Test-Time Scaling with Repeated Sampling Improves Multilingual Text Generation

Anonymous ACL submission

Abstract

Inference-time scaling via repeated sampling has shown promise in reasoning tasks, but its effectiveness in multilingual generation remains underexplored. We evaluate this approach using perplexity- and reward-based verifiers on two multilingual benchmarks: the Aya Evaluation Suite and m-ArenaHard. Our results show consistent quality improvements, with gains exceeding 35% in some cases. While perplexity-based scoring is effective for open-ended prompts, only reward-based verifiers improve performance on tasks requiring reasoning (e.g., math, code). Our results demonstrate the broader utility of repeated sampling for multilingual text generation and underscore the importance of selecting right verifiers for the task.

1 Introduction

Scaling the inference-time compute has significantly improved LLM performance on coding, mathematics, and other reasoning tasks (e.g., Madaan et al., 2023; Wu et al., 2024; Snell et al., 2024; Brown et al., 2024; Bansal et al., 2024; Muennighoff et al., 2025). There are two broad strategies for test-time scaling: (1) with lengthy chains of thought using an LLM trained to *reason* longer (Guo et al., 2025; Muennighoff et al., 2025), or (2) by repeated sampling, where a verifier or a scorer model selects a final answer from generated samples (Brown et al., 2024; Snell et al., 2024).

Despite the success of test-time scaling in reasoning-heavy tasks, its effect on multilingual text generation remains largely unexplored. In this work, we investigate whether the *repeated sampling* strategy can improve multilingual generation, even for benchmarks that do not explicitly focus on reasoning, code, or mathematics. We employ two approaches for selecting the final output: (a) perplexity-based scoring using an autoregressive language model, and (b) reward scoring

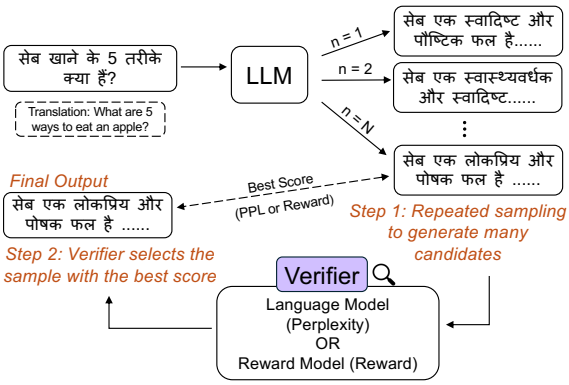

Figure 1: Repeated sampling procedure using a verifier to pick the final answer.

with pre-trained reward models. The perplexity-based method considers only the generated responses, selecting the output with the lowest perplexity score as a proxy for fluency. In contrast, the reward-based method evaluates both the prompt and the generated response together, scoring each prompt-response pair to determine the best output. Figure 1 illustrates this procedure.

Repeated sampling may allow models to explore a wider range of potential outputs. Consequently, in the case of multilingual generation, especially for low-resource languages, the likelihood of generating text that is both high quality and contextually appropriate increases. When model confidence is limited, e.g., for underrepresented languages, initial outputs may not be the best. By selecting from multiple generated candidates using criteria based on fluency (via perplexity) or prompt-response alignment (via reward models), we aim to identify outputs that are more consistent with human judgments or benchmark standards.

We show empirically that repeated sampling with verification improves multilingual text generation. Using two perplexity- and reward-based verifiers each, we observe that output quality improves with the number of samples. On the Aya Evaluation

Suite (Singh et al., 2024), a multilingual benchmark for open-ended generation, both verifier types yield substantial gains; a 2B-parameter Gemma-based reward model achieves up to 35% increase in win rates. However, for m-ArenaHard (Dang et al., 2024), a benchmark focused on programming and mathematics, only reward-based verifiers lead to measurable improvements.

Our contributions are twofold. (1) Ours is the first work to demonstrate that multilingual generation benefits from test-time scaling, highlighting the broader applicability of repeated sampling beyond traditional English-based reasoning tasks. (2) We show that verifier choice is critical: both perplexity- and reward-based scoring helps general multilingual generation, but only reward-based verifiers are effective for tasks requiring reasoning.

2 Experimental Setup

This section outlines our experimental setup, including details on the datasets, languages, LLMs, verifiers, and the evaluation protocol used.

Dataset and Languages. We use two datasets for evaluation. First is the Aya Evaluation Suite (Singh et al., 2024), that contains open-ended conversation-style, culturally grounded prompts for evaluating multilingual LLM generation. We conduct evaluations on nine languages: Arabic, English, Hindi, Punjabi, Portuguese, Russian, Telugu, Turkish, and Chinese.

The second dataset is m-ArenaHard (Dang et al., 2024), the multilingual variant of ArenaHard-Auto (Li et al., 2024). This dataset contains the translated prompts from the ArenaHard-Auto benchmark. With this dataset, we perform evaluation in seven languages: Arabic, Czech, English, Hindi, Japanese, Portuguese, and Vietnamese.

Verifiers. We employ two types of verifiers to score and select generations from repeated sampling. For perplexity-based verification, we select sample that has the lowest perplexity and use the pre-trained variants of `LLaMA-3.1-8B` (Grattafiori et al., 2024) and `Gemma-2B` models (Team, 2024). For reward-based scoring, we use two fine-tuned reward models that are based on the above pre-trained ones: `URM-LLaMA-3.1-8B` (Lou et al., 2024), and `GRM-Gemma-2B-rewardmodel-ft` (Yang et al., 2024b). Both reward models rank high on the RewardBench benchmark (Lambert et al., 2024) and are among the best models in their respective parameter ranges. Note that while the underlying pre-trained models are multilingual, the reward models have only been trained on the English preference data. Appendix A.2 elaborates on our verifiers.

LLMs Evaluated. We evaluate commonly used open-weight multilingual LLMs at both their large and small parameter scales. Specifically, we use Qwen-2.5 (Yang et al., 2024a) at 14B, 72B parameter scales, Llama-3 at 8B and 70B scales (Grattafiori et al., 2024), and aya-expanse at 8B and 32B (Dang et al., 2024).

Evaluation Protocol. We follow prior work and adopt the LLM-as-a-Judge framework for evaluating multilingual generation (Dang et al., 2024), based on MT-Bench (Zheng et al., 2023). The core evaluation metric is the win/loss rate — i.e., how often a model's response is judged better or worse than a baseline. We report the difference (*delta*) between win and loss rates, averaged across all languages. All evaluations use `gemini-2.0-flash` as the judge, which we choose because of its cost efficiency and strong multilingual performance.

Our baseline consists of single-sample generations without repeated sampling, using temperature-based decoding. Due to its stochastic nature, the baseline output may vary across runs, introducing randomness into the delta scores. Comparing against multiple baseline samples is impractically expensive. In a pilot study (fig. 5, appendix), we found that the variance across runs is small (< 2 percentage points), indicating that our findings are robust to baseline selection.

3 Results and Analysis

We plot the impact of number of samples (n) used for repeated sampling vs the average delta score achieved to show the scaling effect. We perform this evaluation for $n = 5, 10, 15, 20, 25$ and then scale it in steps of 25 after that (to $n = 50, 75, 100$).

Figure 2 shows the main scaling plots for the Aya Evaluation Suite and fig. 3 for the m-ArenaHard. For the latter, due to space constraints, we only show the plots for the larger models of each model family. Other plots are presented in appendix fig. 6.

1. Repeated sampling with verifiers improves multilingual generation. For Aya Evaluation Suite, across all model architectures, verifier types, and scales, repeated sampling during inference improves multilingual generation. This effect is evident even with a modest number of samples;

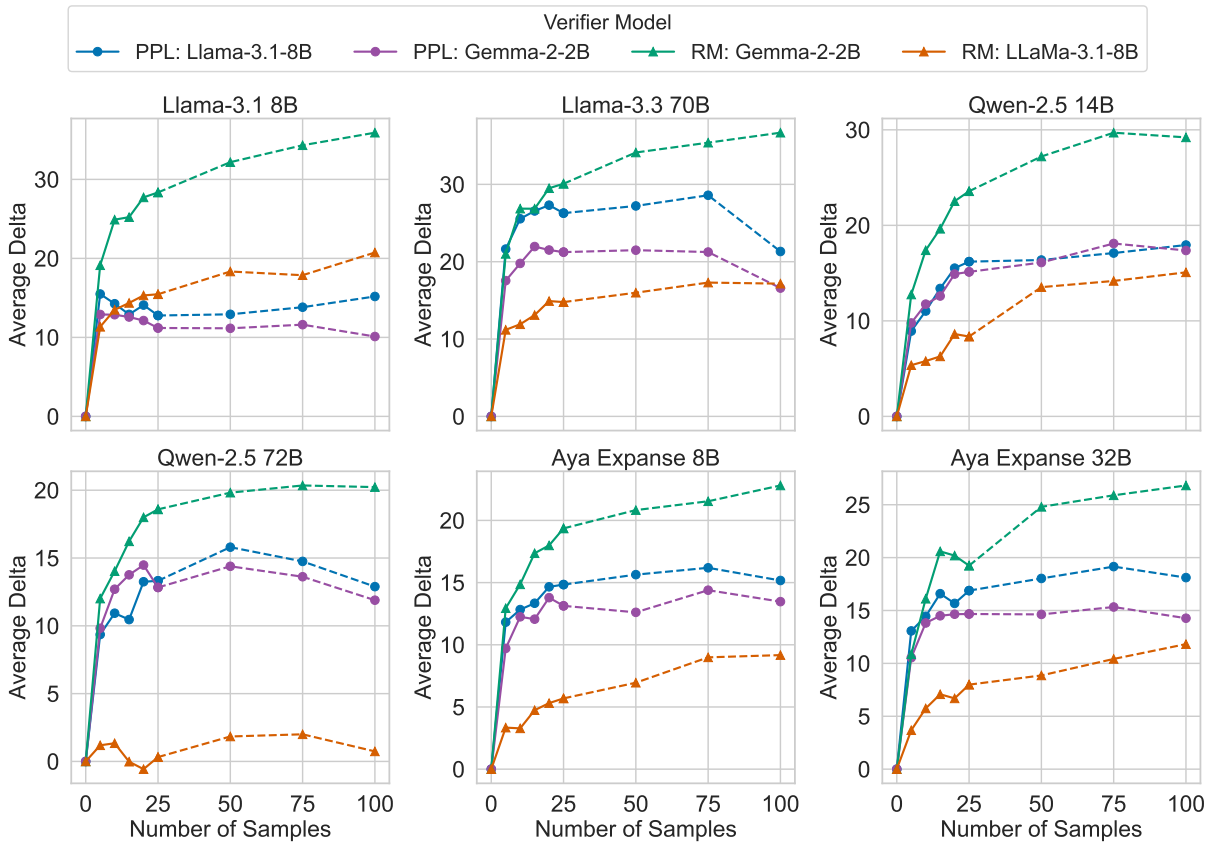

Figure 2: Test-time scaling with repeated sampling for Aya Evaluation Suite. The plots show the difference between win and loss rates (delta). We see that all verifiers — both perplexity-based (PPL) and reward-based (RM) — can improve generation quality.

we see significant performance gains as early as $n = 5$. While improvements continue with more samples, the marginal benefit begins to plateau around $n = 25$ for most models. An exception is the pairing of the `URM-LLaMA-3.1-8B` reward model with the Qwen-2.5 72B model, where gains are minimal. Yet, the consistent upward trend supports the value of repeated sampling for improving generation quality in multilingual settings.

Aya Evaluation Suite vs m-ArenaHard. For m-ArenaHard, only reward-based verifiers consistently improve performance. In fact, perplexity-based verification can even hurt generation quality with more samples (e.g., 70B-parameter Llama-3 using Gemma verifier). This observation aligns with the nature of the tasks: open-ended textual prompts predominate the Aya Evaluation Suite, where fluency correlates strongly with quality. In contrast, m-ArenaHard prompts demand domain-specific reasoning in mathematics or programming, where surface-level fluency is insufficient. Reward models, trained extensively on reasoning-intensive math/coding datasets, are better at distinguishing high-quality responses in these settings.

2. Llama-based verifiers improve Llama-based models. Interestingly, Llama-based verifiers can help select higher quality texts from their post-trained counterparts. This is unexpected, as post-trained models are typically optimized to align with either their own reward signals or their internal fluency preferences, suggesting they might better self-evaluate. Nonetheless, Llama verifiers appear well-suited to identifying strong outputs from related model families.

3. Gemma-based reward model delivers the highest gains. The reward model based on Gemma-2 delivers the highest gains across both benchmarks. At $n = 100$, all generation models improve by over 20%, with Llama-8B and Llama-70B achieving increases exceeding 35%. This is particularly notable given that the Gemma verifier has only 2B parameters. These results indicate that, under repeated sampling, even a relatively small reward model can substantially enhance the output quality of much larger language models. Impor-

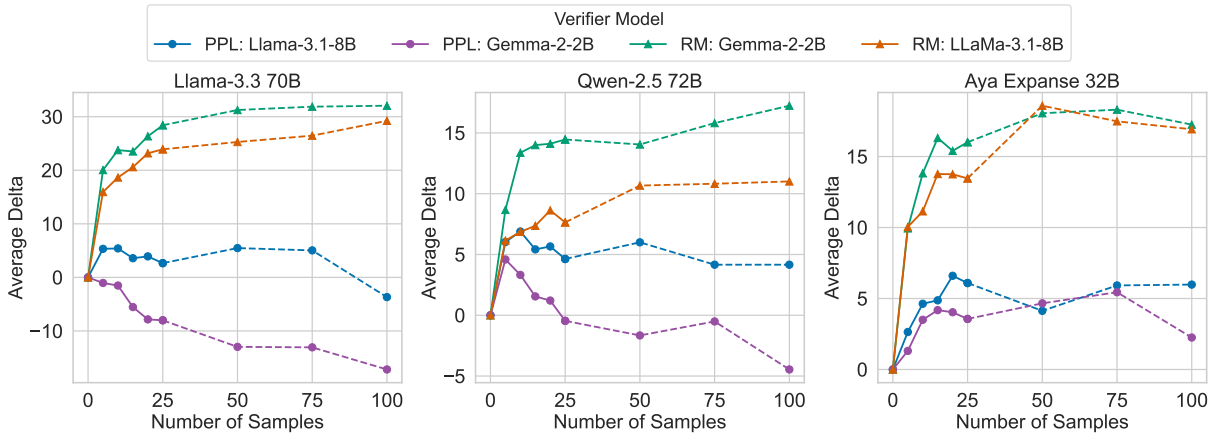

Figure 3: Test-time scaling with repeated sampling for m-ArenaHard. The plots show the difference between win and loss rates (delta). Only reward-model based verifiers improve generation quality.

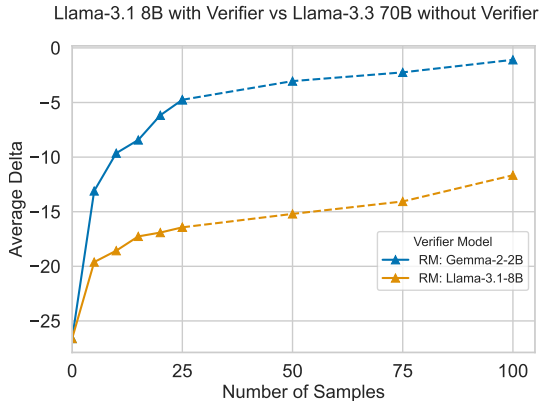

Figure 4: Training-time compute vs Test-time compute.

tantly, the verifier was trained exclusively on English prompt-response data; further improvements may be possible with multilingual training.

4. Verifier choice is critical. As observed on the m-ArenaHard benchmark, reward-based verifiers substantially enhance generation quality, whereas perplexity-based methods often degrade it. Thi highlights the importance of choosing the right verifier for the right task. Moreover, verifier performance can vary across languages, suggesting that language-specific factors influence effectiveness. See for instance the heatmaps provided in appendix fig. 7 and fig. 8.

For example, while the Gemma-based perplexity verifier appears more effective for Qwen models on Hindi, a Llama-based verifier yields stronger results on Arabic. These variations suggest that no single verifier might be universally optimal, and future work could explore strategies for selecting verifiers based on language, model architecture, or task type, and training language-agnostic verifiers.

Train-time Scaling vs Test-time Scaling. One question we ask is whether test-time scaling can compensate for the limitations of smaller models, potentially narrowing the gap with larger models trained using significantly more compute. To explore this, we compare the outputs of Llama-3 8B with repeated sampling (test-time scaling) against Llama-3 70B without sampling (train-time scaling only). As shown by the negative delta values in fig. 4, while repeated sampling does improve the smaller model's outputs, it does not match the quality achieved by the larger model, indicating that test-time scaling alone cannot fully substitute for increased training-time capacity. This suggests that while test-time scaling can enhance performance, especially for smaller models, it might not be a replacement for the representational capacity and capabilities gained through large-scale training.

4 Conclusion

This work establishes inference-time scaling via repeated sampling as a practical and effective strategy for improving multilingual text generation. Our analysis reveals that performance gains are robust across models, verifier types, and languages, with notable improvements achievable even for small-scale verifiers. Crucially, we find that verifier effectiveness can be task and language-dependent: perplexity-based methods work well for open-ended prompts but underperform on domains requiring structured reasoning. These findings highlight the importance of careful verifier selection and open up promising directions for future work, including adaptive strategies that tailor verifier choice to the characteristics of the input and the model.

Limitations

We identify two primary limitations in this work. First, the reward models employed are trained solely on English data. This choice reflects the limited availability of compact, high-performing multilingual reward models. Existing open-weight multilingual reward models, such as those in m-RewardBench, tend to be prohibitively large (e.g., Gemma-2 at 27B), making them impractical as lightweight verifiers in repeated sampling setups. Future work could address this by developing smaller, more efficient multilingual reward models.

Second, due to budget constraints, our evaluation uses a single baseline sample when computing win-rates, which may introduce variance in the results. Although our analysis shows that this variance is relatively small, its impact across different models and languages remains uncertain. Moreover, our use of LLM-as-a-Judge introduces an additional layer of potential bias, as the judgments reflect the preferences and limitations of the underlying judge model. While this approach is practical and widely adopted, it may not fully align with human evaluations, particularly in multilingual or culturally nuanced contexts.

References

Hritik Bansal, Arian Hosseini, Rishabh Agarwal, Vinh Q Tran, and Mehran Kazemi. 2024. Smaller, weaker, yet better: Training llm reasoners via compute-optimal sampling. *arXiv preprint arXiv:2408.16737*.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. 2024. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*.

John Dang, Shivalika Singh, Daniel D'souza, Arash Ahmadian, Alejandro Salamanca, Madeline Smith, Aidan Peppin, Sungjin Hong, Manoj Govindassamy, Terrence Zhao, and 1 others. 2024. Aya expanse: Combining research breakthroughs for a new multilingual frontier. *arXiv preprint arXiv:2412.04261*.

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

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

Srishti Gureja, Lester James V Miranda, Shayekh Bin Islam, Rishabh Maheshwary, Drishti Sharma, Gusti Winata, Nathan Lambert, Sebastian Ruder, Sara Hooker, and Marzieh Fadaee. 2024. M-rewardbench: Evaluating reward models in multilingual settings. *arXiv preprint arXiv:2410.15522*.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, and 1 others. 2024. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. 2024. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*.

Xingzhou Lou, Dong Yan, Wei Shen, Yuzi Yan, Jian Xie, and Junge Zhang. 2024. Uncertainty-aware reward model: Teaching reward models to know what is unknown. *arXiv preprint arXiv:2410.00847*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36:46534–46594.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*.

Shivalika Singh, Freddie Vargus, Daniel Dsouza, Börje F. Karlsson, Abinaya Mahendiran, Wei-Yin Ko, Herumb Shandilya, Jay Patel, Deividas Mataciunas, Laura OMahony, Mike Zhang, Ramith Hettiarachchi, Joseph Wilson, Marina Machado, Luisa Souza Moura, Dominik Krzemiński, Hakimeh Fadaei, Irem Ergün, Ifeoma Okoh, and 14 others. 2024. Aya dataset: An open-access collection for multilingual instruction tuning. *Preprint*, arXiv:2402.06619.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

Gemma Team. 2024. Gemma.

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. 2024. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models. *arXiv preprint arXiv:2408.00724*.

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

Rui Yang, Ruomeng Ding, Yong Lin, Huan Zhang, and Tong Zhang. 2024b. Regularizing hidden states enables learning generalizable reward model for llms. *arXiv preprint arXiv:2406.10216*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

A Other Experimental Details

We provide some additional experimental details regarding the models and the hyperparameters used in our evaluation.

A.1 Dataset and Languages.

We use the Aya Evaluation Suite for conducting all our experiments (Singh et al., 2024).[1] The dataset contains open-ended conversation-style, culturally grounded prompts for evaluating multilingual generation in LLMs. We demonstrate our results on nine languages, namely, Arabic (`ar`), English (`en`), Hindi (`hi`), Punjabi (`pa`), Portuguese (`pt`), Russian (`ru`), Telugu (`te`), Turkish (`tr`), and Chinese (`zh`). To get the Punjabi subset, we translate Hindi examples to Punjabi using Google Translate followed by manual post-editing. Second dataset is the m-ArenaHard dataset which is the multilingual variant of ArenaHard-Auto (Li et al., 2024). This dataset was released by Dang et al. (2024) and contains the translated prompts from the ArenaHard-Auto benchmark. With this dataset, we perform evaluation in seven languages: Arabic (`ar`), Czech (`cs`), English (`en`), Hindi (`hi`), Japanese (`ja`), Portuguese (`po`), and Vietnamese (`vi`). We sample the first 250 prompts for each language for evaluation for each of the datasets.

A.2 Verifiers

We employ two types of verifiers to score generations produced through repeated sampling. For perplexity-based verification, we use the LLaMA-3.1-8B and Gemma-2B models. These models are used in their pre-trained forms, as they serve solely to compute perplexity scores. For reward model-based scoring, we incorporate two classifier reward models that have been fine-tuned on top of these pre-trained architectures. The first is the `URM-LLaMA-3.1-8B` model [2], which achieves the highest overall score among all models with fewer than 8 billion parameters on the RewardBench benchmark (Lambert et al., 2024).[3] The second is the `GRM-Gemma-2B-rewardmodel-ft` [4], a reward model fine-tuned on the Gemma-2B backbone, and the leading performer in the sub-3B parameter category on RewardBench. Using reward models that share the same backbone as the perplexity models helps reduce confounding factors and allows for clearer interpretation of results in our experimental comparisons.

It is important to note that both reward models are trained exclusively on English-language data. We deliberately choose not to explore multilingual reward models for two main reasons. First, the best-performing open-weight multilingual reward models, such as Gemma-27B, are significantly larger in scale, which limits their practicality in repeated sampling scenarios. Second, the objective of this work is to demonstrate that simple, readily available verifiers can already yield meaningful improvements in multilingual generation. That said, developing a smaller yet effective multilingual reward model remains an interesting direction for future research. As an aside, prior work has shown that the English-only `URM-LLaMA-3.1-8B` model can be effective across multiple languages (Gureja et al., 2024).

A.3 Models and Hyperparameters.

We evaluate the most common multilingual, open-weight LLMs at both their large and small parameter scales. Specifically, we use Qwen-2.5 (Yang et al., 2024a) at 14B, 72B parameter scales, Llama-3 at 8B and 70B scales (Grattafiori et al., 2024) [5], and aya-expanse at 8B and 32B (Dang et al., 2024). We abbreviate these names in figures to make them more readable.

[1] https://huggingface.co/datasets/CohereLabs/aya_evaluation_suite

[2] https://huggingface.co/LxzGordon/URM-LLaMa-3.1-8B

[3] https://huggingface.co/spaces/allenai/reward-bench

[4] https://huggingface.co/Ray2333/GRM-Gemma-2B-rewardmodel-ft

[5] We use Llama-3.1 for the 8B model, and Llama-3.3 for the 70B model. Llama-3.3 does not have a corresponding 8B model and is known to be much superior to the Llama-3.1 70B model.

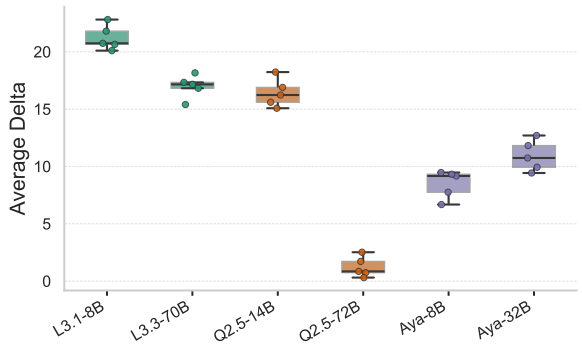

Figure 5: Impact of baseline used for win rate calculation. We use `gemini-2.0-flash` as the judge model and evaluate stability with a reward model as a verifier: `URM-LLaMa-3.1-8B`

Hyperparameters. During our preliminary experiments, we found that temperature and other sampling parameters like top-p did not degrade the generation quality. We therefore fixed the temperature to 0.8, and top-p to 0.95.

GPT-4o vs Gemini for Judge. We used Gemini's 2.0 flash model for calculating win/loss rates. For a subset of our experiments on Aya Evaluation Suite with Llama-based verifiers, we found that on average the delta scores from GPT-4o and Gemini were within 3.0% of each other. Gemini model is 25x cheaper than the GPT-4o model and therefore considering the scale of our evaluation, it made more sense to use the Gemini model.

What baseline response to use? We need to compare the response chosen by the verifier with a baseline sampled response (i.e. when there is no repeated sampling). Due to randomness in sampling, this comparison itself becomes non-deterministic. Of course, ideally, we want to perform the direct comparison with multiple different baseline responses. But this is not practical due to budget constraints.

We find that this does not have a significant impact. See fig. 5 for a box plot that shows the variability of the delta scores when comparing it with different baseline responses. Although there is some variance, but as can be clearly seen, it is within a small range (around 2 % points).

A.4 For Responsible NLP Checklist

We briefly describe details regarding the questions present in the responsible NLP checklist.

1. **B2: Discuss The License For Artifacts and B3: Artifact Use Consistent With Intended Use**: We have primarily used two dataset artifacts in this paper: Aya Evaluation Suite, and m-ArenaHard dataset. The intended use of both of these datasets was for evaluation and benchmarking which completely aligns with the goal of this paper.

   Additionally, we have only used the open-weight models that at the very least provide a permissive license for research evaluations we conduct. Usually, each model provider has their own license (e.g. llama-3.1 license) but all of them provide free and permissive use for academic research under the conditions that their work is properly cited.

2. **C1 Model Size And Budget** We mention model sizes in the section 2 when introducing the models we evaluate. We ran all evaluations on our university machines. The evaluations for smaller models were performed on a number of GPUs: A40, A6000, A100-40GB, and for the larger models we relied more heavily on A100-80 GB. We do not have an exact estimate of the number of GPU hours as they were spread across different clusters. But approximately, we used less than 2000GPU hours of compute.

3. **E1 Information About Use Of Ai Assistants** We primarily used GitHub Co-Pilot for coding assitance through the VS Code extension. Occasionally, we also relied on ChatGPT for code modifications through their web interface.

A.5 On Manual or Alternate Evaluation

While alternative evaluation strategies such as manual annotation or language-specific diagnostics can provide valuable insights, they are difficult to operationalize at scale, particularly in multilingual settings. Manual evaluation requires access to fluent speakers and calibrated annotators across a wide range of low-resource languages, which presents substantial logistical and methodological challenges. Moreover, automatic heuristics like language identification or lexical checks can often fail to capture the nuanced dimensions of generation quality, such as coherence, cultural alignment, or task relevance. Our use of the LLM-as-a-judge framework is practical and widely adopted alternative, and prior work has shown strong correlations between LLM judgments and human preferences

across multiple domains. Still, we acknowledge that relying on a single judge model introduces evaluation bias, especially when the model's proficiency varies across languages. A promising direction to mitigate this limitation is to aggregate judgments from multiple strong multilingual LLMs, leveraging inter-model agreement to increase evaluation reliability.

B Detailed Results and Plots

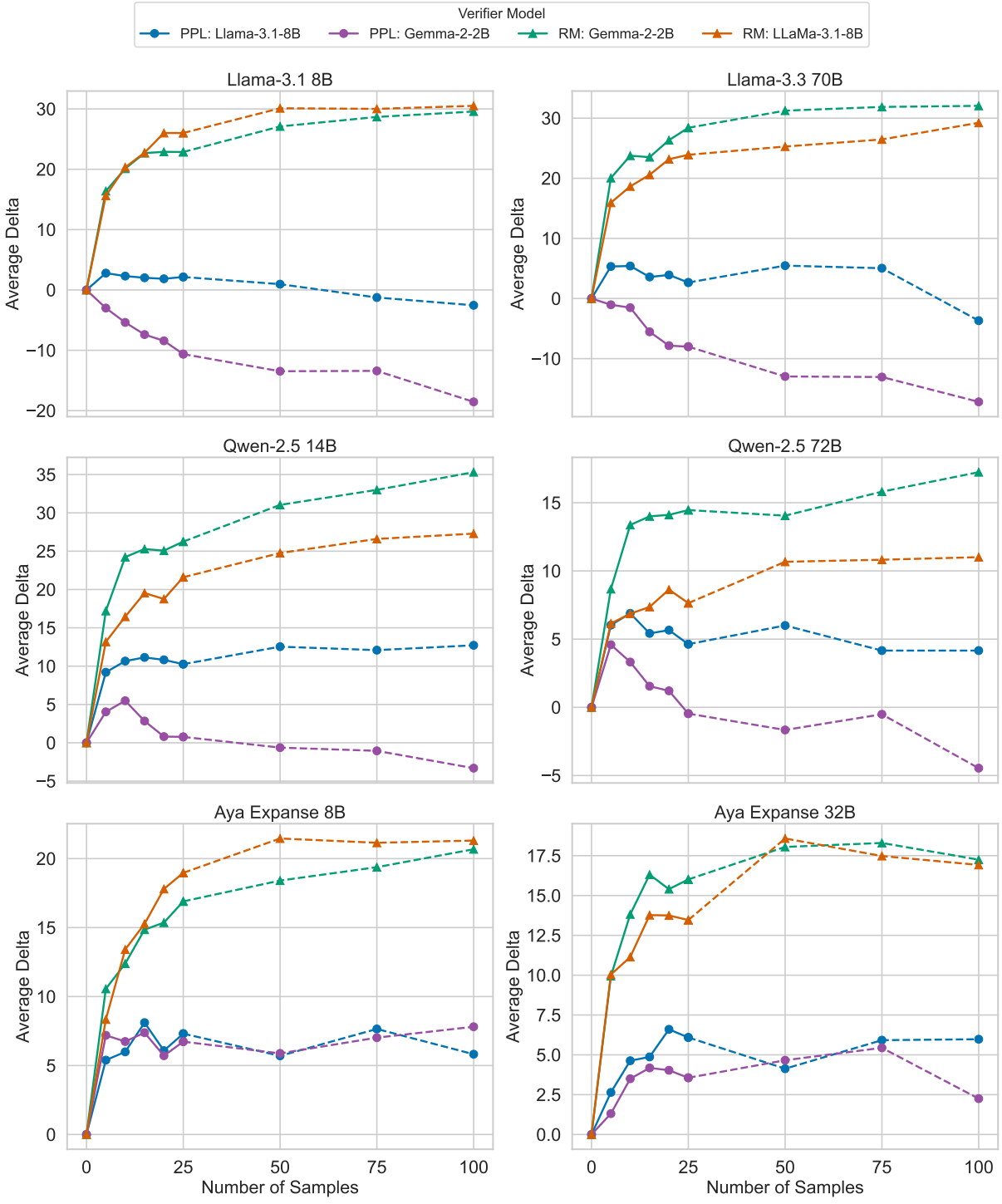

Figure 6: Inference time scaling with repeated sampling for multilingual generation. Results for the m-ArenaHard.

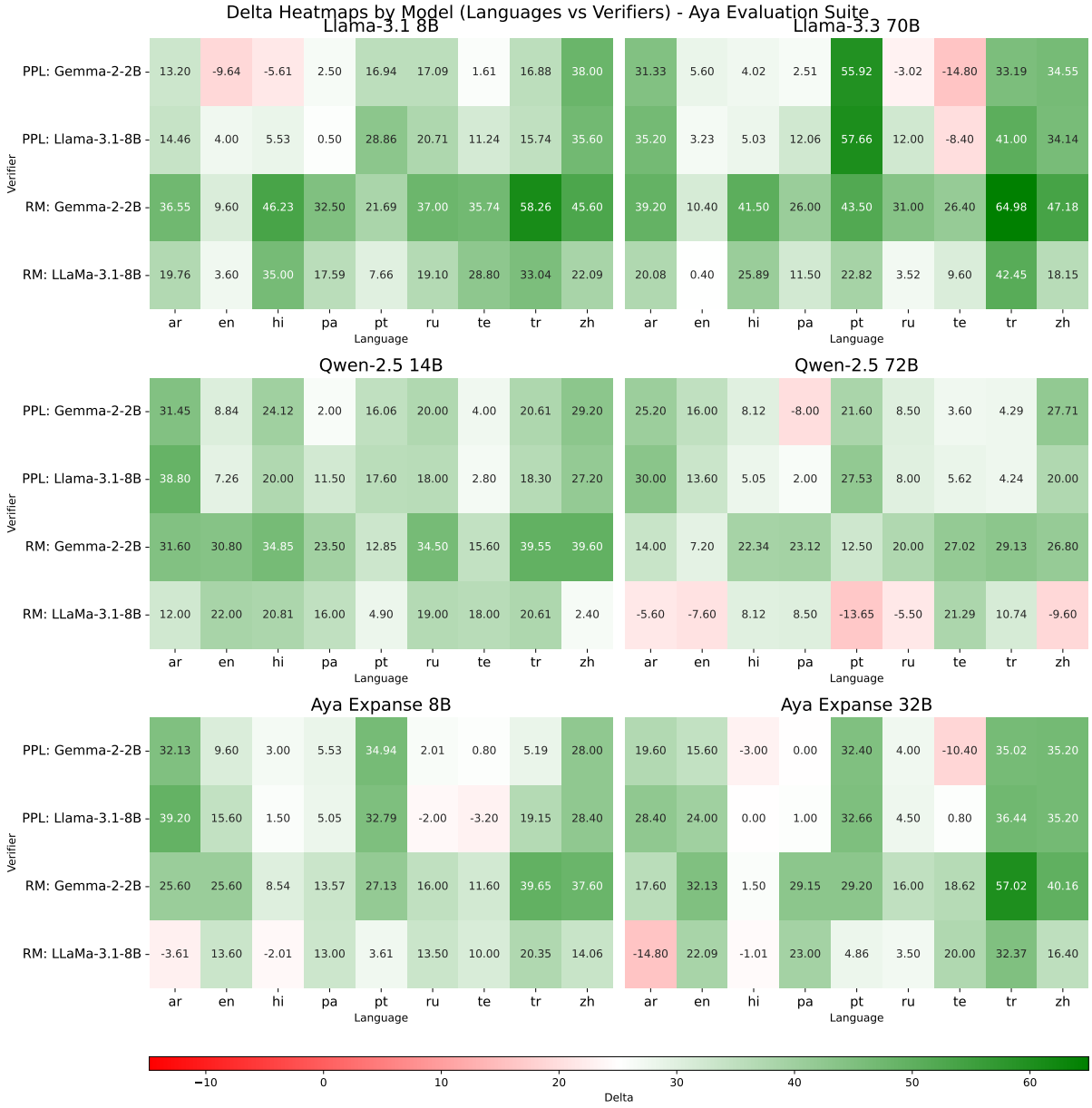

Figure 7: For Aya Evaluation Suite. Heatmaps showing the language-specific results for each model and verifier. The results are shown for $n = 100$.

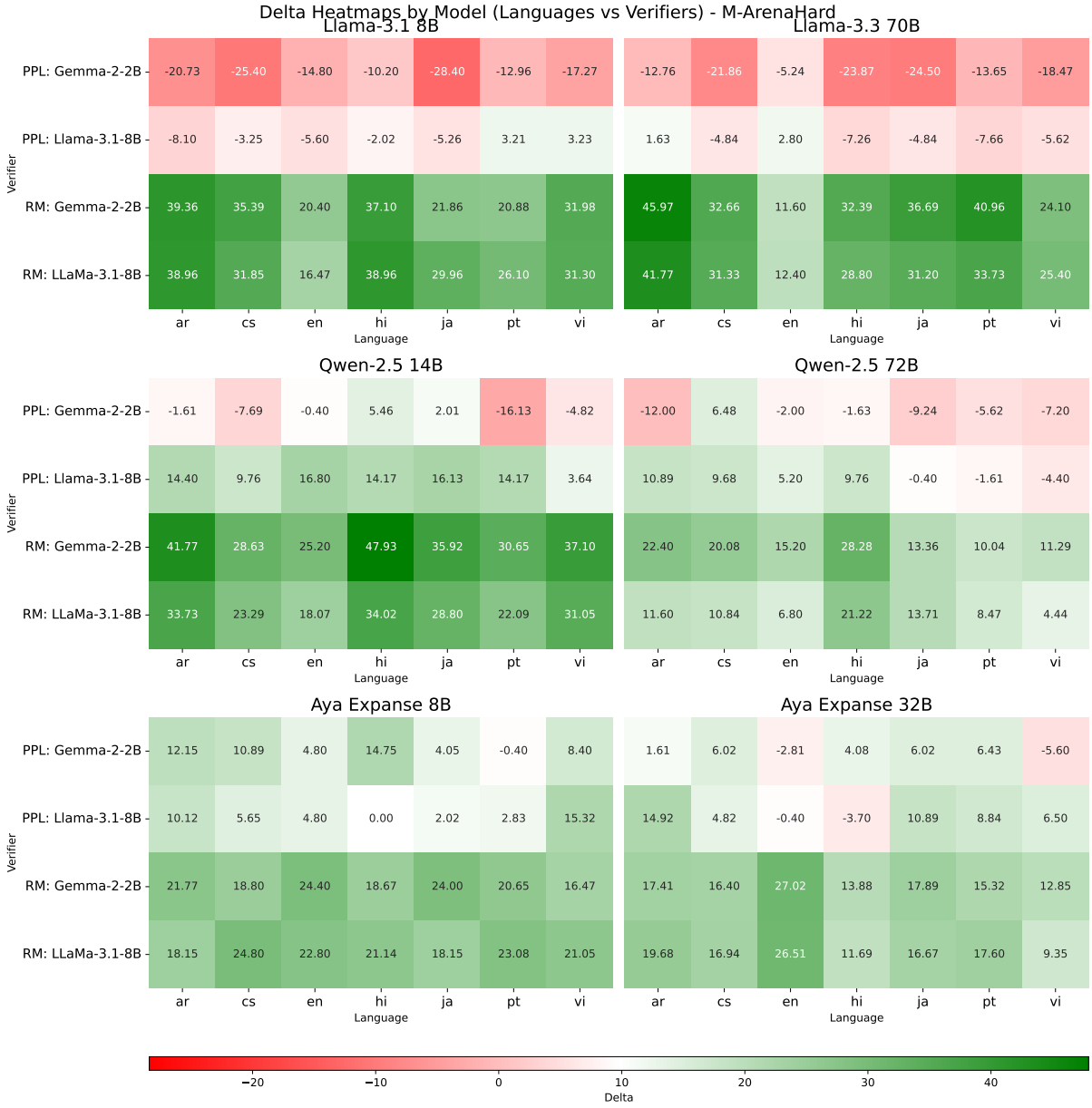

Figure 8: For m-ArenaHard. Heatmaps showing the language-specific results for each model and verifier. The results are shown for $n = 100$.