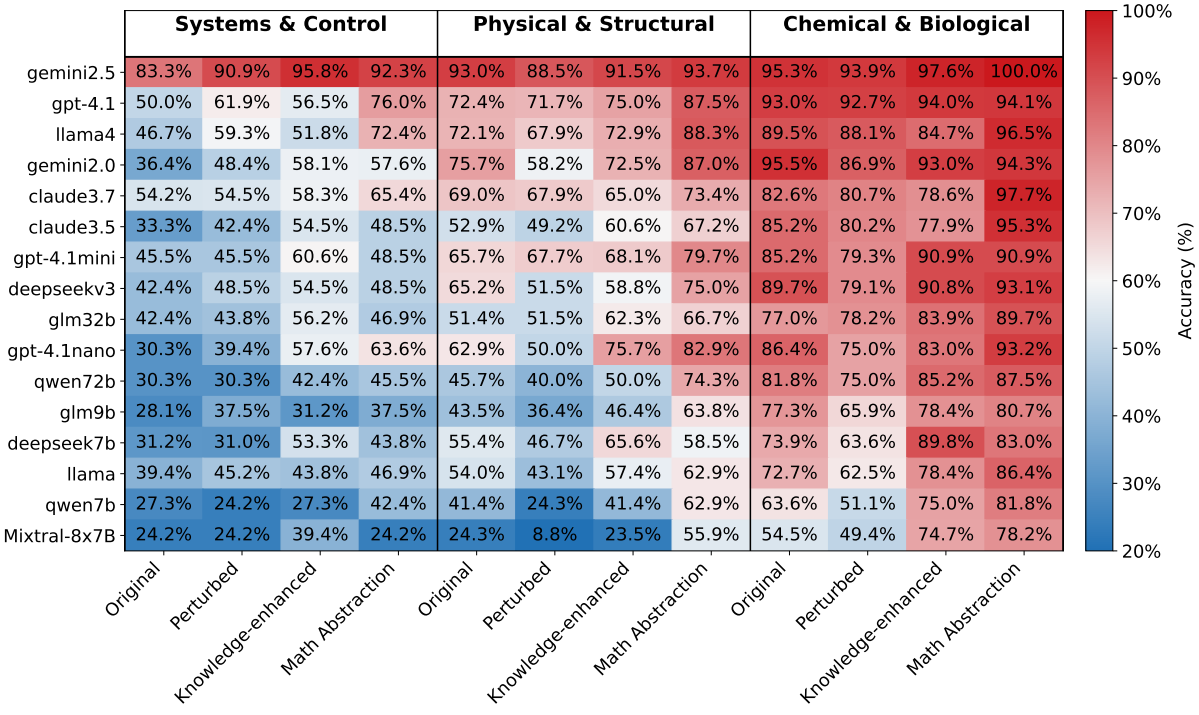


Figure 11: Accuracy across engineering subfields and problem variants in Level 2.