Building Reasoning Trees - Beyond the 80/20 Rule

Aditya Shukla Tom Gribilas Nicholas Stranges

1. Base Paper: Beyond The 80/20 Rule

Wang et al.'s *Beyond the 80/20 Rule: High-Entropy Minority Tokens Drive Effective Reinforcement Learning for LLM Reasoning* explores the role of high-entropy tokens in Chain-of-Thought reasoning (CoT) and Reinforcement Learning with Verifiable Rewards (RLVR). They identify various patterns in token entropies and suggest that the subset of tokens with high entropy act as forks in the Large Language Model's (LLM) reasoning paths. They use these observations to inform a modified RLVR approach targeting only high-entropy tokens (Wang et al., 2025). In our project, we build on these results by investigating entropy distributions stratified by correctness, constructing and analyzing a reasoning tree, and proposing a further modified RLVR approach.

1.1. Entropy as a Measure of Exploration

Wang et al. describe *token-level entropy* during generation as:

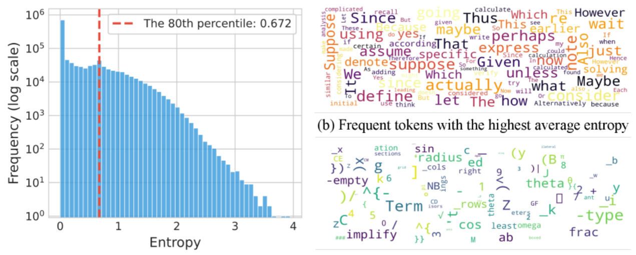
$$H_t := -\sum_{j=1}^{V} p_{t,j} \log p_{t,j}$$

where $(p_{t,1}, \cdots, p_{t,V}) = \pi_\theta(\cdot \mid q, o_{<t}) = \text{Softmax}\left(\frac{\mathbf{z}_t}{T}\right)$.

Here, π_θ denotes an LLM parameterized by θ, q is the input query, and $o_{<t} = (o_1, \ldots, o_{t-1})$ represents the previously generated tokens. V is the vocabulary size, $z_t \in \mathbb{R}^V$ denotes the pre-softmax logits at time step t, $\mathbf{p_t} \in \mathbb{R}^V$ is the corresponding probability distribution over the vocabulary, and $T \in \mathbb{R}$ is the decoding temperature (Wang et al., 2025).

With this definition, their preliminary investigation begins with determining token entropy distributions and patterns. Specifically, they task Qwen-3-8B (Yang et al., 2025) with math, coding, and scientific reasoning problems using Thinking Mode and a decoding temperature of 1. They find a small subset of tokens exhibit high entropies. Somewhat arbitrarily, they define a bound for high-entropy tokens as the 80th percentile, in accordance with the widespread 80/20 rule (Figure 1a). They observe this bound at 0.672, with over half of tokens having an entropy less than 10^{-2}. Tokens below 0.672 are considered low-entropy tokens (Wang et al., 2025).

They observe that low-entropy tokens tend to be perfunctory characters that serve to continue or complete the linguistic structure, adding little to no meaning of their own to the running context. Examples of these include mathematical notation, delimiters, parentheses, and word suffixes. Alternatively, high-entropy tokens are dominated by grammatical conjunctions serving as logical connectors, such as *but*, *thus*, *however*, and *suppose*. Figures 1b and 1c depict word clouds of common high and low entropy tokens. Low-entropy tokens naturally exhibit more determinism than their high-entropy counterparts, leading the authors to interpret the latter as pivotal decision points that determine the trajectory of reasoning among multiple potential pathways (Wang et al., 2025).



(a) Distribution of token entropy (b) Frequent tokens with the highest average entropy (c) Frequent tokens with the lowest average entropy

Figure 1. Generated token entropy patterns (Wang et al., 2025).

Based on their observed behaviour for causing forks in CoT reasoning, Wang et al. refer to high-entropy tokens as forking tokens. To test the importance of high-entropy tokens in reasoning processes, they investigate Qwen3-8B's performance on AIME 2024 and 2025 with varying modifications on decoding temperature. Specifically, they define temperature as a function of token entropy as:

$$T'_t = \begin{cases} T_{high} & \text{if } H_t > h_{\text{threshold}} \\ T_{low} & \text{otherwise} \end{cases}$$

They repeat their tests twice, once holding T_{high} constant at 1 while varying T_{low} and once holding T_{low} constant at 1 while varying T_{high}. Figure 2 depicts the results of this experiment (Wang et al., 2025).

They find that lowering T_{high} degrades model performance compared to lowering T_{low}. Increasing T_{high} is found to substantially improve performance compared to increasing T_{low}. This suggests that assigning higher temperatures to forking tokens may benefit the model. Wang et al. claim that this observation aligns with high-entropy tokens' forking behaviour, with high temperature allowing them to explore diverse reasoning paths (Wang et al., 2025).

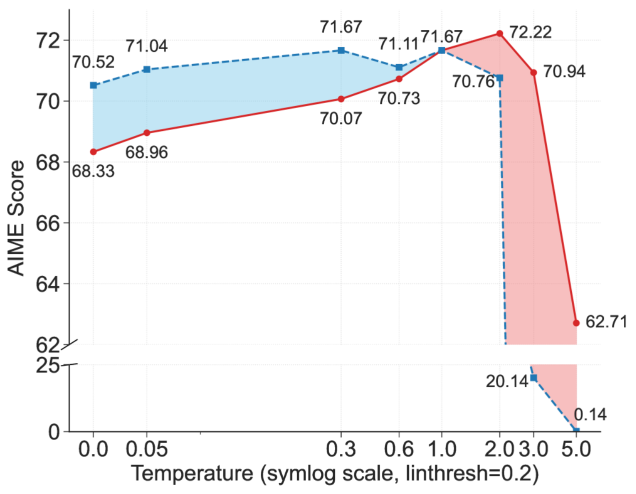


Figure 2. Qwen3-8B average AIME 2024 and 2025 Scores when varying T_{high} with $T_{low} = 1$ (red) and when varying T_{low} with $T_{high} = 1$ (blue). Lowering T_{high} and raising T_{low} deteriorate model performance (Wang et al., 2025)

1.2. Reinforcement Learning with Forking Tokens

Reinforcement Learning (RL) algorithms in LLM post-training have proven to be very effective at improving performance. RL allows models to learn objectives beyond cross entropy loss for next token prediction. Wang et al. focus on using RL to improve model performance on common mathematical reasoning benchmarks. Standard training methods use RLVR, a binary score based on the correctness of the provided answer. One of the first uses of RLVR is to train DeepSeekMath using an RL algorithm called Group Relative Policy Optimization (GRPO) (Shao et al., 2024). The present paper uses a newer version of the GRPO algorithm called Dynamic sAmpling Policy Optimization (DAPO) that does not consider Kullback–Leibler divergence between the reference policy and the current policy (Wang et al., 2025).

Both GRPO and DAPO calculate the objective with respect to groups (denoted G) of completions, allowing each completion to be compared relative to its group. Token-level rewards are averaged over each token in each completion in the group. Rewards are calculated with two values, the policy ratio and the advantage. The policy ratio is denoted:

$$r_t(\theta) = \frac{\pi_\theta(o_t \mid q, o_{<t})}{\pi_{\theta_{\text{old}}}(o_t \mid q, o_{<t})}.$$

and is the ratio of the log probability of the chosen token for the current model and the original reference model. The policy ratio determines the magnitude of the update. Advantage is the mechanism that determines the quality of each token, with correct answers given a reward of 1 and incorrect answers given a reward of 0. Rewards are Z-score normalized across their group and multiplied by the policy ratio to produce the weighted, normalized advantage. DAPO applies clipping to constrain the size of the update and stabilize training (Wang et al., 2025).

To target high-entropy tokens, Wang et al. modify DAPO to only calculate the weighted, normalized advantage for high-entropy tokens. The authors use an indicator function $\mathbb{I}\left[H_t^i \geq \tau_\rho^B\right]$ that evaluates to 1 only if the entropy threshold has been passed. Across three models, Qwen3-8B, Qwen3-14B, and Qwen3-32B, only using high-entropy tokens leads to increased performance on five out of six mathematics reasoning benchmarks tested. Furthermore, improvements from isolating for high-entropy tokens scale as model size increases. For example, the 8B model achieves a 0.53% accuracy increase, whereas the 32B model achieves a 4.10% accuracy increase. The authors attribute this phenomenon to the inability of smaller models to explore effectively, but a rigorous investigation is needed (Wang et al., 2025).

2. Building Trees with Forking Tokens

Wang et al.'s identification of forking tokens in the reasoning process suggests an abstraction of LLM reasoning to a tree (Wang et al., 2025). In this tree, high-entropy forking tokens can act as branches in the reasoning process, allowing the model to explore multiple different paths. This abstraction allows for post-generation analysis, as not all of the tree's children may lead to the same solution and generate with varying lengths. Conceptually, the nodes of the tree are high-entropy tokens and the edges are weighted by the number of tokens between consecutive forks.

The tree generation process is implemented by creating a custom generation template using the Hugging Face library. The token generation process is done as a single pass and each token's entropy is stored. Once a full generation has completed, the top-k highest entropy tokens are stored as nodes and the entropy cutoff is set to the entropy of the kth highest entropy token. The top-k strategy is used instead of the top 20% strategy suggested in the paper because as the amount of tokens in the first pass increases, the size of the tree generated scales exponentially. Specifically, the size of a balanced tree is roughly B^D, where B is the number of branches per node and D is the depth of the tree. The depth of the tree can be considered the number of starting nodes. Reasoning outputs can be thousands of tokens and therefore building a reasoning tree with this many nodes is computationally intractable. For the purposes of this project, branches per node (B) is set to two and the starting depth (D) is set to five.

Once a single pass has been stored in the graph, backtracking from the final token begins. While traversing backward through the tree, if a node that has unused branches is encountered, generation is continued from that node using all of the previous tokens as model context. While generating the new branch, token entropy is checked to determine if newly generated tokens are high-entropy. The backtracking process continues until all possible branches are created or the number of created branches surpasses the maximum branch limit. During the implementation of the backtracking procedure, an important limitation of Hugging Face was discovered. The key-value (KV) cache available in Hugging Face is only designed for append operations and does not support pop operations. This drawback meant that the KV cache needed to be rebuilt each time a new branch started. Future work would include the creation of a KV cache that supports the pop operation.

The maximum branch limit is implemented due to a phenomenon observed during tree generation that we call *entropy explosion*. As the reasoning tree grows in width, branches far from the original path exhibit model degradation. The model's token probability mass functions spread out, leading to higher variance and entropy and causing many tokens in that branch to be considered high-entropy forking tokens. This phenomenon would cause the tree's generation process to diverge, as each branch stemming from these new forking tokens also exhibits this phenomenon. When excessive high-entropy tokens in a single branch are detected, the entropy cutoff is increased to decrease divergence and a maximum branch limit is set to ensure convergence in a specified time frame. We offer no concrete explanation for the entropy explosion phenomenon but we believe it is related to the model generating tokens from highly unlikely context. An example of a reasoning tree is seen in Figure 3. Further examples of generated trees can be seen in Appendix A. Note these visualizations only show forking nodes and not all of the generated tokens.

3. Our Experiments

3.1. Entropy as a Measure of Reasoning Quality

Several works – including Beyond the 80/20 Rule – discover that entropy can be interpreted as a measure of the model's exploration of the solution space (Wang et al., 2025), uncertainty or confidence (Zhu et al., 2025), and its perceived difficulty of a given problem (Scalena et al., 2025). We extend this to explore whether token-level entropy can also serve as a measure of convergence towards the correct answer in verifiable domains.

To test this hypothesis, we use Qwen-1.5B-Math-Instruct, a model instruction-tuned for mathematical reasoning problems. We choose the AIME 2024 dataset, after finding other datasets to be either too difficult (such as AIME2025, OpenR1Math, and CompetitionMath) or too simple (such as MATH500 and GSM8k) and likely to have cross-contaminated into training data.

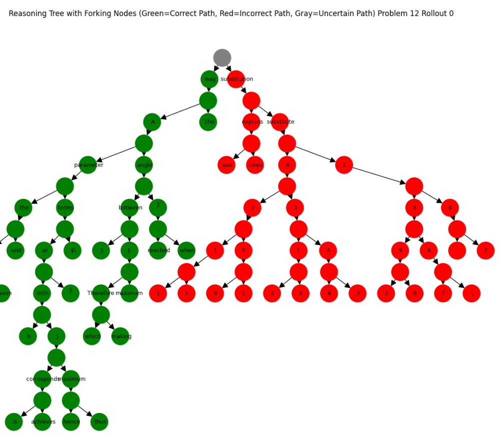

Reasoning Tree with Forking Nodes (Green=Correct Path, Red=Incorrect Path, Gray=Uncertain Path) Problem 12 Rollout 0

Figure 3. Example of a reasoning tree generated from AIME 2024 Q12. Only forking tokens are depicted. A node is coloured green if all of its children yield the correct answer, red if all of its children yield an incorrect answer, and gray if it has both green and red children. The selected token is included in each node. Blank nodes represent placeholder nodes used for backtracking when generating the tree.

For the 30 problems in AIME2024, we generate 50 rollouts for each at a temperature of 0.6, yielding 1500 total rollouts. Responses will naturally vary in length, so the entropy sequences are normalized to ensure fair comparison. We accomplish this by linearly interpolating each sequence to a standard 100-point representation, where each point corresponds to a percentile of generation progress (0% = start, 100% = end).

We utilize the Math-Verify library on Hugging Face to compare mathematical equivalence between the model's boxed answer and the ground truth in the dataset. This enables verification across different but equivalent mathematical representations and allows us to flag each response and rollout as having reached the correct answer (”correct rollout”) or not (”incorrect rollout”).

Within this experimental framework, we compute aggregate statistics for the entropy trajectory for each of these 1500 rollouts. Figure 4 depicts the trend of mean entropy across correct and incorrect rollouts and is statistically assessed with the Mann-Whitney U Test for difference in medians. A non-parametric statistical test is opted for due to highly non-normal data, which violates the assumptions of common statistical tests (Student's t-test, Welch's t-test etc.).

We observe a statistically significant difference between the mean entropy for correct rollouts and incorrect rollouts ($p = 8 \times 10^{-8}$ at 50% of the way through generation). Specifically, we observe the entropy to be consistently high at the very beginning (0%) of generation for both correct and incorrect rollouts, before immediately decreasing and

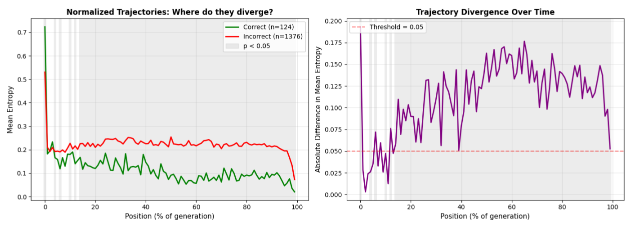


Figure 4. Entropy trajectory for correct and incorrect rollouts. A statistically signifcant difference is observed between average entropies. Gray shading indicates statistical significance.

settling into stable regimes which differ between rollouts by their ultimate correctness. This divergence gains statistical significance at around 15% and persists throughout the remainder of the generation.

This finding validates our hypothesis that the model possesses some innate signal of answer correctness in its token generation entropy. However, earlier works have also demonstrated that entropy can capture a model's perceived difficulty of a given problem. Thus, it is possible that our observed divergence actually stems from the fact that the model found certain problems harder than others, rather than the eventual correctness or incorrectness of its approach. Thus, we conduct a similar experiment with a single problem. We selectively pick one that the model gets correct approximately half the time to nullify any effects due to problem difficulty. Comparing across multiple rollouts generated by our backtracking method, we again find rollouts leading to the correct answer have lower mean entropy than those that reached the incorrect answer. Again, this divergence is statistically significant by the Mann-Whitney U-test ($p = 6 \times 10^{-10}$ at 80% of the way through generation). Note that significant effects occur later in generation when backtracking because forking tokens have a tendency to occur later in generation. Refer to Appendix B for a visualization of this effect.

These findings provide justification for our heuristic-based backtracking approach that leverages the trending mean entropy, a metric that can be computed online during decoding. If we find this trending mean entropy to be above a certain threshold (within the regime of incorrectness), then we may prefer to stop decoding and backtrack to the last high-entropy forking token (as this represents a crucial decision made by the model). From here, we can sample another token from its softmax distribution and begin generating a different response, hoping to instead enter a region of lower mean entropy (within the regime of correctness).

One factor we hypothesize may confound our findings is the effect of including the expected right answer when prompting the model with the problem. We suggest two contradicting possibilities for the effect of such answer conditioning:

Possibility 1: If the model picks the correct approach early on and is progressing towards generating the right answer, knowing what this final answer is supposed to be will likely increase its confidence during generation. This can be likened to when a student that knows the correct answer will be happy to see that their running solution correctly simplifies in a few more steps. In this case, correct answers have lower entropy than incorrect answers, and divergence may be more intense than in the standard case of no conditioning.

Possibility 2: If the model – knowing the expected correct answer – falls into a false sense of confidence and fails to explore the solution space sufficiently (i.e. has lower entropy), it may not find the correct approach to the problem and thus be more likely to reach an incorrect answer. Here, correct rollouts may have higher entropy than the incorrect rollouts due to greater exploration.

Interestingly, we do still observe a statistically significant difference (albeit with a smaller effect) between the trending entropies of correct and incorrect rollouts, but the effect is reversed! As seen in Figure 5, correct rollouts have higher entropy on average throughout generation than incorrect rollouts, suggesting the second possibility may be true. In the standard case with no conditioning, entropy correlates strongly with the model's correctness. In the conditioned case, entropy correlates strongly with the model's exploration. This may be interpreted as a demonstration of the multi-faceted role entropy plays for models that generate probability distributions over a class of possibilities. However, we must note that the statistical significance is weaker in this case as opposed to the non-conditioned case seen earlier.

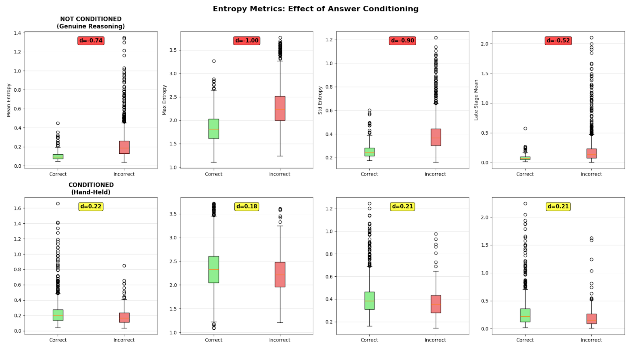


Figure 5. Distributions of mean entropy across correct and incorrect rollouts without answer conditioning (top) and with answer conditioning (bottom).

Another artifact to be noted across all experiments is that the inordinately high entropies seen at the very beginning of generation are always higher for correct rollouts vs incorrect rollouts. This, again, may be evidence of the entropy being a marker of the model's ability to "plan and evaluate" solutions before generating even the first token.

3.2. Impactful Forks and Token-Level Analysis

While Wang et al. define all tokens in the 80th percentile or higher as forking tokens, it is important to note that most do not ultimately impact the model's correctness (Wang et al., 2025). This is exhibited by the tree in Figure 3, in which only 1 forking token (the root fork) impacts correctness. The remaining forks affect the exact path taken, but not the ultimate correctness of the final answer. Motivated by this, we define an *Impactful Fork* as a forking token that directly impacts the correctness of the final output. That is, an impactful fork is a forking token with one child leading to an incorrect answer (red in Figure 3) and one child remaining on the correct path (either green or gray in Figure 3). Gray nodes are considered on the correct path because they have at least one child that achieves the correct answer, meaning it has not yet made a mistake. These impactful forks are the exact token the model makes a mistake. We look for two key insights, whether we can differentiate an impactful fork from a non-impactful fork and if we can determine the correct path within an impactful fork. We hypothesize that these forks can be identified and differentiated from non-impactful forks using distributional information in a similar fashion to how we compare correct and incorrect rollouts. To investigate this, we test Qwen-1.5B-Math-Instruct on AIME problems 0, 9, and 12. These problems are selected to ensure a mixed distribution of correct and incorrect responses (20.92% correct response rate). 25 rollouts are conducted for each question, and the forking token cutoff is established at the 5th highest entropy value in the initial path. This yields roughly 800 reasoning paths per question.

Figure 6 depicts the distributions of entropy between impactful and non-impactful forks. Refer to Appendix C for additional descriptive statistic distribution visualizations. Impactful forks tend to have lower entropy than non-impactful forks. Similarly, they tend to have a larger mean of top 25 probabilities (corresponding to a larger joint probability for a smaller subset of tokens), a higher standard deviation of top 25 probabilities, and a higher maximum probability. Each of these differences exhibits statistically significant results for Mann-Whitney U Test (Entropy: $p = 1.49 \times 10^{-56}$, Mean: $p = 4.91 \times 10^{-44}$, Std: $p = 2.02 \times 10^{-43}$, Max: $p = 3.50 \times 10^{-34}$) and Kolmogorov-Smirnov (KS) Test for difference in sampled distributions (Entropy: $p = 1.02 \times 10^{-59}$, Mean: $p = 5.86 \times 10^{-55}$, Std: $p = 5.95 \times 10^{-41}$, Max: $p = 1.20 \times 10^{-29}$). Therefore, there is an observable difference between impactful and non-impactful forks using only naïve distributional information. It is important to note that the observed difference here may be a product of entropy explosion on incorrect paths.

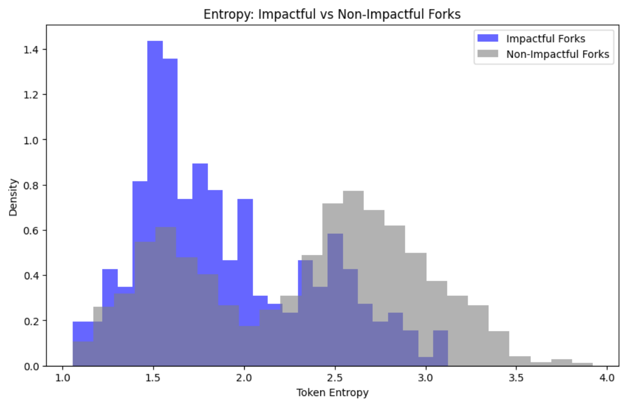


Figure 6. Token entropy distributions of impactful (blue) and non-impactful (gray) forks.

Next, we investigate the difference between correct and incorrect paths within impactful forks. Figure 7 depicts the distributions of selected token probability between correct and incorrect children. Refer to Appendix D for the distributions of selected token probability index. Note that we can not use the same distributional information here as above because both the correct and incorrect path are sampled from the same distribution. Therefore, we use information specific to the token selected for inference. No difference is observed between the correct and incorrect paths in either metric. Wilcoxon-Signed Rank Tests for difference in medians with paired data (Probability: $p = 0.67$, Index: $p = 0.42$) and KS Tests (Probability: $p = 0.68$, Index: 0.57) identify no significant difference between paths. Therefore, the present methods do not identify a difference between correct and incorrect paths from impactful forks.

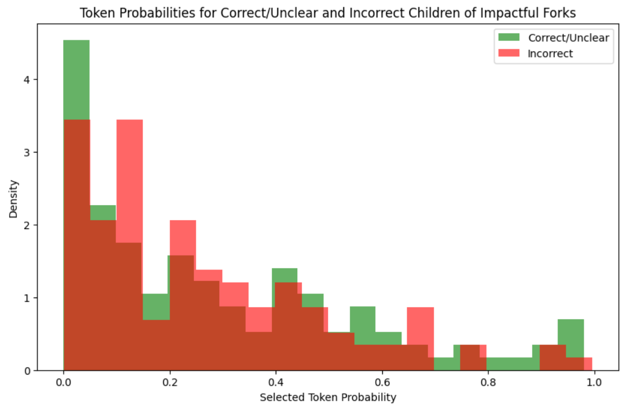


Figure 7. Distribution of selected token probability for correct children (green) and incorrect children (red). No difference is found between the classes.

We identify an observable difference between impactful and non-impactful forks using basic distributional information.

With a similar approach to Wang et al., this can be used to better inform RLVR by targeting impactful forks rather than all forking tokens (Wang et al., 2025). It is unclear whether this will improve RLVR results but is an avenue for future work. Automatic identification of impactful forks may also be used to better inform reasoning algorithms by providing the model with immediate knowledge that a fork is likely impactful. We do not identify a difference between correct and incorrect paths from impactful forks with naïve selected token probabilities. Future research for both subproblems may include using more nuanced representations to classify differences. For example, one may be able to build a classification model for impactful and non-impactful forks using raw softmax probability distributions. Additionally, investigating patterns in token distributions following impactful forks may be useful for retroactively identifying the correct path.

3.3. Length-Adjusted Advantage

When comparing the performance of DAPO on all tokens with DAPO only rewarding the top 20% high-entropy tokens, the authors conclude that there is a correlation between length and mathematics reasoning benchmark performance. The authors mention these results in Table 2 of the paper and is visualized here in Figure 8. In the NeurIPS rebuttal for this paper, the authors show that the frequency of logical connector words increases with their high-entropy targeting methods (Wang et al., 2025). We pose two questions: why does length need to increase for better performance and why is increasing the frequency of logical connector words desirable? Correct answers have different lengths and the desired solution should be as short as possible. Also, why is *wait* needed if the model could solve the problem originally?

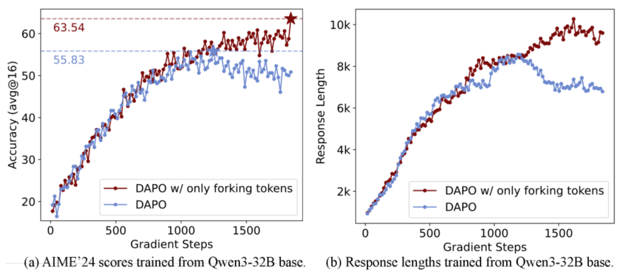


Figure 8. Qwen3-32B (a) average AIME score and (b) response length with vanilla DAPO (blue) and forking-token specific DAPO (red). Wang et al.'s adaptation results in higher accuracy and longer response lengths (Wang et al., 2025)

To address these questions, we propose a length adjusted reward that will reward shorter correct answers the highest. For correct completions, length will be Z-score normalized within its group. This is a measure of how much the length deviates from its group. The sigmoid of the negative of this metric will give a value approaching 1 when the length is much shorter than the rest of the group and approaching 0 when the length is much longer than the rest of the group. A visualization of the length score can be seen in Appendix E. The length reward is added to correct answers, so incorrect answers get a reward of 0 and correct answers get a reward in $(1, 2)$.

$$R^i_{\text{correct}} = 1 + \text{sigmoid}(-\rho^i),$$

$$\text{where} \quad \rho^i = \frac{|o^i| - \text{mean}\Big(\{|o^i|\}_{i=1}^G\Big)}{\text{std}\Big(\{|o^i|\}_{i=1}^G\Big)}.$$

We hypothesize that generating completions using the tree method produces different reward outcomes than standard generation. This hypothesis stems from the fact that completions produced by tree generation are not independent because the completions share the same root context. The data for this experiment are collected using Qwen-1.5B-Math-Instruct on AIME problems 0, 9, and 12 using 960 rollouts per problem with a group size of 32. It is clear that these sampling methods perform differently when comparing Figure 9a and Figure 9b. A KS test between these distributions affirms this observation ($p = 4.86 \times 10^{-84}$. It is hard to know what method would lead to an increase in performance without comparing the performance of post-trained models using these methods.

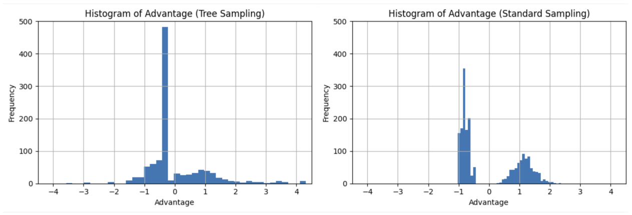


Figure 9. Distribution of advantages using tree sampling (left) and standard sampling (left).

References

Scalena, D., Zotos, L., Fersini, E., Nissim, M., and Üstün, A. EAGER: Entropy-Aware GEneRation for Adaptive Inference-Time Scaling, October 2025. URL `http://arxiv.org/abs/2510.11170`. arXiv:2510.11170 [cs].

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y. K., Wu, Y., and Guo, D. DeepSeek-Math: Pushing the Limits of Mathematical Reasoning in Open Language Models, April 2024. URL `http://arxiv.org/abs/2402.03300`. arXiv:2402.03300 [cs].

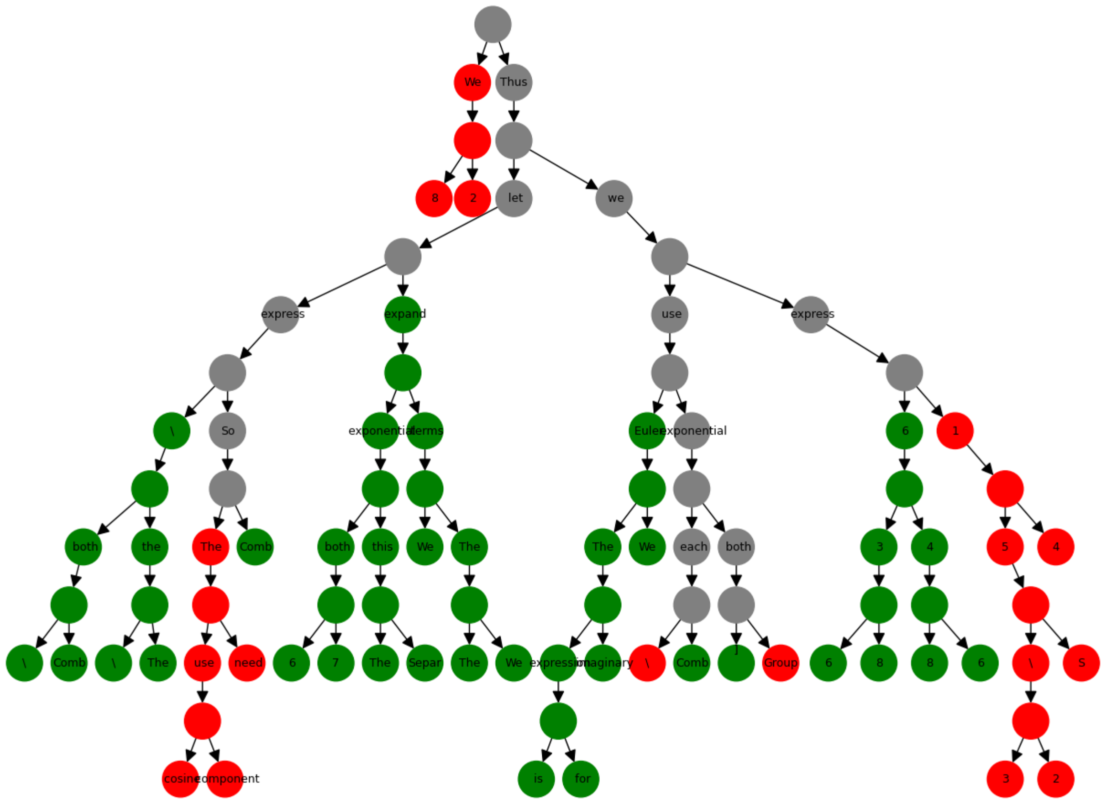
Wang, S., Yu, L., Gao, C., Zheng, C., Liu, S., Lu, R., Dang, K., Chen, X., Yang, J., Zhang, Z., Liu, Y., Yang, A., Zhao, A., Yue, Y., Song, S., Yu, B., Huang, G., and Lin, J. Beyond the 80/20 Rule: High-Entropy Minority Tokens Drive Effective Reinforcement Learning for LLM Reasoning, November 2025. URL http://arxiv.org/abs/2506.01939. arXiv:2506.01939 [cs].

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Zhou, J., Lin, J., Dang, K., Bao, K., Yang, K., Yu, L., Deng, L., Li, M., Xue, M., Li, M., Zhang, P., Wang, P., Zhu, Q., Men, R., Gao, R., Liu, S., Luo, S., Li, T., Tang, T., Yin, W., Ren, X., Wang, X., Zhang, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Zhang, Y., Wan, Y., Liu, Y., Wang, Z., Cui, Z., Zhang, Z., Zhou, Z., and Qiu, Z. Qwen3 Technical Report, May 2025. URL http://arxiv.org/abs/2505.09388. arXiv:2505.09388 [cs].

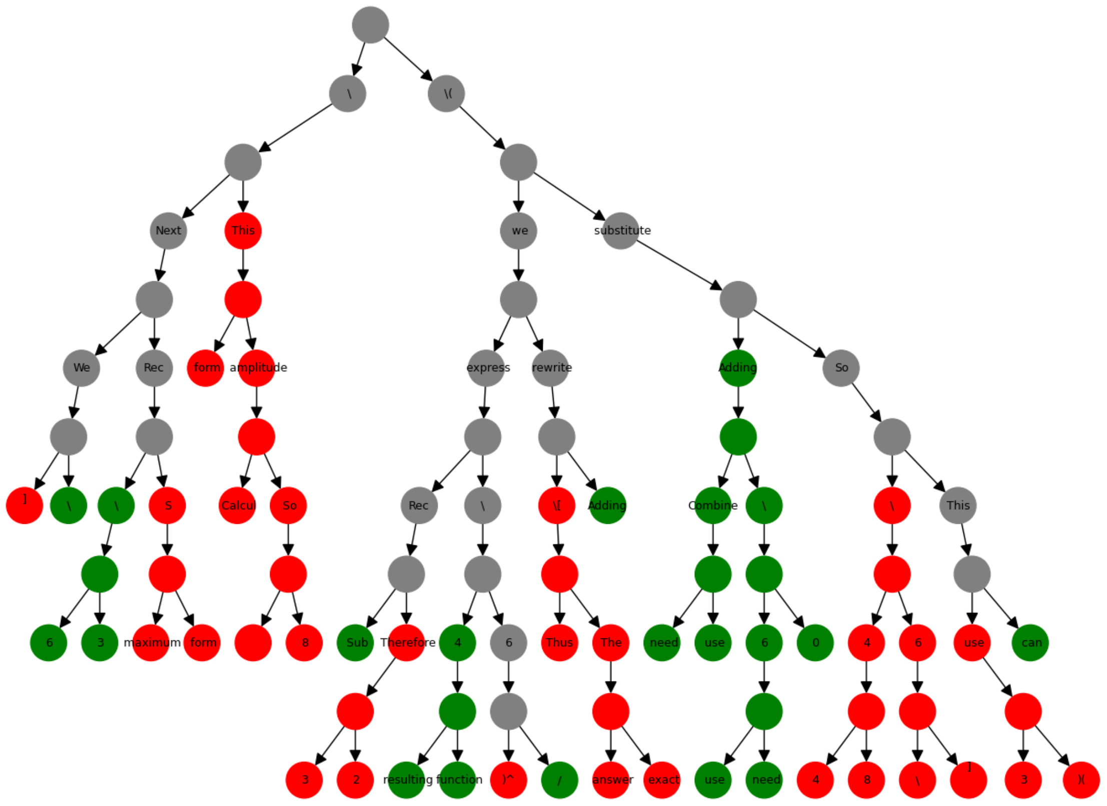
Zhu, Y., Sun, L., Zhao, G., Lin, W., and Zhang, X. Uncertainty Under the Curve: A Sequence-Level Entropy Area Metric for Reasoning LLM, August 2025. URL http://arxiv.org/abs/2508.20384. arXiv:2508.20384 [cs].

A. Additional Examples of Reasoning Trees

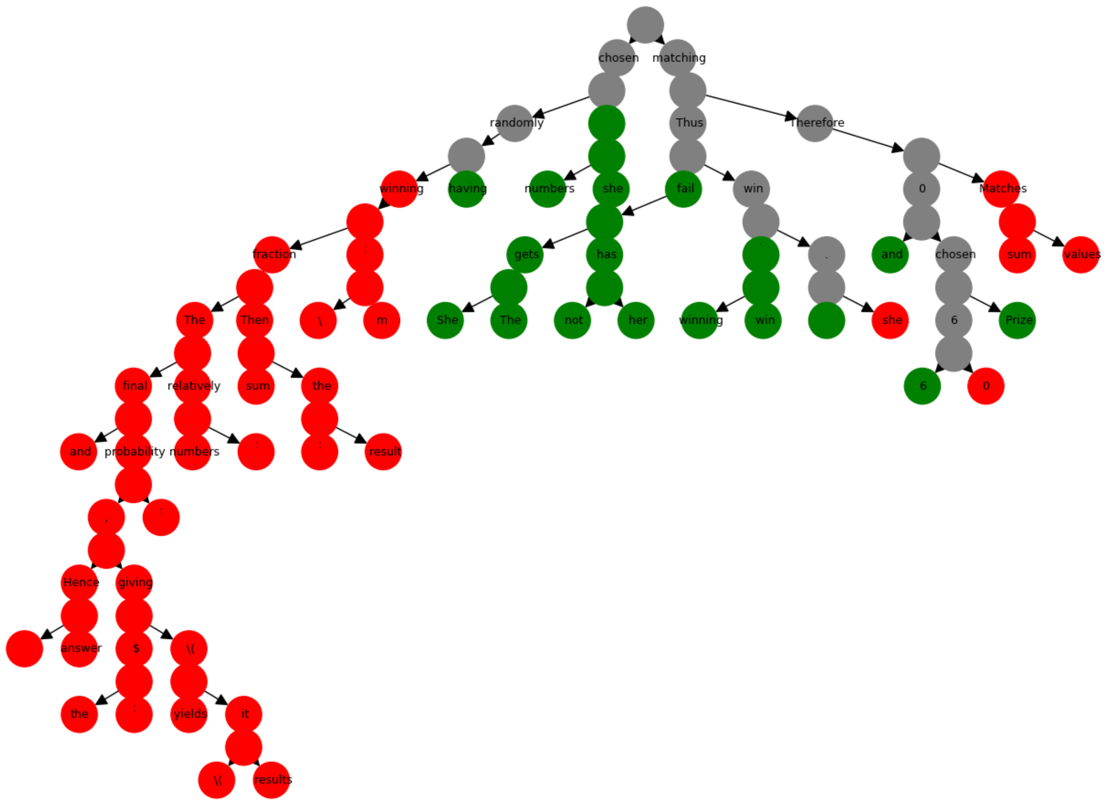
Reasoning Tree with Forking Nodes (Green=Correct Path, Red=Incorrect Path, Gray=Uncertain Path) Problem 12 Rollout 2



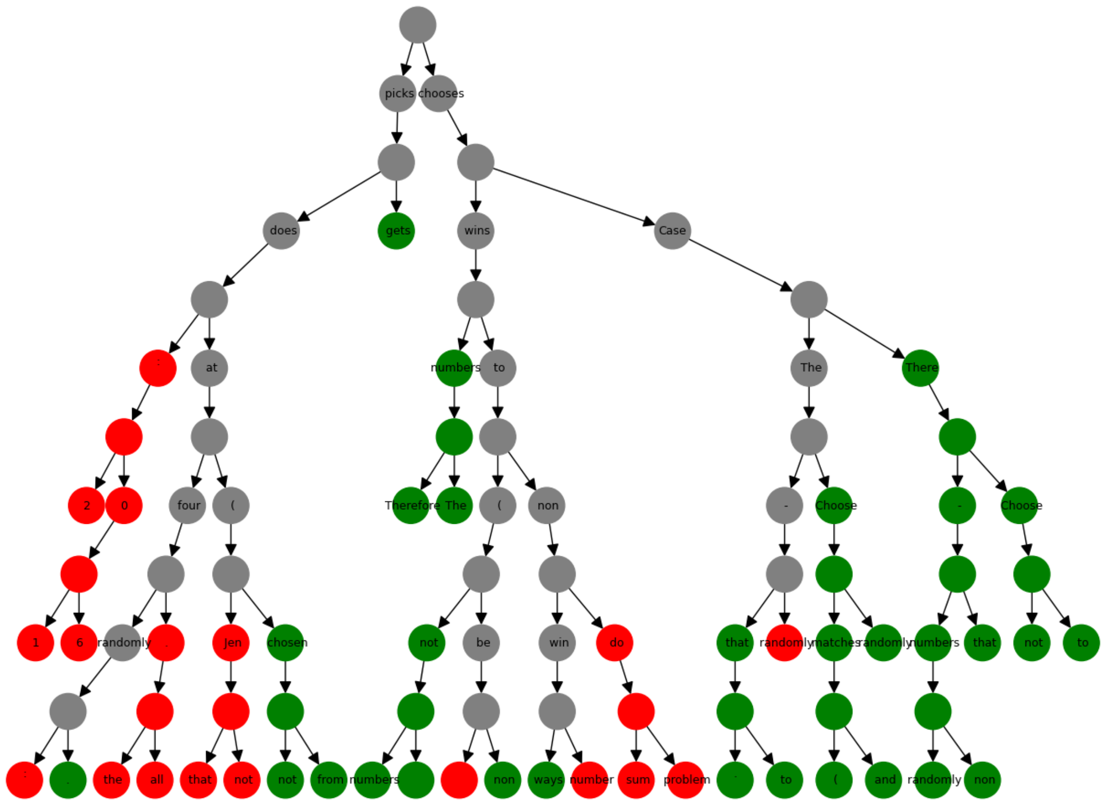
Reasoning Tree with Forking Nodes (Green=Correct Path, Red=Incorrect Path, Gray=Uncertain Path) Problem 12 Rollout 9



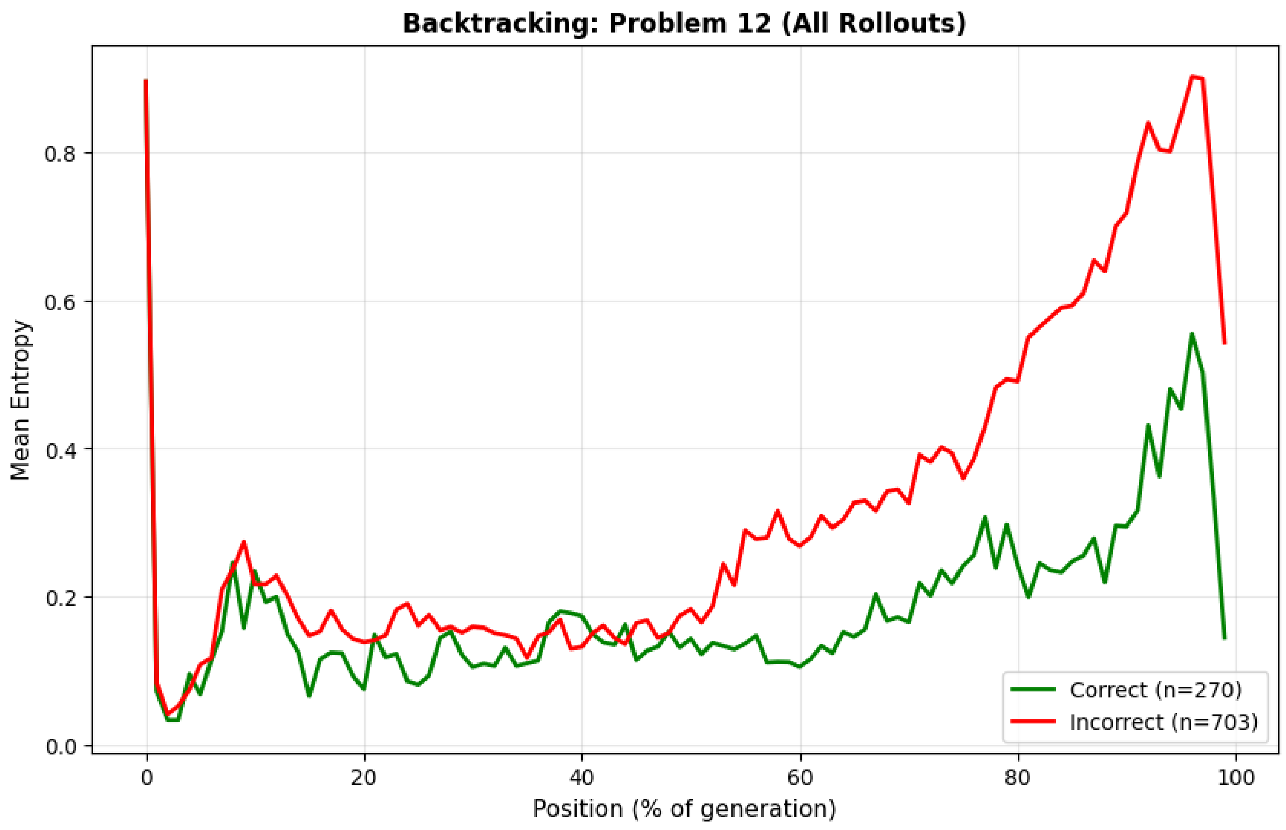
Reasoning Tree with Forking Nodes (Green=Correct Path, Red=Incorrect Path, Gray=Uncertain Path) Problem 9 Rollout 0



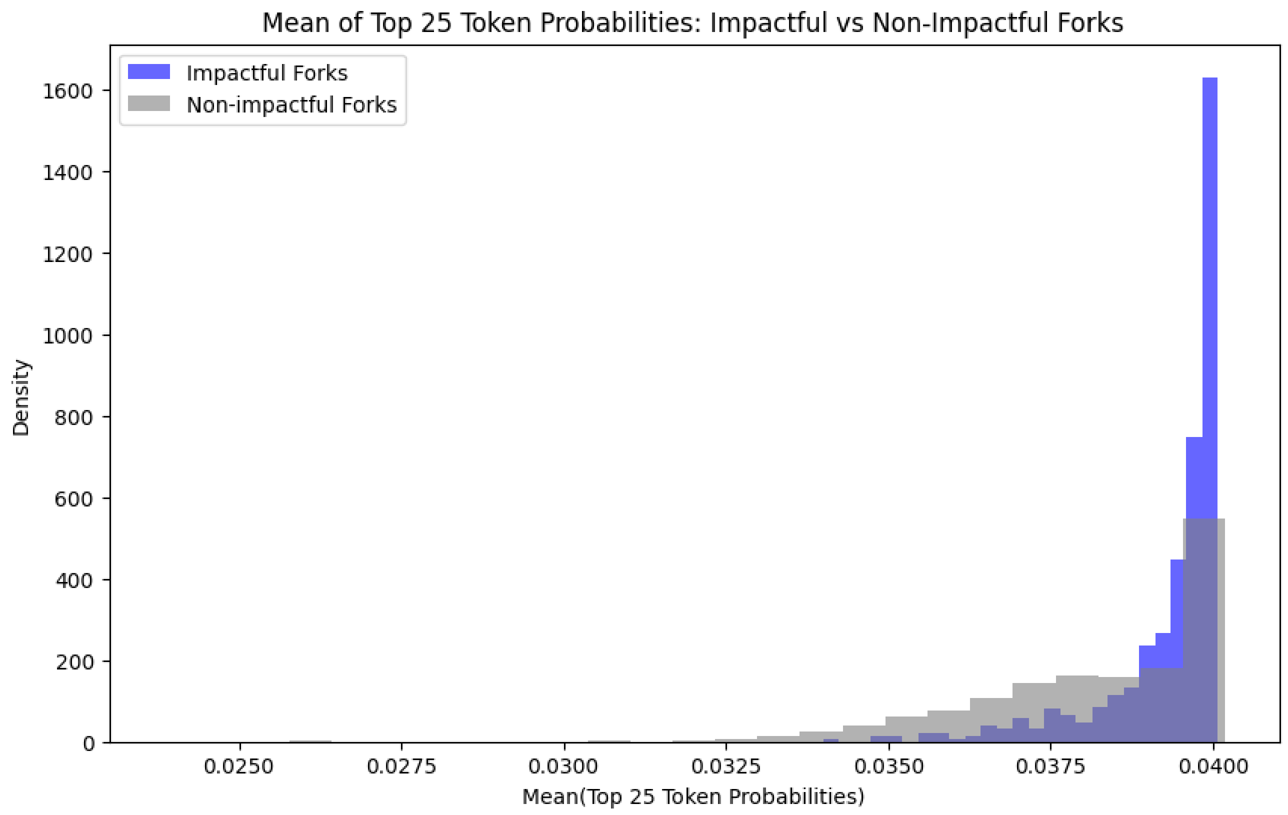
Reasoning Tree with Forking Nodes (Green=Correct Path, Red=Incorrect Path, Gray=Uncertain Path) Problem 9 Rollout 5

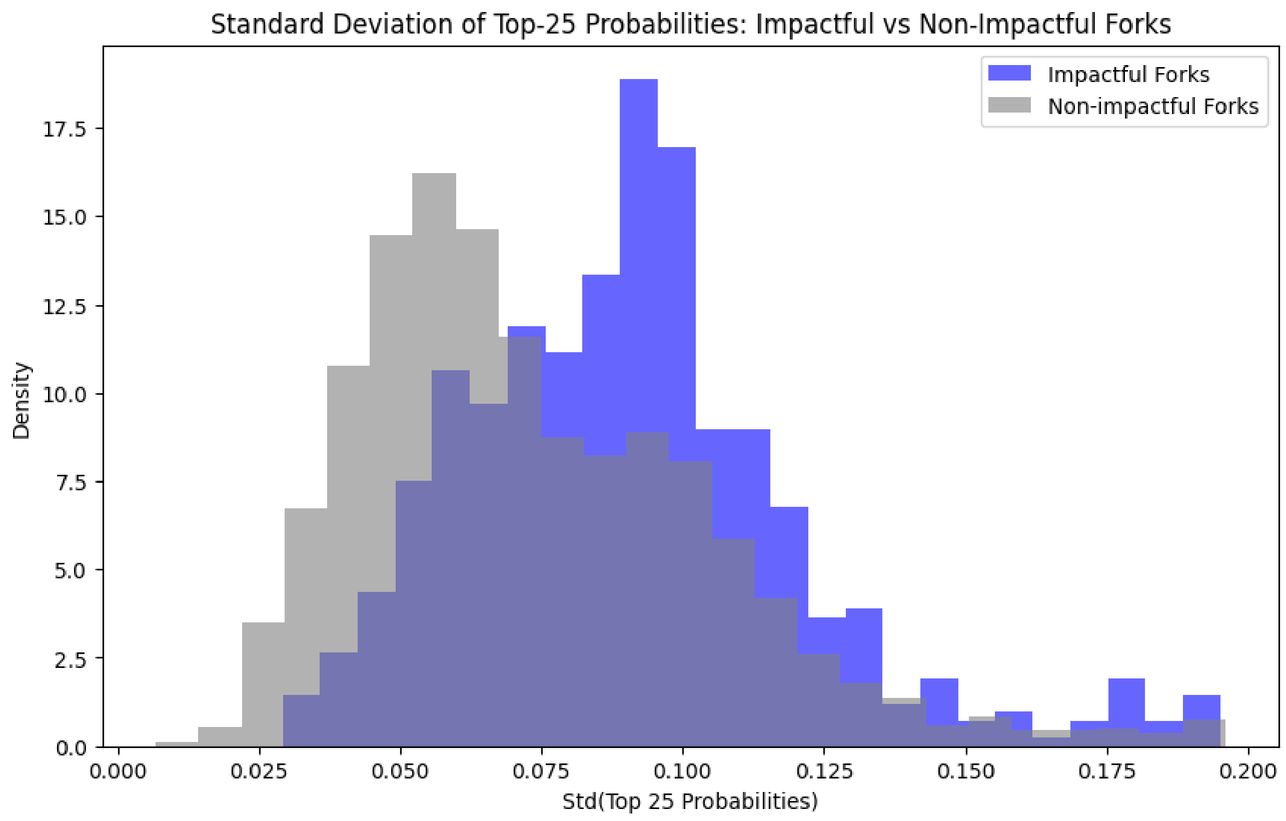


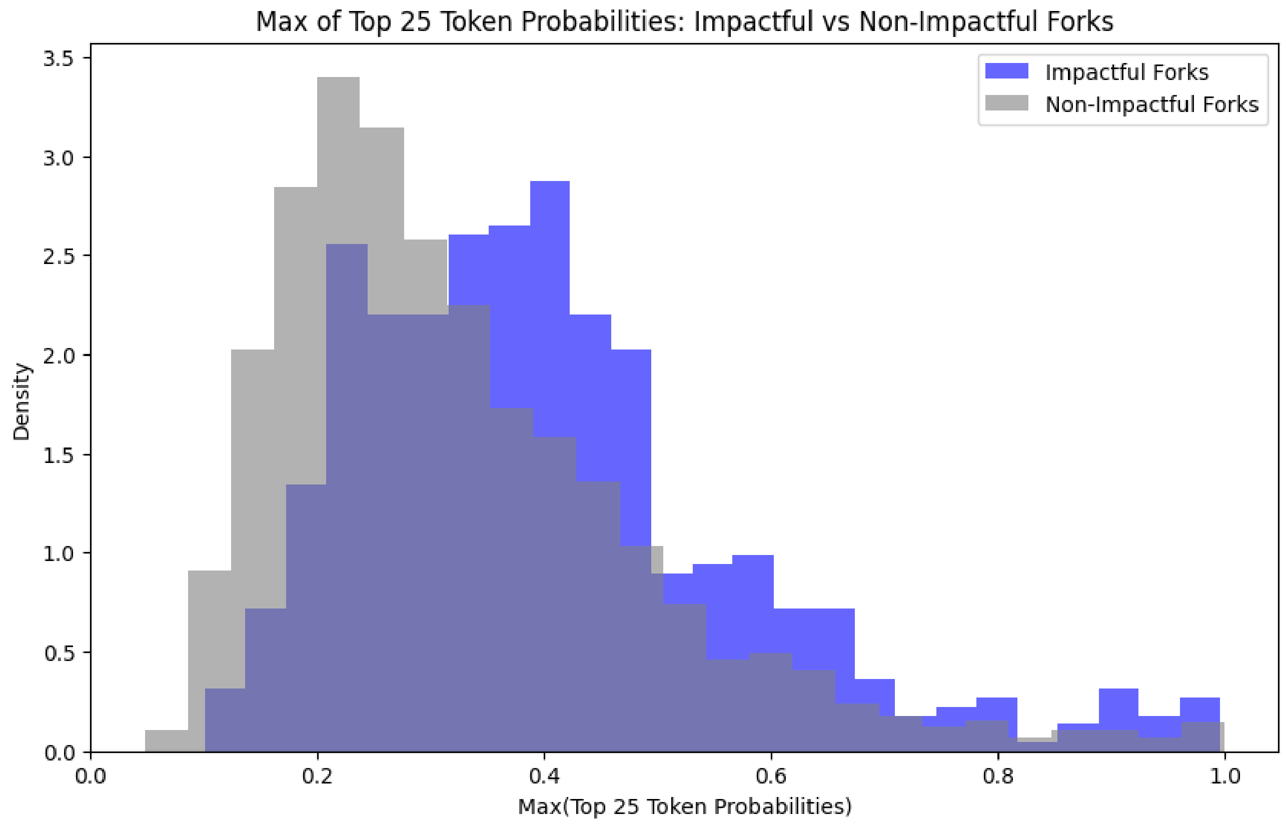
B. Entropy of Correct and Incorrect Rollouts with Backtracking



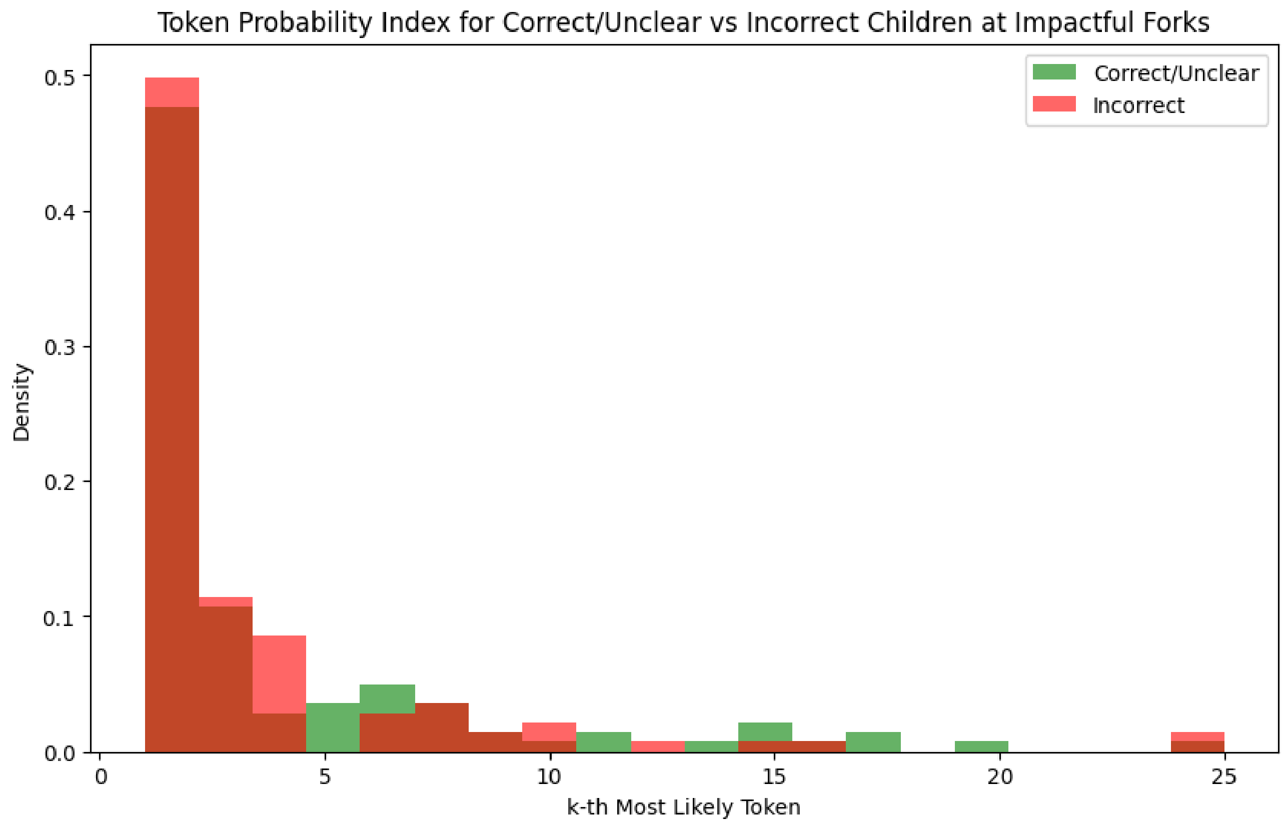
C. Additional Descriptive Statistic Distributions for Impactful and Non-Impactful Forks



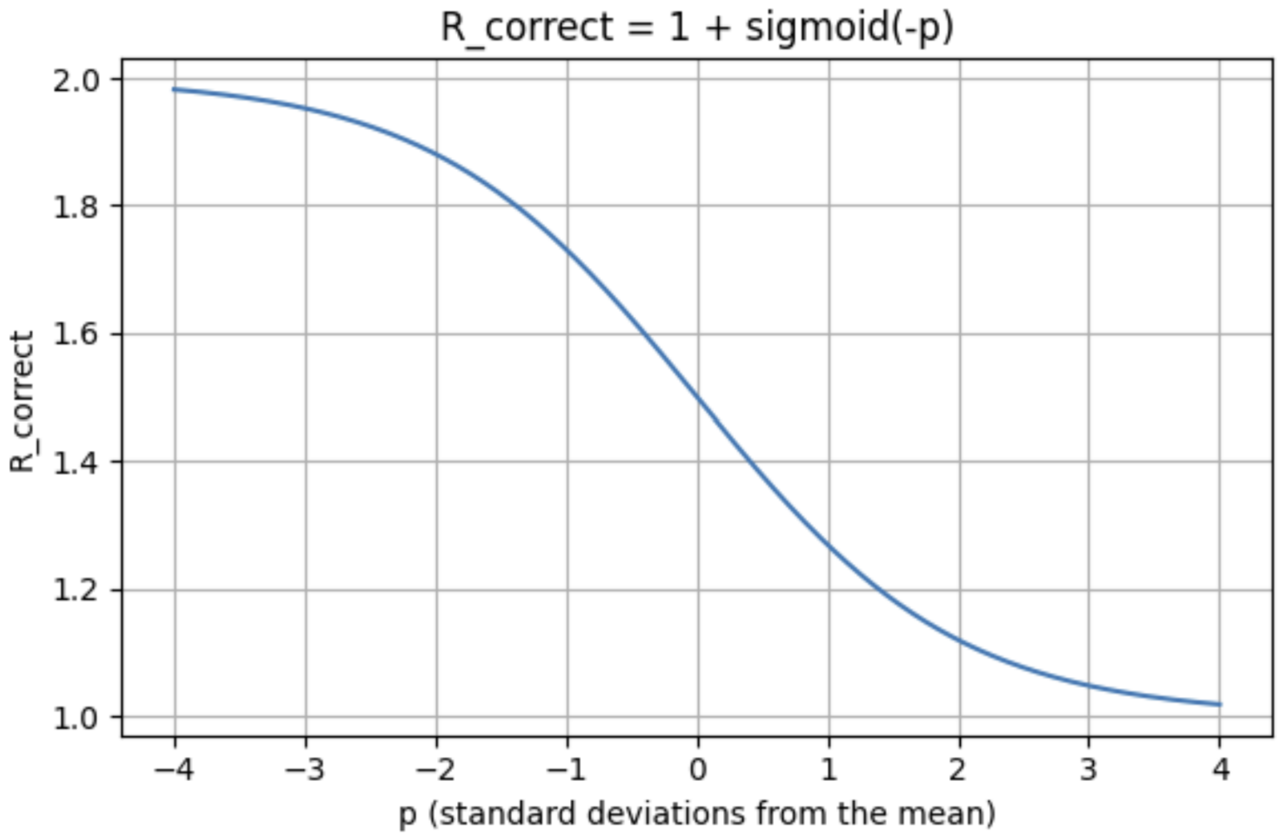

Standard Deviation of Top-25 Probabilities: Impactful vs Non-Impactful Forks
Impactful Forks
Non-impactful Forks
Density
Std(Top 25 Probabilities)
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0.200


Max of Top 25 Token Probabilities: Impactful vs Non-Impactful Forks
Impactful Forks
Non-Impactful Forks
Density
Max(Top 25 Token Probabilities)
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
0.0
0.2
0.4
0.6
0.8
1.0

D. Distribution of Selected Token Probability Index for Correct and Incorrect Children of Impactful Forks



E. Length-Adjusted Advantage Visualization


R_correct = 1 + sigmoid(-p)
R_correct
p (standard deviations from the mean)