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 five LRMs on mathematical, competitive programming, and social reasoning tasks reveal that difficulty perception is not random and its ranking instead highly correlates between different models; we present evidence that models perform better on problems they deem easier and demonstrate how the performance of non-reasoning models is undermined by prompts that understate task difficulty compared to models' own perception; additionally, models are also shown to differ from each other in terms of how they perceive questions that are within and outside of their knowledge. Our findings establish difficulty perception as a concept separate from verbalized confidence in model epistemology while yielding insights into the workings of LLMs and LRMs.

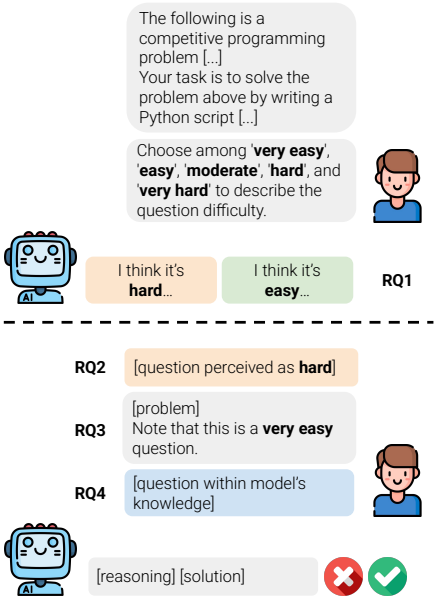

Figure 1: In this paper, using an array of math, coding, and social reasoning tasks, we investigate (in **RQ1**, Section 4.1) properties of the verbalized difficulty perception of models, (in **RQ2**, Section 4.2) the relation between difficulty perception and performance, (in **RQ3**, Section 4.3) how cues of difficulty in prompt affect performance, and (in **RQ4**, Section 4.4) how knowledge affects difficulty perception

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.

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; 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 diffi-

culty 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 a range of tasks including mathematics, coding, and theory of mind. Alongside several other interesting findings, 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, understating the difficulty of a problem in prompt injections undermines performance, while having prior exposure to problems does not lead to consistent adjustments in a model's perception towards that problem. Through these experiments, our research establishes difficulty perception as a key concept in model epistemology and provides insights into properties revolving around this concept.

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.

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; Team, 2025b; Yang et al., 2024; 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 consider three English datasets spanning mathematics, competitive programming, and social reasoning domains. A total of 186 problems are considered:

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**. 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 stratified sampling (with respect to difficulty ratings provided in the benchmark) to obtain 60 problems in the competitive programming subset that resembles the original distribution of difficulty levels.

[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.

3. **Coordination-ToM**. This is part of the LLM Coordination (Agashe et al., 2025) benchmark, which evaluates the ability of LLMs to coordinate in card, cooking, and maze games. We specifically make use of the Theory of Mind (ToM) QA portion of the benchmark, consisting of 66 single-turn, multiple-choice questions on the mentioned games, testing the intention and requirement guessing capabilities of models.

3.2 Models

We consider five popular open- and closed-source 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 (Team, 2025a), and DeepSeek R1 Llama 8B (Guo et al., 2025). For each of these, we consider reasoning budgets of 1024 tokens and 8192 tokens. These budgets are implemented, as per our observation, as soft constraints for the Gemini and GPT models through the Gemini API and effort levels "low" and "medium" in the OpenAI API, while the reasoning tokens of the latter two models are cut off exactly at the indicated budgets. At the cut position, we manually insert the following sentence recommended by 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
```

Additionally we experiment with a non-reasoning version for each of these models: 0-token budget for Gemini 2.5 Flash, "minimal" effort for GPT-5 mini (no option with zero reasoning is available for this model), the non-thinking mode of Qwen3 8B and Qwen3 14B, and Llama 3.1 8B Instruct, respectively.

All inference is done with four random attempts and we report pass@4 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). Using a similar procedure, where "the difficulty of the question", "easy", and "hard" are replaced respectively by "your confidence in successfully solving the question", "low", and "high", we also obtain verbalized confidence for comparison.

Additionally, we also elicit difficulty ratings conditioned on models' answers. Specifically, we prompt models with a conversation including a problem statement, the models' answer, 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 is the relation of difficulty perception to performance? Here we examine the accuracy (we specifically measure pass@4 to account for random variations in the models' answers) of models on problems it identifies as belonging to each difficulty level listed above. The prompts and answers for difficulty ratings in **RQ1** are not included in the prompt for this research question, making sure that models are not biased by such prompts and answers. Again, we compare this alignment between difficulty perception and performance to that between verbalized confidence and performance.

RQ3. How do cues of difficulty in prompts affect LLM reasoning? If model performance is connected to difficulty perception, we expect that performance is sensitive to any manipulation of such perception. A simple form of this manipulation can be to inject *cues of difficulty* along with prompts. Specifically, we add

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

where `{level}` is one of *very easy*, *easy*, *moderately difficult*, *hard*, and *very hard*, after problem statements. These levels are aligned with our difficulty perception extraction in **RQ1**. We perform statistical analysis for this prompt injection by estimating logistic regressions with problem and model fixed effects.[2]

RQ4. How does knowledge affect difficulty perception? Intuitively, if a person has had prior exposure to a solution to a problem, they could be more likely to perceive the problem as easier, since they are able to recall the memorized solution and utilize it. As such, we hypothesize that having knowledge about how to solve problems may reduce the difficulty perception of models on such problems. In order to test this hypothesis, we fine-tune our considered open models on solutions, measure their difficulty perception on the same problems after fine-tuning, and compare it to the perception before any fine-tuning. More details on the fine-tuning process is available in Appendix B.

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.
```

Most figures and tables for robustness check will be included in Appendix E.

[2]Problem and model fixed effects account for any variation in performance coming from any differences across problems and models, respectively, allowing us to isolate the effect of the prompt manipulation itself.

4 Results

4.1 Difficulty Perception

In Figure 2, we first observe that while the specific distributions vary across different models and tasks (for instance, AIME registers ratings towards the harder side, Coordination-ToM leans towards the opposite, and E2H-Codeforces shows a more uniform distribution), **difficulty ratings across different models and datasets commonly do not result in many ratings at the "very easy" and "very hard" extremes**.

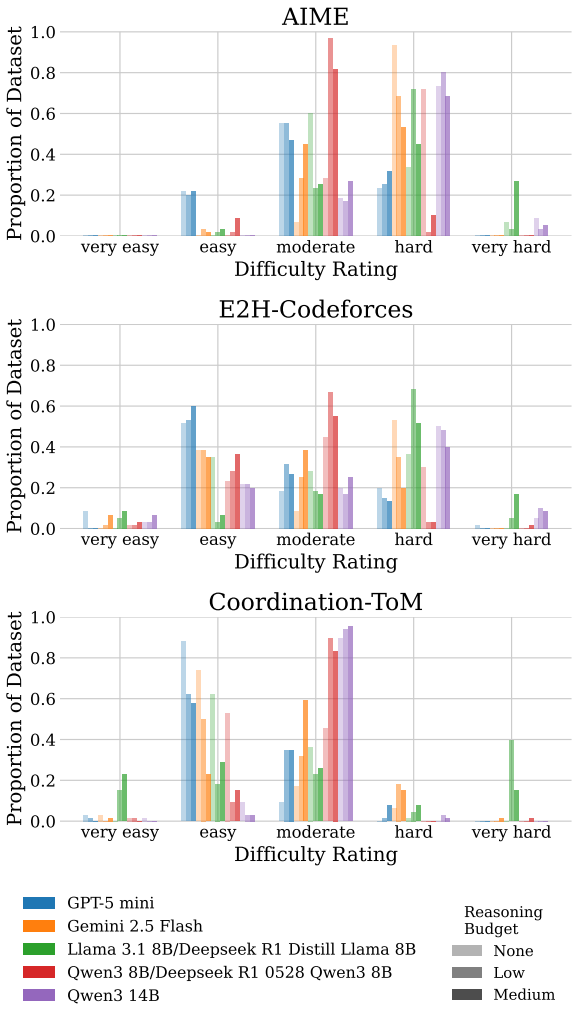

Figure 2: Difficulty perception are typically within the "easy", "moderate", and "hard" categories rather than the "very easy" and "very hard" extremes. The distribution also varies by models and tasks.

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 most correlation

values ranging between +0.3 and +0.9. 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 perception in 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.

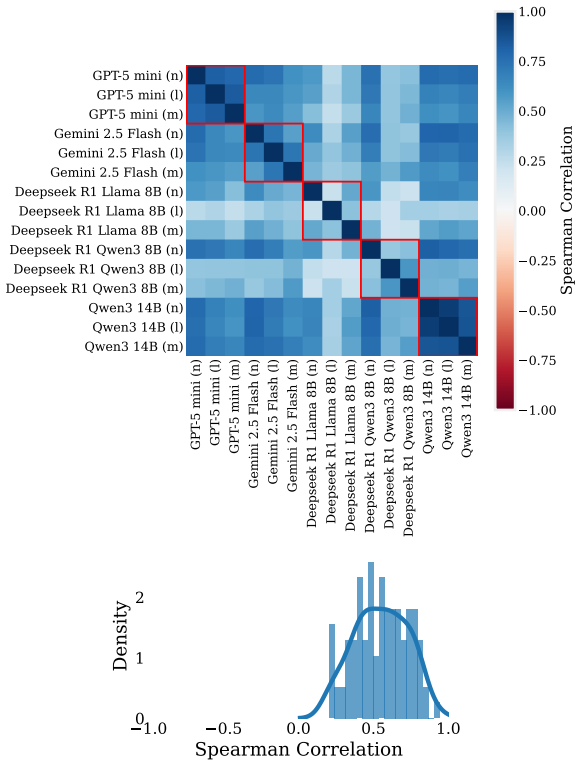

Figure 3: Difficulty ratings are positively rank-correlated between models. This suggests that different models have a common sense of whether one problem is more difficult than another. In the correlation heatmap, n, l, and m stand for none, low, and medium reasoning budgets, respectively.

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 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 -0.202; here Qwen3 14B actually shows a weak positive correlation for the none and low reasoning budgets. This means that a model being confident about a problem only corresponds slightly to the event that it thinks the problem is easy. This means that **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.

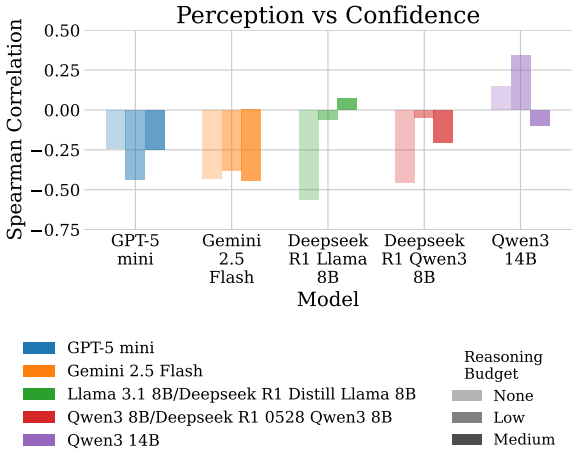

Figure 4: The rank correlation between difficulty perception and verbalized confidence: a negative correlation means that the more confident the model is, the easier it thinks the problem is. A weak correlation here indicates that difficulty perception and confidence are separate concepts from the perspective of the LLMs.

Finally, we compare difficulty perception before and after problem solving (i.e., conditioned on models' answers). Figure 5 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 more uniform compared to the word-based counterpart (distribution plots in Figure A1 and entropy measures in Table A1). However, 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 A3 then shows weak correlations between numerical

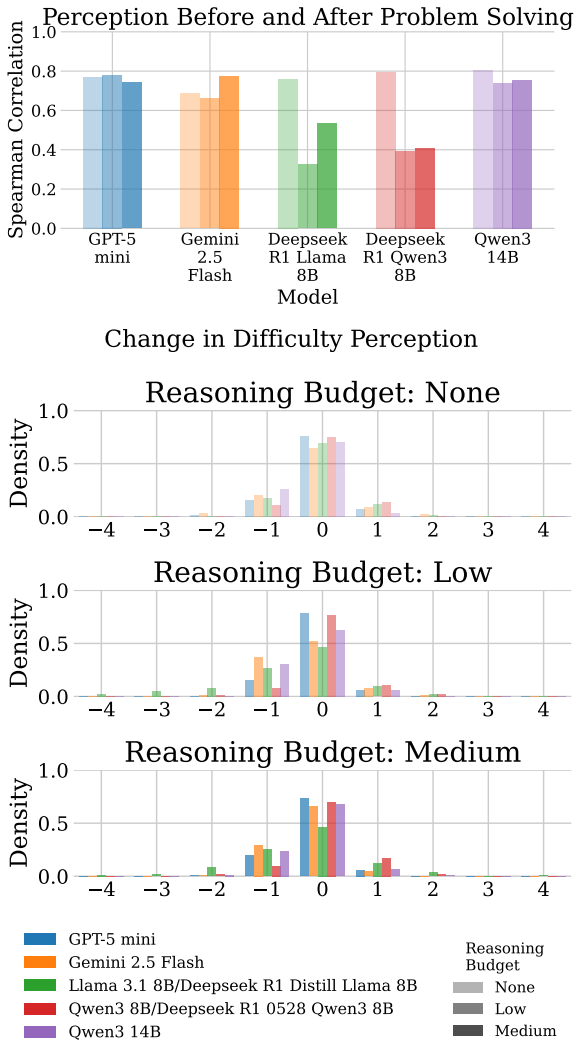

Figure 5: *Top.* The rank correlation between difficulty perception conditioned and unconditioned on models' answers. Strong positive correlation is recorded across multiple models and budget levels. *Bottom.* The distribution of changes in perception after problem solving, where our Likert word-based scale is converted to a 1-to-5 numerical scale. All models and budgets record mostly no changes.

difficulty perception and verbalized confidence, which aligns with our existing findings. Finally, Figure A5 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 of Difficulty Perception to Performance

Figure 6 (*Left*) shows the performance of GPT-5 mini and Qwen3 14B 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. We see a clear trend where **higher performance is attained at problems with easier perception**. For example, Qwen3 14B achieves a 93.8%, 74.4%, and 33.7% mean pass@4 on problems it perceives as "easy", "moderate", and "hard", respectively. This shows that difficulty perception is a measure with intuitive connections to model performance.

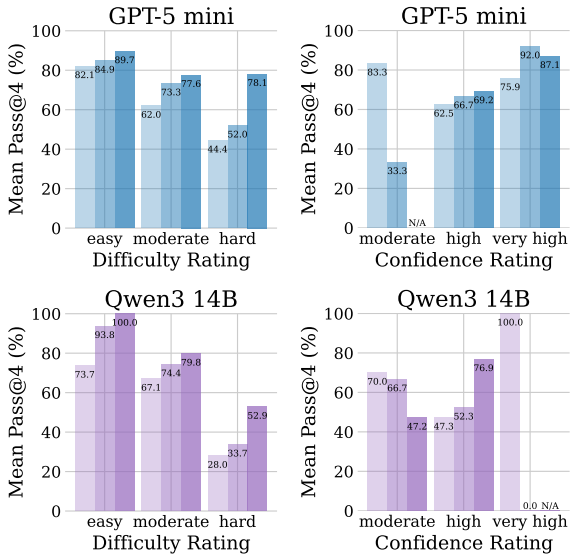

Figure 6: *Left.* Examples of models performing better on problems they perceive as easier. Here all problems from the three datasets we consider are combined for the pass@4 measure, so the mean is weighted. *Right.* Examples of models not necessarily performing better on problems they are more confident about, aligning with the literature. N/A denotes categories where there is no problem with the matching confidence verbalized by the corresponding model. Here we use "moderate", "high", and "very high" instead of "easy", "moderate", and "hard" as in the case of difficulty ratings due to the distribution of verbalized confidence skewing more towards the higher end; for more information see Appendix F.

Importantly, note that verbalized confidence in many cases does not align well with performance (*Right* in Figure 6): problems in which a model is more confident may lead to significantly lower performance. This property of verbalized confidence has been attributed to LLMs reflecting human data

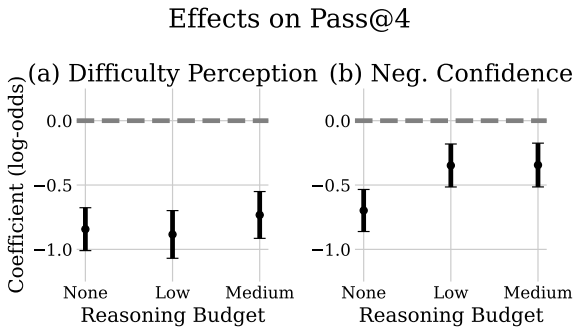

Figure 7: In logistic regression with pass@4 as the dependent variable, both difficulty perception (a) and negative confidence (b) yield negative coefficients, though the former shows a stronger effect. This means each unit increase of difficulty perception lowers the chance of success more than each unit decrease of confidence.

and being overconfident in prior studies (Xiong et al., 2024). Furthermore, Figure 7 presents results of logistic regression on pass@4 from difficulty perception and negative confidence, when both are converted to a 1-to-5 (or negative 1 to negative 5) numerical scale; we see that each unit increase in perceiving the difficulty of a problem is associated more negatively with successfully solving a question than each unit decrease in confidence is.[3] So verbalized **difficulty perception without any calibration technique is more indicative of model performance than the confidence counterpart**, highlighting difficulty perception as a key concept indicative of model performance. More details on the regression can be found in Appendix D.

We also obtained the same findings for numerical perceptions. Figure A2 illustrates examples of difficulty perception-performance and confidence-performance correspondences with GPT-5 mini and Qwen3 14B. Figure A6 demonstrates that difficulty perception still has a stronger effect on pass@4 than confidence does.

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

4.3 The Effect of Difficulty Cues on Performance

We also examine how the gap between the model's perceived difficulty and the difficulty cue provided in the prompt influences performance, by estimating logistic regressions with problem and model fixed effects. Specifically, we define *deviation* as the difference between difficulty perception and the level of the difficulty cue (both translated into a 1-to-5 scale); this means the larger the deviation, the easier the problem is manipulated than perceived by the model. Figure 8(a) shows the coefficient for deviation where pass@4 is the dependent variable, with the unit of analysis being individual problems perceived and solved by models. We find that when no reasoning effort is required, the model performs worse when problems are framed as easier than it perceives them to be. This suggests that **understatements of task difficulty may undermine accuracy**, perhaps by leading the model to allocate insufficient effort. However, **this negative effect disappears once the model is set to engage in reasoning** with a low- or medium-level budget. More details on the regression can be found in Appendix D. As shown in Figure 8(b), we find consistent results with numerical ratings.

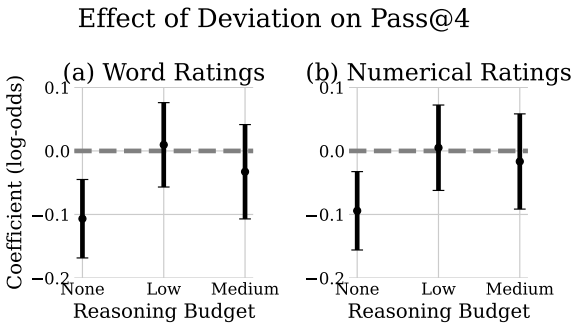

Figure 8: LLMs perform worse when problems are framed as easier than it perceives them to be, when difficulty ratings by words (a) and by numbers (b) are used.

Takeaway 3. Understatements of task difficulty undermines performance for non-reasoning models but has little effect otherwise.

4.4 The Effect of Knowledge on Difficulty Perception

Figure 9 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] These models also include problem and model fixed effects.

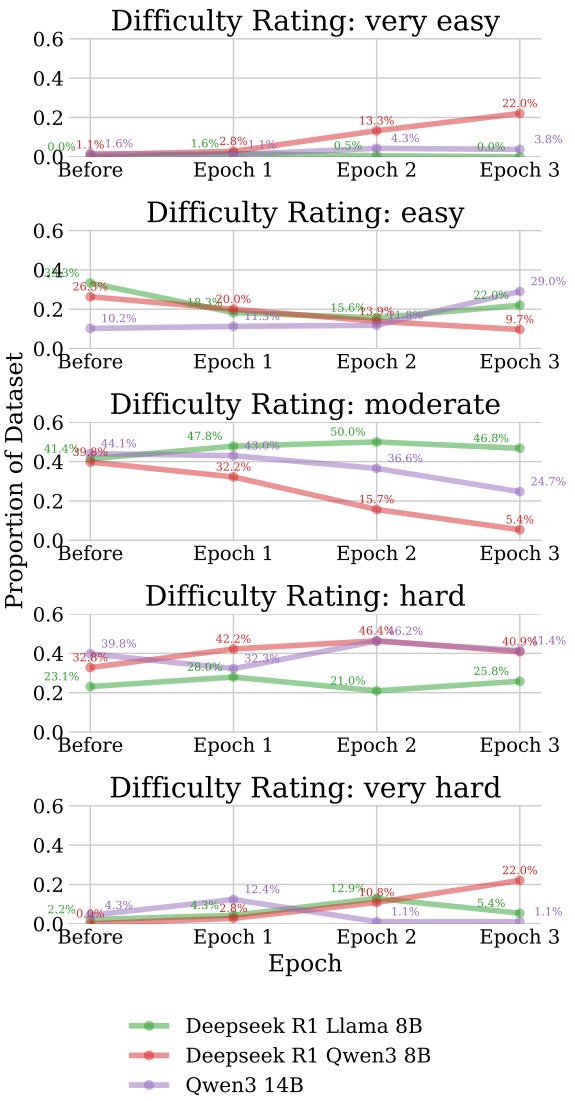

Figure 9: Contrary to our hypothesis, injecting more knowledge into models through fine-tuning does not always lead to easier perception.

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

5 Conclusions

In this study, we proposed the concept of difficulty perception as a novel lens into model epistemology. Through experiments with five large reasoning models and tasks in mathematics, coding, and social reasoning domains, we have shown, alongside several other interesting properties, that (1) difficulty perception is distributed differently but ranked similarly across models, (2) such perception correlates strongly with problem performance, (3) understatement of task difficulty in prompts undermines the performance of models with no reasoning budget, and (4) knowledge does not have a consistent effect on perception. Our findings establish difficulty perception as a promising subject of study with clear separation from verbalized confidence in the epistemology of LLMs and LRMs.

Limitations

A limitation of our study is the amount of data we were able to run experiments on, due to the enormous costs of LRMs with extended thinking traces together with our compute limitation, especially for the prompt injection experiments. Despite this setback, we were still able to examine a variety of tasks from mathematical reasoning to competitive programming and theory of mind. 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? Can probing such representations provide a tool for difficulty perception steering? What are the characteristics of texts produced by models with different difficulty perception and/or prompts containing different difficulty cues? Additionally, how the difficulty perceptions of models line up with a standardized benchmark such as Ding et al. (2024) remains unanswered. These questions will allow for more insights into the working mechanisms of models as well as control methods for trustworthiness.

Ethical Considerations

Our research involves the manipulation of how LLMs and LRMs perceive problem difficulty, which could be misused to degrade the performance of such 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*.

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*.

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

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.

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

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

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 (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 Fine-Tuning Details

To ensure fair comparisons, we only fine-tune open models, with which the exact fine-tuning methods and hyperparameters can be controlled. This appendix describes how we collect chains of thought for fine-tuning and the training configurations.

B.1 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 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).

B.2 Hyperparameters and Training Configuration

Models were (supervised) fine-tuned using **trl** on each dataset seperately for **3 epochs** using a cosine learning-rate schedule with **initial LR =** 2×10^{-4} and **warmup ratio = 0.05**. Optimization employed `paged_adamw_8bit` with **weight decay = 0.01**. We used **per-device batch size = 4** with **gradient accumulation = 8** (effective per-device batch = 32 sequences) and **bfloat16** precision. Parameter-efficient adaptation used **LoRA** with **rank** $r = (32)$, $\alpha = 2r$, and **dropout = 0.05**, targeting attention projections (`q/k/v/o`) and MLP projections (`gate/up/down`). **Gradient clipping** was applied with **max_grad_norm = 1.0** (default), and **gradient checkpointing** was enabled (non-reentrant) to reduce memory overhead. Tokenization followed each model's native chat template without adding a generation prompt. Sequences were **truncated to the tokenizer's maximum context length** and **padded dynamically per batch**; no sequence packing was used. Checkpoints were saved **once per epoch** with a retention cap of **10**; data loading used **4 workers**; device placement was handled by **automatic device mapping**.

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

C Prompt Templates

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

C.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{}.
```

C.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.
```

C.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.
```

D Logistic Regression Details

Tables A2 and A3 provide detailed information on the logistic regression we did in Sections 4.2 and 4.3.

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 A1 shows that the distribution of numerical difficulty perception is more uniform than the word form we have been investigating. Table A1 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 A5 shows that difficulty perceptions before and after problem solving are correlated and mostly do not differ also on absolute terms.
- Figure A2 illustrates the correspondence between easier perception and higher performance for GPT-5 mini and Qwen3 14B.
- Figure 8(b) shows consistency in terms of how cues of difficulty affect model performance.
- Figure A3 shows that the correlation between difficulty perception and verbalized confidence for most cases are weak.

Dataset	Word	Numerical
AIME	1.062	1.320
E2H-Codeforces	1.616	1.843
coordination-TOM	1.076	1.267

Table A1: 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 higher entropy measures.

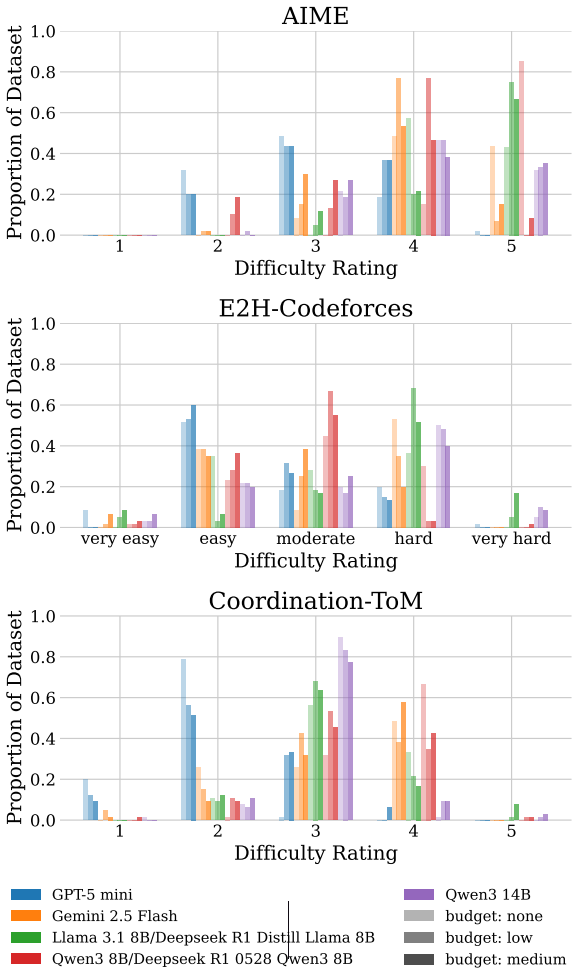

Figure A1: The numerical form of difficulty perception enjoys a more uniform distribution compared to the word-based counterpart. Still, the distribution varies by models and tasks.

F Additional Visualization

Figure A8 shows the distribution of verbalized confidence for each task, both numerical and word-based.

Reasoning Budget	Coefficient	95% Confidence Interval	p-value
Difficulty Perception			
None	*-0.8430*	(-1.010, -0.676)	<0.001
Low	*-0.8839*	(-1.069,-0.699)	<0.001
Medium	*-0.7325*	(-0.914, -0.551)	<0.001
Negative Confidence			
None	*-0.6978*	(-0.861, -0.534)	<0.001
Low	*- 0.3481*	(-0.515, -0.181)	<0.001
Medium	*-0.3446*	(-0.515, -0.175)	<0.001

Table A2: Details of logistic regression on pass@4 from difficulty perception and negative confidence. Statistically significant results ($p < 0.05$) are highlighted in cyan.

Reasoning Budget	Coefficient	Confidence Interval	p-value
Word			
None	*-0.1067*	(-0.168, -0.045)	0.006
Low	0.010	(-0.057, 0.076)	0.777
Medium	-0.033	(-0.107, 0.041)	0.387
Numerical			
None	*-0.094*	(-0.156, -0.033)	0.002
Low	0.005	(-0.062, 0.072)	0.883
Medium	-0.017	(-0.091, 0.058)	0.664

Table A3: Details of logistic regression on pass@4 from deviation, in both word and numerical forms. Statistically significant results ($p < 0.05$) are highlighted in cyan.

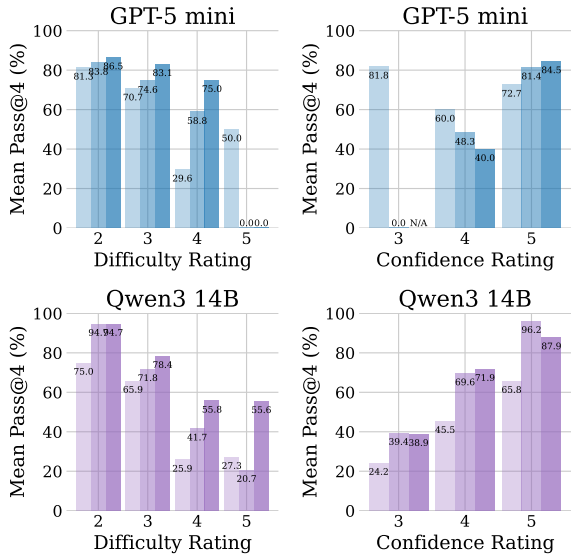

Figure A2: *Left* Models perform better on problems they perceive as easier for the numerical form of perception. *Right.* The same is not often true when it comes to verbalized confidence.

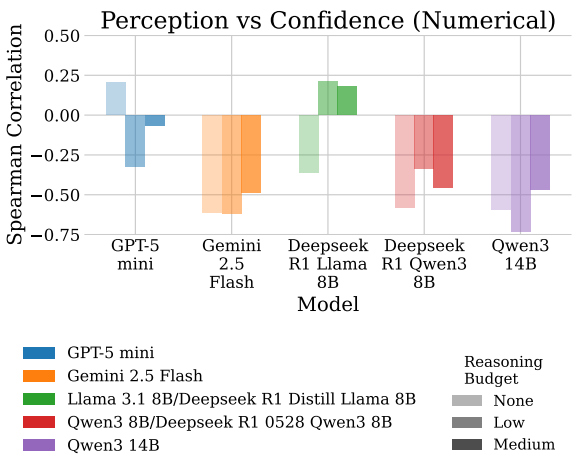

Figure A3: There are weak correlations between numerical difficulty perception and verbalized confidence, aligning with the word perception case.

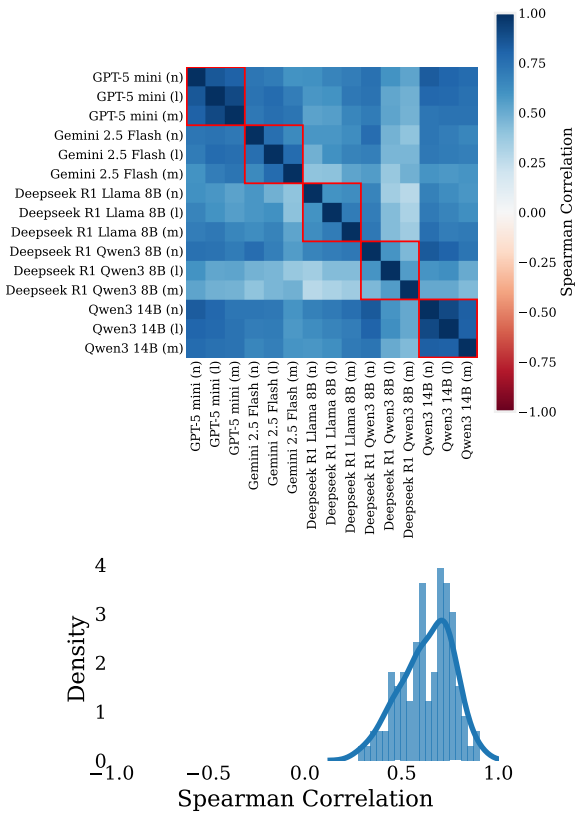

Figure A4: Numerical difficulty ratings are positively rank-correlated between models, and more so than word ratings. In the correlation heatmap, n, l, and m stand for none, low, and medium reasoning budgets, respectively.

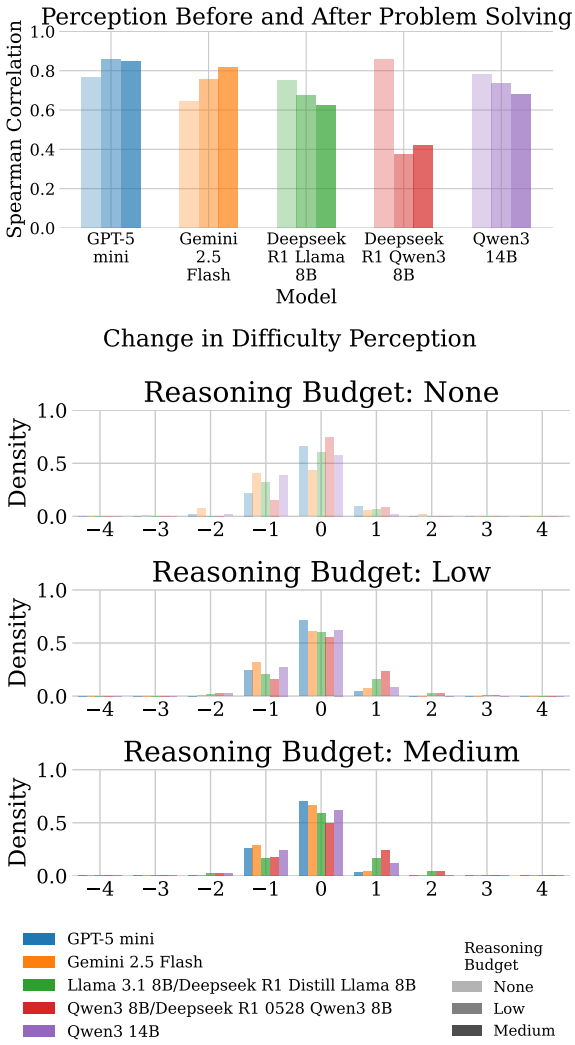

Figure A5: *Top.* The rank correlation between difficulty perception conditioned and unconditioned on models' answers, using numerical perception. Strong positive correlation is recorded across multiple models and budget levels. *Bottom.* The distribution of changes in perception after problem solving. All models and budgets record mostly no changes.

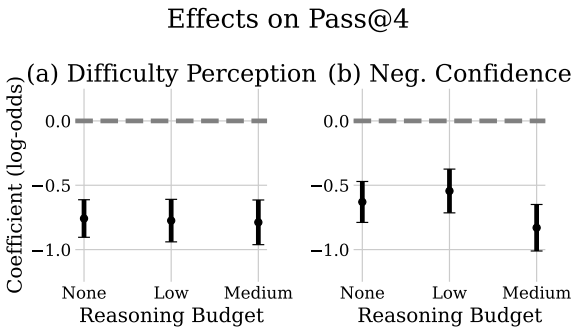

Figure A6: In the numerical form, both difficulty perception and negative confidence still yield negative coefficients. Apart from the case of "medium" reasoning budgets, difficulty perception still has a stronger relation to performance than verbalized confidence.

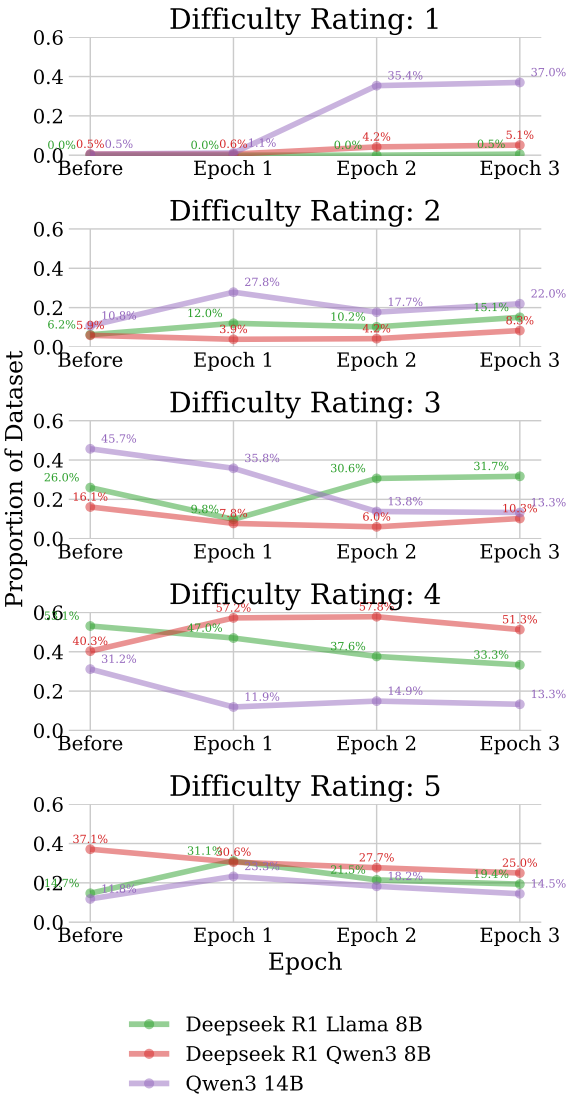

Figure A7: Injecting more knowledge into models through fine-tuning also does not always lead to easier perception in the numerical case.

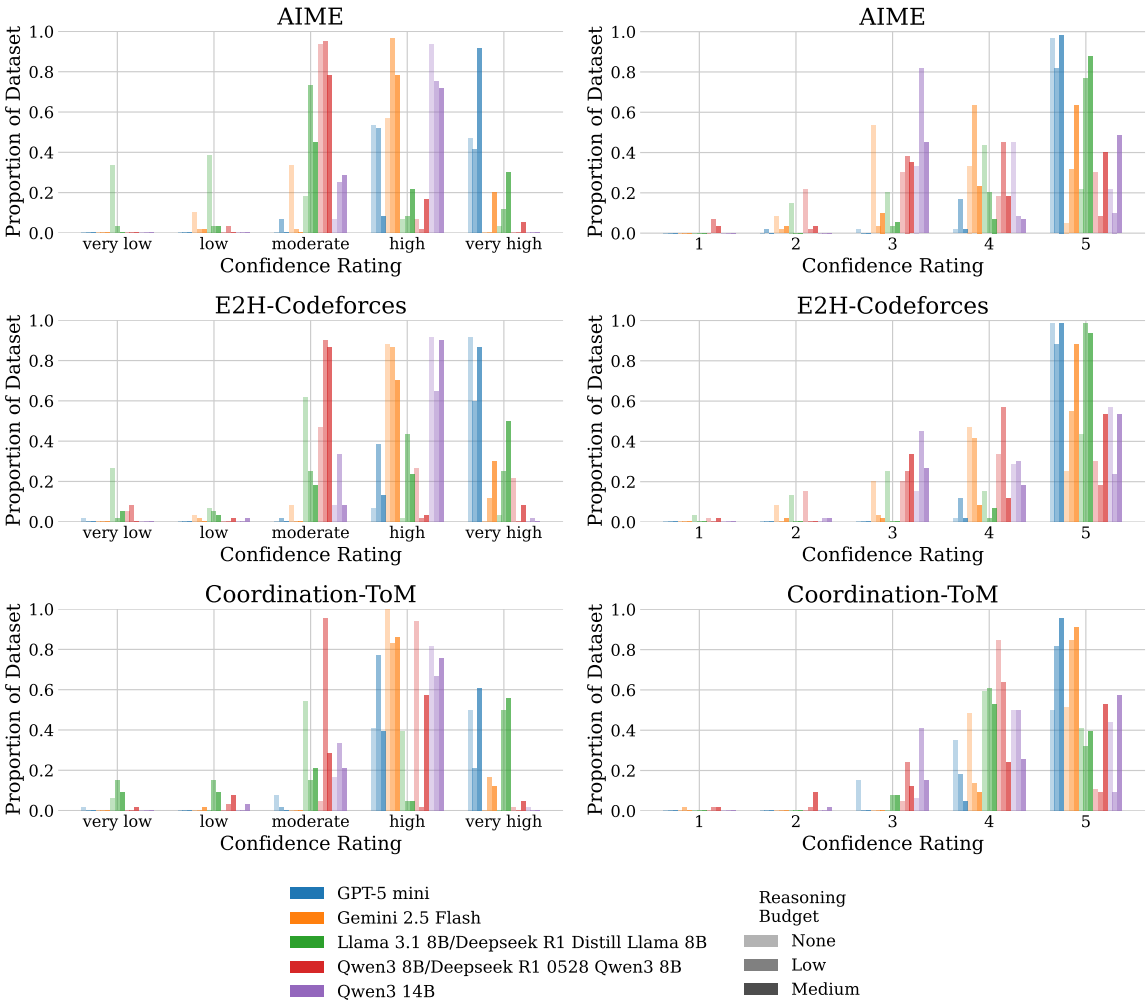

Figure A8: The distributions of verbalized confidence concentrate on “moderate”, “high”, and “very high” as well as 3, 4, and 5.