BERT-as-a-Judge: A Robust Alternative to Lexical Methods for Efficient Reference-Based LLM Evaluation

Hippolyte Gisserot-Boukhlef[A,C], Nicolas Boizard[B,C], Emmanuel Malherbe[A], Céline Hudelot[C] and Pierre Colombo[C,D]

[A]Artefact Research Center
[B]Diabolocom
[C]MICS, CentraleSupélec, Université Paris-Saclay
[D]Cohere

Accurate evaluation is central to the large language model (LLM) ecosystem, guiding model selection and downstream adoption across diverse use cases. In practice, however, evaluating generative outputs typically relies on rigid lexical methods to extract and assess answers, which can conflate a model's true problem-solving ability with its compliance with predefined formatting guidelines. While recent LLM-as-a-Judge approaches mitigate this issue by assessing semantic correctness rather than strict structural conformity, they also introduce substantial computational overhead, making evaluation costly. In this work, we first systematically investigate the limitations of lexical evaluation through a large-scale empirical study spanning 36 models and 15 downstream tasks, demonstrating that such methods correlate poorly with human judgments. To address this limitation, we introduce BERT-as-a-Judge, an encoder-driven approach for assessing answer correctness in reference-based generative settings, robust to variations in output phrasing, and requiring only lightweight training on synthetically annotated question-candidate-reference triplets. We show that it consistently outperforms the lexical baseline while matching the performance of much larger LLM judges, providing a compelling tradeoff between the two and enabling reliable, scalable evaluation. Finally, through extensive experimentation, we provide detailed insights into BERT-as-a-Judge's performance to offer practical guidance for practitioners, and release all project artifacts to foster downstream adoption.

Keywords: *language model evaluation, reference-based metrics, encoder models*

1. Introduction

Evaluation lies at the core of the large language model (LLM) ecosystem. In recent years, considerable effort has been devoted to rigorously and fairly assessing model performance across a wide range of tasks, to guide model selection and downstream adoption [1, 2]. For instruction-tuned models (optimized for human interaction and question answering), evaluation is typically conducted in zero-shot generative settings [3, 4], in which models are prompted to directly generate an answer without access to task-specific examples.

While conceptually straightforward, this setup poses two challenges for evaluation: reliably extracting the model's predicted answer for comparison with a reference, and performing the comparison itself. The former arises from answer formatting variations, such as "The answer is X" versus "Answer: X", the latter occurs when comparing outputs like "2.00" versus "2$", both of which should be treated as equivalent. A common mitigation strategy is to enforce constrained output formats via prompting, enabling answers to be extracted with regular expressions (regex) [5, 6], and then rely on metrics beyond exact match, such as ROUGE [7], BERTScore [8], or Math-Verify [9], thereby avoiding errors caused by formatting inconsistencies or lexical variations. Although more flexible than strict exact

Published at the Conference on Language Modeling (COLM) 2026
Corresponding author(s): hippolyte.gisserot-boukhlef@centralesupelec.fr
Artefact Research Center, Paris, France

match, these metrics can still fail to accurately capture answer correctness, especially since models often do not strictly follow prescribed output formats, making reliable answer parsing difficult. Such deviations may stem from differences in model scale, instruction-tuning data mixtures, or alignment strategies, and can artificially deflate measured downstream performance. While formatting adherence is itself an important capability, particularly for instruction following and structured generation [4], it should not confound the evaluation of orthogonal competencies such as factual knowledge, mathematical reasoning, or reading comprehension [10, 11].

Recently, LLM-as-a-Judge frameworks have emerged as a compelling alternative [12, 13]. By delegating answer comparison to a separate language model, these approaches reduce dependence on rigid formatting constraints and can correctly credit semantically valid but structurally unconventional responses. However, they introduce substantial computational overhead and additional sources of variance, including sensitivity to the choice of judge model and prompt design [14].

Question. How can we measure a model's core problem-solving ability without relying on output formatting or expensive inference?

Contributions. In this work, we make the following three contributions:

- Through a comprehensive empirical study across a diverse set of models and tasks, we show that lexical evaluation exhibits weak correlation with human judgments (section 3).
- To address this limitation, we introduce BERT-as-a-Judge, an encoder-driven approach for evaluating generative models in reference-based settings, leveraging the strength of bidirectional attention for text classification (Figure 1). We show that BERT-as-a-Judge consistently outperforms lexical evaluation and even surpasses LLM-as-a-Judge under comparable inference conditions (section 4).
- We provide detailed insights into BERT-as-a-Judge's performance through an extensive set of experiments, offering practical guidance for downstream applications (section 5). Additionally, we release the packaged code[1] along with the full set of generated and annotated data, covering outputs from 36 models across 15 tasks, and open-source all post fine-tuning checkpoints used in our experiments.[2]

2. Experimental Protocol

2.1. Answer Generation

Tasks. The backbone of LLM evaluation consists of tasks whose outputs can be unambiguously judged as correct or incorrect, providing an objective basis for model assessment [15, 16, 17, 18, 19]. In this work, we focus on three families of widely used benchmarks:

- *Multiple-choice*, in which models are given a question along with a set of options: MMLU [10], MMLU-Pro [20], TruthfulQA [21], ARC-Easy/Challenge [22], and GPQA [23].
- *Context extraction*, where models must provide answers grounded in a given passage by citing relevant evidence: SQuAD-v2 [24], HotpotQA [25], DROP [26], and CoQA [27].
- *Open-form mathematics*, in which models generate a final closed-form answer in free text: GSM8K [11], MATH [28], AsDiv [29], AIME 24 [30], and AIME 25 [31].

[1] https://github.com/artefactory/BERT-as-a-Judge
[2] https://huggingface.co/collections/artefactory/bert-as-a-judge

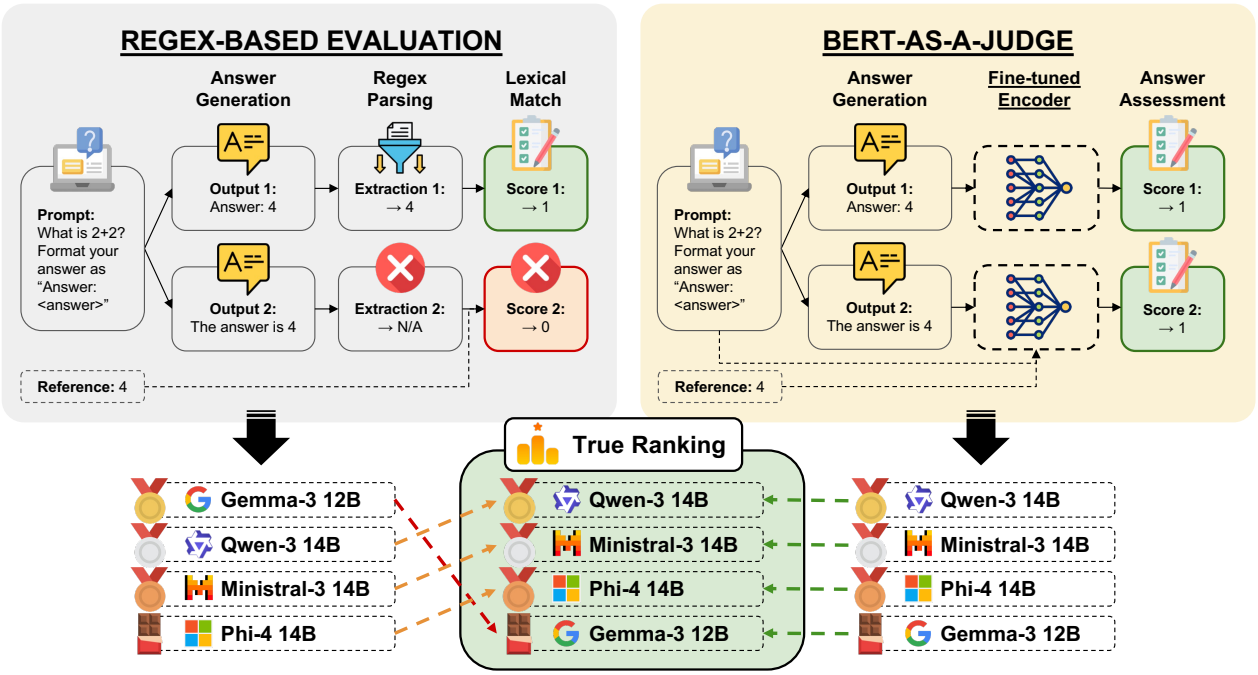


Figure 1 | Comparison between regex-based (lexical) evaluation and BERT-as-a-Judge. Top: illustration of both approaches with simple examples. Bottom: model rankings for four similarly sized models from different families, computed via task-wise Borda count.

Models. We perform inference across a broad range of recent open-weight instruction-tuned model families, spanning from 135M to 70B parameters. Our study includes 36 models in total: Llama-3 (1B, 3B, 8B, 70B) [15], Qwen-3 (600M, 4B, 8B, 14B, 32B) [16], Gemma-3 (1B, 4B, 12B, 27B) [32], Falcon-3 (1B, 3B, 7B) [33], Phi-4 (3.8B, 14B) [34, 35], SmolLM-2 and 3 (135M, 360M, 1.7B, 3B) [36, 37], OLMo-3 (7B, 32B) [17], Ministral-3 (3B, 8B, 14B) [38], LFM-2 (350M, 700M, 1.2B, 2.6B) [38], EuroLLM (1.7B, 9B, 22B) [39, 40, 18], and Apertus (8B, 70B) [19].

Generation parameters. For each task-model pair, responses are produced in a zero-shot setting using greedy decoding, with a maximum generation length of 2048 tokens. For experimental purposes, models are prompted to conclude their outputs in the format "Final answer: [answer]" to facilitate downstream regex parsing and ensure fair comparison between model- and regex-based assessment methods.

2.2. Labeling

Synthetic labeling. For annotation, we employ Nemotron-Super-v1.5 [41] as an automatic evaluator. The model is provided with the question, the candidate answer, and the reference answer, and is asked to determine whether the candidate response is correct given the available information.[3] Inference is conducted using greedy decoding in non-reasoning mode.

Human labeling. To validate the reliability of the synthetic labeling approach, we perform human annotation on a subset of the data. Specifically, we randomly sample instances from the generated

[3]The full evaluation prompt is provided in Appendix A.

dataset and have them independently labeled by a pool of 11 human evaluators, totaling 3,212 annotations, and resulting in an overall average agreement of 97.5% with the synthetic labels.[4]

2.3. Evaluation Methods

BERT-as-a-Judge. We propose to train a BERT-like encoder model on labeled question-candidate-reference triplets constructed as described in subsection 2.1 and subsection 2.2, leveraging its bidirectional attention mechanism well suited for structured text classification [42]. We construct the training mixture from the tasks described in subsection 2.1 that provide an explicit training split, namely MMLU, ARC-Easy, ARC-Challenge, SQuAD-v2, HotpotQA, GSM8K, and Math. The training dataset is constructed to balance the number of samples across task categories and models, resulting in approximately 1M synthetically labeled samples in total. We initialize the encoder from EuroBERT 210M [43] and fine-tune it for one epoch using binary cross-entropy. We employ a learning rate of 2×10^{-5}, following the authors' recommendations for sequence classification, along with a 5% warmup ratio and a linear decay schedule. Training is conducted on 8 MI250x GPUs, yielding an effective batch size of 32, taking approximately 20 GPU hours per run.

Baselines. We compare BERT-as-a-Judge to the following baselines:

- *Regex:* Extracts answers using a regular expression based on the pattern "Final answer: [answer]" and evaluates multiple-choice tasks with exact match, context extraction with ROUGE-L [7], and open-form math with Math-Verify [9]. For answer parsing, we build on the regex rules provided by the lm-evaluation-harness framework [6], adapting them to our prompting format and the range of evaluated models.
- *LLM-as-a-Judge:* Uses a generative model to determine whether a candidate response matches the reference answer for a given question, following the procedure described in subsection 2.1. To keep inference costs comparable to the encoder, we use a model of similar scale by default (Qwen-3 0.6B), prompting it to respond directly with "True" or "False". Larger LLM judges and more flexible prompting strategies are also evaluated in section 5.

2.4. Assessment of Evaluation Quality

Metric. We assess each method by its accuracy against synthetic labels from Nemotron-Super-v1.5,[5] reflecting how well it predicts whether a given answer is correct or incorrect.[6]

Benchmarks. We assess all evaluation methods on the full set of tasks introduced in subsection 2.1, including both the test splits of tasks used during encoder training (subsection 2.3) and the tasks reserved exclusively for out-of-domain evaluation.

3. Limitations of Regex-Based Evaluation

This section analyzes the impact of regex-based evaluation on measured downstream performance, noting discrepancies from both formatting-related parsing failures and post-parsing matching er-

[4] Further details are provided in Appendix C.

[5] For methods producing scores between 0 and 1 (e.g., encoder models with soft probabilities), we use a default threshold of 0.5.

[6] To estimate performance with respect to human annotations, we apply a correction based on the observed agreement between human and synthetic labels (subsection 2.2); results are reported in Appendix C.

rors. Specifically, we quantify parsing failure rates across a range of models in Figure 2 and assess performance deltas relative to ground-truth labels in Table 1.

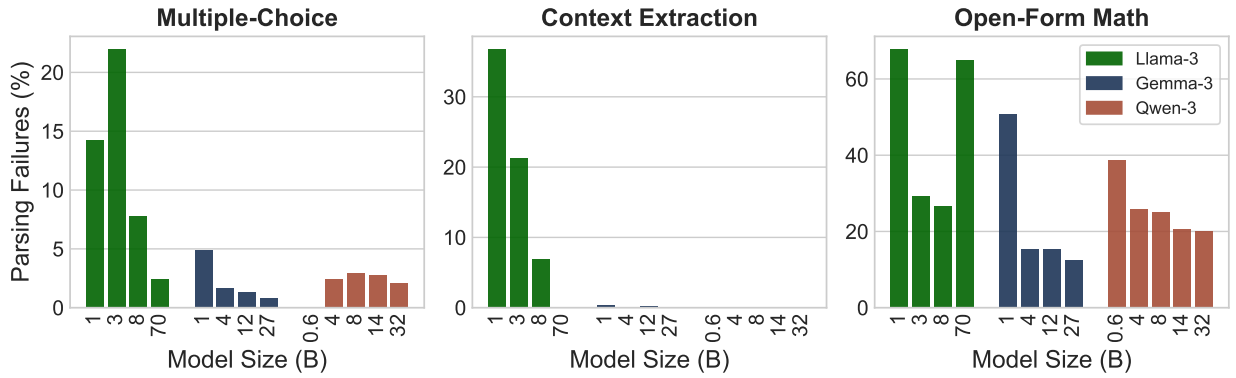


Figure 2 | Quantification of regex parsing failures. Values represent the failure rate, defined as the percentage of instances with unparsable outputs. Results are shown for the Llama-3, Gemma-3, and Qwen-3 model families and aggregated by task category.

Model scale, family, and task type have a high impact on output formatting. Figure 2 shows that larger models tend to produce fewer formatting errors, as illustrated by the Llama-3 models on context extraction and Qwen-3 on open-form math tasks. Model family also plays a significant role: Qwen-3 and Gemma-3 consistently achieve near-perfect formatting compliance on context extraction, whereas smaller Llama-3 models exhibit substantially higher failure rates. Task type further impacts formatting accuracy. Open-form math proves the most challenging, with Llama-3 70B generating incorrectly formatted outputs over 60% of the time and Qwen-3 32B around 20%, while multiple-choice and context extraction tasks are much easier, with mid- to large-scale models often achieving near-zero failure rates.

Regex-based evaluation distorts performance measurements. Table 1 illustrates the risks of relying on regex-based evaluation, showing substantial negative deltas in measured performance across a broad set of models. Notably, even models with high formatting compliance (e.g., Gemma-3 family on context extraction tasks) suffer from substantial underestimation, often due to overly verbose outputs that technically follow formatting rules but fail lexical matching.[7] Overall, regex-based assessment significantly distorts leaderboard rankings; for instance, Qwen-3 32B drops 18 positions while Gemma-3 4B climbs 6 on multiple-choice tasks, with these shifts often representing mere artifacts of formatting quirks and rigid lexical matching rather than true differences in capability.

4. Encoder-Based Evaluation

In this section, motivated by the observation that regex-based evaluation fails to accurately reflect true model performance across a wide range of benchmarks (section 3), we train a BERT-as-a-Judge encoder model to assess answer correctness following the methodology described in subsection 2.3. We then assess the proposed approach using the setup detailed in subsection 2.4 and report the results in Table 2. To further support our analysis, Table 3 shows performance for models whose outputs were excluded from the training mixture.

[7] More details are provided in Appendix E.

Family	Size	Multiple-Choice		Context Extraction		Open-Form Math	
		ΔAccuracy	ΔRank	ΔAccuracy	ΔRank	ΔAccuracy	ΔRank
Llama-3	1B	-0.9 (-2.6/+1.7)	↑4.8	-23.9 (-18.1/-5.8)	↓1.2	-11.9 (-11.1/-0.8)	↓0.8
	3B	-13.1 (-8.5/-4.6)	↓0.5	-27.5 (-15.6/-11.9)	↓0.4	-7.0 (-2.6/-4.4)	↑1.7
	8B	-23.3 (-0.9/-22.3)	↓6.4	-21.0 (-4.0/-17.0)	↑3.5	-5.4 (-0.8/-4.6)	↑4.2
	70B	-1.1 (-1.1/-0.1)	↑1.3	-18.4 (-0.1/-18.3)	↑6.5	-30.4 (-29.0/-1.4)	↓13.3
Gemma-3	1B	-0.1 (-0.7/+0.5)	↑5.7	-12.3 (-0.1/-12.2)	↑4.6	-11.3 (-6.8/-4.5)	↓1.2
	4B	+0.3 (-0.2/+0.5)	↑6.2	-30.8 (-0.0/-30.8)	↓1.5	-11.8 (-1.2/-10.6)	↓0.1
	12B	-0.1 (-0.2/+0.1)	↑5.3	-27.6 (-0.1/-27.4)	↓4.2	-10.3 (-1.4/-8.9)	↑1.6
	27B	+0.1 (-0.1/+0.2)	↑4.7	-28.7 (-0.0/-28.7)	↓2.1	-10.8 (-1.1/-9.7)	↑0.4
Qwen-3	0.6B	-20.8 (-0.0/-20.8)	↓2.8	-13.8 (-0.0/-13.8)	↑2.8	-10.8 (-5.8/-5.0)	↓0.5
	4B	-3.2 (-0.5/-2.7)	↑3.5	-20.2 (-0.0/-20.2)	↑7.0	-12.0 (-5.3/-6.7)	↑1.0
	8B	-7.0 (-0.6/-6.3)	↑0.7	-29.7 (-0.0/-29.7)	↓7.2	-15.2 (-7.9/-7.3)	↓0.2
	14B	-19.3 (-0.7/-18.6)	↓13.2	-20.9 (-0.0/-20.9)	↑0.5	-15.0 (-7.8/-7.2)	↑0.8
	32B	-23.8 (-0.5/-23.3)	↓17.6	-25.1 (-0.0/-25.1)	↓1.1	-10.3 (-2.6/-7.7)	↑2.8

Table 1 | Impact of regex-based evaluation on measured model performance. The "ΔAccuracy" columns report the difference between regex-based and ground-truth (synthetic-label) accuracy, with values in parentheses indicating the contributions from parsing failures and post-parsing matching, respectively. The "ΔRank" columns show the corresponding average change in ranking across the 36 evaluated models, where greener values indicate upward rank changes and redder values indicate downward rank changes.

BERT-as-a-Judge shows the strongest alignment with human judgments. As shown in Table 2, our trained encoder achieves the highest accuracy against ground-truth labels across all benchmarks. This advantage holds across task types, reaching near-perfect alignment on multiple-choice datasets (e.g., 99.7% on ARC-Easy and 98.5% on MMLU) and remaining high on complex-output tasks (98.8% on GSM8K and 93.9% on MATH). It also consistently outperforms the regex-based method by substantial margins (e.g., +21.1% on CoQA, +20.3% on MATH, +10.4% on ARC-Challenge), demonstrating that a dedicated encoder more reliably captures answer correctness than rigid lexical heuristics, while remaining computationally efficient (≈200 ms per sample on an Apple M1 CPU).

BERT-as-a-Judge is robust to out-of-domain tasks. Beyond its strong overall accuracy, the encoder-based method maintains high accuracy even on tasks excluded from the training mixture (e.g., 98.6% on TruthfulQA, 88.1% on CoQA, and 95.3% on ASDiv), highlighting its strong generalization ability across the three task categories considered (Table 2).

BERT-as-a-Judge generalizes to unseen models. Table 3 shows that removing generations from specific models in the training mixture has minimal impact on downstream assessment quality for those excluded instances. This demonstrates the strong generalization ability of our approach, indicating it can be safely extended to additional model families outside the training mixture and supporting broader adoption.

Typical encoder scales appear insufficient for LLM-as-a-Judge. Even at three times the size of our encoder, the baseline LLM judge (Qwen-3 0.6B) consistently produces the weakest results, substantially underperforming the regex baseline across all task categories (Table 2). For instance, on ARC-Challenge, it achieves only 50.2% accuracy versus 89.0% for regex, with similar gaps on

Task	Regex	LLM-Judge	BERT-Judge
Multiple-Choice			
ARC-Challenge	89.0	50.2	**99.4**
ARC-Easy	88.2	54.0	**99.7**
MMLU	88.1	50.3	**98.5**
GPQA	86.5	66.2	**93.5**
MMLU-Pro	88.8	57.1	**96.5**
TruthfulQA	92.5	54.5	**98.6**
Context Extraction			
HotpotQA	75.6	70.0	**90.9**
SQuAD-v2	72.3	62.5	**89.3**
CoQA	67.0	75.2	**88.1**
DROP	77.0	69.3	**88.6**
Open-Form Math			
GSM8K	94.4	71.3	**98.8**
Math	73.4	58.9	**93.7**
AIME24	87.8	77.9	**90.0**
AIME25	**91.8**	83.0	91.4
ASDiv	89.2	75.5	**95.3**

Table 2 | Accuracies of evaluation methods against ground-truth labels across tasks, averaged over models. Dashed lines separate test-only tasks from those with a training split. Bold indicates the highest accuracy per task.

Size	Multiple-Choice		Context Extraction		Open-Form Math	
	ID	OOD	ID	OOD	ID	OOD
Ministral-3						
3B	97.0	96.9	83.7	82.5	81.4	86.5
8B	97.9	97.8	87.9	87.1	83.5	84.8
14B	98.2	98.3	89.1	88.6	83.5	82.6
LFM-2						
0.35B	94.8	94.1	90.5	88.6	97.1	96.9
0.7B	96.8	96.7	87.4	84.4	97.4	97.0
1.2B	97.5	97.1	91.2	90.9	94.6	94.4
2.6B	97.9	97.8	87.1	86.0	93.9	93.7
EuroLLM						
1.7B	93.4	93.1	91.5	91.2	98.5	98.4
9B	98.6	98.5	90.7	90.2	94.5	94.1
22B	98.2	98.1	90.6	90.6	91.5	91.1
Apertus						
8B	97.6	97.4	90.3	89.9	97.5	97.4
70B	98.1	98.0	89.5	89.5	97.2	97.1

Table 3 | Assessment accuracy on out-of-domain models. "ID" denotes training on all model outputs, while "OOD" excludes specific models from the training mixture. Results are aggregated by task category.

context extraction tasks (62.5% vs. 72.3% on SQuAD-v2). These results suggest that the evaluation capabilities of LLM-as-a-Judge do not hold for generative models under 1B parameters, highlighting the critical role of scale for such methods. We examine this limitation further in section 5.

5. Experimental Analysis

In this section, we further investigate the properties of our encoder-based evaluation method through a series of complementary analyses.

BERT-as-a-Judge consistently outperforms LLM-as-a-Judge across a wide range of inference budgets. As shown in Table 2, generative evaluation performs poorly at small scales (0.6B parameters). To complement these findings, we conduct a more extensive comparison by varying the judge family (Qwen-3, Gemma-3), model size (0.6B to 32B for Qwen-3 and 1B to 27B for Gemma-3), and inference budget (allowing or not intermediate chain-of-thought tokens before producing the final assessment). Figure 3 shows that our encoder matches the performance of the top-performing LLM judges in the defined setup, while remaining drastically less computationally expensive in terms of inference FLOPs.[8]

BERT-as-a-Judge is training-efficient. By default, we train encoder models on 1M question-candidate-reference triplets. In this experiment, we evaluate lighter configurations: 500K,

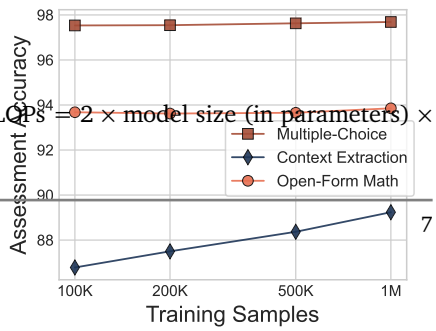


Figure 4 | BERT-as-a-Judge evalua-

[8]Inference FLOPs are estimated using the formula from **(author?)** [44]: FLOPs = 2 × model size (in parameters) × number of generated tokens.

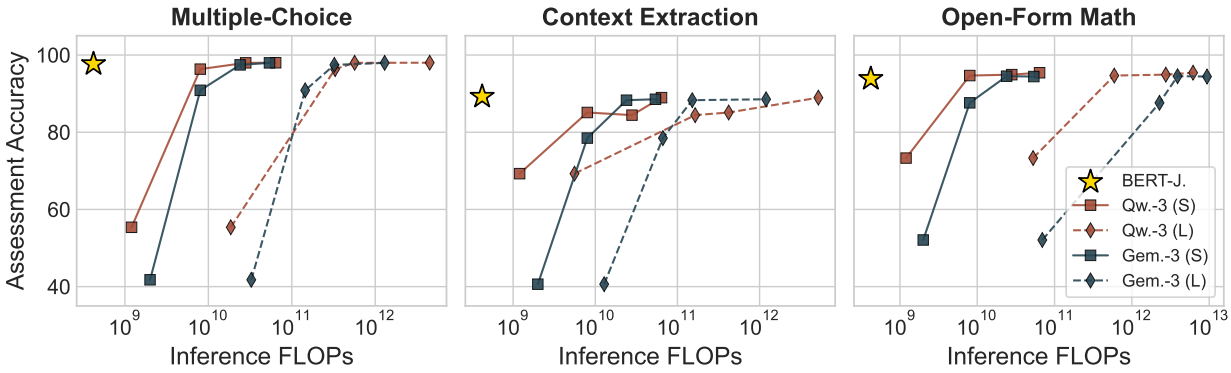


Figure 3 | Comparison between encoder-based evaluation and LLM judges from the Qwen-3 and Gemma-3 families across different model sizes and inference budgets. "S" ("short") denotes the default generation setup, in which the model answers directly with "True" or "False", while "L" ("long") allows the generation of intermediate chain-of-thought tokens before the final judgment.

200K, and 100K samples (Figure 4). Remarkably, 100K training samples are sufficient to accurately evaluate multiple-choice and open-form math tasks, with no significant improvement observed beyond this point. Gains are more noticeable for context extraction, as expected, since this task category requires more than simple candidate-reference matching and often demands understanding of the context provided by the question. Overall, with just 2 GPU hours of training (corresponding to the 100K-sample configuration), our encoder achieves a high assessment accuracy, making it well-suited for settings with limited data and computational resources.

Task Category	Regex	BERT-J.	Regex+ BERT-J.
Multiple-Choice	88.8	**97.7**	90.5
Context Extraction	73.0	**89.2**	75.2
Open-Form Math	87.3	**93.9**	89.9

Table 4 | Comparison of hybrid answer evaluation (Regex+BERT-J.) with standalone regex and BERT-as-a-Judge. Bold values indicate the highest accuracy in each row.

Task Category	Regex	BERT-Judge	
		w/ Q.	w/o Q.
Multiple-Choice	88.8	**97.7**	97.3
Context Extraction	73.0	**89.2**	84.2
Open-Form Math	87.3	**93.9**	**93.9**

Table 5 | Impact of including (w/ Q.) or excluding (w/o Q.) the question in the encoder training prompt. Bold values denote the highest accuracy for each task category.

Combining BERT-as-a-Judge with regex offers an efficient compromise. In this experiment, we use BERT-as-a-Judge as a fallback when regex parsing fails. While it does not reach the performance of the standalone encoder, this hybrid approach substantially improves over regex alone. The results demonstrate that selectively applying the encoder can recover a significant portion of assessment accuracy while keeping computational overhead low (for example, reducing total compute by a factor of five for a model with 20% regex failures).

Removing the question from the prompt yields a controlled performance decrease. As shown in Table 5, omitting the question during encoder training (leaving only the candidate and reference) reduces overall assessment accuracy by removing some contextual information. However, this decrease is well-controlled across tasks. The question-free encoder still outperforms regex and remains close to the full-prompt setup, particularly on multiple-choice and open-form math tasks, while also reducing runtime due to shorter prompts and enabling application to any task with fully textual outputs, including multimodal tasks. The performance gap is slightly larger for context extraction, where the question provides critical information, underscoring the encoder's context-aware capabilities.

Task	Test Set (Form.)			Test Set (Free)		
	Regex	BERT-J. (Form.)	BERT-J. (Free)	Regex	BERT-J. (Form.)	BERT-J. (Free)
Multiple-Choice	88.8	97.7	97.4	–	94.0	97.6
Context Extraction	73.0	89.2	85.8	–	84.3	91.6
Open-Form Math	87.3	93.9	93.7	–	93.1	93.5

Table 6 | Encoder's robustness to answer formatting. "Test Set (Form.)" denotes test sets with formatted answers, while "Test Set (Free)" contains unformatted answers. "Regex" columns show regex-based results, and "BERT-J. (Form.)" and "BERT-J. (Free)" report accuracies for BERT-as-a-Judge encoders trained on formatted and unformatted answers, respectively. Regex results are omitted for the free-format test set, as answers cannot be reliably parsed.

BERT-as-a-Judge is robust to variations in answer formatting guidelines. In our core experiments, and to ensure a fair comparison with the regex baseline, we train and evaluate the encoder on answers formatted to facilitate lexical parsing (subsection 2.1). In practice, however, users may follow custom guidelines or allow free-form responses.[9] Table 6 shows that under cross-formatting evaluation, meaning free-to-formatted (training on free-form, evaluating on formatted answers) and formatted-to-free (the reverse), we observe a slight performance drop compared to aligned settings. Nevertheless, the encoder still substantially outperforms regex, demonstrating strong robustness to variations in answer formatting. As expected, the free-to-formatted encoder consistently outperforms the formatted-to-free variant, benefiting from exposure to a wider range of formats during training and making it the preferred choice for downstream applications.

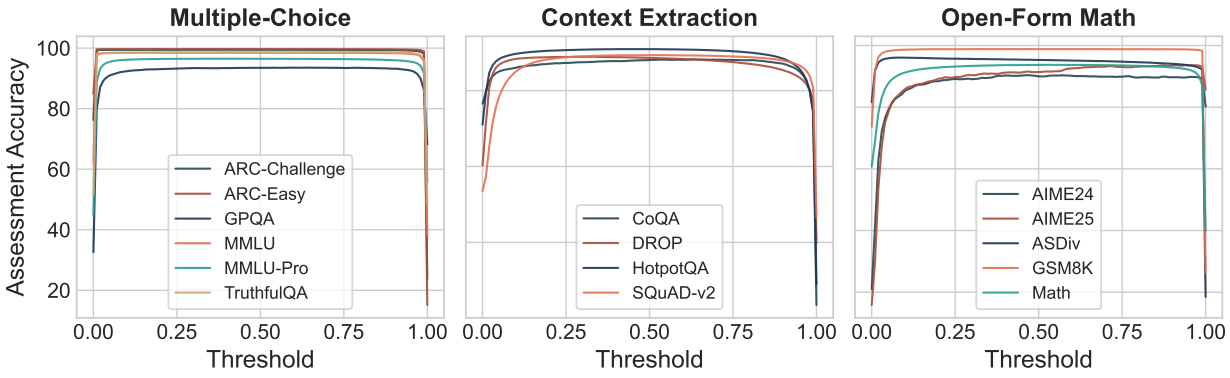


Figure 5 | Effect of score thresholding on BERT-as-a-Judge downstream assessment accuracy across the three task categories, averaged over all models.

[9]Further details are provided in Appendix A.

BERT-as-a-Judge is robust to decision threshold variations. Encoder classifiers output continuous sigmoid probabilities, requiring a decision threshold for discrete evaluation. While our main experiments (section 4) use a standard 0.5 threshold, Figure 5 demonstrates that accuracies remain remarkably stable across a broad spectrum of threshold values for all task categories. This invariance indicates strong separation between classes, enabling reliable off-the-shelf deployment without the need for task-specific threshold tuning.

6. Related Work

Traditional LLM evaluation. Pretrained language models have traditionally been evaluated using log-likelihood [45] or few-shot generation [46, 47, 48, 49, 50]. With the rise of instruction-tuned models, zero-shot generative evaluation has become standard [3, 51, 16, 18]. This paradigm typically enforces structured outputs via prompting [5, 6], followed by rule-based comparison to references using deterministic metrics such as exact match, ROUGE [7], Math-Verify [9], or Code-Eval [52], making evaluation highly sensitive to surface-level formatting.

Model-based evaluation. Both lexical parsing and matching introduce limitations in capturing semantic correctness and robustness. Lexical overlap does not guarantee semantic equivalence, motivating neural metrics such as BERTScore [8] and InfoLM [53] for general text generation, as well as task-specific evaluators like COMET [54, 55, 56], MetricX [57, 58], and BLEURT [59]. Additionally, reliably extracting model outputs is challenging when formatting is inconsistent [60, 61]. To mitigate these issues, LLM-as-a-Judge approaches [12, 13, 62, 63, 64] directly assess candidate-reference equivalence across tasks, offering greater robustness to formatting artifacts, albeit at a substantial computational cost.

7. Conclusion

In this work, we show that standard evaluation protocols often conflate a model's underlying problem-solving ability with its compliance to formatting constraints. Across extensive experiments spanning diverse models and tasks, we demonstrate that regex-based evaluation can substantially underestimate true performance. To address this, we propose BERT-as-a-Judge, a lightweight encoder-based framework that better captures semantic correctness, aligns more closely with human judgment, and avoids the high computational cost of LLM-as-a-Judge methods, enabling efficient and more reliable evaluation.

8. Limitations and Future Work

While BERT-as-a-Judge shows strong alignment with human judgments and effectively mitigates the limitations of lexical assessment, our study focuses on a specific subset of evaluation settings, namely, English benchmarks with objectively verifiable answers, where correctness can be clearly defined.

Building on these results and the existing literature, a natural next step is to broaden the scope of encoder-based evaluation toward more general-purpose settings. This includes expanding beyond fact-based and structured tasks to open-ended generation scenarios such as summarization, machine translation, code generation, and instruction following. Additionally, adapting the framework to multilingual contexts would further improve its applicability across diverse use cases. As foundation models continue to evolve toward multimodal capabilities, extending this approach to handle vision and speech inputs also presents a promising avenue. Exploring such cross-modal evaluation settings

(e.g., visual question answering, image captioning, or speech-based reasoning) could help move toward a unified and efficient evaluation paradigm applicable across tasks and modalities.

Ethics Statement

In conducting this research, we recognize the critical importance of fair and reliable evaluation in the LLM ecosystem. Evaluation metrics that are closely aligned with human judgments are essential to ensure that model comparisons accurately reflect real-world capabilities across the widest possible range of tasks. At the same time, the increasing scale of model evaluation, driven by more models, longer outputs, and a growing number of benchmark tasks, can impose substantial computational costs, raising concerns about accessibility and environmental impact. Our work emphasizes the development of lightweight, encoder-based evaluation methods that maintain high correlation with human judgments while minimizing compute requirements. By prioritizing both fairness and efficiency, we aim to support responsible, reproducible, and scalable evaluation practices in the broader LLM research community.

Acknowledgments

We sincerely thank the 11 human annotators whose careful work was essential in validating the correlation between our synthetic labeling strategy and human judgments. Their contributions ensured the reliability and rigor of our evaluation results.

References

[1] Percy Liang, Rishi Bommasani, Tony Lee, et al. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*, 2022.

[2] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[3] Jason Wei, Maarten Bosma, Vincent Zhao, et al. Finetuned language models are zero-shot learners. *International Conference on Learning Representations*, 2022.

[4] Long Ouyang, Jeffrey Wu, Xu Jiang, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 2022.

[5] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Alexander Cosgrove, Christopher D Manning, Christopher Re, Diana Acosta-Navas, Drew Arad Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue WANG, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri S. Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Andrew Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. Holistic evaluation of language models. *Transactions on Machine Learning Research*, 2023. Featured Certification, Expert Certification.

[6] Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024.

[7] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[8] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

[9] Hugging Face. Introducing the math verify leaderboard, 2024. Accessed: 2026-03-05.

[10] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[11] 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. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[12] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023.

[13] Xuechen Wang et al. Judgelm: Fine-tuned large language models are scalable judges. *arXiv preprint arXiv:2310.17631*, 2023.

[14] Nicolas Boizard, Hippolyte Gisserot-Boukhlef, Kevin El-Haddad, Céline Hudelot, and Pierre Colombo. When does reasoning matter? a controlled study of reasoning's contribution to model performance. *arXiv preprint arXiv:2509.22193*, 2025.

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

[16] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[17] Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025.

[18] Miguel Moura Ramos, Duarte M Alves, Hippolyte Gisserot-Boukhlef, João Alves, Pedro Henrique Martins, Patrick Fernandes, José Pombal, Nuno M Guerreiro, Ricardo Rei, Nicolas Boizard, et al. Eurollm-22b: Technical report. *arXiv preprint arXiv:2602.05879*, 2026.

[19] Project Apertus, Alejandro Hernández-Cano, Alexander Hägele, Allen Hao Huang, Angelika Romanou, Antoni-Joan Solergibert, Barna Pasztor, Bettina Messmer, Dhia Garbaya, Eduard Frank Ďurech, et al. Apertus: Democratizing open and compliant llms for global language environments. *arXiv preprint arXiv:2509.14233*, 2025.

[20] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290, 2024.

[21] Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods, 2021.

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

[23] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First conference on language modeling*, 2024.

[24] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[25] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[26] Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proc. of NAACL*, 2019.

[27] Siva Reddy, Danqi Chen, and Christopher D. Manning. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266, 2019.

[28] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

[29] Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. A diverse corpus for evaluating and developing english math word problem solvers. In *Proceedings of the 58th annual meeting of the Association for Computational Linguistics*, pages 975–984, 2020.

[30] Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2024, 2024.

[31] Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2025, 2025.

[32] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, Gaël Liu, Francesco Visin, Kathleen Kenealy, Lucas Beyer, Xiaohai Zhai, Anton Tsitsulin, Robert Busa-Fekete, Alex Feng, Noveen Sachdeva, Benjamin Coleman, Yi Gao, Basil Mustafa, Iain Barr, Emilio Parisotto, David Tian, Matan Eyal, Colin Cherry, Jan-Thorsten Peter, Danila Sinopalnikov, Surya Bhupatiraju, Rishabh Agarwal, Mehran Kazemi, Dan Malkin, Ravin Kumar, David Vilar, Idan Brusilovsky, Jiaming Luo, Andreas Steiner, Abe Friesen, Abhanshu Sharma, Abheesht Sharma, Adi Mayrav Gilady, Adrian Goedeckemeyer, Alaa Saade, Alex Feng, Alexander Kolesnikov, Alexei Bendebury, Alvin Abdagic, Amit Vadi, András György, André Susano Pinto, Anil Das, Ankur Bapna, Antoine Miech, Antoine Yang, Antonia Paterson, Ashish Shenoy, Ayan Chakrabarti, Bilal Piot, Bo Wu, Bobak Shahriari, Bryce Petrini, Charlie Chen, Charline Le Lan, Christopher A. Choquette-Choo, CJ Carey, Cormac Brick, Daniel Deutsch, Danielle Eisenbud, Dee Cattle, Derek Cheng, Dimitris Paparas, Divyashree Shivakumar Sreepathihalli, Doug Reid, Dustin Tran, Dustin Zelle, Eric Noland, Erwin Huizenga, Eugene Kharitonov, Frederick Liu, Gagik Amirkhanyan, Glenn Cameron, Hadi Hashemi, Hanna Klimczak-Plucińska, Harman Singh, Harsh Mehta, Harshal Tushar Lehri, Hussein Hazimeh, Ian Ballantyne, Idan Szpektor, Ivan Nardini, Jean Pouget-Abadie, Jetha Chan, Joe Stanton, John Wieting, Jonathan Lai, Jordi Orbay, Joseph Fernandez, Josh Newlan, Ju yeong Ji, Jyotinder Singh, Kat Black, Kathy Yu, Kevin Hui, Kiran Vodrahalli, Klaus Greff, Linhai Qiu, Marcella Valentine, Marina Coelho, Marvin Ritter, Matt Hoffman, Matthew Watson, Mayank Chaturvedi, Michael Moynihan, Min Ma, Nabila Babar, Natasha Noy, Nathan Byrd, Nick Roy, Nikola Momchev, Nilay Chauhan, Noveen Sachdeva, Oskar Bunyan, Pankil Botarda, Paul Caron, Paul Kishan Rubenstein, Phil Culliton, Philipp Schmid, Pier Giuseppe Sessa, Pingmei Xu, Piotr Stanczyk, Pouya Tafti, Rakesh Shivanna, Renjie Wu, Renke Pan, Reza Rokni, Rob Willoughby, Rohith Vallu, Ryan Mullins, Sammy Jerome, Sara Smoot, Sertan Girgin, Shariq Iqbal, Shashir Reddy, Shruti Sheth, Siim Põder, Sijal Bhatnagar, Sindhu Raghuram Panyam, Sivan Eiger, Susan Zhang, Tianqi Liu, Trevor Yacovone, Tyler Liechty, Uday Kalra, Utku Evci, Vedant Misra, Vincent Roseberry, Vlad Feinberg, Vlad Kolesnikov, Woohyun Han, Woosuk Kwon, Xi Chen, Yinlam Chow, Yuvein Zhu, Zichuan Wei, Zoltan Egyed, Victor Cotruta, Minh Giang, Phoebe Kirk, Anand Rao, Kat Black, Nabila Babar, Jessica Lo, Erica Moreira, Luiz Gustavo Martins, Omar Sanseviero, Lucas Gonzalez, Zach Gleicher, Tris Warkentin, Vahab Mirrokni, Evan Senter, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, Yossi Matias, D. Sculley, Slav Petrov, Noah Fiedel, Noam Shazeer, Oriol Vinyals, Jeff

Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Jean-Baptiste Alayrac, Rohan Anil, Dmitry, Lepikhin, Sebastian Borgeaud, Olivier Bachem, Armand Joulin, Alek Andreev, Cassidy Hardin, Robert Dadashi, and Léonard Hussenot. Gemma 3 technical report, 2025.

[33] TII Team. The falcon 3 family of open models, December 2024.

[34] Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

[35] Abdelrahman Abouelenin, Atabak Ashfaq, Adam Atkinson, Hany Awadalla, Nguyen Bach, Jianmin Bao, Alon Benhaim, Martin Cai, Vishrav Chaudhary, Congcong Chen, et al. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras. *arXiv preprint arXiv:2503.01743*, 2025.

[36] Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf. Smollm2: When smol goes big – data-centric training of a small language model, 2025.

[37] Elie Bakouch, Loubna Ben Allal, Anton Lozhkov, Nouamane Tazi, Lewis Tunstall, Carlos Miguel Patiño, Edward Beeching, Aymeric Roucher, Aksel Joonas Reedi, Quentin Gallouédec, Kashif Rasul, Nathan Habib, Clémentine Fourrier, Hynek Kydlicek, Guilherme Penedo, Hugo Larcher, Mathieu Morlon, Vaibhav Srivastav, Joshua Lochner, Xuan-Son Nguyen, Colin Raffel, Leandro von Werra, and Thomas Wolf. SmolLM3: smol, multilingual, long-context reasoner, 2025.

[38] Alexander H Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, et al. Ministral 3. *arXiv preprint arXiv:2601.08584*, 2026.

[39] Pedro Henrique Martins, Patrick Fernandes, João Alves, Nuno M Guerreiro, Ricardo Rei, Duarte M Alves, José Pombal, Amin Farajian, Manuel Faysse, Mateusz Klimaszewski, et al. Eurollm: Multilingual language models for europe. *Procedia Computer Science*, 255:53–62, 2025.

[40] Pedro Henrique Martins, João Alves, Patrick Fernandes, Nuno M Guerreiro, Ricardo Rei, Amin Farajian, Mateusz Klimaszewski, Duarte M Alves, José Pombal, Nicolas Boizard, et al. Eurollm-9b: Technical report. *arXiv preprint arXiv:2506.04079*, 2025.

[41] Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, Ido Shahaf, Oren Tropp, Ehud Karpas, Ran Zilberstein, Jiaqi Zeng, Soumye Singhal, Alexander Bukharin, Yian Zhang, Tugrul Konuk, Gerald Shen, Ameya Sunil Mahabaleshwarkar, Bilal Kartal, Yoshi Suhara, Olivier Delalleau, Zijia Chen, Zhilin Wang, David Mosallanezhad, Adi Renduchintala, Haifeng Qian, Dima Rekesh, Fei Jia, Somshubra Majumdar, Vahid Noroozi, Wasi Uddin Ahmad, Sean Narenthiran, Aleksander Ficek, Mehrzad Samadi, Jocelyn Huang, Siddhartha Jain, Igor Gitman, Ivan Moshkov, Wei Du, Shubham Toshniwal, George Armstrong, Branislav Kisacanin, Matvei Novikov, Daria Gitman, Evelina Bakhturina, Jane Polak Scowcroft, John Kamalu, Dan Su, Kezhi Kong, Markus Kliegl, Rabeeh Karimi, Ying Lin, Sanjeev Satheesh, Jupinder Parmar, Pritam Gundecha, Brandon Norick, Joseph Jennings, Shrimai Prabhumoye, Syeda Nahida Akter, Mostofa Patwary, Abhinav

Khattar, Deepak Narayanan, Roger Waleffe, Jimmy Zhang, Bor-Yiing Su, Guyue Huang, Terry Kong, Parth Chadha, Sahil Jain, Christine Harvey, Elad Segal, Jining Huang, Sergey Kashirsky, Robert McQueen, Izzy Putterman, George Lam, Arun Venkatesan, Sherry Wu, Vinh Nguyen, Manoj Kilaru, Andrew Wang, Anna Warno, Abhilash Somasamudramath, Sandip Bhaskar, Maka Dong, Nave Assaf, Shahar Mor, Omer Ullman Argov, Scot Junkin, Oleksandr Romanenko, Pedro Larroy, Monika Katariya, Marco Rovinelli, Viji Balas, Nicholas Edelman, Anahita Bhiwandiwalla, Muthu Subramaniam, Smita Ithape, Karthik Ramamoorthy, Yuting Wu, Suguna Varshini Velury, Omri Almog, Joyjit Daw, Denys Fridman, Erick Galinkin, Michael Evans, Katherine Luna, Leon Derczynski, Nikki Pope, Eileen Long, Seth Schneider, Guillermo Siman, Tomasz Grzegorzek, Pablo Ribalta, Monika Katariya, Joey Conway, Trisha Saar, Ann Guan, Krzysztof Pawelec, Shyamala Prayaga, Oleksii Kuchaiev, Boris Ginsburg, Oluwatobi Olabiyi, Kari Briski, Jonathan Cohen, Bryan Catanzaro, Jonah Alben, Yonatan Geifman, Eric Chung, and Chris Alexiuk. Llama-nemotron: Efficient reasoning models, 2025.

[42] Junyan Zhang, Yiming Huang, Shuliang Liu, Yubo Gao, and Xuming Hu. Do bert-like bidirectional models still perform better on text classification in the era of llms. *arXiv preprint arXiv:2505.18215*, 1, 2025.

[43] Nicolas Boizard, Hippolyte Gisserot-Boukhlef, Duarte M. Alves, André Martins, Ayoub Hammal, Caio Corro, Céline Hudelot, Emmanuel Malherbe, Etienne Malaboeuf, Fanny Jourdan, Gabriel Hautreux, João Alves, Kevin El-Haddad, Manuel Faysse, Maxime Peyrard, Nuno M. Guerreiro, Patrick Fernandes, Ricardo Rei, and Pierre Colombo. Eurobert: Scaling multilingual encoders for european languages, 2025.

[44] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[45] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[46] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[47] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

[48] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of machine learning research*, 24(240):1–113, 2023.

[49] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[50] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[51] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024.

[52] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[53] Pierre Jean A Colombo, Chloé Clavel, and Pablo Piantanida. Infolm: A new metric to evaluate summarization & data2text generation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 10554–10562, 2022.

[54] Ricardo Rei, José G. C. de Souza, Duarte Alves, Chrysoula Zerva, Ana C Farinha, Taisiya Glushkova, Alon Lavie, Luisa Coheur, and André F. T. Martins. COMET-22: Unbabel-IST 2022 submission for the metrics shared task. In Philipp Koehn, Loïc Barrault, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Tom Kocmi, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Marco Turchi, and Marcos Zampieri, editors, *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 578–585, Abu Dhabi, United Arab Emirates (Hybrid), December 2022. Association for Computational Linguistics.

[55] Ricardo Rei, Marcos Treviso, Nuno M. Guerreiro, Chrysoula Zerva, Ana C Farinha, Christine Maroti, José G. C. de Souza, Taisiya Glushkova, Duarte Alves, Luisa Coheur, Alon Lavie, and André F. T. Martins. CometKiwi: IST-unbabel 2022 submission for the quality estimation shared task. In Philipp Koehn, Loïc Barrault, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Tom Kocmi, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Marco Turchi, and Marcos Zampieri, editors, *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 634–645, Abu Dhabi, United Arab Emirates (Hybrid), December 2022. Association for Computational Linguistics.

[56] Nuno M Guerreiro, Ricardo Rei, Daan van Stigt, Luisa Coheur, Pierre Colombo, and André FT Martins. xcomet: Transparent machine translation evaluation through fine-grained error detection. *Transactions of the Association for Computational Linguistics*, 12:979–995, 2024.

[57] Juraj Juraska, Mara Finkelstein, Daniel Deutsch, Aditya Siddhant, Mehdi Mirzazadeh, and Markus Freitag. Metricx-23: The google submission to the wmt 2023 metrics shared task. In *Proceedings of the Eighth Conference on Machine Translation*, pages 756–767, 2023.

[58] Juraj Juraska, Tobias Domhan, Mara Finkelstein, Tetsuji Nakagawa, Geza Kovacs, Daniel Deutsch, Pidong Wang, and Markus Freitag. Metricx-25 and gemspaneval: Google translate submissions to the wmt25 evaluation shared task. In *Proceedings of the Tenth Conference on Machine Translation*, pages 957–968, 2025.

[59] Thibault Sellam, Dipanjan Das, and Ankur Parikh. BLEURT: Learning robust metrics for text generation. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online, July 2020. Association for Computational Linguistics.

[60] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023.

[61] Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. Generalizing verifiable instruction following, 2025.

[62] Anna Bavaresco, Raffaella Bernardi, Leonardo Bertolazzi, Desmond Elliott, Raquel Fernández, Albert Gatt, Esam Ghaleb, Mario Giulianelli, Michael Hanna, Alexander Koller, et al. Llms instead of human judges? a large scale empirical study across 20 nlp evaluation tasks. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 238–255, 2025.

[63] Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, et al. Prometheus: Inducing fine-grained evaluation capability in language models. In *The Twelfth International Conference on Learning Representations*, 2023.

[64] Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4334–4353, 2024.

A. Prompting Details

In this section, we detail how model outputs are generated for both answer generation (Table 7) and answer assessment (Table 9 , Table 10). For answer generation, we also describe the suffixes used to impose different formatting instructions (Table 8). Throughout the main text, we use the soft configuration by default, as it allows both regex-based answer parsing and the inclusion of intermediate chain-of-thought tokens, which improve answer quality (Appendix B). For regex parsing under both the soft and strict formatting constraints, we use the pattern "Final answer:\\s*(.+)", which provides a general and flexible mechanism for extracting the predicted answer.

Task Category	**Generation Prompt**

Multiple-Choice	`Answer the following multiple-choice question.`
	
	`Question: {question}`
	
	`Choices:`
	`A) {choice_1}`
	`B) {choice_2}`
	`C) {choice_3}`
	`D) {choice_4}`
	`[...]`

Context Extraction	`Answer the question based on the provided context.`
	
	`Context:`
	`{context}`
	
	`Question: {question}`

Open-Form Math	`{question}`

Table 7 | Base generation prompts for each task category.

Task Category	Formatting Instruction	Generation Suffix
Multiple-Choice	Free	None
	Soft	`Conclude your response with "Final answer: X", where X is the letter of the correct choice.`
	Strict	`Respond only with the exact format "Final answer: X", where X is the letter of the correct choice.`
Context Extraction	Free	None
	Soft	`Conclude your response with "Final answer: X", where X is the exact span from the context that answers the question.`
	Strict	`Respond only with the exact format "Final answer: X", where X is the exact span from the context that answers the question.`
Open-Form Math	Free	None
	Soft	`Conclude your response with "Final answer: X", where X is the computed solution.`
	Strict	`Respond only with the exact format "Final answer: X", where X is the computed solution.`

Table 8 | Generation suffixes used across task categories for all formatting strategies.

Assessment Prompt

```
You are an expert evaluator. Your task is to determine whether the CANDIDATE response
correctly answers the QUESTION.
Judge the CANDIDATE as correct only if its final answer, disregarding any intermediate
reasoning or explanation, is semantically equivalent to the
REFERENCE with respect to the QUESTION.
Base your judgment solely on the information given. Do not rely on external knowledge.

[QUESTION starts here]
{question}
[QUESTION ends here]

[REFERENCE starts here]
{reference}
[REFERENCE ends here]

[CANDIDATE starts here]
{candidate}
[CANDIDATE ends here]

Conclude your response with exactly one of the following:
- "Final answer: True" if the CANDIDATE is correct
- "Final answer: False" if the CANDIDATE is incorrect
```

Table 9 | Answer assessment prompt for LLM judges, allowing intermediate token generation before the final judgment. This involves Nemotron-Super-v1.5 for label generation (subsection 2.2) and generative judges evaluated under inference budget L (Figure 3).

Assessment Prompt

```
You are an expert evaluator. Your task is to determine whether the CANDIDATE response
correctly answers the QUESTION.
Judge the CANDIDATE as correct only if its final answer, disregarding any intermediate
reasoning or explanation, is semantically equivalent to the
REFERENCE with respect to the QUESTION.
Base your judgment solely on the information given. Do not rely on external knowledge.

[QUESTION starts here]
{question}
[QUESTION ends here]

[REFERENCE starts here]
{reference}
[REFERENCE ends here]

[CANDIDATE starts here]
{candidate}
[CANDIDATE ends here]

Respond with exactly one of the following strings (add no additional text):
- "Final answer: True" if the CANDIDATE is correct
- "Final answer: False" if the CANDIDATE is incorrect
```

Table 10 | Prompt used for direct assessment by LLM judges, applied to generative judges evaluated under inference budget S (Figure 3).

B. Effect of Generation Mode on Downstream Performance

In this section, we compare different answer production modes to assess their impact on model performance. Specifically, answers are generated under three formatting regimes:[10]

- *Log-likelihood*: Candidate answers are iteratively appended to the prompt, and the model's prediction is derived from the sequence with the highest log-likelihood.[11]
- *Strict*: The model is prompted to respond exactly with "Final answer: [answer]".
- *Soft*: The model is prompted to conclude its response with "Final answer: [answer]" but may reason before answering.
- *Free*: The model may answer in any format.

The results are reported in Table 11.

Task	Log-lik.	Generative		
		Strict	Soft	Free
Multiple-Choice				
ARC-Challenge	46.3	74.4	**76.2**	75.0
ARC-Easy	65.0	84.6	**84.9**	84.4
GPQA	25.9	**32.3**	31.8	28.8
MMLU	39.9	59.4	**62.0**	60.5
MMLU-Pro	21.1	38.0	**44.3**	42.6
TruthfulQA	33.2	51.3	**51.6**	51.3
Context Extraction				
CoQA	–	71.3	76.5	**86.7**
DROP	–	48.4	60.2	**64.5**
HotpotQA	–	66.0	71.0	**82.4**
SQuAD-v2	–	44.7	53.5	**62.0**
Open-Form Math				
AIME24	–	18.1	**19.8**	18.1
AIME25	–	13.2	14.4	**15.7**
ASDiv	–	68.2	81.5	**82.1**
GSM8K	–	43.1	73.6	**73.6**
Math	–	49.3	**60.2**	57.3

Table 11 | Comparison of evaluation modes across all benchmarks. Results are averaged over all models, and bold values indicate the best performance for each task.

Models demonstrate greater capacity in generative mode. We first examine multiple-choice tasks by comparing generative evaluation against the log-likelihood approach. Our results indicate that the likelihood-based setup consistently and severely impairs performance across all evaluated multiple-choice benchmarks (e.g., -22.1% on MMLU and -29.9% on ARC-Challenge). This suggests that while likelihood evaluation offers a convenient, regex-free parsing mechanism, it significantly bottlenecks the model's inherent problem-solving capabilities compared to generative inference.

Strict formatting constraints impair performance. Setting aside likelihood-based evaluation, we compare strict and soft generative prompting strategies. We observe that strict prompting yields the lowest overall performance. While it performs comparably to the soft method on multiple-choice tasks, it significantly degrades performance on tasks requiring more complex outputs (e.g., -11.8% on DROP and -30.5% on GSM8K). This pronounced drop highlights the importance of allowing intermediate chain-of-thought generation to fully leverage the model's problem-solving capacity.

[10] Full prompting details are provided in Appendix A.
[11] Applies only to multiple-choice tasks.

C. Human-Synthetic Label Agreement

This section complements subsection 2.2 in the main text by presenting detailed results of the human annotations. We report human-synthetic average agreement per task category (Table 12), showing consistently high agreement across categories, and analyze how this agreement impacts downstream performance measurement.

Task Category	Accuracy (%)
Context Extraction	96.83
Multiple-Choice	96.81
Open-Form Math	98.70
Average	**97.45**

Table 12 | Accuracy between human and synthetic labels across task categories.

As described in subsection 2.4, the reported accuracies in the main text are computed using synthetic labels generated by Nemotron-Super-v1.5. To estimate performance with respect to human annotations, we can apply a correction based on the observed agreement between human and synthetic labels. Let A_H denote the accuracy with respect to human labels (unknown), A_S the accuracy against synthetic labels, and ρ the agreement rate between synthetic and human judgments. Let Y_H, Y_S, and $\hat{Y}$ be random variables representing, for a given example, the human label, the synthetic label, and the predicted label, respectively.

$$
\begin{aligned}
A_H &= \mathbb{P}\left(\hat{Y} = Y_H\right) && (1)\\
&= \mathbb{P}\left(\hat{Y} = Y_H | Y_H = Y_S\right)\mathbb{P}\left(Y_H = Y_S\right) + \mathbb{P}\left(\hat{Y} = Y_H | Y_H \neq Y_S\right)\mathbb{P}\left(Y_H \neq Y_S\right) && (2)\\
&= \mathbb{P}\left(\hat{Y} = Y_S | Y_H = Y_S\right)\mathbb{P}\left(Y_H = Y_S\right) + \mathbb{P}\left(\hat{Y} \neq Y_S | Y_H \neq Y_S\right)\mathbb{P}\left(Y_H \neq Y_S\right) && (3)\\
&= \mathbb{P}\left(\hat{Y} = Y_S\right)\mathbb{P}\left(Y_H = Y_S\right) + \mathbb{P}\left(\hat{Y} \neq Y_S\right)\mathbb{P}\left(Y_H \neq Y_S\right) && (4)\\
&= A_S\rho + (1 - A_S)(1 - \rho) && (5)\\
&= (2\rho - 1)\, A_S + 1 - \rho && (6)
\end{aligned}
$$

The final expression (Equation 6)[12] can be interpreted as pulling the estimated accuracy A_H toward random guessing. If human and synthetic labels are uncorrelated ($\rho = 0.5$), the estimated accuracy drops to a random guess, regardless of A_S (see Figure 6 for illustration).

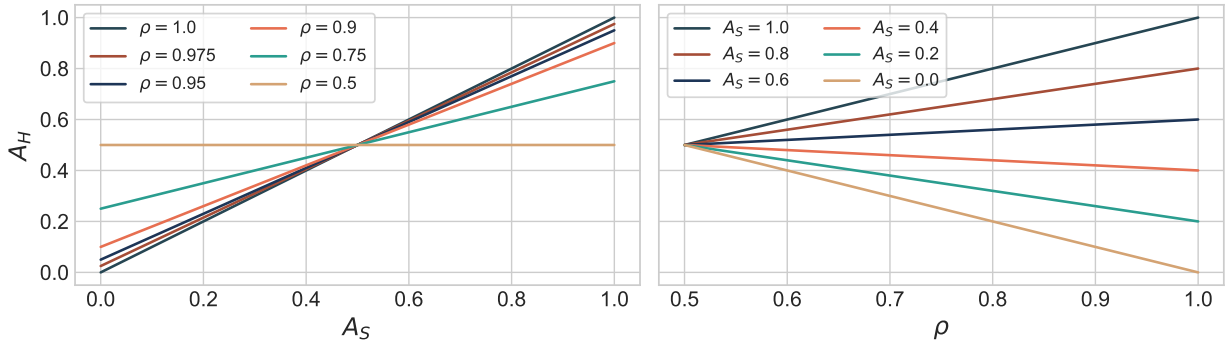


Figure 6 | Sensitivity of the A_H estimate to variations in A_S and ρ.

[12] In Equation 4, we assume $(\hat{Y}, Y_S) \perp (Y_H = Y_S)$, in line with our empirical observations. Intuitively, this means that agreement between the predicted and synthetic labels, $\hat{Y}$ and Y_S, is independent of whether the synthetic label Y_S matches the human label Y_H.

D. Detailed Results

D.1. Regex Parsing Failures

In this section, we extend Figure 2 from the main text by presenting disaggregated regex parsing failure rates across models and tasks.

Family	Size	ARC Challenge	ARC Easy	GPQA	MMLU	MMLU Pro	Truthful QA
Apertus	8B	0.2	0.1	4.9	0.8	4.5	0.0
	70B	0.1	0.1	21.2	2.6	12.3	1.5
EuroLLM	1.7B	97.6	98.7	83.0	94.1	69.9	66.3
	9B	0.0	0.0	11.8	1.2	11.8	0.0
	22B	0.1	0.0	7.4	0.4	3.0	0.0
Falcon-3	1B	5.1	4.0	11.2	5.5	31.9	3.7
	3B	0.0	0.2	2.0	0.2	2.5	0.1
	7B	0.0	0.0	1.1	0.1	1.3	0.1
Gemma-3	1B	0.1	0.0	12.5	2.3	14.5	0.0
	4B	0.0	0.0	6.2	0.3	3.3	0.0
	12B	0.0	0.0	4.9	0.1	3.0	0.0
	27B	0.0	0.0	3.6	0.1	1.3	0.0
LFM-2	0.35B	0.4	0.4	7.1	1.2	11.3	4.0
	0.7B	2.9	2.8	18.1	6.1	15.5	1.6
	1.2B	0.5	0.5	6.7	1.0	5.4	0.4
	2.6B	0.0	0.2	10.0	1.4	9.9	0.6
Llama-3	1B	5.0	3.6	36.4	9.0	26.0	5.4
	3B	8.1	5.2	32.4	21.8	47.4	16.9
	8B	0.1	0.3	25.2	2.6	16.1	2.3
	70B	0.0	0.0	7.4	0.9	6.0	0.0
Ministral-3	3B	0.1	0.0	23.9	1.2	15.1	0.0
	8B	0.0	0.0	25.0	1.4	10.3	0.0
	14B	0.2	0.1	16.5	1.3	8.6	0.0
OLMo-3	7B	0.7	0.4	22.3	1.9	15.8	0.0
	32B	0.3	0.0	51.1	1.6	17.4	0.0
Phi-4	3.6B	0.0	0.0	2.5	0.2	3.0	0.0
	14B	0.0	0.0	0.9	0.0	0.6	0.0
Qwen-3	0.6B	0.0	0.0	0.0	0.1	0.0	0.0
	4B	0.0	0.0	9.6	0.4	4.6	0.0
	8B	0.1	0.0	10.9	0.7	5.9	0.0
	14B	0.1	0.0	11.4	0.3	4.7	0.0
	32B	0.0	0.0	7.8	0.1	4.3	0.0
SmolLM-2/3	0.135B	98.4	98.4	97.8	97.6	97.8	98.3
	0.36B	6.7	5.2	43.1	12.2	37.0	7.1
	1.7B	0.0	0.0	0.2	0.2	0.5	0.2
	3B	0.7	0.5	19.4	5.1	16.9	1.6

Table 13 | Parsing failure rates on multiple-choice benchmarks.

Family	Size	CoQA	DROP	HotpotQA	SQuAD-v2
Apertus	8B	0.0	0.2	0.1	0.1
	70B	0.0	0.3	0.0	0.1
EuroLLM	1.7B	41.4	39.7	16.3	38.6
	9B	0.0	0.0	0.0	0.0
	22B	0.2	0.0	0.3	0.0
Falcon-3	1B	20.6	6.2	3.9	5.2
	3B	2.0	0.8	1.2	3.6
	7B	0.6	0.5	0.2	0.2
Gemma-3	1B	0.0	0.8	0.1	0.3
	4B	0.0	0.0	0.1	0.0
	12B	0.2	0.0	0.3	0.1
	27B	0.0	0.0	0.0	0.0
LFM-2	0.35B	1.8	2.2	4.1	3.5
	0.7B	20.8	1.3	0.5	1.1
	1.2B	0.2	0.5	1.5	0.2
	2.6B	1.8	2.8	1.6	3.2
Llama-3	1B	31.2	53.0	28.9	33.9
	3B	34.6	5.3	17.1	27.9
	8B	6.4	7.7	3.5	9.8
	70B	0.0	0.0	0.0	0.3
Ministral-3	3B	0.0	0.1	0.1	0.1
	8B	0.0	0.0	0.0	0.0
	14B	0.0	0.0	0.0	0.3
OLMo-3	7B	0.0	2.2	0.5	1.8
	32B	0.0	0.1	0.1	2.3
Phi-4	3.6B	0.0	0.0	0.0	0.0
	14B	0.0	0.0	0.0	0.0
Qwen-3	0.6B	0.0	0.0	0.0	0.0
	4B	0.0	0.0	0.0	0.0
	8B	0.0	0.1	0.0	0.0
	14B	0.0	0.0	0.0	0.0
	32B	0.0	0.0	0.0	0.0
SmolLM-2/3	0.135B	47.8	46.1	26.0	50.8
	0.36B	54.2	44.2	24.8	48.1
	1.7B	1.2	0.1	0.4	3.3
	3B	0.4	0.7	0.1	0.3

Table 14 | Parsing failure rates on context extraction benchmarks.

Family	Size	AIME24	AIME25	ASDiv	GSM8K	Math
Apertus	8B	50.0	46.7	0.9	0.8	13.3
	70B	26.7	36.7	1.3	0.4	15.5
EuroLLM	1.7B	70.0	76.7	5.3	6.3	54.1
	9B	63.3	50.0	16.9	16.2	43.7
	22B	33.3	30.0	2.5	6.8	11.7
Falcon-3	1B	83.3	90.0	69.0	59.9	77.7
	3B	46.7	40.0	0.9	0.5	16.2
	7B	13.3	16.7	0.0	0.2	2.9
Gemma-3	1B	93.3	83.3	10.3	18.9	47.5
	4B	50.0	16.7	0.3	0.7	8.9
	12B	33.3	33.3	0.1	0.2	9.4
	27B	33.3	23.3	0.2	0.2	5.2
LFM-2	0.35B	73.3	66.7	3.2	4.5	56.2
	0.7B	60.0	66.7	1.3	2.8	29.3
	1.2B	33.3	33.3	5.1	12.2	15.1
	2.6B	33.3	43.3	0.3	0.4	13.3
Llama-3	1B	100.0	100.0	25.9	28.1	84.9
	3B	53.3	70.0	1.8	1.3	20.2
	8B	53.3	56.7	3.1	1.2	18.5
	70B	100.0	100.0	6.7	25.4	92.1
Ministral-3	3B	96.7	86.7	0.8	1.6	26.6
	8B	86.7	90.0	0.7	1.4	19.1
	14B	93.3	86.7	0.4	1.1	23.4
OLMo-3	7B	60.0	46.7	0.3	1.0	11.1
	32B	80.0	83.3	0.3	1.3	15.8
Phi-4	3.6B	40.0	23.3	0.1	0.1	18.7
	14B	16.7	16.7	0.0	0.0	3.2
Qwen-3	0.6B	83.3	73.3	0.5	2.0	34.5
	4B	56.7	63.3	0.2	0.6	8.7
	8B	56.7	60.0	0.3	0.1	7.9
	14B	50.0	46.7	0.1	0.0	6.1
	32B	50.0	43.3	0.1	0.6	6.3
SmolLM-2/3	0.135B	70.0	66.7	36.4	40.6	68.2
	0.36B	70.0	63.3	15.8	22.6	52.3
	1.7B	63.3	80.0	2.4	5.8	38.7
	3B	50.0	43.3	0.9	1.4	20.4

Table 15 | Parsing failure rates on open-form math benchmarks.

D.2. Impact of Regex-Based Evaluation on Performance Measurement

This section extends Table 1 from the main text by providing a breakdown of how regex-based evaluation affects downstream measured performance across every model and task.

Family	Size	CoQA		DROP		HotpotQA		SQuAD-v2	
		ΔAccuracy	ΔRank	ΔAccuracy	ΔRank	ΔAccuracy	ΔRank	ΔAccuracy	ΔRank
Apertus	8B	-30.2 (-0.0/-30.2)	↓3.0	-21.8 (-0.0/-21.8)	↑1.5	-20.6 (-0.1/-20.5)	↑2.0	-19.2 (-0.0/-19.2)	↑10.0
	70B	-41.4 (-0.0/-41.4)	↓7.5	-21.9 (-0.1/-21.9)	↑3.0	-22.7 (-0.0/-22.7)	↓3.0	-25.3 (-0.0/-25.3)	↓1.0
EuroLLM	1.7B	-49.6 (-30.6/-19.0)	↓8.0	-19.5 (-11.6/-7.9)	↓1.0	-37.2 (-9.9/-27.3)	↓6.0	-30.0 (-16.6/-13.4)	↓7.0
	9B	-27.6 (-0.0/-27.6)	↓1.0	-18.9 (-0.0/-18.9)	↑1.0	-15.9 (-0.0/-15.9)	↑5.0	-22.6 (-0.0/-22.6)	↑6.0
	22B	-24.8 (-0.2/-24.6)	↑1.0	-25.4 (-0.0/-25.4)	0.0	-16.3 (-0.2/-16.1)	↑1.0	-29.2 (-0.0/-29.2)	0.0
Falcon-3	1B	-42.6 (-16.2/-26.4)	↓7.5	-11.5 (-1.7/-9.8)	0.0	-17.4 (-2.2/-15.3)	0.0	-19.3 (-2.1/-17.2)	0.0
	3B	-34.0 (-2.0/-32.0)	↓1.0	-18.6 (-0.5/-18.1)	0.0	-23.4 (-1.0/-22.4)	↓1.0	-21.7 (-1.7/-20.0)	↑2.0
	7B	-19.8 (-0.4/-19.4)	↑4.0	-14.6 (-0.3/-14.3)	↑6.0	-12.4 (-0.2/-12.2)	↑4.0	-18.0 (-0.1/-17.9)	↑14.0
Gemma-3	1B	-19.0 (-0.0/-19.0)	↑10.5	-6.0 (-0.1/-5.9)	↑2.0	-9.9 (-0.1/-9.8)	↑2.0	-14.1 (-0.1/-14.0)	↑4.0
	4B	-43.0 (-0.0/-43.0)	↓5.0	-27.4 (-0.0/-27.4)	↑2.0	-23.8 (-0.1/-23.8)	0.0	-29.0 (-0.0/-28.9)	↓3.0
	12B	-28.8 (-0.2/-28.6)	↓5.0	-31.8 (-0.0/-31.8)	↓5.0	-20.1 (-0.3/-19.9)	↓7.0	-29.5 (-0.1/-29.4)	0.0
	27B	-28.8 (-0.0/-28.8)	↓1.0	-31.1 (-0.0/-31.1)	↓4.0	-25.4 (-0.0/-25.4)	↓2.5	-29.4 (-0.0/-29.4)	↓1.0
LFM-2	0.35B	-20.0 (-1.4/-18.6)	↑8.5	-3.7 (-0.6/-3.1)	↑3.0	-8.2 (-2.0/-6.1)	↑3.0	-10.3 (-1.5/-8.8)	↑5.0
	0.7B	-33.6 (-15.8/-17.8)	↑1.0	-5.5 (-0.4/-5.1)	↑2.0	-10.3 (-0.3/-10.0)	↑2.0	-11.4 (-0.6/-10.7)	↑3.0
	1.2B	-14.8 (-0.0/-14.8)	↑14.5	-8.9 (-0.1/-8.8)	↑2.0	-9.9 (-1.2/-8.7)	↑7.0	-13.2 (-0.0/-13.1)	↑13.0
	2.6B	-29.6 (-1.6/-28.0)	↑2.5	-28.6 (-1.2/-27.4)	↓2.0	-22.6 (-1.1/-21.5)	0.0	-33.5 (-1.2/-32.4)	↓9.0
Llama-3	1B	-30.0 (-21.0/-9.0)	↑3.0	-24.1 (-20.4/-3.7)	↓6.0	-22.0 (-17.2/-4.8)	0.0	-19.7 (-13.9/-5.8)	↓2.0
	3B	-40.4 (-29.2/-11.2)	↓4.5	-17.9 (-3.0/-14.9)	↑8.0	-22.6 (-13.4/-9.3)	0.0	-29.1 (-17.0/-12.1)	↓5.0
	8B	-23.6 (-4.6/-19.0)	↓1.0	-21.5 (-3.7/-17.8)	↑3.0	-17.1 (-2.0/-15.1)	↑3.0	-21.9 (-5.9/-16.0)	↑9.0
	70B	-17.2 (-0.0/-17.2)	↑7.0	-17.0 (-0.0/-17.0)	↑3.0	-11.5 (-0.0/-11.5)	↑10.0	-27.8 (-0.3/-27.5)	↑6.0
Ministral	3B	-36.8 (-0.0/-36.8)	↓1.0	-32.9 (-0.0/-32.9)	↓6.0	-29.8 (-0.0/-29.7)	↓5.0	-30.6 (-0.0/-30.6)	↓8.0
	8B	-30.0 (-0.0/-30.0)	↑2.0	-35.0 (-0.0/-35.0)	↓9.0	-24.9 (-0.0/-24.9)	↓6.0	-34.0 (-0.0/-34.0)	↓6.0
	14B	-27.4 (-0.0/-27.4)	↓2.0	-30.7 (-0.0/-30.7)	↓5.0	-23.4 (-0.0/-23.4)	↓9.0	-36.3 (-0.0/-36.3)	↓7.0
OLMo-3	7B	-17.6 (-0.0/-17.6)	↑5.0	-22.5 (-0.8/-21.7)	↑4.0	-11.7 (-0.0/-11.6)	↑9.5	-26.4 (-0.2/-26.2)	↑7.0
	32B	-24.8 (-0.0/-24.8)	↓3.5	-27.3 (-0.0/-27.2)	↓2.0	-16.1 (-0.0/-16.0)	0.0	-35.6 (-2.2/-33.4)	↓4.0
Phi-4	3.6B	-42.0 (-0.0/-42.0)	↓8.5	-23.2 (-0.0/-23.2)	↑1.0	-29.4 (-0.0/-29.4)	↓3.0	-35.1 (-0.0/-35.1)	↓12.0
	14B	-36.0 (-0.0/-36.0)	↓0.5	-32.0 (-0.0/-32.0)	↓4.0	-27.8 (-0.0/-27.8)	↓8.0	-45.5 (-0.0/-45.5)	↓18.0
Qwen-3	0.6B	-27.2 (-0.0/-27.2)	↑4.0	-9.4 (-0.0/-9.4)	↑1.0	-7.3 (-0.0/-7.3)	↑3.0	-11.1 (-0.0/-11.1)	↑3.0
	4B	-23.0 (-0.0/-23.0)	↑6.0	-20.0 (-0.0/-20.0)	↑6.0	-12.9 (-0.0/-12.9)	↑7.0	-25.0 (-0.0/-25.0)	↑9.0
	8B	-30.6 (-0.0/-30.6)	↓8.0	-34.5 (-0.0/-34.4)	↓9.0	-19.7 (-0.0/-19.7)	↓4.0	-34.2 (-0.0/-34.2)	↓8.0
	14B	-19.8 (-0.0/-19.8)	0.0	-19.9 (-0.0/-19.9)	↓1.0	-12.8 (-0.0/-12.8)	0.0	-31.1 (-0.0/-31.1)	↑3.0
	32B	-23.0 (-0.0/-23.0)	↓2.5	-25.3 (-0.0/-25.3)	↑2.0	-17.1 (-0.0/-17.1)	↓2.0	-35.1 (-0.0/-35.1)	↓2.0
SmolLM-2/3	0.135B	-38.0 (-24.2/-13.8)	0.0	-11.5 (-6.8/-4.7)	↑1.0	-31.7 (-10.1/-21.6)	↓1.0	-23.9 (-12.5/-11.4)	0.0
	0.36B	-55.0 (-35.2/-19.8)	↓6.0	-17.6 (-11.4/-6.1)	↓2.0	-36.5 (-12.7/-23.8)	↓4.0	-27.9 (-16.0/-11.9)	↓3.0
	1.7B	-25.2 (-1.0/-24.2)	↑6.5	-8.4 (-0.1/-8.3)	0.0	-14.7 (-0.3/-14.4)	↑3.0	-18.3 (-1.5/-16.8)	0.0
	3B	-35.2 (-0.4/-34.8)	↑1.0	-20.5 (-0.1/-20.3)	↑4.5	-22.3 (-0.0/-22.2)	0.0	-25.4 (-0.0/-25.3)	↑2.0

Table 16 | Effect of regex-based evaluation on performance measurement for context extraction benchmarks.

Family	Size	ARC-Challenge		ARC-Easy		GPQA		MMLU		MMLU-Pro		TruthfulQA	
		ΔAccuracy	ΔRank	ΔAccuracy	ΔRank	ΔAccuracy	ΔRank	ΔAccuracy	ΔRank	ΔAccuracy	ΔRank	ΔAccuracy	ΔRank
Apertus	8B	0.0 (-0.1/+0.1)	↑3.0	0.0 (-0.0/0.0)	↑6.0	+5.8 (-0.0/+5.8)	↑16.0	-0.9 (-0.1/-0.8)	↑4.0	+0.9 (-0.5/+1.4)	↑4.0	-0.5 (-0.0/-0.5)	0.0
	70B	-1.8 (-0.1/-1.7)	↑3.0	-1.5 (-0.0/-1.5)	↑2.0	-1.3 (-1.3/0.0)	↑8.5	-3.5 (-0.4/-3.1)	↑3.0	-2.9 (-1.2/-1.7)	↑1.0	-3.3 (-0.5/-2.8)	↓2.0
EuroLLM	1.7B	-29.8 (-28.2/-1.5)	↓1.0	-29.2 (-28.4/-0.8)	↓1.0	-28.3 (-19.2/-9.2)	↓17.5	-27.4 (-24.6/-2.9)	↓1.0	-22.4 (-9.3/-13.1)	↓7.0	-29.0 (-16.3/-12.7)	↓5.0
	9B	-0.3 (-0.0/-0.3)	↑4.0	-0.0 (-0.0/-0.0)	↑6.0	-8.9 (-2.7/-6.3)	↑0.5	-1.6 (-0.5/-1.2)	↑2.0	-8.2 (-2.7/-5.5)	0.0	+0.2 (-0.0/+0.2)	↑3.0
	22B	-8.6 (-0.1/-8.5)	↓6.0	-7.7 (-0.0/-7.7)	↓8.0	-9.4 (-1.1/-8.3)	↓2.5	-12.7 (-0.1/-12.6)	↓8.0	-10.0 (-0.7/-9.3)	↓2.0	-6.2 (-0.0/-6.2)	↓1.0
Falcon-3	1B	-26.7 (-2.6/-24.1)	0.0	-33.1 (-3.0/-30.1)	0.0	-14.7 (-2.2/-12.5)	↓10.5	-15.4 (-1.7/-13.7)	↑2.0	-13.8 (-6.3/-7.5)	0.0	-6.6 (-0.6/-6.0)	↑3.0
	3B	-3.2 (-0.0/-3.2)	↑2.0	-2.2 (-0.1/-2.1)	↑2.0	-4.5 (-0.2/-4.2)	↑7.0	-3.9 (-0.1/-3.9)	↑2.0	-7.5 (-0.4/-7.1)	↓1.0	-1.1 (-0.1/-1.0)	↑2.0
	7B	-5.2 (-0.0/-5.2)	↓2.0	-6.6 (-0.0/-6.6)	↓4.0	-6.0 (-0.7/-5.4)	↑2.0	-6.7 (-0.0/-6.7)	↓1.0	-7.1 (-0.4/-6.7)	↑1.0	-2.0 (-0.0/-2.0)	↑1.0
Gemma-3	1B	+0.1 (-0.1/+0.2)	↑5.0	-0.6 (-0.0/-0.6)	↑5.0	+2.5 (-1.6/+4.0)	↑11.0	-1.0 (-0.6/-0.4)	↑4.0	-2.2 (-1.8/-0.5)	↑5.0	+0.5 (-0.0/+0.5)	↑4.0
	4B	+0.2 (-0.0/+0.2)	↑3.0	0.0 (-0.0/0.0)	↑6.0	+1.1 (-0.4/+1.6)	↑12.0	+0.9 (-0.1/+0.9)	↑7.0	-0.4 (-0.5/+0.1)	↑6.0	+0.2 (-0.0/+0.2)	↑3.0
	12B	+0.3 (-0.0/+0.3)	↑6.0	0.0 (-0.0/0.0)	↑4.0	-0.4 (-0.4/0.0)	↑7.0	0.0 (-0.0/0.0)	↑6.0	-1.1 (-0.5/-0.6)	↑5.0	+0.6 (-0.0/+0.6)	↑4.0
	27B	+0.3 (-0.0/+0.3)	↑5.0	+0.2 (-0.0/+0.2)	↑4.0	+0.2 (-0.2/+0.4)	↑6.0	+0.3 (-0.0/+0.3)	↑4.0	-0.9 (-0.4/-0.5)	↑5.0	+0.2 (-0.0/+0.2)	↑4.0
LFM-2	0.35B	-36.1 (-0.2/-35.9)	↑1.0	-50.0 (-0.4/-49.6)	↑1.0	-9.8 (-0.4/-9.4)	↑3.5	-29.9 (-0.5/-29.4)	↑1.0	-15.0 (-1.8/-13.2)	0.0	-20.3 (-2.0/-18.4)	↓1.0
	0.7B	-16.6 (-1.5/-15.1)	↓2.0	-21.6 (-2.4/-19.2)	↓1.0	-11.8 (-3.6/-8.3)	↑1.0	-23.5 (-2.9/-20.6)	↓3.0	-14.4 (-3.1/-11.3)	↓1.0	-6.6 (-1.1/-5.5)	↑2.0
	1.2B	-5.7 (-0.3/-5.5)	↑2.0	-4.6 (-0.4/-4.2)	↑2.0	-4.9 (-0.7/-4.2)	↑3.0	-10.1 (-0.4/-9.8)	↓1.0	-13.5 (-0.9/-12.6)	↓2.0	-9.9 (-0.0/-9.9)	↓2.0
	2.6B	-0.1 (-0.0/-0.1)	↑5.5	-0.2 (-0.0/-0.2)	↑6.5	-5.4 (-1.1/-4.2)	↑3.5	-1.3 (-0.3/-1.0)	↑6.0	-4.5 (-1.7/-2.8)	↑3.0	-2.1 (-0.1/-2.0)	↑1.0
Llama-3	1B	-1.9 (-2.4/+0.5)	↑5.0	-1.9 (-2.1/+0.2)	↑5.0	+2.0 (-3.8/+5.8)	↑9.5	-1.2 (-2.5/+1.4)	↑5.0	-1.7 (-3.2/+1.5)	↑4.0	-1.0 (-1.6/+0.6)	0.0
	3B	-14.8 (-6.5/-8.3)	↓1.0	-13.0 (-4.3/-8.8)	↓1.0	-2.2 (-5.8/+3.6)	↑8.0	-17.9 (-11.7/-6.2)	↓4.0	-14.7 (-14.0/-0.8)	↓2.0	-15.9 (-8.7/-7.2)	↓3.0
	8B	-38.1 (-0.0/-38.1)	↓11.0	-32.5 (-0.2/-32.4)	↓10.5	-5.6 (-2.0/-3.6)	↑4.0	-26.4 (-0.5/-25.9)	↓9.0	-14.3 (-2.0/-12.3)	↓4.0	-22.8 (-1.0/-21.8)	↓8.0
	70B	+0.2 (-0.0/+0.2)	↑3.0	+0.1 (-0.0/+0.1)	0.0	-3.3 (-2.7/-0.7)	0.0	-0.6 (-0.7/+0.1)	0.0	-3.5 (-2.9/-0.6)	↑3.0	+0.4 (-0.0/+0.4)	↑2.0
Ministral	3B	-2.0 (-0.0/-2.0)	↑3.0	-0.7 (-0.0/-0.7)	↑4.0	-11.4 (-6.7/-4.7)	↑0.5	-2.1 (-0.3/-1.7)	↑2.0	-8.0 (-5.1/-2.9)	↑3.0	-1.5 (-0.0/-1.5)	↑0.5
	8B	-4.4 (-0.0/-4.4)	0.0	-2.6 (-0.0/-2.6)	↑2.0	-8.7 (-5.6/-3.1)	↑2.0	-8.3 (-0.6/-7.7)	↑1.0	-12.3 (-3.1/-9.2)	↑1.0	-9.1 (-0.0/-9.1)	↓5.0
	14B	-1.8 (-0.0/-1.8)	0.0	-0.9 (-0.1/-0.8)	0.0	-10.3 (-3.8/-6.5)	↓1.0	-5.7 (-0.5/-5.1)	0.0	-11.3 (-3.0/-8.3)	↑1.0	-8.1 (-0.0/-8.1)	↓2.0
OLMo-3	7B	-0.3 (-0.2/-0.1)	↑5.0	-0.0 (-0.0/-0.0)	↑7.0	-8.0 (-6.7/-1.3)	↑1.5	-0.9 (-0.5/-0.4)	↑6.0	-8.4 (-5.8/-2.6)	↑1.0	+0.4 (-0.0/+0.4)	↑3.0
	32B	+0.2 (-0.1/+0.3)	↑3.0	0.0 (-0.0/0.0)	↑4.0	-25.9 (-22.3/-3.6)	↓11.0	-1.6 (-0.6/-1.1)	↑4.0	-13.0 (-9.5/-3.5)	↓2.0	0.0 (-0.0/0.0)	0.0
Phi-4	3.6B	-0.4 (-0.0/-0.4)	↑5.0	-0.7 (-0.0/-0.7)	↑4.0	+3.1 (-0.2/+3.3)	↑14.0	-3.0 (-0.1/-2.9)	↑4.0	-1.9 (-0.4/-1.6)	↑4.0	-0.1 (-0.0/-0.1)	↑1.5
	14B	-6.7 (-0.0/-6.7)	↓7.5	-8.0 (-0.0/-8.0)	↓13.0	-10.5 (-0.2/-10.3)	↓1.0	-12.3 (-0.0/-12.3)	↓4.0	-6.5 (-0.1/-6.4)	↓1.0	-10.0 (-0.0/-10.0)	↓5.0
Qwen-3	0.6B	-31.2 (-0.0/-31.2)	↓2.0	-43.3 (-0.0/-43.3)	↓2.0	-14.7 (-0.0/-14.7)	↓11.0	-19.3 (-0.0/-19.3)	0.0	-8.0 (-0.0/-8.0)	0.0	-8.4 (-0.0/-8.4)	↓2.0
	4B	-0.2 (-0.0/-0.2)	↑4.0	+0.1 (-0.0/+0.1)	↑5.0	-7.1 (-2.0/-5.1)	↑4.0	-3.1 (-0.1/-3.0)	↑3.0	-8.5 (-0.9/-7.6)	↑2.0	-0.1 (-0.0/-0.1)	↑3.0
	8B	-2.2 (-0.0/-2.2)	↑3.0	-0.9 (-0.0/-0.9)	↑2.0	-11.6 (-2.0/-9.6)	↓2.0	-7.8 (-0.3/-7.5)	0.0	-15.1 (-1.5/-13.7)	↓1.0	-4.0 (-0.0/-4.0)	↑2.0
	14B	-16.0 (-0.0/-16.0)	↓18.0	-11.2 (-0.0/-11.2)	↓18.0	-26.1 (-2.9/-23.2)	↓15.0	-21.6 (-0.0/-21.5)	↓14.0	-30.4 (-1.4/-29.0)	↓10.0	-10.3 (-0.0/-10.3)	↓4.0
	32B	-19.5 (-0.0/-19.5)	↓20.5	-13.0 (-0.0/-13.0)	↓20.0	-38.6 (-1.3/-37.3)	↓26.0	-27.9 (-0.0/-27.9)	↓19.0	-37.7 (-1.3/-36.4)	↓19.0	-5.8 (-0.0/-5.8)	↓1.0
SmolLM-2/3	0.135B	-23.2 (-22.9/-0.3)	↑1.0	-24.0 (-23.7/-0.3)	↑2.5	-20.8 (-20.5/-0.2)	↓6.0	-22.7 (-22.0/-0.7)	0.0	-11.9 (-11.6/-0.3)	↓1.0	-18.2 (-17.9/-0.4)	↑1.0
	0.36B	-23.9 (-1.7/-22.2)	↑2.0	-26.5 (-1.3/-25.2)	↑1.5	-21.2 (-10.0/-11.2)	↓6.0	-24.1 (-2.8/-21.4)	↑2.0	-9.2 (-4.0/-5.2)	↑3.0	-22.3 (-0.9/-21.4)	0.0
	1.7B	-57.2 (-0.0/-57.2)	↓7.0	-79.2 (-0.0/-79.2)	↓9.0	-27.7 (-0.2/-27.5)	↓15.5	-44.9 (-0.1/-44.8)	↓7.0	-15.9 (-0.1/-15.8)	↓1.0	-18.4 (-0.1/-18.2)	0.0
	3B	-0.3 (-0.1/-0.3)	↑4.5	-0.4 (-0.1/-0.3)	↑6.0	-6.9 (-3.3/-3.6)	↑0.5	-3.0 (-1.7/-1.3)	↑3.0	-4.9 (-3.1/-1.8)	↑2.0	-1.7 (-0.4/-1.3)	↑1.0

Table 17 | Effect of regex-based evaluation on performance measurement for multiple-choice benchmarks.

Family	Size	AIME24		AIME25		ASDiv		GSM8K		Math	
		ΔAccuracy	ΔRank	ΔAccuracy	ΔRank	ΔAccuracy	ΔRank	ΔAccuracy	ΔRank	ΔAccuracy	ΔRank
Apertus	8B	0.0 (-0.0/0.0)	↑4.0	0.0 (-0.0/0.0)	↑5.0	-5.1 (-0.1/-5.0)	↑3.0	-0.4 (-0.2/-0.2)	↑3.0	-10.2 (-1.2/-9.0)	↑6.0
	70B	0.0 (-0.0/0.0)	↑4.0	0.0 (-0.0/0.0)	↑5.0	-5.8 (-0.7/-5.1)	↑1.0	-0.3 (-0.0/-0.3)	↑2.0	-13.3 (-1.3/-12.0)	↑7.0
EuroLLM	1.7B	0.0 (-0.0/0.0)	↑4.0	0.0 (-0.0/0.0)	↑5.0	-1.7 (-0.9/-0.8)	↑1.0	-0.8 (-0.2/-0.7)	↑1.0	-2.7 (-2.0/-0.7)	0.0
	9B	-10.0 (-6.7/-3.3)	↓7.5	-3.3 (-3.3/0.0)	↓3.0	-29.5 (-11.9/-17.6)	↓4.0	-27.5 (-10.7/-16.8)	↓6.0	-30.2 (-15.5/-14.7)	↓1.0
	22B	-20.0 (-3.3/-16.7)	↓3.0	-6.7 (-0.0/-6.7)	↓7.5	-32.3 (-1.9/-30.4)	↓8.0	-29.7 (-5.4/-24.3)	↓8.0	-38.6 (-4.2/-34.4)	↓7.5
Falcon-3	1B	-3.3 (-3.3/0.0)	↓1.0	-3.3 (-0.0/-3.3)	↓3.0	-51.1 (-45.0/-6.1)	↓4.0	-36.7 (-31.1/-5.6)	↓4.0	-22.6 (-18.8/-3.8)	↓1.0
	3B	-6.7 (-0.0/-6.7)	↓4.0	-6.7 (-0.0/-6.7)	↓7.5	-6.7 (-0.6/-6.1)	0.0	-1.9 (-0.2/-1.7)	↑1.0	-24.7 (-4.1/-20.5)	↑1.0
	7B	-6.7 (-0.0/-6.7)	↑4.5	-6.7 (-0.0/-6.7)	↑3.5	-5.8 (-0.0/-5.8)	↑5.0	-0.2 (-0.1/-0.1)	↑2.0	-24.9 (-0.8/-24.1)	↑3.0
Gemma-3	1B	-6.7 (-6.7/0.0)	↓4.0	-3.3 (-3.3/0.0)	↓3.0	-10.8 (-5.2/-5.7)	↓1.0	-6.7 (-5.5/-1.3)	↑2.0	-28.7 (-13.3/-15.5)	0.0
	4B	-6.7 (-0.0/-6.7)	↑3.5	-16.7 (-3.3/-13.3)	↓3.5	-7.2 (-0.0/-7.1)	↓4.5	-1.3 (-0.3/-1.0)	↑1.0	-27.3 (-2.5/-24.8)	↑3.0
	12B	-6.7 (-3.3/-3.3)	↑9.0	-6.7 (-0.0/-6.7)	↑5.5	-7.2 (-0.0/-7.2)	↓6.5	-1.1 (-0.1/-1.0)	↑2.0	-29.9 (-3.6/-26.3)	↓2.0
	27B	-10.0 (-3.3/-6.7)	↑6.5	-6.7 (-0.0/-6.7)	↑4.5	-7.5 (-0.1/-7.4)	↓10.0	-0.8 (-0.2/-0.6)	0.0	-29.0 (-2.0/-27.0)	↑1.0
LFM-2	0.35B	0.0 (-0.0/0.0)	↑4.0	0.0 (-0.0/0.0)	↑5.0	-19.5 (-1.7/-17.7)	0.0	-22.8 (-1.2/-21.6)	↓1.0	-24.0 (-14.0/-10.0)	↓1.0
	0.7B	0.0 (-0.0/0.0)	↑4.0	0.0 (-0.0/0.0)	↑5.0	-5.3 (-0.8/-4.5)	↑3.0	-1.4 (-1.1/-0.3)	↑1.0	-16.8 (-7.5/-9.3)	↑3.0
	1.2B	-10.0 (-3.3/-6.7)	↓7.5	-3.3 (-0.0/-3.3)	↓3.0	-8.5 (-3.6/-4.9)	↑1.0	-7.1 (-6.7/-0.3)	↑1.0	-20.6 (-5.0/-15.6)	↑3.0
	2.6B	-13.3 (-3.3/-10.0)	↑5.0	-13.3 (-10.0/-3.3)	↓3.0	-4.7 (-0.0/-4.6)	↑12.0	-0.6 (-0.1/-0.5)	↑1.0	-30.6 (-6.9/-23.7)	↑2.0
Llama-3	1B	-3.3 (-3.3/0.0)	↓1.0	-3.3 (-3.3/0.0)	↓3.0	-15.8 (-14.5/-1.3)	0.0	-8.7 (-8.7/-0.0)	↑2.0	-28.1 (-25.5/-2.7)	↓2.0
	3B	-6.7 (-6.7/0.0)	↑4.5	-3.3 (-0.0/-3.3)	↓3.0	-4.7 (-1.2/-3.5)	↑1.0	-0.5 (-0.4/-0.1)	↑3.0	-19.6 (-4.5/-15.1)	↑3.0
	8B	0.0 (-0.0/0.0)	↑9.5	0.0 (-0.0/0.0)	↑5.0	-6.7 (-1.6/-5.1)	↑2.0	-0.5 (-0.2/-0.2)	↑3.0	-19.8 (-2.4/-17.5)	↑1.5
	70B	-33.3 (-33.3/0.0)	↓17.0	-10.0 (-10.0/0.0)	↓9.5	-11.4 (-5.9/-5.5)	↓6.0	-24.1 (-24.2/+0.1)	↓18.0	-73.1 (-71.5/-1.6)	↓16.0
Ministral	3B	-36.7 (-36.7/-0.0)	↓11.5	-20.0 (-20.0/0.0)	↓5.0	-6.3 (-0.4/-5.9)	↑1.0	-0.6 (-0.3/-0.3)	↑2.0	-34.9 (-14.4/-20.5)	↓3.0
	8B	-40.0 (-40.0/0.0)	↓9.0	-36.7 (-36.7/-0.0)	↓9.5	-6.1 (-0.4/-5.7)	↑5.0	-1.9 (-0.9/-1.0)	0.0	-33.7 (-10.7/-23.0)	↓3.0
	14B	-50.0 (-50.0/0.0)	↓16.5	-33.3 (-33.3/-0.0)	↓8.5	-5.9 (-0.2/-5.7)	↑3.0	-1.1 (-0.7/-0.5)	↑0.5	-37.9 (-15.9/-22.0)	↓6.5
OLMo-3	7B	-30.0 (-20.0/-10.0)	↓5.5	-13.3 (-13.3/0.0)	↑3.0	-6.4 (-0.1/-6.3)	↑0.5	-0.7 (-0.2/-0.5)	↑1.0	-30.3 (-5.5/-24.8)	↓0.5
	32B	-36.7 (-36.7/-0.0)	↓6.0	-23.3 (-23.3/-0.0)	↓2.0	-5.9 (-0.0/-5.9)	↑2.0	-0.8 (-0.5/-0.3)	↓1.0	-31.7 (-9.5/-22.1)	↓3.0
Phi-4	3.6B	-6.7 (-6.7/0.0)	↓4.0	-3.3 (-3.3/-0.0)	↑2.5	-5.6 (-0.0/-5.6)	↑6.0	-1.5 (-0.0/-1.5)	0.0	-28.7 (-9.8/-18.8)	↑1.0
	14B	-16.7 (-10.0/-6.7)	↑4.5	-6.7 (-6.7/0.0)	↑3.5	-6.7 (-0.0/-6.7)	↓3.5	-1.3 (-0.0/-1.3)	↑1.0	-28.3 (-1.5/-26.8)	↑2.0
Qwen-3	0.6B	-10.0 (-10.0/0.0)	↑0.5	-3.3 (-3.3/0.0)	↓3.0	-4.3 (-0.1/-4.2)	↑3.0	-3.4 (-0.5/-3.0)	↑1.0	-33.0 (-15.4/-17.7)	↓4.0
	4B	-13.3 (-13.3/-0.0)	↑3.5	-10.0 (-10.0/0.0)	↑4.5	-7.3 (-0.1/-7.2)	↓5.0	-1.1 (-0.4/-0.7)	0.0	-28.4 (-2.8/-25.6)	↑2.0
	8B	-16.7 (-16.7/0.0)	↑3.5	-23.3 (-20.0/-3.3)	↓5.0	-7.0 (-0.2/-6.8)	↓3.0	-1.4 (-0.0/-1.4)	↑1.5	-27.7 (-2.7/-25.0)	↑2.0
	14B	-20.0 (-16.7/-3.3)	↓1.0	-20.0 (-20.0/0.0)	↓1.0	-6.5 (-0.0/-6.5)	↑1.0	-0.2 (-0.0/-0.2)	↑3.0	-28.3 (-2.2/-26.1)	↑2.0
	32B	-10.0 (-6.7/-3.3)	↑5.5	-6.7 (-3.3/-3.3)	↑5.5	-5.8 (-0.0/-5.8)	0.0	-0.8 (-0.4/-0.5)	↓1.0	-28.3 (-2.5/-25.8)	↑4.0
SmolLM-2/3	0.135B	0.0 (-0.0/0.0)	↑4.0	0.0 (-0.0/0.0)	↑5.0	-3.1 (-3.0/-0.1)	0.0	-0.1 (-0.6/+0.5)	0.0	-1.7 (-1.7/-0.0)	0.0
	0.36B	0.0 (-0.0/0.0)	↑4.0	0.0 (-0.0/0.0)	↑5.0	-3.3 (-2.4/-1.0)	↑1.0	-1.7 (-1.1/-0.6)	0.0	-4.0 (-3.2/-0.8)	↑1.0
	1.7B	0.0 (-0.0/0.0)	↑10.0	0.0 (-0.0/0.0)	↑5.0	-4.7 (-1.8/-2.9)	↑2.0	-2.4 (-1.7/-0.6)	↑2.0	-14.2 (-7.9/-6.3)	↑3.0
	3B	-10.0 (-3.3/-6.7)	↑0.5	-3.3 (-3.3/0.0)	↑5.5	-5.7 (-0.3/-5.4)	↑2.0	-0.4 (-0.2/-0.2)	↑2.0	-33.2 (-12.6/-20.6)	0.0

Table 18 | Effect of regex-based evaluation on performance measurement for open-form math benchmarks.

D.3. BERT-as-a-Judge vs. Regex

In this section, we provide a detailed, unaggregated comparison of regex-based evaluation and BERT-as-a-Judge, reported by model and task. The results are based on the default BERT-as-a-Judge configuration, trained on 1M question-candidate-reference triplets, with candidates generated under the soft-constraint instruction (Appendix A).

Family	Size	ARC Challenge	ARC Easy	GPQA	MMLU	MMLU Pro	Truthful QA
Apertus	8B	99.4	99.6	92.4	98.2	96.7	99.1
	70B	99.5	99.7	95.1	98.4	96.6	99.0
EuroLLM	1.7B	99.5	99.7	92.4	97.0	88.5	83.0
	9B	99.9	100.0	94.9	99.3	97.5	99.8
	22B	99.7	99.9	94.4	98.4	97.2	99.9
Falcon-3	1B	98.4	98.9	92.4	97.9	95.0	97.6
	3B	99.7	99.7	94.6	99.2	97.1	99.8
	7B	99.9	99.9	96.2	99.5	97.7	99.6
Gemma-3	1B	99.4	99.8	90.6	97.5	96.5	99.5
	4B	99.8	99.9	96.0	98.3	97.0	99.8
	12B	99.6	99.9	94.6	99.1	97.6	99.4
	27B	99.7	99.8	95.5	98.9	98.0	99.8
LFM-2	0.35B	98.8	98.6	89.3	95.2	93.2	93.5
	0.7B	99.1	99.2	91.5	96.8	96.1	98.2
	1.2B	98.7	99.4	94.4	97.5	96.6	98.2
	2.6B	99.5	99.7	94.9	98.5	96.1	99.0
Llama-3	1B	99.1	99.5	91.5	97.7	96.5	99.1
	3B	99.5	99.7	93.5	98.0	96.5	99.1
	8B	99.9	99.8	94.0	98.9	97.2	99.0
	70B	99.7	99.8	94.4	99.0	97.6	98.4
Ministral-3	3B	99.5	99.8	90.4	98.7	94.0	99.3
	8B	99.7	99.8	93.5	99.1	96.3	99.3
	14B	99.7	99.9	94.2	99.2	97.2	99.0
OLMo-3	7B	99.4	99.8	92.6	98.7	94.1	99.6
	32B	99.5	99.9	77.7	99.2	91.4	99.8
Phi-4	3.6B	99.7	99.9	94.2	99.4	97.8	99.6
	14B	99.9	100.0	96.0	99.2	98.0	99.0
Qwen-3	0.6B	99.8	99.9	95.8	99.1	97.4	100.0
	4B	99.4	99.7	95.5	98.8	97.5	99.6
	8B	99.0	99.5	94.2	98.8	97.2	99.4
	14B	99.7	99.9	94.0	99.2	97.8	99.6
	32B	99.4	99.7	96.7	99.1	97.9	99.5
SmolLM-2/3	0.135B	97.6	98.9	88.6	96.7	97.0	96.7
	0.36B	99.5	99.4	97.3	98.7	98.1	98.7
	1.7B	99.9	99.9	97.5	99.5	98.7	99.9
	3B	99.5	99.7	94.4	98.6	97.0	98.8

Table 19 | BERT-as-a-Judge assessment accuracy on multiple-choice benchmarks.

Family	Size	ARC Challenge	ARC Easy	GPQA	MMLU	MMLU Pro	Truthful QA
Apertus	8B	99.0	99.2	91.5	96.4	96.2	97.8
	70B	97.7	97.9	93.8	94.9	94.5	96.2
EuroLLM	1.7B	70.2	70.8	71.7	72.6	77.6	71.0
	9B	99.5	99.9	89.7	98.0	91.3	99.8
	22B	91.2	92.1	85.3	85.8	87.3	93.5
Falcon-3	1B	73.3	66.8	84.8	84.3	85.8	93.1
	3B	96.6	97.7	93.3	95.5	91.1	98.7
	7B	94.6	93.3	90.4	92.8	91.8	97.8
Gemma-3	1B	98.7	98.9	86.4	95.3	94.1	99.5
	4B	99.8	99.9	95.8	98.2	96.9	99.8
	12B	99.6	99.9	94.2	99.1	97.6	99.4
	27B	99.7	99.8	95.8	98.9	98.1	99.8
LFM-2	0.35B	63.9	49.8	87.9	69.6	84.7	79.4
	0.7B	82.5	78.0	86.4	75.4	84.8	92.4
	1.2B	92.1	94.2	89.7	87.5	85.1	87.6
	2.6B	99.2	99.4	91.1	97.5	94.5	96.9
Llama-3	1B	96.9	97.6	90.4	95.9	94.7	97.8
	3B	84.2	86.4	89.7	79.9	83.4	83.8
	8B	61.8	67.3	90.4	72.7	84.3	76.7
	70B	99.7	99.8	92.6	98.0	95.6	97.4
Ministral-3	3B	98.0	99.1	88.6	97.3	91.8	97.3
	8B	95.5	97.3	89.5	91.5	87.5	90.5
	14B	98.2	98.9	89.3	94.2	88.6	90.5
OLMo-3	7B	99.2	99.8	91.1	98.2	91.4	99.6
	32B	99.5	99.9	74.1	98.0	86.9	99.5
Phi-4	3.6B	99.2	99.1	94.2	96.2	95.9	99.1
	14B	93.3	92.0	86.4	87.0	92.4	88.5
Qwen-3	0.6B	68.8	56.7	83.9	80.0	89.7	91.6
	4B	98.6	99.5	90.6	95.2	90.3	99.6
	8B	96.1	98.1	85.7	91.2	84.1	94.7
	14B	83.8	88.7	72.5	78.1	69.3	89.2
	32B	80.0	86.5	60.9	71.7	62.0	93.5
SmolLM-2/3	0.135B	76.8	76.0	79.2	77.3	88.1	81.8
	0.36B	76.1	73.5	78.3	75.8	90.6	77.7
	1.7B	42.8	20.8	71.4	55.1	84.1	81.6
	3B	99.0	99.2	88.6	95.6	93.7	96.3

Table 20 | Regex assessment accuracy on multiple-choice benchmarks.

Family	Size	CoQA	DROP	HotpotQA	SQuAD-v2
Apertus	8B	90.2	89.1	91.9	89.9
	70B	87.4	89.8	91.8	89.0
EuroLLM	1.7B	91.4	91.3	92.1	91.2
	9B	90.0	89.6	93.5	89.6
	22B	90.4	89.2	93.3	89.4
Falcon-3	1B	90.4	90.2	90.4	91.5
	3B	88.6	89.3	89.0	89.0
	7B	92.8	92.2	93.8	89.1
Gemma-3	1B	84.8	90.1	91.0	91.1
	4B	75.4	83.5	88.9	87.8
	12B	90.6	85.2	91.8	85.2
	27B	87.2	85.4	88.7	83.9
LFM-2	0.35B	88.8	90.4	90.7	92.1
	0.7B	83.4	89.0	85.2	92.1
	1.2B	91.4	90.5	91.5	91.5
	2.6B	85.6	85.2	88.8	88.9
Llama-3	1B	91.6	89.4	94.5	92.7
	3B	88.4	88.8	91.1	89.7
	8B	88.4	87.8	92.8	88.3
	70B	89.8	89.0	90.5	87.3
Ministral-3	3B	75.6	85.0	86.4	87.7
	8B	87.2	87.1	89.9	87.5
	14B	88.6	89.3	91.0	87.7
OLMo-3	7B	92.2	85.0	92.1	88.2
	32B	92.0	81.7	91.5	87.3
Phi-4	3.6B	90.0	90.5	90.8	89.8
	14B	82.2	89.2	89.4	88.2
Qwen-3	0.6B	86.0	90.5	92.9	91.5
	4B	85.4	88.8	91.4	90.2
	8B	91.6	84.6	90.5	89.0
	14B	93.0	90.6	93.5	87.3
	32B	91.2	89.2	92.2	88.4
SmolLM-2/3	0.135B	86.8	92.0	88.3	91.1
	0.36B	85.2	89.7	90.1	91.2
	1.7B	91.8	93.5	93.8	92.0
	3B	84.8	88.7	88.1	89.9

Table 21 | BERT-as-a-Judge assessment accuracy on context extraction benchmarks.

Family	Size	CoQA	DROP	HotpotQA	SQuAD-v2
Apertus	8B	67.0	75.7	73.7	78.3
	70B	56.6	76.3	72.5	72.8
EuroLLM	1.7B	49.2	80.0	60.5	69.0
	9B	70.8	79.9	80.5	76.1
	22B	72.4	73.0	80.2	69.4
Falcon-3	1B	55.4	85.1	77.5	78.8
	3B	61.6	79.2	71.3	76.1
	7B	77.4	83.9	83.8	80.1
Gemma-3	1B	75.0	88.8	83.4	82.9
	4B	54.6	70.3	71.2	69.5
	12B	68.8	66.9	76.0	69.0
	27B	68.4	67.4	69.6	68.3
LFM-2	0.35B	76.0	90.2	84.0	86.0
	0.7B	62.0	86.6	78.1	84.2
	1.2B	80.4	86.3	82.9	84.3
	2.6B	66.4	69.7	72.2	64.7
Llama-3	1B	66.4	75.1	74.8	79.0
	3B	56.8	79.4	72.8	69.6
	8B	74.0	77.1	79.9	76.5
	70B	78.4	81.1	82.3	70.2
Ministral-3	3B	62.8	65.5	66.6	67.5
	8B	67.2	63.8	71.0	64.5
	14B	70.6	68.2	73.0	62.3
OLMo-3	7B	78.4	74.8	83.3	71.8
	32B	72.0	70.3	78.9	62.5
Phi-4	3.6B	56.8	75.0	66.5	63.4
	14B	63.2	66.9	68.7	53.0
Qwen-3	0.6B	70.4	88.9	86.6	86.7
	4B	71.4	76.8	81.0	73.2
	8B	67.8	62.5	75.9	64.2
	14B	77.8	79.2	83.3	67.5
	32B	75.4	73.0	79.1	63.2
SmolLM-2/3	0.135B	60.8	87.4	67.0	75.6
	0.36B	45.0	81.8	61.4	71.3
	1.7B	72.4	90.5	81.5	80.1
	3B	62.8	76.9	71.5	73.0

Table 22 | Regex assessment accuracy on context extraction benchmarks.

Family	Size	AIME24	AIME25	ASDiv	GSM8K	Math
Apertus	8B	100.0	100.0	95.4	98.6	93.3
	70B	100.0	100.0	94.5	99.1	92.6
EuroLLM	1.7B	100.0	100.0	97.8	97.6	97.0
	9B	90.0	96.7	95.4	98.6	91.8
	22B	80.0	93.3	95.1	98.9	90.2
Falcon-3	1B	96.7	96.7	95.6	97.9	90.6
	3B	96.7	93.3	95.4	99.4	93.3
	7B	93.3	93.3	95.2	99.8	93.9
Gemma-3	1B	93.3	100.0	93.5	96.5	90.4
	4B	90.0	83.3	95.2	98.6	93.6
	12B	96.7	93.3	94.8	98.9	94.9
	27B	90.0	93.3	95.1	99.2	95.4
LFM-2	0.35B	100.0	100.0	96.2	97.2	92.3
	0.7B	100.0	100.0	94.8	98.9	93.1
	1.2B	90.0	96.7	94.6	98.9	92.9
	2.6B	90.0	90.0	95.2	99.2	94.9
Llama-3	1B	96.7	96.7	93.7	97.6	92.9
	3B	96.7	96.7	94.9	99.2	93.0
	8B	100.0	100.0	94.0	99.0	92.5
	70B	93.3	93.3	95.3	99.6	94.5
Ministral-3	3B	66.7	56.7	94.4	99.2	90.3
	8B	66.7	63.3	94.9	99.8	92.8
	14B	70.0	60.0	95.3	99.6	92.5
OLMo-3	7B	80.0	80.0	95.5	98.9	94.7
	32B	60.0	66.7	95.6	99.3	93.6
Phi-4	3.6B	96.7	100.0	95.4	99.5	94.6
	14B	83.3	100.0	95.2	99.7	95.9
Qwen-3	0.6B	86.7	100.0	95.7	97.0	92.4
	4B	86.7	93.3	94.9	99.2	95.3
	8B	83.3	83.3	95.0	99.4	95.9
	14B	80.0	80.0	95.4	99.7	95.4
	32B	93.3	93.3	95.3	99.6	95.8
SmolLM-2/3	0.135B	100.0	100.0	98.3	98.6	97.7
	0.36B	100.0	100.0	98.0	98.9	95.7
	1.7B	100.0	100.0	95.8	98.0	93.1
	3B	93.3	96.7	95.2	99.0	94.6

Table 23 | BERT-as-a-Judge assessment accuracy on open-form math benchmarks.

Family	Size	AIME24	AIME25	ASDiv	GSM8K	Math
Apertus	8B	100.0	100.0	94.0	99.2	88.4
	70B	100.0	100.0	92.9	99.7	85.0
EuroLLM	1.7B	100.0	100.0	96.4	97.5	96.5
	9B	90.0	96.7	70.2	72.5	69.4
	22B	80.0	93.3	66.9	70.3	61.1
Falcon-3	1B	96.7	96.7	48.6	63.3	77.0
	3B	93.3	93.3	92.5	97.8	74.7
	7B	93.3	93.3	92.5	99.8	74.5
Gemma-3	1B	93.3	96.7	88.7	93.1	71.1
	4B	93.3	83.3	91.5	98.6	72.3
	12B	93.3	93.3	91.9	98.9	69.8
	27B	90.0	93.3	91.5	99.2	70.6
LFM-2	0.35B	100.0	100.0	79.8	76.6	75.9
	0.7B	100.0	100.0	93.1	97.9	81.8
	1.2B	90.0	96.7	90.5	92.6	78.1
	2.6B	86.7	86.7	92.8	99.4	69.2
Llama-3	1B	96.7	96.7	83.7	90.8	71.8
	3B	93.3	96.7	93.8	99.4	79.9
	8B	100.0	100.0	91.9	99.4	79.9
	70B	66.7	90.0	87.3	75.7	26.9
Ministral-3	3B	63.3	80.0	91.4	99.4	64.9
	8B	60.0	63.3	91.7	98.1	66.0
	14B	50.0	66.7	91.8	98.9	62.0
OLMo-3	7B	70.0	86.7	92.2	99.3	69.3
	32B	63.3	76.7	92.4	99.2	68.3
Phi-4	3.6B	93.3	96.7	92.2	98.2	70.7
	14B	83.3	93.3	92.1	98.7	71.3
Qwen-3	0.6B	90.0	96.7	93.9	96.4	66.8
	4B	86.7	90.0	91.3	98.9	71.3
	8B	83.3	76.7	91.7	98.5	71.8
	14B	80.0	80.0	91.9	99.6	71.3
	32B	90.0	93.3	91.8	99.2	71.3
SmolLM-2/3	0.135B	100.0	100.0	95.9	98.7	97.5
	0.36B	100.0	100.0	95.5	98.0	94.6
	1.7B	100.0	100.0	94.0	96.9	85.1
	3B	90.0	96.7	92.6	99.6	66.3

Table 24 | Regex assessment accuracy on open-form math benchmarks.

E. Illustrative Examples

In this section, we present examples of common failure cases in regex-based evaluation. Table 25 illustrates a case where parsing fails despite the model producing a correct answer, while Table 26 shows a case where parsing succeeds but additional formatting introduced by the model prevents correct assessment against the reference.

Question	Answer the question based on the provided context. Context: A psychological identity relates to self-image (one's mental model of oneself), self-esteem, and individuality. Consequently, Weinreich gives the definition "A person's identity is defined as the totality of one's self-construal, in which how one construes oneself in the present expresses the continuity between how one construes oneself as one was in the past and how one construes oneself as one aspires to be in the future"; this allows for definitions of aspects of identity, such as: "One's ethnic identity is defined as that part of the totality of one's self-construal made up of those dimensions that express the continuity between one's construal of past ancestry and one's future aspirations in relation to ethnicity" (Weinreich, 1986a). The description or representation of individual and group identity is a central task for psychologists, sociologists and anthropologists and those of other disciplines where "identity" needs to be mapped and defined. How should one describe the identity of another, in ways which encompass both their idiosyncratic qualities and their group memberships or identifications, both of which can shift according to circumstance? Following on from the work of Kelly, Erikson, Tajfel and others Weinreich's Identity Structure Analysis (ISA), is "a structural representation of the individual's existential experience, in which the relationships between self and other agents are organised in relatively stable structures over time ... with the emphasis on the socio-cultural milieu in which self relates to other agents and institutions" (Weinreich and Saunderson, (eds) 2003, p1). Using constructs drawn from the salient discourses of the individual, the group and cultural norms, the practical operationalisation of ISA provides a methodology that maps how these are used by the individual, applied across time and milieus by the "situated self" to appraise self and other agents and institutions (for example, resulting in the individual's evaluation of self and significant others and institutions).[citation needed] Question: What is the ISA? Conclude your response with "Final answer: X", where X is the exact span from the context that answers the question.
Reference	Identity Structure Analysis (ISA)
Candidate	The ISA, or Identity Structure Analysis, is a methodology developed by Weinreich that maps how individuals perceive themselves and others in relation to their social environment, including their own group memberships and identifications, as well as the cultural norms and discourses that shape these perceptions.

Table 25 | Example 488 from the CoQA dataset, along with the answer from Llama-3 1B

Question	We have a triangle $\triangle ABC$ such that $AB = 6,$ $BC = 8,$ and $CA = 10.$ If AD is an angle bisector such that D is on $BC,$ then find the value of $AD^2.$ Conclude your response with "Final answer: X", where X is the computed solution.
Reference	45
Candidate	To solve for \(AD^2 \) in \(\triangle ABC\) where \(AB = 6 \), \(BC = 8 \), and \(CA = 10 \), and \(AD \) is the angle bisector of \(\angle BAC\) with \(D \) on \(BC \), we can use the Angle Bisector Theorem and the Angle Bisector Length Formula. ### Step-by-Step Solution: 1. **Angle Bisector Theorem:** The Angle Bisector Theorem states that the angle bisector divides the opposite side into segments proportional to the adjacent sides. Therefore, if \(D \) divides \(BC \) into segments \(BD \) and \(DC \), then: \[\frac{BD}{DC} = \frac{AB}{AC} = \frac{6}{10} = \frac{3}{5} \] [...] 4. **Calculate \(AD^2 \):** \[AD^2 = 6 \cdot 10 \left(1 - \frac{8^2}{(6 + 10)^2} \right) \] \[AD^2 = 60 \left(1 - \frac{64}{16^2} \right) \] \[AD^2 = 60 \left(1 - \frac{64}{256} \right) \] \[AD^2 = 60 \left(1 - \frac{1}{4} \right) \] \[AD^2 = 60 \left(\frac{3}{4} \right) \] \[AD^2 = 60 \cdot 0.75 \] \[AD^2 = 45 \] Final answer: \(\boxed{45} \)

Table 26 | Example 2070 from the MATH dataset, along with the answer from Falcon-3 7B