Difficulty Perception in the Reasoning of LLMs

Anonymous ACL submission

Abstract

How do large language models (LLMs) perceive task difficulty, and how does this perception shape their problem-solving capabilities? While existing work on the epistemology of LLMs has mainly focused on confidence, difficulty perception offers a novel perspective on models' knowledge and reasoning process. This question becomes especially relevant in the context of large reasoning models (LRMs), where test-time compute can be allocated in the form of special thinking tokens depending on problem difficulty. Our experiments with six LRMs on mathematical, competitive programming, science, and social reasoning tasks reveal that difficulty perception is not random; instead, its ranking highly correlates between different models; we present evidence that models perform better on problems they deem easier, a correspondence stronger than that of verbalized confidence. We also show that cues overstating problem difficulty in prompts can cause reasoning inefficiency. Our findings establish difficulty perception as a concept separate from verbalized confidence in model epistemology while highlighting risks from simple prompt injections with hints of difficulty.

1 Introduction

As large language models (LLMs) are widely used in various domains such as mathematics (Imani et al., 2023), the sciences (Jablonka et al., 2024; Ziems et al., 2024), coding (Guo et al., 2024), medicine (Thirunavukarasu et al., 2023), and social interactions (Liu et al., 2024), it is important to probe their knowledge and reasoning processes. Existing works on model epistemology (Steup and Neta, 2005), however, have largely focused on the elicitation and calibration of models' *confidence* (Tian et al., 2023; Xiong et al., 2024) and how models respond to uncertainty markers in prompts (Zhou et al., 2023). Yet a key gap remains in understanding how models perceive and interpret problem *difficulty* itself.

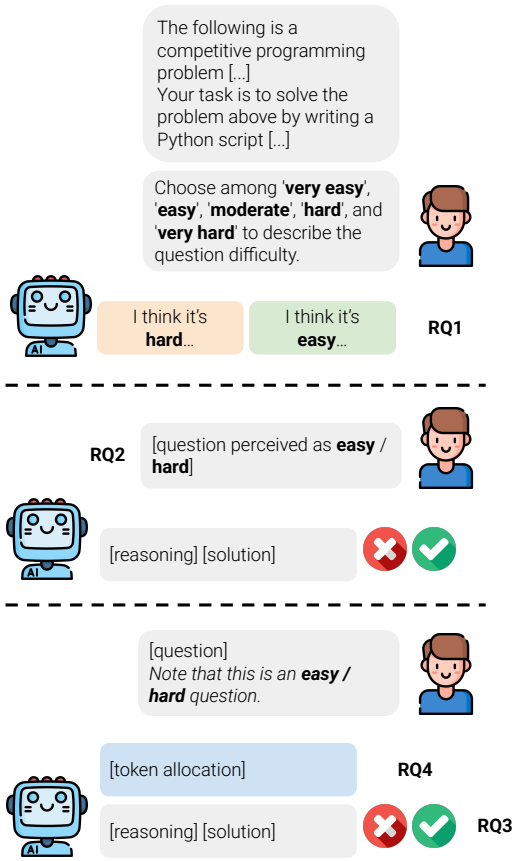

Figure 1: In this paper, using an array of math, coding, science, and social reasoning tasks, we investigate (in **RQ1**, Section 4.1) properties of models' verbalized difficulty perception, (in **RQ2**, Section 4.2) the relation between difficulty perception and performance, and (in Section 4.3) how cues of difficulty in prompt affect performance (**RQ3**) and token allocation (**RQ4**).

This gap is particularly salient with the emergence of large reasoning models (LRMs)—models which produce special reasoning tokens before their final answer (Jaech et al., 2024; Guo et al., 2025; Qwen Team, 2025b). They have achieved impressive performance on a range of tasks, thanks

to the allocation of reasoning tokens on difficult problems. Accordingly, it is crucial to understand how test-time scaling shapes models' perceived difficulty and stands against the manipulation of such perception.

In this paper, we introduce and investigate the concept of difficulty perception in a series of experiments on various tasks spanning mathematics, coding, science and social reasoning. We show that difficulty perceptions are not arbitrary but instead correlated across different models; such perceptions are separate from verbalized confidence and match models' performance better; furthermore, prompts that overstate problem difficulty result in significantly higher token usage while performance remains the same or worsens. Through these experiments, our research establishes difficulty perception as a key concept in model epistemology, while also highlighting efficiency risks from simple prompt injections with hints of difficulty.

2 Related Work

Epistemology of LLMs Epistemology is the study of knowledge (Steup and Neta, 2005), which can provide more transparency in the decision-making process of black-box systems like LLMs. In the context of LLM research, much focus has been placed on uncertainty or confidence towards their own responses. The majority of existing works studied the elicitation of model's confidence in various environments and techniques to calibrate such confidence to match actual performance (Tian et al., 2023; Xiong et al., 2024; He et al., 2025). Zhou et al. (2023) instead examined how epistemic markers in prompts can affect model capabilities. In this paper, we propose a new perspective on model epistemology: difficulty perception. Through the elicitation of difficulty ratings and prompting with difficulty markers, our methods utilize elements of previous work in a new context. Our experiments establish difficulty perception as a concept separate from verbalized confidence and enjoying a higher level of correspondence to performance. Our behavioral approach is complementary to Lugoloobi and Russell (2025), which linearly probes internal activations to find difficulty representations.

Sycophancy in LLMs LLMs are known to conform to bias factors by users rather than truth, a phenomenon known as sycophancy. Sharma et al. (2024) investigated sycophancy in situations involving a user's stated preference, questioning, opinion, and mistakes. Some works in the literature look into mitigation methods, such as finetuning on synthetic data (Wei et al., 2023) or finetuning specific components of LLMs responsible for sycophancy (Chen et al., 2024). Our study of how difficulty cues can affect models echoes existing findings on sycophancy, where such cues can also be considered part of a user's belief.

Reasoning models Recently, language models capable of producing designated reasoning or thinking tokens have gained traction, as they unlock a new dimension of scaling through such tokens (Jaech et al., 2024; Guo et al., 2025; Qwen Team, 2025b; Yang et al., 2024; Qwen Team, 2025a). Through the premise that test-time compute, i.e., the amount of reasoning tokens, can be expended according to problem difficulty, these models have claimed superior performance on olympiad mathematics, science, and coding benchmarks compared to their vanilla instruction-tuned counterparts. In this paper, we examine how the performance of these models can be manipulated through difficulty cues at various reasoning budgets.

3 Methodology

3.1 Data

In this study, we draw on four English-language datasets that collectively span mathematics, competitive programming, science, and social reasoning, yielding a diverse and comprehensive benchmark of 658 problems.

1. **AIME** (Lin, 2025). This dataset consists of 60 olympiad-level problems from the 2024 and 2025 editions of the American Invitational Mathematics Examination, covering various topics in number theory, geometry, combinatorics, etc.

2. **E2H-Codeforces** and **E2H-ARC**. This is part of the Easy2Hard benchmark (Ding et al., 2024), which contains problems in various domains accompanied by difficulty levels derived from multiple sources (e.g., human ratings and LLM leaderboards)[1]. We perform

[1]Note that while Ding et al. (2024) employed GPT-4 to verify the derived difficulty levels, our purpose is different (we perform model analysis instead of building a dataset) and we study difficulty perception of a wider range of models rather than a single one.

stratified sampling (with respect to difficulty ratings provided in the benchmark) to obtain 200 problems in the competitive programming (Codeforces) subset and another 200 in the multiple-choice science question-answering subset (ARC (Clark et al., 2018)) that resemble the original difficulty level distribution.

3. **LLM Coordination** (Agashe et al., 2025). This benchmark evaluates the ability of LLMs to coordinate in card, cooking, and maze games. We make use of the QA portion of the benchmark, consisting of 198 single-turn, multiple-choice questions involving theory-of-mind, joint planning, and environment comprehension capabilities.

3.2 Models

We consider six popular reasoning models: Gemini 2.5 Flash (Comanici et al., 2025), GPT-5 mini (OpenAI, 2025), DeepSeek R1 0528 Qwen3 8B (Guo et al., 2025), Qwen3 {14B, 32B} (Qwen Team, 2025a), and DeepSeek R1 Llama 8B (Guo et al., 2025). For each of these, we consider reasoning budgets of 0, 1024, and 8192 tokens for comprehensiveness. The non-zero budgets are implemented as soft constraints through the Gemini API and effort levels "low" and "medium" in the OpenAI API, while the reasoning tokens of the open models are cut off exactly at the indicated budgets. At the cutoff position, we manually insert the following sentence recommended by Qwen Team (2025a) and continue the generation:

```
Considering the limited time by the user,
I have to give the solution based on the
thinking directly now.\n</think>.\n\n
```

We note that for the 0-token budget, we use "minimal" effort for GPT-5 mini (no option with zero reasoning is available for this model) while Llama 3.1 8B Instruct and Qwen3 8B are used as the non-reasoning version for the Deepseek distilled models.

All inference is done with four trials (to account for random variations) and we report pass@1 for performance. See Appendix A for more details on inference.

3.3 Research Questions and Experiments

RQ1. How do models perceive difficulty? Characterizing the distribution of difficulty ratings is crucial for establishing how models internally categorize task difficulty, revealing whether models share a similar ordering of problem difficulty or exhibit divergent internal notions of what counts as "difficult." To do this, we elicit difficulty ratings on a Likert scale (Joshi et al., 2015) from models through the inclusion of the following prompt after problem statements.

```
Without solving the question above,
choose a word among 'very easy', 'easy',
'moderate', 'hard', and 'very hard'
which best describes the difficulty of
the question.
```

For each pair of model and problem, we sample eight times and determine the difficulty perception as the majority vote, i.e., self-consistency (Wang et al., 2023). We also compare perceived difficulty to verbalized confidence with similar prompts: "the difficulty of the question", "easy", and "hard" are replaced respectively by "your confidence in successfully solving the question", "low", and "high".

Additionally, we also elicit difficulty ratings conditioned on models' reasoning processes. Specifically, we prompt models with a conversation including a problem statement, the models' solution, and a second user message asking

```
Now that you have answered the question,
choose a word among 'very easy', 'easy',
'moderate', 'hard', and 'very hard'
which best describes the difficulty of
the question.
```

RQ2. What do perceived difficulty and performance relate? Here we examine the accuracy of models on problems it identifies as belonging to each difficulty level listed above. The inference here is independent of the difficulty ratings process in **RQ1**, ensuring that models are not biased by their own ratings. Again, here we also make comparisons to verbalized confidence; we do not utilize other forms of confidence estimation (e.g., using logits of answer token) in order to ensure fair and direct comparisons to the form of perceived difficulty we are proposing.

RQ3. How do cues of difficulty in prompts affect LLM reasoning? If model performance is indeed connected to perceived difficulty, a natural question is whether models are sensitive to manip-

ulation of such perception. One simple form of this manipulation can be to inject *cues of difficulty* along with prompts. Specifically, after problem statements we add

```
Note that this is a(n) {level} question.
```

where `{level}` is one of *very easy*, *easy*, *moderately difficult*, *hard*, and *very hard*. These levels are aligned with our difficulty perception extraction in **RQ1**. We report average performance changes when models are prompted with cues that either overstate, similarly state, or understate problem difficulties compared to models' own perception.

RQ4. Token count analyses. Here we examine how models allocate tokens according to their own difficulty perception and cues of difficulty as stated in prompts.

Robustness Check For each experiment, we examine whether our findings still hold when *numerical* difficulty ratings are elicited instead of the word counterpart. In particular, we use the prompt

```
Without solving the question above,
choose a single integer from 1 to 5
which best describes the difficulty of
the question, where 1 is the easiest and
5 is the hardest.
```

to extract difficulty ratings before problem solving and mention the same numeric scale to obtain perception after problem solving:

```
Now that you have answered the question,
choose a single integer from 1 to 5
which best describes the difficulty of
the question, where 1 is the easiest and
5 is the hardest.
```

Figures and tables for robustness check will be included in Appendix E. While outside the primary scope of this work, we also include an exploration of the alignment between model-perceived difficulty and human judgment (E2H-Codeforces), alongside ratings derived from LLM leaderboards (E2H-ARC), in Appendix B. Note that we also perform some fine-tuning experiments (investigating how knowledge exposure affects difficulty perception) with limited findings (Appendix C).

4 Results

4.1 Difficulty Perception

We begin by examining how different models perceive task difficulty across datasets and reasoning budgets. In Figure 2, we first observe that perceived difficulty distributions vary across different models and tasks: for instance, E2H-ARC registers ratings towards the easier side while E2H-Codeforces problems result in ratings more concentrated around "moderate"). However, the "very hard" extreme is rare for all datasets.

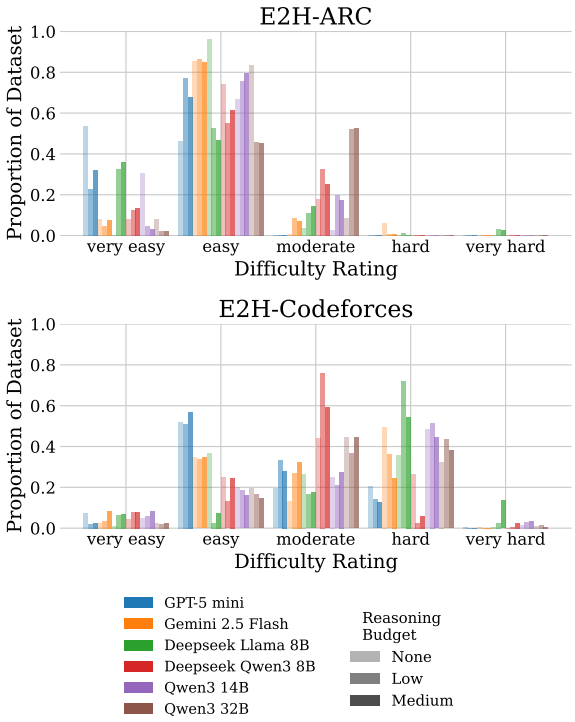

Figure 2: Perceived difficulty distribution varies by models and tasks.

In Figure 3, we illustrate the Spearman correlation between the difficulty ratings by different models. Despite aforementioned results that models have different difficulty perception distributions, we see that **perception is still highly correlated between different models**, with 90% of the correlation values lying above 0.513 and a median of 0.662. This means that if a problem A is perceived as easier than a problem B for a model, we are likely to see the same ranking by a different model. Finally, we also see that different reasoning budgets (none, low, medium) with the same model sometimes do, but not always, result in a high correlation among each other (enclosed in red) compared to other models.

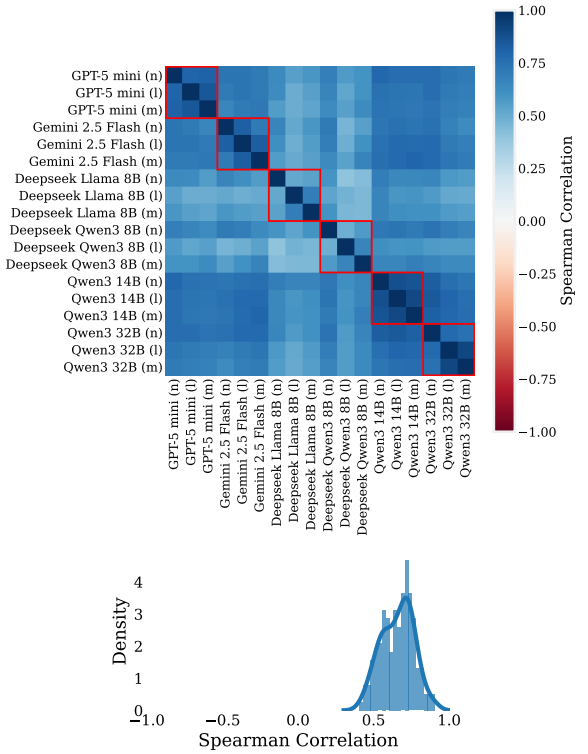

Figure 3: Difficulty ratings are positively rank-correlated between models. This suggests that different models have a common sense of whether one problem is more difficult than another. In the correlation heatmap, n, l, and m stand for none, low, and medium reasoning budgets, respectively.

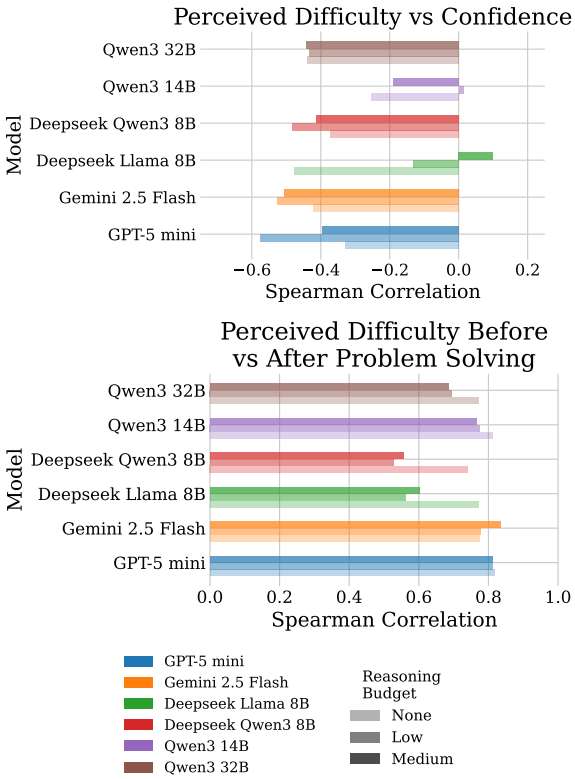

Figure 4: *Top.* A weak correlation here indicates that difficulty perception and confidence are separate concepts from the perspective of the LLMs. *Bottom.* Strong positive correlation between perceived difficulty conditioned and unconditioned on models' solution is recorded across multiple models and budget levels.

Next, we investigate how verbalized confidence (Tian et al., 2023; Xiong et al., 2024; He et al., 2025) and verbalized difficulty perception (the concept we propose) are related. Figure 4 (*Top*) illustrates the Spearman correlation between how models rate the difficulty of each problem and how confident they are about successfully solving the same problem. In most cases, we observe a low negative correlation, with the mean of -0.34. This means a model being confident about solving a problem only corresponds slightly to the event that it thinks the problem is easy. As such **verbalized confidence and difficulty perception are separate concepts** from the perspective of LLMs, which highlights the importance of studying difficulty perception in its own right.

Finally, we compare difficulty perception before and after problem solving (i.e., conditioned on models' own solution). Figure 4 (*Bottom*) shows (1) the Spearman correlation between difficulty perception before and after problem solving and (2) the distribution of changes in perception, with the unit of analysis being individual problems; here the word-based scale from "very easy" to "very hard" is converted into a numerical scale from 1 to 5, and a problem being rated "very hard" when a model is conditioned on its answer but "hard" when the model is unconditioned will result in a change of $+1$, for instance. We observe a strong positive correlation as well as a mode at 0 (with the majority of instances ranging from -1 to $+1$) in the change histograms for most models and reasoning budget, which indicates that **models mostly do not change how they perceive problem difficulty after they have solved the problem**.

As for robustness check, we find that numerical perception results in distributions that are somewhat more uniform compared to the word-based counterpart (distribution plots in Figure A3 and entropy measures in Table A3). There are still very strong rank correlations between the perceptions of different models (Figure A4), with the majority of correlation values greater than 0.5. Figure A5 (*Top*) then shows weak correlations between numerical difficulty perception and verbalized confidence,

which aligns with our existing findings. Finally, Figure A5 (*Bottom*) confirms consistent trends in the numerical case for perceptions before and after problem solving.

Takeaway 1. Difficulty perceptions are not arbitrary; instead its ranking is highly correlated across models. Verbalized confidence and perceived difficulty are separate concepts with weak correlation. Difficulty perception of models towards a problem often persists before and after problem solving.

4.2 Relation Between Difficulty Perception and Performance

Figure 5 shows the performance of Qwen3 32B at different reasoning budgets on problems they deem *easy, moderately difficult, or hard* as well as those they have *moderate, high, and very high confidence about*. We see a clear trend where **higher performance is attained at problems with easier perception**: the model achieves a 96.1%, 73.1%, and 29.0% pass@1 on problems it perceives as "easy", "moderate", and "hard", respectively. This shows that difficulty perception is a measure with intuitive connections to model performance.

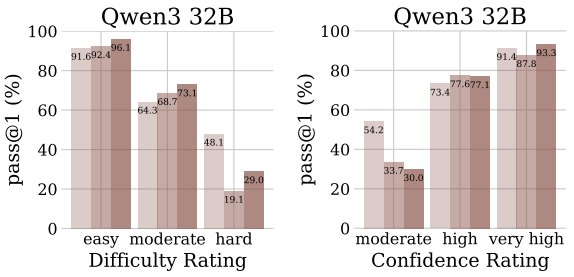

Figure 5: Models perform better on problems they perceive as easier, similar to how models perform better on problems they are more confident about.

	Effort		
Metric	None	Low	Medium
Per. Difficulty	-0.650	-0.655	-0.601
Verbal. Confidence	0.357	0.351	0.540

Table 1: Perceived difficulty is more correlated to performance compared to verbalized confidence across effort levels. Here the unit of analysis is each combination of (task, model, perceived difficulty level), resulting in 78 to 93 data points for each correlation calculation.

Importantly, we note that verbalized confidence is not as predictive of model performance as perceived difficulty. Concretely, Table 1 presents the Spearman correlation between pass@1 and difficulty perception as well as verbalized confidence, when both are converted to a 1-to-5 numerical scale. While perceived difficulty consistently results in strong negative correlations above 0.6 in absolute terms, the confidence counterparts range from 0.36 to 0.54. This observation is intriguing as *perceived difficulty* and *confidence* are conceptually similar metrics. We attribute this empirical discrepancy to (1) LLMs reflecting human data and being overconfident, as discussed in prior studies (Xiong et al., 2024), and (2) difficulty markers being potentially included in pretraining data more as accurate, objective problem labels (e.g., "Difficulty level: easy") rather than self-expressions of uncertainty (e.g., "I am 99% sure this is correct.")

We obtained similar findings for numerical perceptions. Figure A6 illustrates examples of difficulty perception-performance and confidence-performance correspondences with Qwen3 32B. Table A4 shows the same trend as above with performance correlations.

Takeaway 2. Models perform better on problems they perceive as easier. Difficulty perception correlates with performance more than verbalized confidence.

4.3 How Difficulty Cues Affect Performance and Token Allocation

Table 2 shows how performance changes when cues are given. We consider a variable that is the difference between the injected cue and a model's own perceived difficulty of problems; this variable is negative when the cue understates difficulty and positive when the cue overstates difficulty. Overall, models exhibit minor variations in performance, with most improvements and drops being within a 5% margin. With non-zero effort levels, the Deepseek-distilled Llama model becomes an outlier where performance deteriorates more significantly, up to -8.7%, as the injected cue further overstates problem difficulty.

Does this imply that models are only slightly sensitive to difficulty cues? We investigate further through token counts analyses. Figure 6 (*Middle*) shows percentage changes in the number of tokens that each model allocates when cues are introduced, compared to a baseline without prompt injection

	Cue - Perceived Difficulty						
Model	-3	-2	-1	0	1	2	3
Effort: none							
Deepseek Qwen3 8B	-1.0	-1.2	-1.1	-0.5	-0.2	-0.0	-0.9
Deepseek Llama 8B	-1.3	0.5	0.3	0.8	-0.1	-1.8	-1.5
Qwen3 14B	-1.1	-1.2	-0.5	0.3	0.6	0.7	-0.3
Qwen3 32B	2.1	-0.2	0.3	-0.2	0.5	-0.9	0.4
GPT-5 mini	-2.7	-1.6	0.2	0.6	0.4	1.0	-0.3
Gemini 2.5 Flash	0.4	-2.1	-0.5	-0.0	0.5	0.0	-0.5
Effort: low							
Deepseek Qwen3 8B	-2.8	-1.4	-1.9	-2.8	-1.8	-3.2	-3.6
Deepseek Llama 8B	3.3	1.6	-0.5	-0.8	-2.3	-3.8	-5.3
Qwen3 14B	-2.0	-2.1	-0.1	-0.8	-0.4	0.2	-0.5
Qwen3 32B	1.9	0.1	0.4	0.6	1.0	0.3	-0.4
GPT-5 mini	-2.2	-4.8	-0.1	-0.3	-1.5	-1.5	-0.3
Gemini 2.5 Flash	-0.5	0.8	1.4	0.9	1.0	0.7	0.5
Effort: medium							
Deepseek Qwen3 8B	2.8	-0.1	0.1	-1.4	-1.8	-2.4	-3.0
Deepseek Llama 8B	-0.7	0.1	-2.9	-3.6	-3.2	-6.0	-8.7
Qwen3 14B	1.3	0.9	0.2	-0.3	0.5	0.3	0.3
Qwen3 32B	4.6	-1.6	0.4	0.1	0.4	0.5	0.0
GPT-5 mini	-3.0	-5.9	-1.4	-0.0	-1.0	-0.6	-0.2
Gemini 2.5 Flash	1.8	-2.5	0.3	0.6	-0.1	0.0	-0.8

Table 2: Performance-wise, models are mostly resistant to difficulty cues, but less so as more reasoning budget is allowed. Numbers here are changes in pass@1 (%) compared to when no cue is given.

(Figure 6 (*Top*)). Among the considered models, we see that Qwen and Deepseek-distilled Qwen models use more tokens when prompted with more overstating cues: at cue-perception difference +3, these models allocate from 15% to 20% more tokens. This indicates **inefficiency, where more resources are used while performance stays similar or becomes worse**, as seen in Table 2. At the same time, while the Deepseek-distilled Llama model actually allocates less compute (up to 5% fewer tokens) with more overstating cues, performance worsens by up to 8.7% as mentioned above. We note that this pattern of inefficiency applies also when we give models zero reasoning budgets (Figure 6 (*Bottom*)). Deepseek Llama 8B here especially shows much more variations in token counts, with increases of up to 60%. As such, models, non-reasoning or reasoning alike, **suffer from ineffective overthinking when cues overstating difficulty are injected**.

We also examine the generated texts themselves. The $+3$ (overstating) manipulation consistently increases the model's use of "easier" language. In contrast, the -3 (understating) manipulation shows a weaker and noisier reduction in "easy" wording compared to the original responses. Table A5 provides representative examples where the manipulation introduces these cues in the manipulated response and removes similar cues from the original response.

Finally, we highlight a practical risk: excessive resource allocation can occur due to prompts mentioning difficulty, whether as benign use cases (e.g., a student prompting "I am solving this problem and it seems really difficult") or deliberate attacks (e.g., hidden texts on a website or system prompts through an API provider saying "the following content is extremely difficult to summarize/extract information from"). These manipulations require no black-box access (unlike steering approaches such as Lugoloobi and Russell (2025)) yet are still impactful. We call for the development of models that are robust to such scenarios, not only at the performance level (which stays stable as in Table 2) but also in terms of token consumption.

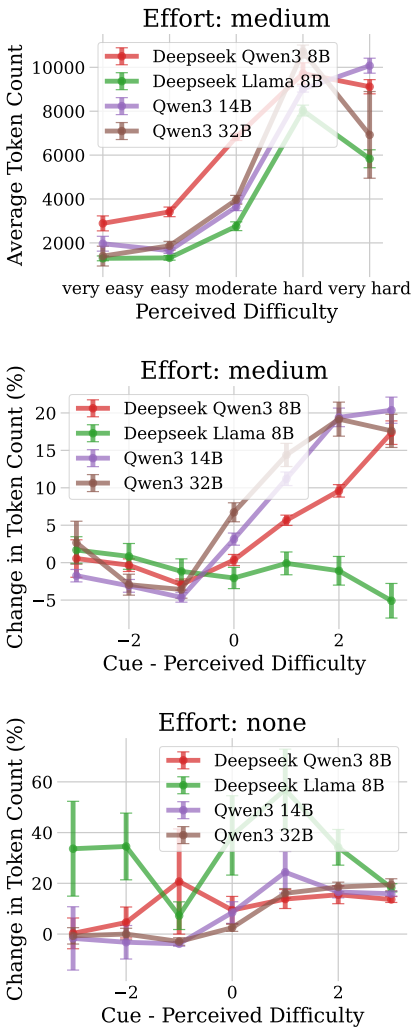

Figure 6: *Top.* Models allocate more tokens to problems they perceive as harder; the higher error margin in the "very hard" category simply reflects the fact that the category is rare, as mentioned in Section 4.1. *Middle.* Qwen and Deepseek-distilled Qwen models allocate up to 20% more tokens when given cues overstating problem difficulty. Here only the open-weight models are shown, since we do not have access to the reasoning tokens of GPT-5 mini and Gemini 2.5 Flash. *Bottom.* Deepseek Llama 8B are extremely sensitive to difficulty cues with a zero reasoning budget, while Qwen and Deepseek-distilled Qwen models again spend up to 20% more tokens on prompts with overstating cues.

Takeaway 3. Prompt injections that overstate problem difficulty make models allocate more tokens while performance either stays stable or declines.

In the Appendix, we show consistent findings for numerical perception in Figure A8 and Table A1.

5 Conclusions

In this study, we proposed the concept of difficulty perception as a novel lens into model epistemology. Through comprehensive experiments with six large reasoning models and tasks in mathematics, coding, science, and social reasoning domains, we have shown that (1) perceived difficulty is consistent across models, (2) such perception correlates strongly with problem performance and more so than verbalized confidence, and (3) prompts with cues overstating problem difficulty consistently decreases reasoning efficiency, for both reasoning and non-reasoning models alike. Our findings establish difficulty perception as a promising subject of study with clear separation from verbalized confidence in the epistemology of LLMs, while implying risks of efficiency declination from simple difficulty-hinted prompt injections, whether benign or malicious.

Limitations

A limitation of our study is the number of mathematical problems we were able to run experiments on (60 AIME problems), due to the enormous costs of LRMs with extended thinking traces together with our compute limitation, especially for the prompt injection experiments. We plan to extend this work with more math problems from E2H-AMC (another subset of the Easy2Hard benchmark with various competition problems at all levels). We encourage the community to build upon our work and investigate properties and uses of difficulty perception in various tasks in natural language processing.

Furthermore, our research opens up a series of interesting questions to be answered: do models internally represent difficulty and can probing such representations provide a tool for difficulty perception steering (c.f., Lugoloobi and Russell (2025))? What are the characteristics of texts produced by models with different difficulty perception and/or prompts containing different difficulty cues? These questions will allow for more insights into the working mechanisms of models as well as control methods for trustworthiness.

Ethical Considerations

Our research describes prompt injections that could be misused to degrade the efficiency of reasoning models. We call for responsible use of models and the development of models that are more robust to such manipulation.

References

Saaket Agashe, Yue Fan, Anthony Reyna, and Xin Eric Wang. 2025. LLM-coordination: Evaluating and analyzing multi-agent coordination abilities in large language models. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 8038–8057, Albuquerque, New Mexico. Association for Computational Linguistics.

Wei Chen, Zhen Huang, Liang Xie, Binbin Lin, Houqiang Li, Le Lu, Xinmei Tian, Deng Cai, Yonggang Zhang, Wenxiao Wang, Xu Shen, and Jieping Ye. 2024. From yes-men to truth-tellers: Addressing sycophancy in large language models with pinpoint tuning. In *Forty-first International Conference on Machine Learning*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Mucong Ding, Chenghao Deng, Jocelyn Choo, Zichu Wu, Aakriti Agrawal, Avi Schwarzschild, Tianyi Zhou, Tom Goldstein, John Langford, Anima Anandkumar, and Furong Huang. 2024. Easy2hard-bench: Standardized difficulty labels for profiling LLM performance and generalization. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

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*.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Yu Wu, YK Li, and 1 others. 2024. Deepseek-coder: When the large language model meets programming–the rise of code intelligence. *arXiv preprint arXiv:2401.14196*.

Zhitao He, Sandeep Polisetty, Zhiyuan Fan, Yuchen Huang, Shujin Wu, and Yi R. Fung. 2025. MM-Boundary: Advancing MLLM knowledge boundary awareness through reasoning step confidence calibration. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16427–16444, Vienna, Austria. Association for Computational Linguistics.

Shima Imani, Liang Du, and Harsh Shrivastava. 2023. MathPrompter: Mathematical reasoning using large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 37–42, Toronto, Canada. Association for Computational Linguistics.

Kevin Maik Jablonka, Philippe Schwaller, Andres Ortega-Guerrero, and Berend Smit. 2024. Leveraging large language models for predictive chemistry. *Nature Machine Intelligence*, 6(2):161–169.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Ankur Joshi, Saket Kale, Satish Chandel, and D Kumar Pal. 2015. Likert scale: Explored and explained. *British journal of applied science & technology*, 7(4):396.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Yen-Ting Lin. 2025. Aime 2025 dataset. Accessed: 2025-08-25.

Ruibo Liu, Ruixin Yang, Chenyan Jia, Ge Zhang, Diyi Yang, and Soroush Vosoughi. 2024. Training socially aligned language models on simulated social interactions. In *The Twelfth International Conference on Learning Representations*.

William Lugoloobi and Chris Russell. 2025. Llms encode how difficult problems are. *arXiv preprint arXiv:2510.18147*.

OpenAI. 2025. Introducing GPT-5. Accessed: 2025-08-25.

Qwen Team. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Qwen Team. 2025b. Qwq-32b: Embracing the power of reinforcement learning.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Esin DURMUS, Zac Hatfield-Dodds, Scott R Johnston, Shauna M Kravec, Timothy Maxwell, Sam McCandlish, Kamal Ndousse, Oliver Rausch, Nicholas Schiefer, Da Yan, Miranda Zhang, and Ethan Perez. 2024. Towards understanding sycophancy in language models. In *The Twelfth International Conference on Learning Representations*.

Matthias Steup and Ram Neta. 2005. Epistemology.

Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. 2023. Large language models in medicine. *Nature medicine*, 29(8):1930–1940.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher Manning. 2023. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5433–5442, Singapore. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Jerry Wei, Da Huang, Yifeng Lu, Denny Zhou, and Quoc V Le. 2023. Simple synthetic data reduces sycophancy in large language models. *arXiv preprint arXiv:2308.03958*.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, YIFEI LI, Jie Fu, Junxian He, and Bryan Hooi. 2024. Can LLMs express their uncertainty? an empirical evaluation of confidence elicitation in LLMs. In *The Twelfth International Conference on Learning Representations*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Kaitlyn Zhou, Dan Jurafsky, and Tatsunori Hashimoto. 2023. Navigating the grey area: How expressions of uncertainty and overconfidence affect language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5506–5524, Singapore. Association for Computational Linguistics.

Caleb Ziems, William Held, Omar Shaikh, Jiaao Chen, Zhehao Zhang, and Diyi Yang. 2024. Can large language models transform computational social science? *Computational Linguistics*, 50(1):237–291.

A Inference Details

For the open-source models (Llama 3.1 8B, Qwen3 {8B, 14B}, and the Deepseek-R1 series) and Gemini 2.5 Flash, we adopt the recommended parameters (Qwen Team, 2025a): Temperature=0.6, TopP=0.95, and TopK=20 for thinking mode and Temperature=0.7, TopP=0.8, and TopK=20 for non-thinking mode. For GPT-5 mini, we use the default parameters provided by their official API since they are mostly not customizable. We generously cap generation at 40,000 tokens for thinking tokens and regular tokens combined.

Our inference was done on the Gemini API for Gemini 2.5 Flash, OpenAI API for GPT-5 mini, and vLLM (Kwon et al., 2023) for the open-source models. A single run through all of our experiments (four trials for each problem and model pair) amount to around $500 in API credits, while our local inference on a 4xRTX A5000 server takes around 400 GPU hours, where the main expense is on the prompt injection/manipulation experiment.

In order to ensure valid responses on difficulty perception, we made use of structured output tools provided by the OpenAI API and vLLM.

B Comparison with Human Difficulty Judgements

In Figure A1, we show that verbalized difficulty judgements from models largely align with how humans judge difficulty in E2H-Codeforces. However, we see very weak positive correlations with the ratings derived from LLM leaderboards as in the case of E2H-ARC.

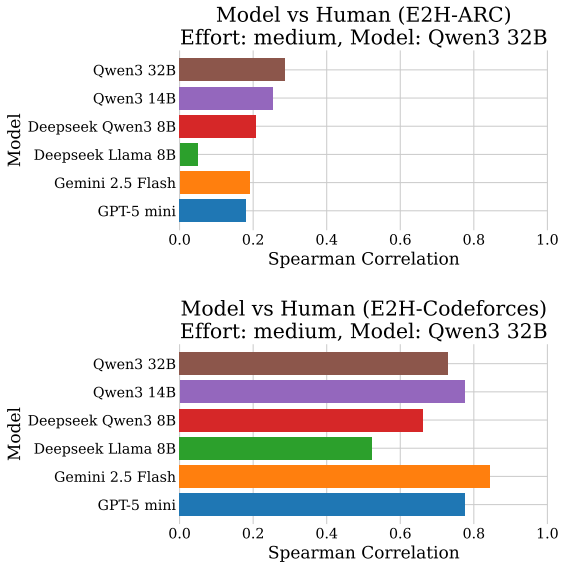

Figure A1: Models largely agree with human judgements on E2H-Codeforces, but deviate from leaderboard-derived ratings on E2H-Codeforces

C Fine-tunning

To ensure fair comparisons, we only fine-tune open models, with which the exact fine-tuning methods

	Cue - Perceived Difficulty						
Model	-3	-2	-1	0	1	2	3
Effort: none							
Deepseek Qwen3 8B	-1.5	-0.8	-0.3	-0.9	-0.6	0.2	-0.6
Deepseek Llama 8B	0.4	0.8	0.9	0.0	-0.7	-1.8	-2.9
Qwen3 14B	0.1	-2.0	-0.3	0.2	0.6	0.7	0.0
Qwen3 32B	0.4	-0.3	0.3	-0.2	0.7	-0.7	0.0
GPT-5 mini	2.1	-1.6	-0.7	0.8	0.7	0.1	0.5
Gemini 2.5 Flash	-1.8	-1.3	-0.4	0.6	0.4	0.1	0.7
Effort: low							
Deepseek Qwen3 8B	-2.6	-0.9	-1.8	-2.5	-2.4	-2.2	-5.7
Deepseek Llama 8B	0.5	-2.0	-1.5	0.6	-1.7	-1.7	-3.1
Qwen3 14B	-1.0	-1.7	-0.5	-0.8	-0.3	-0.4	-0.9
Qwen3 32B	2.6	0.2	0.2	0.7	0.9	-0.1	0.3
GPT-5 mini	-4.8	-4.0	-1.0	-0.5	-0.9	-1.2	-0.2
Gemini 2.5 Flash	-0.4	0.7	1.9	0.3	1.1	0.9	0.2
Effort: medium							
Deepseek Qwen3 8B	1.5	0.1	-0.8	-1.6	-3.1	-2.6	-2.2
Deepseek Llama 8B	0.1	-3.1	-3.2	-4.1	-3.8	-4.9	-6.9
Qwen3 14B	1.3	1.2	-0.0	-0.6	0.8	0.3	-0.5
Qwen3 32B	2.2	-0.4	0.2	-0.1	0.7	0.0	0.3
GPT-5 mini	-2.0	-6.4	-1.4	-0.2	-0.8	-0.8	-0.1
Gemini 2.5 Flash	-1.6	-0.3	0.1	0.3	-0.4	0.3	-0.8

Table A1: In the numerical case, prompt injections with difficulty cues also cause minor variations, which is somewhat more significant for medium budget and Deepseek Llama 8B.

and hyperparameters can be controlled. This appendix describe the results, explains how we collect chains of thought for fine-tuning and gives the training configurations we used. Also note the fact that the fine tunning experimentation was done on a subset of our dataset.

C.1 The Effect of Knowledge on Difficulty Perception

Figure A2 visualizes the distribution of difficulty perception through each epoch of fine-tuning. Surprisingly, we found that the **change in distribution is not always towards the easier direction**. While Qwen3 8B has an increased proportion of "very easy" and "very hard" predictions (both to 22%) with a decrease in "moderate" predictions, Qwen3 14B only experiences a significant increase by 19.4% in "moderate"; at the same time, Llama 3.1 8B has a decrease of 11.3% in "easy" but the change in all other categories stays within 10%. As such, the considered models do not show a uniform trend in adjusting their difficulty perception based on their knowledge, against our cognitive intuition. Figure A7 gives similar observations in the numerical case.

Takeaway Models do not adjust their difficulty perception based on their knowledge the same way.

C.2 Solution Derivation for Fine-Tuning

Coordination–Theory of Mind (ToM) Data Generation. Supervision targets were produced with a multi-turn chat prompt comprising: (i) a **system** block that required *short, faithful, checkable* justifications and provided game-specific hints for *Hanabi*, *Overcooked*, and *CollabGames* (generic cases used collaboration-style guidance); (ii) **user/assistant** stubs establishing the environment and directive (assistant stubs: "I understand the environment. Please tell me how I can help." and "I understand. Please provide the scenario/question."); and (iii) a final **user** turn appending the question with a suffix that **quoted** the exact gold answer ("`{answer_text}`") and instructed the model to repeat it verbatim in `Answer:`.

The generation loop invoked `gpt-4.1`, parsed `Explanation` and `Answer`, then forced the answer

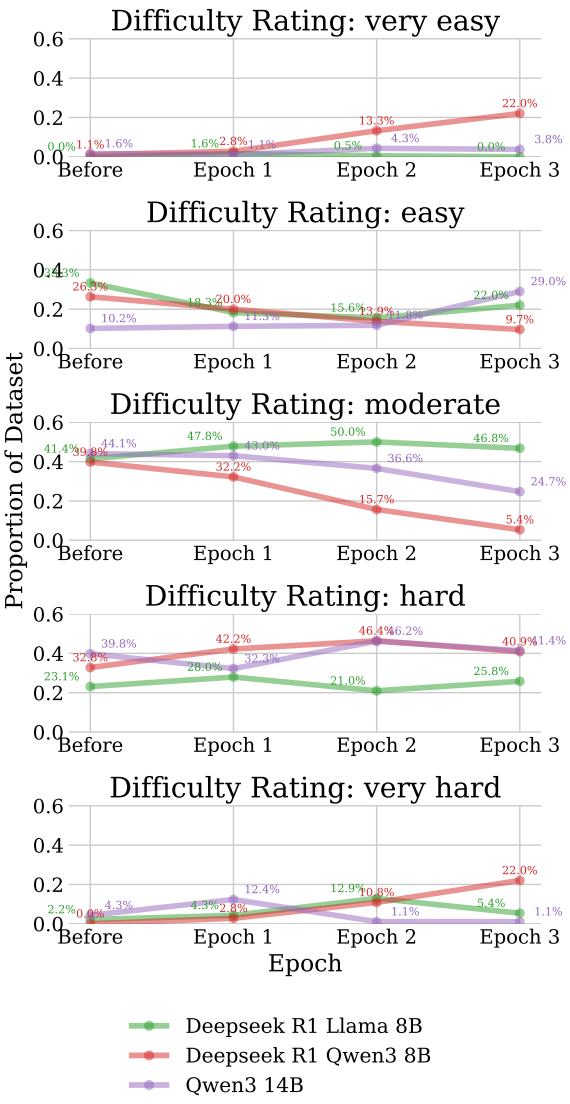

Figure A2: Contrary to our hypothesis, injecting more knowledge into models through fine-tuning does not always lead to easier perception.

to the gold text, and validated that both lines were present, contained no banned/meta phrasing, and matched the gold answer; failures retried up to 10 attempts. Successful pairs were written to CSV and serialized for SFT as a **5-turn chat** input with a **two-line label**:

```
Explanation: <TOM_Explanation>
Answer: <normalized answer text>
```

Example Supervision Signal (GPT-4.1).

> *Hanabi ToM question, 5-turn chat prompt; brief rationale followed by the selected answer.*
>
> **Explanation:** The next playable card for the Green stack is Green 2, and Bob's Card 1 is Green 2. Revealing rank 2 will identify this card as immediately playable, following the convention to give Play Clues for cards that can be played right away.
>
> **Answer:** Reveal Bob's rank 2 cards.

AIME (2024–2025) Data Management. The problems were obtained from the Mathematical Association of America's AIME 2024 and 2025, and the solutions were manually curated from AoPS to maximize per-problem multiplicity. For 2024, we collected **30 problems with 73 training examples** with unique solutions, and for 2025, **62 solutions across 30 problems**. The two years were then combined into a single corpus of 135 instances.

E2H-Codeforces Data Management. Beginning with the E2H seed dataset, each problem was augmented into multiple chat-style training pairs to capture alternative solution strategies. Redundancy was controlled via duplicate-solution screening, reducing the pool from **180 to 154** solutions (26 removed).

C.3 Training Configuration

We fine-tuned models for **3 epochs** using the `trl` library with the `paged_adamw_8bit` optimizer and a cosine learning rate schedule. We utilized **QLoRA** for parameter-efficient adaptation, quantizing base weights to 4-bit `nf4` with double quantization and `bfloat16` compute. LoRA adapters were applied to all attention and MLP modules. Detailed hyperparameters are listed in Table A2.

Hyperparameter	Value
Learning Rate	2×10^{-4}
Scheduler	Cosine (0.05 warmup)
Optimizer	Paged AdamW 8-bit
Batch Size	32 (effective)
LoRA Rank (r)	32
LoRA Alpha (α)	64
LoRA Dropout	0.05
Target Modules	Attention (q,k,v,o) & MLP
Max Context	Tokenizer Limit (No Packing)

Table A2: Hyperparameters used for SFT.

QLoRA configuration. Base weights were quantized to **4-bit `nf4`** with double quantization and **bfloat16** compute; all other hyperparameters as above.

D Prompt Templates

In this appendix, we describe the prompt templates used for each of the considered datasets.

D.1 AIME

We include guidance on the answer range from the official AIME problem sheets, and use a standard chain-of-thought prompt.

```
{problem statement}
Your final answer should be an integer ranging from 0 to 999, inclusive. Please reason step by step, and put your final answer within \boxed{}.
```

D.2 E2H-Codeforces

We follow the formatting used on Codeforces, where the programming problems were sourced from.

```
The following is a competitive programming problem.

# {problem name}

{problem statement}

## Input
{input spec}

## Output
{output spec}

## Example
### Input
{sample inputs}

### Output
{sample outputs}
```

We add the following information to specify the task and answer formatting:

```
Your task is to solve the problem above by writing a Python script that reads input from standard input (stdin) and prints the desired outputs to standard output (stdout). Please ensure your solution is efficient and handles all edge cases. Make sure to include any necessary imports and helper functions. Your code should be ready to run and should not include any additional explanations or comments.
```

D.3 Coordination-TOM

We utilize the exact same prompts provided by the original authors of LLM Coordination (Agashe et al., 2025).

```
USER: {game description}
ASSISTANT: I understand the game. Please tell me how I can help.
USER: {directive}
ASSISTANT: I understand. Please provide the scenario.
USER: {question} Think step by step.
```

E Robustness Check

Here we present a series of evidence that **while difficult perception in the numerical form has some distinct properties, several of our findings persist**.

- Figure A3 shows that the distribution of numerical difficulty perception is more uniform than the word form we have been investigating. Table A3 concretizes this observation with the average entropy across all models and reasoning budgets for each form.
- In Figure A4, we again observe a rank correlation between the ratings of different models, where the distribution is now skewed more towards 1, with the majority of Spearman correlation values greater than 0.5.
- Figure A6 illustrates the correspondence between easier perception and higher performance for Qwen3 32B.
- Figure A5 shows that the correlation between difficulty perception and verbalized confidence as well as difficulty perceptions before and after problem solving.
- Figure A8 and Table A1 demonstrates the inefficiency caused by overstating difficulty cues for the numerical case.

Dataset	Word	Numerical
AIME	1.043	1.294
E2H-ARC	1.000	1.036
E2H-Codeforces	1.654	1.880
LLM Coordination	1.374	1.555

Table A3: The average entropy of difficulty perception distributions in the word and numerical forms on each dataset. The numerical form results in more spread-out distributions and accordingly somewhat higher entropy measures.

	Effort		
Metric	None	Low	Medium
Per. Difficulty	-0.631	-0.687	-0.733
Verbal. Confidence	0.507	0.473	0.610

Table A4: For the numerical case, perceived difficulty is also more correlated to performance compared to verbalized confidence across effort levels. Correlations are calculated based on 75 to 100 data points for each setting.

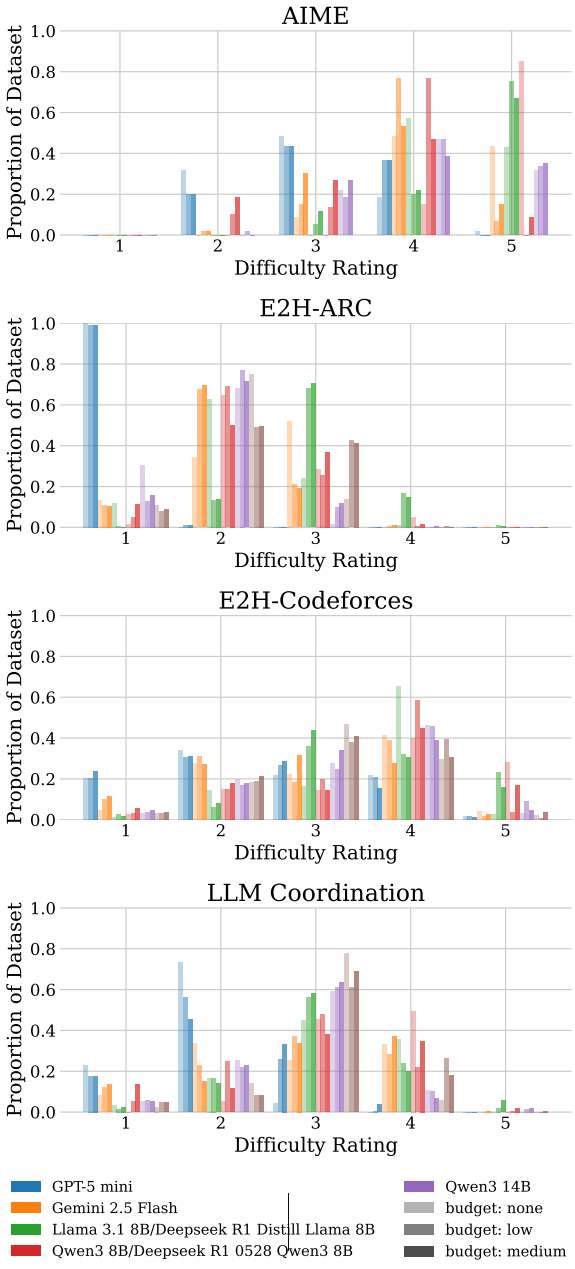

Figure A3: The numerical form of difficulty perception enjoys a more uniform distribution compared to the word-based counterpart. Still, the distribution varies by models and tasks.

Condition	Mode	Term	problem_id	Sentence
Overstating (+3)	introduced	hard	E2H-Codeforces_91	"But why is the problem rated as hard?"
Overstating (+3)	introduced	easy	E2H-Codeforces_91	"This seems too easy…"
Overstating (+3)	removed	easy	E2H-Codeforces_153	"But perhaps the easiest way is to…"
Understating (−3)	introduced	easy	E2H-Codeforces_57	"Wait that seems too easy."
Understating (−3)	removed	easy	E2H-Codeforces_128	"Once we have that, grouping is easy."
Understating (−3)	removed	hard	E2H-Codeforces_97	"This seems like a very hard problem."

Table A5: Comparison of manipulated and original thoughts.

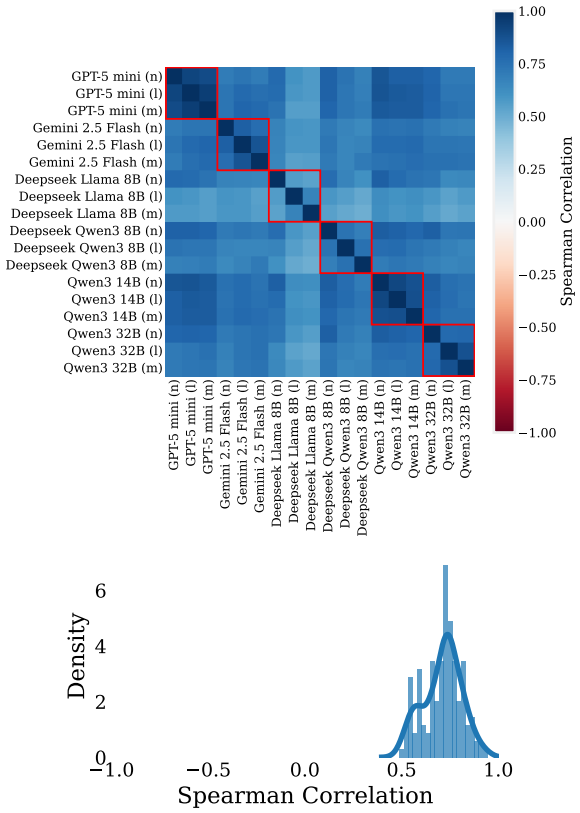

Figure A4: Numerical difficulty ratings are positively rank-correlated between models, and more so than word ratings. In the correlation heatmap, n, l, and m stand for none, low, and medium reasoning budgets, respectively.

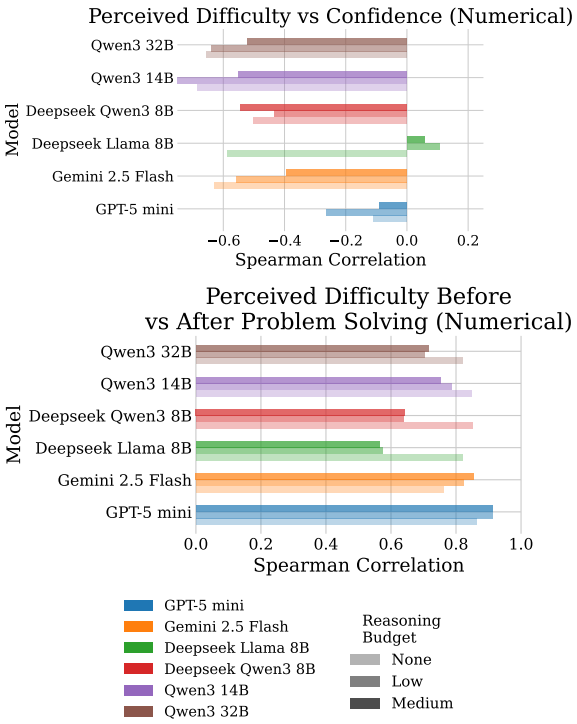

Figure A5: *Top.* Numerical perceived difficulty overall shows more correlation to verbalized confidence compared to word-based counterparts. *Bottom.* We still observe strong positive correlation between perceived difficulty conditioned and unconditioned on models' solution is recorded across multiple models and budget levels.

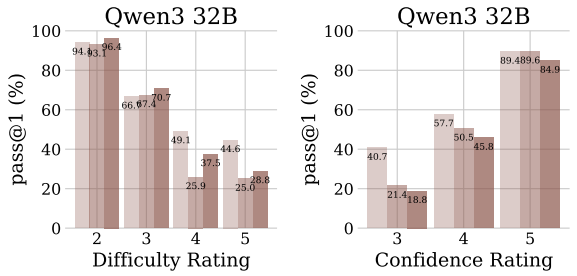

Figure A6: For the numerical form of perceived difficulty, we also see that models perform better on problems they perceive as easier.

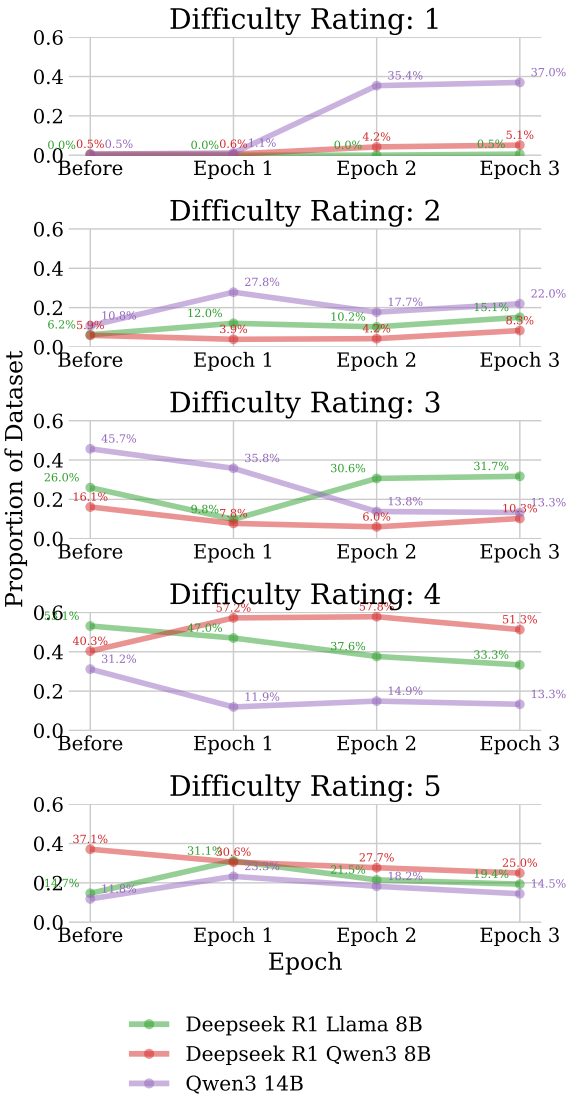

Figure A7: Injecting more knowledge into models through fine-tuning also does not always lead to easier perception in the numerical case.

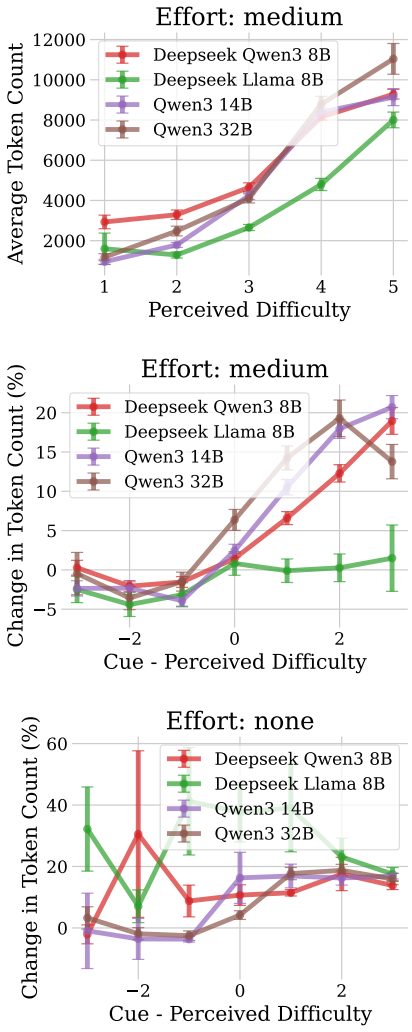

Figure A8: Replication of Figure 6 for the numerical case, with the same findings