

BoN Appetit Team at LeWiDi-2025: Best-of-N Test-time Scaling Can Not Stomach Annotation Disagreements (Yet)

Tomas Ruiz^{1,2} Siyao Peng^{1,3} Barbara Plank^{1,3} Carsten Schwemmer^{1,2}

¹Ludwig Maximilian University of Munich, Germany

²Computational Social Sciences ³MaiNLP & MCML

{t.ruiz, siyao.peng, b.plank, carsten.schwemmer}@lmu.de

Abstract

Test-time scaling is a family of techniques to improve LLM outputs at inference time by performing extra computation. To the best of our knowledge, test-time scaling has been limited to domains with verifiably correct answers, like mathematics and coding. We transfer test-time scaling to the LeWiDi-2025 tasks to evaluate annotation disagreements. We experiment with three test-time scaling methods: two benchmark algorithms (Model Averaging and Majority Voting), and a Best-of-N (BoN) sampling method. The two benchmark methods improve LLM performance consistently on the LeWiDi tasks, but the BoN method does not. Our experiments suggest that the BoN method does not currently transfer from mathematics to LeWiDi tasks, and we analyze potential reasons for this gap.

1 Introduction

Supervised learning typically assumes a single fixed label per example. However, prior work documents substantial interpretative variability in human annotations, with annotators often disagreeing on labels (Roß et al., 2016; Warner and Hirschberg, 2012; Baan et al., 2022), especially for subjective Natural Language Processing (NLP) tasks (Ovstedt Alm, 2011). Plank (2022) and Cabitza et al. (2023) argue that this variability is informative rather than problematic and Röttger et al. (2022) suggests that variability should be explicitly integrated into the annotation processes.

The shared task **Learning With Disagreement** (LeWiDi) 2025 (Leonardelli et al., 2025) tackles this opportunity and provides four datasets with annotator-level metadata and label variation. We document the datasets in detail in subsection 3.6. The datasets support two different tasks: (1) Perspectivist task: Predicting the label of each individual annotator. (2) Soft-label task: Predicting the distribution of human annotations for a single

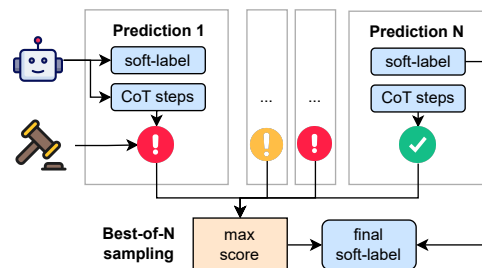


Figure 1: **Best-of-N sampling with step-wise scores.** For each problem a reasoning LLM generates N soft-labels and Chains-of-Thought (CoTs). Next, an LLM-as-a-judge scores each step in the CoT for correctness, and BoN selects the soft-label with the highest total score. *Takeaway:* Sampling multiple times increases the chances for a good prediction.

problem instance. This distribution is known as a **soft-label**, or a *human judgement distribution* (Nie et al., 2020).

In the previous iteration of the LeWiDi shared task (Leonardelli et al., 2023), many teams trained encoder-based models like BERT (Devlin et al., 2019) directly on soft-labels. However, the innovations in generalist Large Language Models (LLMs) and the rise of “reasoning” capabilities (Wei et al., 2022; OpenAI, 2025; Yang et al., 2025; DeepSeek-AI et al., 2025) motivated us to answer the following question:

“Can reasoning LLMs handle interpretative variability and annotation disagreement effectively at inference time?”

To answer this question, we turn to **test-time scaling** methods, like BoN sampling, which improve the LLMs performance by spending more compute per problem (Cobbe et al., 2021; Shen et al., 2021). These methods have been very successful in mathematics and coding, but have not been applied yet to NLP tasks with annotation disagreement, as far as we know. In this paper, we

take established test-time scaling methods and apply them to the LeWiDi tasks.

Our **contributions** are:

- A metric named prediction diversity, used to analyze the performance of test-time scaling methods on soft-label tasks. We show that it tracks problem difficulty on the LeWiDi tasks.
- We show that Model Averaging and Majority Voting consistently improve LLMs performance across all LeWiDi datasets.
- Finally, we show that BoN sampling with step-wise scores *does not work well* on the LeWiDi tasks, and analyze potential causes.

2 Related Work

2.1 Learning Interpretative Variability

Modeling the diverse perspectives that human annotators have on the same problems is important to prevent minority voices from being ignored (Leonardelli et al., 2021). Prior work on modeling annotator disagreement has explored various techniques, such as using separate model heads for each annotator (Mostafazadeh Davani et al., 2022), learning specific representations for annotators (Mokhberian et al., 2024), separating stable opinions from annotation mistakes (Gordon et al., 2021; Weber-Genzel et al., 2024), and using soft-labels to aid learning (Fornaciari et al., 2021; Uma et al., 2020). To evaluate models on soft-labels, Rizzi et al. (2024) propose using the Manhattan or Euclidean distance rather than the Cross-Entropy loss. In terms of quantifying the diversity of soft-labels, Singh et al. (2024) proposed the Jensen-Shannon Divergence in the context of ensemble classification.

2.2 Test-Time Scaling

Test-time scaling methods improve the performance of LLMs by spending more compute per problem instance. One approach is to refine an initial response iteratively with self-feedback (Madaan et al., 2023), or improve the response by following a set of rules (a *constitution*, Bai et al. 2022). Another common test-time scaling approach is **Best-of-N sampling**, where multiple solutions are sampled in parallel, and a verifier model scores or ranks the solutions to select the best one (Cobbe et al., 2021; Shen et al., 2021). The scores are computed based solely on the outcome

(correct or incorrect) of the task (*Outcome Reward Model*). But the scores can also be computed for the correctness of individual reasoning steps used to arrive at the answer (*Process Reward Model*). Lightman et al. (2024) showed that scoring individual steps in a Chain-of-Thought (CoT) for correctness, and discarding CoTs with faulty steps improves the performance of LLMs on the MATH dataset (Hendrycks et al., 2021). In their work, the scoring annotations were provided by humans. Follow-up work replaced the human scores with *automated scoring*, using either Monte Carlo (MC) sampling (Wang et al., 2024) or an LLM judge (*LLM-as-a-judge*, Zheng et al., 2023). Research by Zhang et al. (2025) and Zheng et al. (2025) showed that using LLMs to provide these scores generalizes well and is competitive with training a custom model.

A different approach that leverages diversity plus selection is *Mixture-of-Experts (MoE)*: multiple parallel expert subnetworks, with a gate that selects a few experts per input. Both MoE and test-time scaling are independent approaches that can be combined during model evaluation, *e.g.* as did Comanici et al. (2025) for the SWE-Bench (Jimenez et al., 2024).

3 Method

Nomenclature: A dataset is a collection of problem instances (*problem* in short). We sample a reasoning LLM N times to solve a problem. Each *sample* contains a *prediction* and a *CoT*. A prediction could be text, a soft-label or a list of integers (perspectivist task).

Our test-time scaling method is not novel but rather a combination of methods already established in the literature. Our innovation is to apply it to a new domain: the LeWiDi-2025 tasks. We refer to the method as *BoN sampling with step-wise scores*, or just *BoN sampling* in short. The method, shown in Figure 1, consists of three steps: (1) A reasoning LLM generates N samples for a problem. (2) A judge LLM scores each CoT-step in each sample for correctness. (3) We choose the sample with the best score for the final prediction. We explain all method details in subsection 3.1 and subsection 3.2. Table 1 is an overview of all methods we run experimentally, which include different baselines and benchmarks.

Method	Samples	Use CoT
Most Frequent	-	-
Simple Sampling	1	✗
<i>Model Averaging</i>	<i>N</i>	✗
<i>Majority Voting</i>	<i>N</i>	✗
BoN Oracle	<i>N</i>	✗
BoN + SWS	<i>N</i>	✓

Table 1: **Methods Overview.** The first two methods are baselines (subsection 3.4). The next two methods are our own benchmarks (subsection 3.5). The BoN Oracle (subsection 3.3) is a performance upper bound on our proposed method BoN + SWS (Step-Wise Scores, subsection 3.2). The models we submitted to the shared task are in *italic*.

3.1 LLM Setup

Prompts We prompt a reasoning LLM to solve the soft-label and perspectivist tasks directly. For example, in the soft-label task, we present the dataset (e.g. sarcasm detection), and instruct the model to predict the human soft-labels (snippet in Listing 1, full prompt in Listing 12).

Listing 1: Prompt Snippet (Soft-label Task)

```
Below is a context+response pair where human annotators rated the sarcasm level of the 'response' ranging from 1 (not at all) to 6 (completely). Please guess the distribution of ratings and output it in the final_response field in JSON format [...]
```

In the perspectivist task, we instead instruct the model to predict the label for each annotator (snippet in Listing 2, full prompt in Listing 13).

Listing 2: Prompt Snippet (Perspectivist Task)

```
Below is a context+response pair [...]. Please guess the rating given by each annotator and output them all in a list, in the same order as the annotators. [...]
```

We include a prompt section that explicitly instructs the model to reason about the *diverse perspectives and interpretations* that annotators could have. This improved performance by a small, but statistically significant margin, so we included it in all later experiments (see Appendix H).

3.2 BoN Sampling with Step-Wise Scores

In BoN sampling we score each of the N model samples and select the best for the final prediction. The score for a sample depends on the correctness of each step in its CoT. Our BoN sampling method

borrowed heavily from Lightman et al. (2024), so in Table 2 we summarize the differences and similarities between both.

	Lightman et al.	Ours
Domain	Math	LeWiDi Tasks
Model	GPT-4	Qwen3-32B
Scorer	Human(s)	LLM-as-a-judge
Reduction	Product	Mean
Sampling	Best-of-N	
Scores	bad=0, okay=0, good=1	

Table 2: **Comparison to Lightman et al. (2024).** First 4 rows are differences, last 2 rows are similarities.

First, we split the CoT into logical steps (details in Appendix B), and then score each step as either “great”, “okay”, or “bad” in line with Lightman et al. (2024). They used human annotations to train a scoring model, but we use an *LLM-as-a-judge* instead to provide the scores directly, as suggested by Zheng et al. (2025). The prompt for the LLM-as-a-judge is based on their scoring instructions (snippet in Listing 3, full prompt in Listing 14).

Listing 3: Prompt Snippet (LLM-as-a-judge)

```
Your goal is to grade an LLM's step-by-step solution to a problem. The model will often say things that look ok at first, but will turn out to be wrong on closer inspection - stay vigilant! Please mark each step with (great, okay, bad). [...]
```

We follow Lightman et al. (2024) to convert the three scores to numbers (bad=0, okay=0, good=1), and average all the step-wise scores to compute a *prediction-level score*. Lightman et al. (2024) used a product reduction, but in our experiments, mean reduction outperformed product. This step-wise scoring is repeated for $N = 10$ model samples and their corresponding CoTs. We select the one with the highest prediction-level score as the final prediction.

Models Previous research in hate speech detection and Natural Language Inference (NLI) showed that *explanations* are useful to judge the plausibility and correctness of annotations and model predictions (Mathew et al., 2020; Jiang et al., 2023; Weber-Genzel et al., 2024). We hypothesize that reasoning about annotator disagreements also requires a *deliberative* and *explanatory* approach that considers multiple interpretations and weights their likelihoods. Therefore, we use a reasoning LLM

for our experiments (Qwen3-32B, Yang et al. 2025). For LLM-as-a-judge we use a model from a different family besides Qwen3 (DeepSeek-R1-0528-Qwen3-8B, DeepSeek-AI et al. 2025). Sampling parameters are detailed in Appendix A.

3.3 Upper Bound on Performance

We determine the upper bound on performance of any BoN sampling method by computing a so-called **BoN oracle**. The BoN oracle is a hypothetical model that always selects the best prediction among N predictions (in our case, lowest distance). We compute the oracle in the *training* set by choosing the soft-label among the N predictions with the lowest distance to the human soft-label. However, for a dataset with unknown human soft-labels, we cannot compute the oracle. The oracle is an analytical tool to determine the BoN performance ceiling, rather than an algorithm to use in practice. The oracle soft-label p_o for a set of predictions P (size N) is defined as:

$$p_o = \arg \min_{p \in P} W(p, p_h) \quad (1)$$

where p_h is the human soft-label. We compute it for both the soft-label and perspectivist tasks.

3.4 Baselines

The LeWiDi 2025 shared task proposed the **Most Frequent Baseline**. In the soft-label task, this is the mean label value for each label across all training problems. In the perspectivist task, it is the most frequent label for each individual annotator. Our own basic baseline is the performance of the LLM without any test-time scaling, *i.e.* with a single sample per problem ($N = 1$). We call this **Simple Sampling**.

3.5 Test-Time Scaling Benchmarks

BoN sampling uses a lot more compute per problem than Simple Sampling (N times more). To benchmark BoN sampling fairly, we compare it with two test-time scaling algorithms that also create a single prediction out of N predictions.

Soft-label task We benchmark against *Model Averaging*, where all N soft-labels p^n are averaged into a single soft-label \bar{p} . The resulting soft-label \bar{p} is a valid probability distribution. Each entry i of \bar{p} is defined as:

$$\bar{p}_i = \frac{1}{N} \sum_{n=1}^N p_i^n \quad (2)$$

Perspectivist task We benchmark against *Majority Voting*, where we sample the model N times per problem, each prediction resulting in a label per annotator, and then select the most frequent label (within the N predictions) for each annotator.

3.6 LeWiDi-2025 Datasets

We report the datasets as provided by the LeWiDi-2025 shared task. All datasets provide some level of annotator-level metadata like gender, age, nationality, education and more.

The Conversational Sarcasm Corpus (CSC):

The CSC dataset by Jang and Frassinelli (2024) is a dataset for sarcasm detection with 7,036 entries (5,628 train, 704 dev, 704 test). Each entry consists of a context+response pair, where the response is rated for sarcasm on a 6-point Likert scale, by either 4 or 6 annotators.

The MultiPico dataset (MP):

The MP by Casola et al. (2024) is a dataset for irony detection with 18,778 entries (12,017 train, 3,005 dev, 3,756 test). Each entry consists of a post-reply pair from Twitter and Reddit, and the reply’s irony is rated as either ironic (1) or not ironic (0) by between 2 and 21 annotators.

The Paraphrase Detection dataset (PAR):

The Paraphrase is a dataset by the MainLP lab¹ for paraphrasing detection with 500 entries (400 train, 50 dev, 50 test). Each entry has two questions from Quora Question Pairs (QQP), and annotators rate how strongly the questions are paraphrases of one another from -5 to 5. Each entry is rated by 4 annotators.

The VariErr NLI dataset (VEN):

VariErrNLI by Weber-Genzel et al. (2024) is a dataset for NLI with 500 samples (400 train, 50 dev, 50 test). Annotators can assign any and multiple of the NLI categories (entailment, contradiction, neutral) for each entry. Each entry is annotated by 4 annotators.

3.7 Metrics

Soft-label Task As suggested by the LeWiDi task, we report *Manhattan Distance* for the MP and VEN datasets, and *Wasserstein Distance* for the CSC and PAR datasets. Both distances are exactly equivalent when applied to binary datasets (Rizzi et al., 2024). The Wasserstein distance measures the minimum “work” needed to transform one

¹<https://mainlp.github.io/>

probability distribution into another, where “work” equals the amount of mass moved times the distance.

Perspectivist Task For the perspectivist task, we report *Error Rate* (1 - accuracy) for the MP and VEN datasets, and *Absolute Distance* for the CSC and PAR datasets. We divide the Absolute Distance by the range of the Likert scale, in line with the LeWiDi organizers. Both metrics are exactly equivalent when applied to binary datasets.

Prediction Diversity BoN sampling requires a diverse set of predictions for each problem. Otherwise, if all predictions were the same (or very similar), it would not matter which one is selected, and BoN sampling would provide no improvement over Simple Sampling. Therefore, we quantify the variability of the soft-labels across the N predictions for each problem, and call this the *prediction diversity*.

We implement this as the average pair-wise distance between all N soft-labels for a single problem. For the LeWiDi datasets, we use the Wasserstein distance because it can capture distances in Likert scales. We do not compare soft-labels to themselves, because the distance is 0. This is why we divide by $N(N - 1)$ rather than by N^2 . The formula for diversity D is:

$$D(P) = \frac{1}{N(N - 1)} \sum_{i=1}^N \sum_{j \neq i}^N W(p^i, p^j) \quad (3)$$

where P is the set of N soft-labels, and $W(p^i, p^j)$ is the Wasserstein distance between the soft-labels p^i and p^j . Note that measuring the diversity of a set of predictions P is different from measuring the spread of a single soft-label (*i.e.* measuring the entropy of the soft-label).

Problem Difficulty Classically, problem difficulty is measured as the percentage of correct answers over N attempts. For soft-label tasks, it can instead be defined as the distance between predictions and human soft-label across N attempts. In the LeWiDi task, we posit a relationship between prediction diversity and problem difficulty: low diversity arises when the model perceives no ambiguity (the problem is easy or only one interpretation is considered), while high diversity arises when multiple plausible interpretations exist and the model’s N predictions vary. Since both prediction diversity and distances (Wasserstein, Manhattan) are computable, their correlation is empirically measurable.

4 Results

In Table 3 we summarize the results on the test, taken from the LeWiDi leaderboard², since we have no access to the test set ground truth. In Table 4 we present a performance overview of all methods for the LeWiDi datasets. We did not train on the train set, so we used it for evaluation.

Dataset	Task	
	Soft-label (↓)	Perspectivist (↓)
CSC	0.928	0.231
MP	0.466	0.414
PAR	1.797	0.228
VEN	0.356	0.272
Avg. Rank	5th	6th
Out of	15	11
Method	Model Averaging	Majority Voting

Table 3: Results on the **test** set of the LeWiDi datasets (lower is better). Values are from the LeWiDi leaderboard. We submitted our best performing methods, Model Averaging and Majority Voting.

4.1 Test-Time Scaling Benchmarks

We first report the performance of all methods except BoN sampling. The orange bar in Figure 2 is the performance *Simple Sampling* with Qwen3-32B. It outperforms the *Most Frequent Baseline* on 3 out of 4 datasets in the soft-label task, but only in 2 out of 4 datasets in the perspectivist task. The test-time scaling benchmarks (green) are *Model Averaging* and *Majority Voting*. Both methods consistently improve performance over Simple Sampling across datasets and tasks. We discuss the effects of Model Averaging on soft-label entropy in Appendix E and compare it with naive soft-label smoothing. The *BoN oracle* (red) is meant to show the performance ceiling of any BoN sampling method. Its strong performance indicates that, at least *theoretically*, a good BoN sampling method can achieve very good performance on the LeWiDi tasks.

4.2 Best-of-N Sampling with Step-Wise Scores

The BoN sampling method has inconsistent performance in the LeWiDi datasets, as shown in Figure 3. Performance is often flat with the number of samples N , or varies wildly with judge model. *E.g.* in the MP dataset, the Deepseek judge is consistently *worse* (higher distance) than Simple Sampling on both tasks (soft-label and perspectivist). BoN sampling is only competitive with the benchmarks (red

²LeWiDi leaderboard: <https://le-wi-di.github.io/>

Method	Soft-label Task				Perspectivist Task			
	CSC	Par	MP	VEN	CSC	Par	MP	VEN
Most Frequent Baseline	1.14	2.89	0.26	0.27	0.21	0.36	0.30	0.33
Simple Sampling	1.00	1.96	0.26	0.22	0.24	0.25	0.43	0.32
<i>Model Averaging</i>	0.91	1.78	0.24	0.20	-	-	-	-
<i>Majority Voting</i>	-	-	-	-	0.23	0.25	0.40	0.30
BoN Sampling + SWS	1.01	1.93	0.26	0.22	0.24	0.25	0.42	0.32
BoN Oracle	0.51	1.29	0.11	0.11	0.15	0.18	0.18	0.16
Metric (\downarrow)	Wasserstein			Manhattan	Abs. Dist.		Error Rate	

Table 4: Results on the **train** set of the LeWiDi datasets. In **bold** is the best performing method by column. BoN sampling underperforms the test-time scaling benchmarks, even though the BoN oracle suggests a high performance ceiling. We submitted to the LeWiDi shared task the Model Averaging (soft-label task) and Majority Voting (perspectivist task) methods, since they performed best.

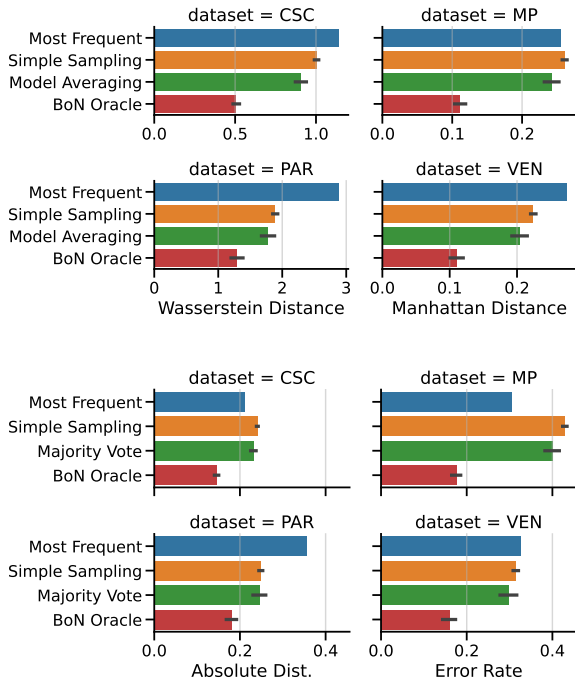


Figure 2: **Test-time Scaling Benchmarks**. Top: soft-label task. Bottom: perspectivist task. Distance metric on the x-axis (lower is better). Model Averaging and Majority Voting (green) are consistently better than Simple Sampling (orange) in the soft-label and perspectivist tasks, respectively.

horizontal lines) in a single case (perspectivist task, PAR dataset, Qwen3-32B judge). These inconsistent results raise the question why step-wise scoring is not effective in the LeWiDi tasks. For the BoN sampling numbers in Table 3, we report the Qwen3-32B judge, because it performs slightly better than the Deepseek judge on the perspectivist task. For the LeWiDi shared task, we submitted the predictions for Model Averaging and Majority Voting, rather than BoN sampling.

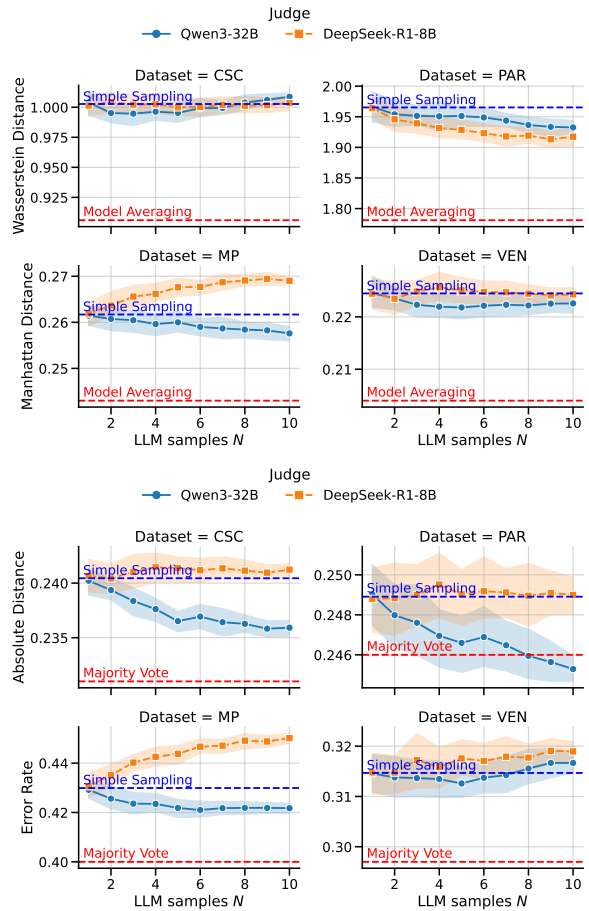


Figure 3: **Best-of-N Sampling on LeWiDi Tasks**. Top: soft-label task. Bottom: perspectivist task. Distance metric on the y-axis (lower is better). Higher N should lead to better performance, but does not. No consistent pattern emerges across datasets and tasks. In red are the test-time scaling benchmarks, which BoN generally does not beat. The shaded areas show the 0.25 and 0.75 quantiles.

4.3 Prediction Diversity

Back to the LeWiDi tasks, we empirically observe that prediction diversity correlates with model performance (Figure 4): diversity increases for difficult problems and decreases for easier ones. For analysis, we binned prediction diversity into five quantiles (but the trends hold for any number of bins). We document the distribution of prediction diversity across datasets in Appendix F. Prediction diversity strongly affects test-time scaling methods, as shown in Figure 5: the BoN oracle performance (the upper bound for any BoN sampling method) increases with diversity. The same applies for Model Averaging.

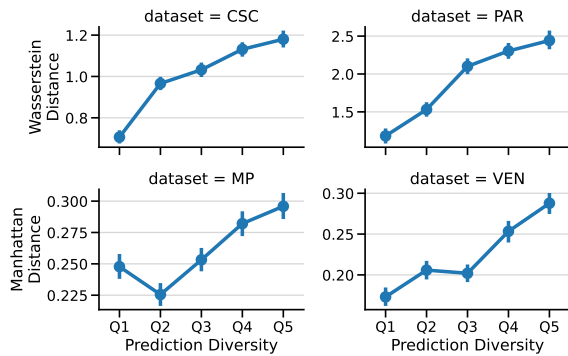


Figure 4: Model performance (lower is better) varies with prediction diversity and is related to the difficulty of the problem.

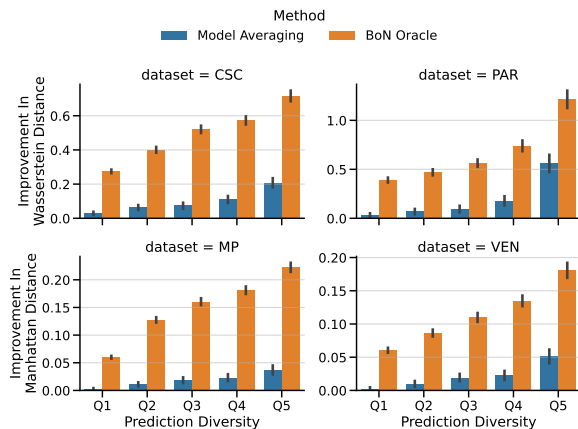


Figure 5: High prediction diversity leads to better performance of test-time scaling methods. Both the upper bound (BoN oracle) and Model Averaging benefit from higher prediction diversity. The y-axis shows the improvement over Simple Sampling.

Table 5 shows that Model Averaging achieves a significant fraction of the theoretical performance gains dictated by the BoN oracle. For example, in the top quantile of the PAR dataset, Model Averag-

ing achieves 46% of the performance gains of the BoN oracle.

Dataset	Prediction Diversity				
	Q1	Q2	Q3	Q4	Q5
CSC	0.11	0.16	0.14	0.19	0.29
PAR	0.09	0.14	0.17	0.24	0.46
MP	0.03	0.09	0.12	0.13	0.17
VEN	0.03	0.11	0.18	0.18	0.27

Table 5: Fraction of the BoN oracle performance gains that Model Averaging achieves for different datasets and prediction diversities.

5 Discussion

5.1 BoN Sampling Underperformance

We were surprised by the underperformance of BoN sampling in the LeWiDi datasets. To verify that we had not made a mistake in our implementation of BoN sampling, we ran our method on two math datasets (PRM800K and AIME), as shown in Figure 6.

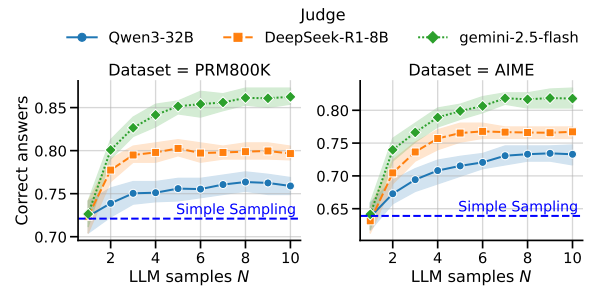


Figure 6: **Best-of-N Sampling in Mathematics:** The performance of BoN sampling (Correct answers, higher is better) improves with the number of samples N and with stronger judges. The shaded area shows the 0.25 and 0.75 quantiles: improvements are consistent.

The results are in line with Lightman et al. (2024). Using the best judge and $N = 10$ samples, the rate of correct answers jumps by 14% on PRM800K and by 18% in AIME. More information about both datasets is in Appendix C.

Why is BoN sampling effective in math, but not in the LeWiDi tasks? We think the LeWiDi tasks are not inherently harder or more intractable than math problems. The gap we observe is a failure of *cross-domain generalization*. For example, we observed that the shift in domain introduces unexpected side-effects:

1. We found qualitative evidence of the LLM being **more vague** in its formulation of CoT

steps in LeWiDi tasks (see Appendix I). When steps are vague, it is harder for a judge to discriminate between good and bad steps. During *post-training*, the Qwen3 model was likely never rewarded for summarizing precise arguments around interpretative variation and different perspectives. In contrast, we know that Qwen3 has been post-trained to reason on “[...] *math, code, logical reasoning, and general STEM problems.*” (Yang et al., 2025). We find same in the technical reports for Deepseek R1 and Gemini-2.5 (DeepSeek-AI et al., 2025; Comanici et al., 2025).

2. We empirically observe that LLMs and judges both spend a higher **compute budget** (*i.e.* they produce more tokens) on reasoning when solving the mathematical tasks than on the LeWiDi tasks as shown in Figure 7. Since reasoning capabilities are learned during post-training, we hypothesize that this difference is also caused by the standard post-training recipe.

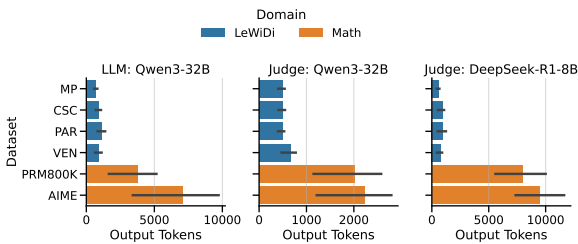


Figure 7: **Compute Budget** used by LLM and judges on different domains. Error bars are the 0.25 and 0.75 quantiles: they show large variability in output length. The models invest an **order of magnitude** more compute budget into solving AIME problems than in the LeWiDi tasks. Both Qwen3 and Deepseek-R1-8B show this bias.

5.2 Logical Steps in LeWiDi Tasks

One might argue that step-wise scoring requires a clear boundary between correct and incorrect steps, which is lacking in tasks with strong interpretative variation. We argue against this for two reasons:

1. Mathematical problem solving is also not always clear-cut. Lightman et al. (2024) show many steps add no insight or progress, leading them to use an “okay” label alongside “great” and “bad”.

2. LeWiDi tasks define correctness precisely (*e.g.*, Wasserstein Distance 0). Steps that are logical, plausible, and advance a prediction are “great”, while vague or unsound steps are “bad”. Previous perspectivist research has also leveraged explanations to judge the validity of annotations (Weber-Genzel et al., 2024).

We see no theoretical conflict between perspectivism and step-wise scoring. Rather, adjusting the method to incorporate perspectivist principles is an avenue for future work. For example, using different step labels like “plausible”, “implausible”, “vague”, etc.

6 Conclusion

We present a systematic evaluation of three test-time scaling methods on the LeWiDi tasks. Our key findings are: (1) Our prediction diversity metric correlates with test-time scaling performance and problem difficulty on the LeWiDi soft-label task. (2) Model Averaging and Majority Voting consistently improve LLM performance across the LeWiDi tasks. (3) BoN sampling with step-wise scores does not transfer from the domain of mathematics to the LeWiDi tasks, potentially due to vague reasoning steps and lower reasoning compute used. We hypothesize that this difference is caused by the post-training recipes of current reasoning LLMs, which lean towards mathematical and logical reasoning. The performance on datasets with annotation disagreements could potentially be improved by including similar tasks in the post-training recipe.

Limitations

We articulated the limitations of BoN sampling with step-wise scores in the LeWiDi tasks. We do not explore prompt optimization thoroughly, because we think that methods should be robust over different prompts. In terms of the prediction diversity metric, we suggest that authors evaluate the correlation with problem difficulty on their own datasets, since we showed an empirical rather than theoretical relationship.

Acknowledgements

This research is funded by the Bavarian Research Institute for Digital Transformation (bidt).

References

- Joris Baan, Wilker Aziz, Barbara Plank, and Raquel Fernandez. 2022. [Stop measuring calibration when humans disagree](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1892–1915, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, and 32 others. 2022. [Constitutional ai: Harmlessness from ai feedback](#). *Preprint*, arXiv:2212.08073.
- Federico Cabitza, Andrea Campagner, and Valerio Basile. 2023. [Toward a perspectivist turn in ground truthing for predictive computing](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(6):6860–6868.
- Silvia Casola, Simona Frenda, Soda Marem Lo, Erhan Sezerer, Antonio Uva, Valerio Basile, Cristina Bosco, Alessandro Pedrani, Chiara Rubagotti, Viviana Patti, and Davide Bernardi. 2024. [MultiPICo: Multilingual perspectivist irony corpus](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16008–16021, Bangkok, Thailand. Association for Computational Linguistics.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. [Training verifiers to solve math word problems](#). *Preprint*, arXiv:2110.14168.
- Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marris, Sam Petulla, Colin Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szpektor, Nan-Jiang Jiang, and 3290 others. 2025. [Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities](#). *Preprint*, arXiv:2507.06261.
- DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. [Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning](#). *Preprint*, arXiv:2501.12948.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of deep bidirectional transformers for language understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Tommaso Fornaciari, Alexandra Uma, Silviu Paun, Barbara Plank, Dirk Hovy, and Massimo Poesio. 2021. [Beyond black & white: Leveraging annotator disagreement via soft-label multi-task learning](#). In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2591–2597, Online. Association for Computational Linguistics.
- Mitchell L Gordon, Kaitlyn Zhou, Kayur Patel, Tatsunori Hashimoto, and Michael S Bernstein. 2021. [The disagreement deconvolution: Bringing machine learning performance metrics in line with reality](#). In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–14.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. [Measuring mathematical problem solving with the MATH dataset](#). In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.
- Hyewon Jang and Diego Frassinelli. 2024. [Generalizable sarcasm detection is just around the corner, of course!](#) In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4238–4249, Mexico City, Mexico. Association for Computational Linguistics.
- Nan-Jiang Jiang, Chenhao Tan, and Marie-Catherine de Marneffe. 2023. [Ecologically valid explanations for label variation in NLI](#). In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10622–10633, Singapore. Association for Computational Linguistics.

- Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R Narasimhan. 2024. [SWE-bench: Can language models resolve real-world github issues?](#) In *The Twelfth International Conference on Learning Representations*.
- W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, and I. Stoica. 2023. [Efficient memory management for large language model serving with pagedattention](#). In *Symposium on Operating Systems Principles*.
- Elisa Leonardelli, Gavin Abercrombie, Dina Almanea, Valerio Basile, Tommaso Fornaciari, Barbara Plank, Verena Rieser, Alexandra Uma, and Massimo Poesio. 2023. [SemEval-2023 task 11: Learning with disagreements \(LeWiDi\)](#). In *Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023)*, pages 2304–2318, Toronto, Canada. Association for Computational Linguistics.
- Elisa Leonardelli, Silvia Casola, Siyao Peng, Giulia Rizzi, Valerio Basile, Elisabetta Fersini, Diego Frassinelli, Hyewon Jang, Maja Pavlovic, Barbara Plank, and Massimo Poesio. 2025. [Lewidi-2025 at nlperspectives: third edition of the learning with disagreements shared task](#). In *Proceedings of the 4th Workshop on Perspectivist Approaches to NLP (NLPerspectives)*. Association for Computational Linguistics.
- Elisa Leonardelli, Stefano Menini, Alessio Palmero Aprosio, Marco Guerini, and Sara Tonelli. 2021. [Agreeing to disagree: Annotating offensive language datasets with annotators’ disagreement](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10528–10539, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. [Let’s verify step by step](#). In *The Twelfth International Conference on Learning Representations*.
- Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhunoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. [Self-refine: Iterative refinement with self-feedback](#). In *Thirty-seventh Conference on Neural Information Processing Systems*.
- B. Mathew, P. Saha, S. M. Yimam, C. Biemann, P. Goyal, and A. Mukherjee. 2020. [Hatexplain: A benchmark dataset for explainable hate speech detection](#). In *AAAI Conference on Artificial Intelligence*, pages 14867 – 14875.
- Negar Mokherian, Myri Marmarelis, Frederic Hopp, Valerio Basile, Fred Morstatter, and Kristina Lerman. 2024. [Capturing perspectives of crowdsourced annotators in subjective learning tasks](#). In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7337–7349, Mexico City, Mexico. Association for Computational Linguistics.
- Aida Mostafazadeh Davani, Mark Díaz, and Vinodkumar Prabhakaran. 2022. [Dealing with disagreements: Looking beyond the majority vote in subjective annotations](#). *Transactions of the Association for Computational Linguistics*, 10:92–110.
- Yixin Nie, Xiang Zhou, and Mohit Bansal. 2020. [What can we learn from collective human opinions on natural language inference data?](#) In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9131–9143, Online. Association for Computational Linguistics.
- OpenAI. 2025. [OpenAI o3 and o4-mini System Card](#). Technical report, OpenAI. Technical report — system card.
- Cecilia Ovesdotter Alm. 2011. [Subjective natural language problems: Motivations, applications, characterizations, and implications](#). In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 107–112, Portland, Oregon, USA. Association for Computational Linguistics.
- Barbara Plank. 2022. [The “problem” of human label variation: On ground truth in data, modeling and evaluation](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10671–10682, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Giulia Rizzi, Elisa Leonardelli, Massimo Poesio, Alexandra Uma, Maja Pavlovic, Silviu Paun, Paolo Rosso, and Elisabetta Fersini. 2024. [Soft metrics for evaluation with disagreements: an assessment](#). In *Proceedings of the 3rd Workshop on Perspectivist Approaches to NLP (NLPerspectives) @ LREC-COLING 2024*, pages 84–94, Torino, Italia. ELRA and ICCL.
- Paul Röttger, Bertie Vidgen, Dirk Hovy, and Janet Pierrehumbert. 2022. [Two contrasting data annotation paradigms for subjective NLP tasks](#). In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 175–190, Seattle, United States. Association for Computational Linguistics.
- Björn Roß, Michael Rist, Guillermo Carbonell, Benjamin Cabrera, Nils Kurowsky, and Michael Wojatzki. 2016. [Measuring the reliability of hate speech annotations: The case of the european refugee crisis](#).
- Jianhao Shen, Yichun Yin, Lin Li, Lifeng Shang, Xin Jiang, Ming Zhang, and Qun Liu. 2021. [Generate &](#)

- rank: A multi-task framework for math word problems. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2269–2279, Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Avyav Singh, Ekaterina Shutova, and Helen Yanakoudakis. 2024. Learning new tasks from a few examples with soft-label prototypes. In *Proceedings of the 9th Workshop on Representation Learning for NLP (RepL4NLP-2024)*, pages 215–236, Bangkok, Thailand. Association for Computational Linguistics.
- Alexandra Uma, Tommaso Fornaciari, Dirk Hovy, Silviu Paun, Barbara Plank, and Massimo Poesio. 2020. A case for soft loss functions. 8:173–177.
- Hemish Veeraboina. 2023. Aime problem set 1983-2024.
- Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439, Bangkok, Thailand. Association for Computational Linguistics.
- William Warner and Julia Hirschberg. 2012. Detecting hate speech on the world wide web. In *Proceedings of the Second Workshop on Language in Social Media*, pages 19–26, Montréal, Canada. Association for Computational Linguistics.
- Leon Weber-Genzel, Siyao Peng, Marie-Catherine De Marneffe, and Barbara Plank. 2024. VariErr NLI: Separating annotation error from human label variation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2256–2269, Bangkok, Thailand. Association for Computational Linguistics.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.
- Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. The lessons of developing process reward models in mathematical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 10495–10516, Vienna, Austria. Association for Computational Linguistics.
- Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. ProcessBench: Identifying process errors in mathematical reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1009–1024, Vienna, Austria. Association for Computational Linguistics.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

A Sampling Parameters

Yang et al. (2025) suggest two different parameter configurations for Qwen3: for thinking and non-thinking modes. In early experiments we found almost no difference in performance between both configurations, but observed less variation with the non-thinking configuration, so we used those parameters in our experiments: top-k=20, top-p=0.8, temperature=0.7, presence-penalty=1.5. We used the same parameters for the Deepseek-R1-8B model. For Gemini-2.5-flash we used the default parameters documented in Google’s documentation: top-k=64, top-p=0.95, temperature=1.0³.

B Splitting the CoT into Steps

To score each step of a CoT for correctness, it must be first split into steps. We instruct the model to answer using a structured format (JSON) with separate fields for the prediction and the CoT steps, as shown in Listing 4.

Listing 4: Output format for the LLM

```
{
  'steps': [
    '<step 1 text>',
    '<step 2 text>',
    ...
  ],
  'final_response': '<text>'
}
```

We found that this approach to get logical steps is more robust than two alternatives: (1) Using *string matching* (e.g. on double line breaks) to split a CoT into steps, because it produces overly granular, incoherent steps, where e.g. a bulleted list becomes a step on its own. (2) Using a separate LLM to *reformat* the CoT into logical steps, because the reformatting model sometimes rephrases and truncates the original CoT instead of only reformatting it. We think that the original LLM is best positioned to split its own reasoning process into coherent, logical steps.

One might argue these steps are *constructed ex-post* and do not reflect the model’s true reasoning. However, during a math exam, students are allowed to sketch ungraded work on separate sheets, and then present a clean step-by-step solution. We follow this same principle, and our math experiments show that the ex-post steps are expressive enough to discriminate good and bad reasoning.

³<https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-5-pro>

C Mathematical Datasets

Our method for providing step-wise scores is *completely automated* and requires no human annotations for the CoT steps at all. As a **sanity check** that our test-time scaling implementation is correct, we also include in our BoN evaluation two datasets with mathematical problems, where we expect step-wise scoring to perform very well. The datasets are: (1) High-school math problems and solutions compiled in the **PRM800K** dataset by Lightman et al. (2024). The problems are originally from the MATH dataset by Hendrycks et al. (2021). (2) Mathematical problems given to the top 2.5% to 5% of high-school students in the US from the American Invitational Mathematics Examination (AIME) compiled by Veeraboina (2023) and ranging from 1983 to 2024. The AIME problems are generally more difficult than those in PRM800K. Many math problems are solved by Qwen3-32B in 10/10 samples, which makes BoN sampling unnecessary. We skipped these problems in our BoN evaluation, which is why the horizontal line for Simple Sampling is relatively low in Figure 6.

D LLM Compliance

When using an LLM with structured output, we need to measure its adherence to the output format of the prompt. We call this **compliance**. As we show in Table 6, the compliance level varies by dataset and by task.

Dataset	Perspectivist	Soft-label
MP	100.0	100.0
CSC	100.0	99.3
VEN	100.0	93.9
PAR	100.0	86.2

Table 6: Percentage of compliant predictions sorted by dataset from highest to lowest. The PAR soft-label task is difficult because the weight of 11 classes must sum to 1.0.

We observe near-perfect compliance for the CSC and MP datasets. The VEN dataset has lower compliance because of the nested structure of the predictions (one for each NLI category). The lowest compliance is in the PAR datasets, which has 11 classes (-5 to 5, including 0). We found that the LLM outputs correct JSON for PAR, but often the soft-labels did not sum exactly to 1. We experimented with enforcing **structured outputs** in vLLM, but initial experiments showed that the LLM would

sometimes output infinite newline characters until it reached the output token limit, which is valid JSON, so we dropped this constraint.

E Model Averaging and Entropy

Model Averaging has an **adaptive** flattening effect on the model’s soft-labels: When the model identified a consensus interpretation (regime of low prediction diversity), Model Averaging keeps soft-labels intact (*e.g.* peaky). And when the model’s answers are diverse, Model Averaging flattens the soft-labels, which has a hedging effect. We compare Model Averaging with a naive **smoothing** method, which flattens a soft-label by averaging it with the uniform distribution, therefore increasing the entropy of the soft-label.

Figure 8 shows the entropy of the soft-labels for Qwen3-32B, for different datasets and sampling methods. It shows that smoothing the soft-labels does not automatically improve the model performance and that Model Averaging is much more adaptive than the naive smoothing.

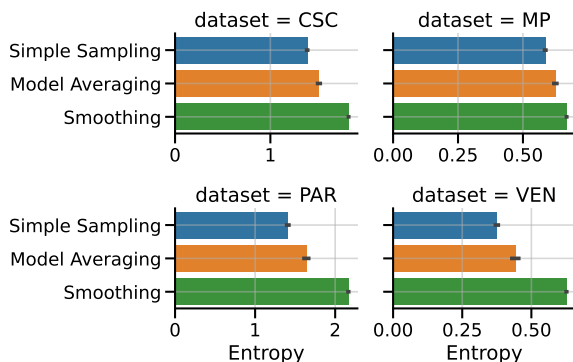


Figure 8: **Entropy of soft-labels:** We observe that both smoothing (green) and Model Averaging (orange) increase the entropy of the soft-labels, but only Model Averaging improves the model performance.

F Prediction Diversity

The distribution of prediction diversity by dataset is shown in Figure 9. We observe that it is distributed with a single peak in the center, and sometimes has a right tail.

G Compute Infrastructure

We use the vLLM engine (Kwon et al., 2023) to run the models, because of its high throughput, which help us compute N samples per example in parallel. vLLM can also be configured to parse the CoT and return them separately from the final answer. All

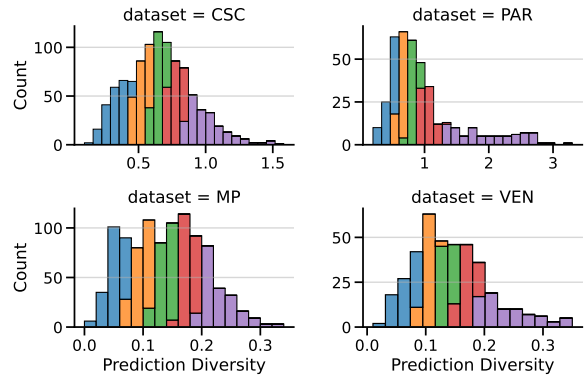


Figure 9: Distribution of prediction diversity by dataset. The distributions follow a normal-like distribution, and the PAR dataset shows a longer tail to the right. The colors indicate the quantiles of the distribution.

our experiments are run on a single NVIDIA H100 GPU, except for the Qwen3-32B model, which is run on two GPUs. We called Gemini-2.5-flash over the Google Cloud API.

H Prompt Ablations

We created two prompt variations that could potentially affect performance for interpretative tasks: (1) One variant provides a *dictionary definition*⁴ of sarcasm (for CSC), or irony (for MP). (2) The second variant explicitly instructs the model to consider different *perspectives and interpretations*.

Listing 5: Prompt Section Defining Sarcasm

Use this definition for sarcasm: "The use of remarks that clearly mean the opposite of what they say, made in order to hurt someone's feelings or to criticize something in a humorous way"

Listing 6: Prompt Section to Consider Perspectives

Think about the perspectives that different annotators might have and how they could potentially interpret the post-reply pair.

We perform an ablation analysis to determine the impact of the two prompt variants: (1) first, we remove the prompt section that defines irony and sarcasm and (2) we remove the prompt section about considering diverse perspectives.

As shown in Table 7 for **ablation 1**, we observe mixed effects: In the CSC dataset, including the definition of sarcasm improves performance, while in the MP dataset, including the definition of irony

⁴Online dictionary: <https://dictionary.cambridge.org/>

decreases performance. We compute the 95% confidence interval of mean performance using the bootstrap method to rule out the possibility that performance differences between prompts are a sampling artifact.

For ablation 2, we observe that prompting the model to consider diverse perspectives improves performance in 3 out of 4 datasets (CSC, MP, VEN). In the PAR dataset, performance is not affected by the prompt section on diverse perspectives.

I Vague CoT Steps

Below is a qualitative comparison of two responses to the same problem by Qwen3-32B. [Listing 7](#) describes the problem (ID=637 of the CSC dataset on sarcasm detection). [Listing 8](#) shows a response with very vague CoT steps, which are not strictly wrong, but are so general that they could be applied to any or all problems. [Listing 9](#) shows a different response with very specific CoT steps that directly refer to the problem statement and are easier to judge.

Note that these are the structured logical steps, rather than the raw CoT. The raw CoT for the vague answer is in [Listing 10](#) and the one for the concrete answer is in [Listing 11](#). Neither of the raw CoT are vague, so there is no reason why the model should generate vague logical steps from either of them. We observed this “vagueness” behavior particularly in the CSC dataset.

Listing 7: Problem Description (Sarcasm Detection)

```
{
  "context": "You and Steve went out last night and drank a lot. The next day, Steve excitedly says that he fell in love with the girl he met last night and that he is going to quit his job and join her on a backpacking trip. Steve says, \"this girl is totally amazing and I'm not gonna miss the opportunity to be with her. I'm quitting my job and traveling with her!\"",
  "response": "Man, you are fucking serious right now? You're going to quit your job and travel with some girl you just met last night while drunk? No girl is worth quitting your job and screwing your life over, especially not a girl you just met last night."
}
```

Listing 8: Vague Logical Steps

```
"Analyze the context and response to identify elements of sarcasm based on the given definition.",
```

```
"Determine if the response uses remarks that mean the opposite of what they say to criticize or hurt someone humorously.",
```

```
"Assess the likelihood of different annotator interpretations, considering variations in perception of sarcasm.",
```

```
"Assign probabilities to each rating (1-6) based on the analysis, ensuring the sum equals 1.0."
```

Listing 9: Specific Logical Steps

```
"Analyze the context and response to determine if sarcasm is present. The response directly criticizes Steve's decision without evident irony or mock praise.",
```

```
"Consider that the response uses rhetorical questions and direct statements to express disapproval, which aligns more with blunt criticism than sarcasm.",
```

```
"Assess potential annotator perspectives: most may view the response as non-sarcastic (rating 1), while others might detect mild sarcasm in the tone (ratings 2-3).",
```

```
"Distribute probabilities accordingly, allocating higher weights to lower sarcasm ratings and minimal probabilities to higher ones."
```

J Prompts

Prompt	Dataset							
	CSC		MP		PAR		VEN	
	low	high	low	high	low	high	low	high
Default	1.002	1.013	0.256	0.258	1.838	1.920	0.217	0.227
-Definition (ablation 1)	1.057	1.071	0.246	0.248	-	-	-	-
-Perspectives (ablation 2)	1.027	1.040	0.263	0.265	1.840	1.924	0.236	0.247

Table 7: Two Prompt Ablations: The numbers are mean performance in terms of Wasserstein Distance (lower is better). The column “low” and “high” are the bounds of the 95% confidence intervals on the performance computed with the bootstrap method. Ablation 1 shows that including/excluding the definition of irony / sarcasm has mixed effects. Ablation 2 shows that not instructing the model to consider diverse perspectives has negative consequences in 3 out of 4 datasets.

Listing 10: Raw CoT for Vague Answer

Okay, let's tackle this problem step by step. First, I need to understand the task. The user provided a context and a response pair, and I have to guess how human annotators would rate the sarcasm level of the response on a scale from 1 to 6. The final answer should be a probability distribution over these ratings, with each probability summing to 1.0.

The definition given for sarcasm is important here: using remarks that mean the opposite to hurt or criticize humorously. So, I need to analyze if the response fits this definition and how different annotators might perceive it differently.

Looking at the context: Steve met a girl after drinking a lot and decided to quit his job to travel with her. The response is a friend (the speaker) reacting to Steve's decision. The response says, "Man, you are fucking serious right now? You're going to quit your job and travel with some girl you just met last night while drunk? No girl is worth quitting your job and screwing your life over, especially not a girl you just met last night."

First, I need to determine if there's any sarcasm in the response. Sarcasm often involves irony or mock praise. In this case, the speaker is clearly criticizing Steve's impulsive decision. The phrases like "you're fucking serious" and "screwing your life over" indicate frustration or disapproval. However, there's no direct use of words that mean the opposite. Instead, the speaker is being straightforward in their criticism. They are pointing out the irrationality of Steve's actions without obvious irony.

But maybe some annotators could interpret parts of the response as sarcastic. For example, when the speaker says "especially not a girl you just met last night," they might be emphasizing the absurdity, which could be seen as a form of understatement or rhetorical question. However, under the given definition, sarcasm requires the remark to clearly mean the opposite. Here, the speaker is directly stating their opinion without such inversion.

Considering different perspectives: Some annotators might think that the speaker is being sarcastic in their tone, even if the words aren't technically ironic. Others might see it as direct criticism without sarcasm. Since sarcasm can be subtle, there's a chance that a portion of annotators might rate it as low-level sarcasm (like 2 or 3), but most might see it as non-sarcastic (rating 1). A few might misinterpret and give higher ratings, thinking the speaker is being sardonic, but that's less likely given the explicit criticism.

I should distribute the probabilities accordingly. Most of the probability mass would be on 1 (not at all sarcastic) and 2 (slightly sarcastic). Maybe a small portion on 3 for those who see some underlying sarcasm. Higher ratings (4-6) would be minimal since the response is pretty straightforward. Let me check if the sum adds up to 1.0 and adjust the probabilities.

Listing 11: Raw CoT for Concrete Answer

Okay, let's tackle this problem. So, the task is to guess the distribution of sarcasm ratings given a context and response pair. The possible ratings are from 1 to 6, with 0 not allowed. The final answer needs to be a JSON object where each key is the rating (as a string) and the value is the probability that an annotator gave that rating.

First, I need to understand what sarcasm means here. The definition provided says it's when someone says the opposite of what they mean, usually to criticize or hurt someone in a humorous way. So, I have to analyze the response to see if it uses sarcasm and how different people might rate it.

Looking at the context: Steve met a girl while drunk, fell in love, and wants to quit his job to travel with her. The response is from someone else reacting to Steve's decision. The response says, "Man, you are fucking serious right now? You're going to quit your job and travel with some girl you just met last night while drunk? No girl is worth quitting your job and screwing your life over, especially not a girl you just met last night."

Now, I need to check if this response is sarcastic. Sarcasm often involves irony or mock praise. Here, the speaker is clearly criticizing Steve's impulsive decision. They're questioning his seriousness and pointing out the impracticality. There's no obvious use of words that mean the opposite. Instead, it's straightforward criticism without humor. The tone is more disapproving than sarcastic.

So, most annotators might rate this as low sarcasm (like 1 or 2). Some might think there's a bit of sarcasm in the rhetorical question ("you are fucking serious"), but that's more shock than sarcasm. Others might see it as direct advice rather than sarcastic. Maybe a few could interpret the repeated emphasis on "just met" as mocking, but that's a stretch.

I should consider possible distributions. If the response isn't sarcastic, the majority would rate 1 (not at all). Some might give 2 for slight sarcasm. Very few might go higher if they think there's hidden mockery. Let's say 70% rate 1, 20% rate 2, and 10% split between 3 and maybe 4. But wait, the sum has to be 1.0. Let me adjust. Maybe 0.7 for 1, 0.2 for 2, 0.05 for 3, and 0.05 for 4. That adds up to 1.0.

Alternatively, perhaps even less for higher numbers. Another approach: since the response is direct and not using irony, most will rate 1. A small percentage might see some sarcasm in the phrasing. Maybe 0.8 for 1, 0.1 for 2, and 0.1 distributed among others. But ensuring the total is 1.0. Let's try:

```
- 1: 0.7
- 2: 0.2
- 3: 0.05
- 4: 0.03
- 5: 0.01
- 6: 0.01
```

That sums to 1.0. But maybe even lower for higher numbers. Alternatively, maybe only 1 and 2. Let's check again. The response is critical but not sarcastic. So most would rate 1. Some might rate 2 if they think there's a hint of sarcasm. Let's say 0.8 for 1, 0.15 for 2, and 0.05 for 3. That's 1.0. Or maybe 0.9 for 1, 0.05 for 2, 0.05 for 3. Hmm. It's tricky. To be safe, I'll go with a high concentration on 1, some on 2, and minimal elsewhere. So final distribution might look like {'1': 0.7, '2': 0.2, '3': 0.05, '4': 0.03, '5': 0.01, '6': 0.01}. That adds up to 1.0.

Listing 12: LLM prompt for the CSC dataset (soft-label task)

```
[Instructions]

Solve the task below. Format your response into logical steps that can be verified by a third-party, and
format each step in JSON format like this:

{{
  'steps': [
    '<text of step 1>',
    '<text of step 2>',
    ...,
    '<text of step n>'
  ],
  'final_response': '<response>'
}}

Please simplify your final response as much as possible, and avoid any unnecessary text there.

[Task]

Below is a context+response pair where human annotators rated the sarcasm level of the 'response' ranging
from 1 (not at all) to 6 (completely). Please guess the distribution of ratings and output it in the
final_response field in JSON format like this: {'0': p0, '1': p1, '2': p2, '3': p3, '4': p4, '5': p5, '6':
p6}, where each p0, ..., p6 is a probability. Note that p0 is always 0.0 because the rating 0 is not
allowed. The sum of probabilities must equal 1.0.

Use this definition for sarcasm: "The use of remarks that clearly mean the opposite of what they say, made
in order to hurt someone's feelings or to criticize something in a humorous way"

Think about the perspectives that different annotators might have and how they could potentially interpret
the context+response pair.

[Context+Response Pair]

{text}
```

Listing 13: LLM prompt for the CSC dataset (perspectivist task)

```
[Instructions]

Solve the task below. Format your response into logical steps that can be verified by a third-party, and
format each step in JSON format like this:

{{
  'steps': [
    '<text of step 1>',
    '<text of step 2>',
    ...,
    '<text of step n>'
  ],
  'final_response': '<response>'
}}

Please simplify your final response as much as possible, and avoid any unnecessary text there.

[Task]

Below is a context+response pair where human annotators rated the sarcasm level of the 'response' ranging
from 1 (not at all) to 6 (completely). Please guess the rating given by each annotator output them all in a
list in the final_response field, in the same order as the annotators.

Use this definition for sarcasm: "The use of remarks that clearly mean the opposite of what they say, made
in order to hurt someone's feelings or to criticize something in a humorous way"

Think about the perspectives that different annotators might have and how they could potentially interpret
the context+response pair.

[Annotator Metadata]

{annotator_metadata}

[Context+Response Pair]

{text}
```

Listing 14: Judge prompt for scoring reasoning steps (soft-label task)

```
[Chain-of-Thought Evaluation]

Overview: Your goal is to grade an LLM's step-by-step solution to a problem. The model will often say things that look ok at first, but will turn out to be wrong on closer inspection - stay vigilant! Please mark each step with (great, okay, bad).

Instructions: A "great" step is anything a smart student would try. Most of the time it's a clear cut step forward towards solving the problem. But it could also be a sub-optimal choice, as long as it looks like something a reasonably smart human might say while trying to solve the problem. An "okay" step is anything that's reasonable for a person to say, but it's not offering any insight, doesn't further the solution by exploring an option, performing a calculation, or offering an idea for the next step. A "bad" step is one that confidently says something incorrect, is off-topic/weird, leads the solution into a clear dead-end, or is not explained clearly enough for a human to follow along with (even if it is correct).

[Output Format]
Only output the step idx and the rating, like this example below.
[
  {"idx": 0, "rating": "great"},
  {"idx": 1, "rating": "ok"},
  {"idx": 2, "rating": "bad"},
  {"idx": 3, "rating": "great"},
  ...
]
Please verify that the number of steps is the same as in the input.

[LLM Problem]

<problem>
{PROBLEM}
</problem>

[LLM Chain-of-Thought Steps]

<steps>
{STEPS}
</steps>

<final_response>
{FINAL_RESPONSE}
</final_response>
```