Do LLMs Encode Functional Importance of Reasoning Tokens ?

Janvijay Singh **Dilek Hakkani-Tür**
The Grainger College of Engineering
University of Illinois Urbana Champaign
{jvsingh2, dilek}@illinois.edu

Abstract

Large language models solve complex tasks by generating long reasoning chains, achieving higher accuracy at the cost of increased computational cost and reduced ability to isolate functionally relevant reasoning. Prior work on compact reasoning shortens such chains through probabilistic sampling, heuristics, or supervision from frontier models, but offers limited insight into whether models internally encode token-level functional importance for answer generation. We address this gap diagnostically and propose *greedy pruning*, a likelihood-preserving deletion procedure that iteratively removes reasoning tokens whose removal minimally degrades model likelihood under a specified objective, yielding length-controlled reasoning chains. We evaluate pruned reasoning in a distillation framework and show that students trained on pruned chains outperform a frontier-model–supervised compression baseline at matched reasoning lengths. Finally, our analysis reveals systematic pruning patterns and shows that attention scores can predict greedy pruning ranks, further suggesting that models encode a nontrivial functional importance structure over reasoning tokens.[1]

1 Introduction

Large Language Models (LLMs) rely on long, explicit reasoning chains to solve complex tasks in mathematics, science, and multi-step decision making (Wei et al., 2022; El-Kishky, 2024; Guo et al., 2025a). While long reasoning chains improve performance, their length incurs substantial costs, including higher inference latency, increased training and memory requirements, and greater difficulty in isolating parts of the reasoning that are functionally responsible for the final answer. This trade-off has motivated a growing line of work on producing more efficient and compact reasoning chains that preserve task performance (Liu et al., 2024; Kang et al., 2025; Aggarwal and Welleck, 2025; Feng et al., 2025). Most existing approaches to compact reasoning achieve compression by explicitly exploring the space of reasoning chains and selecting shorter alternatives. One prominent mechanism is temperature sampling, in which methods generate multiple candidate chains and train models to produce shorter chains among them (Hassid et al., 2025; Aggarwal and Welleck, 2025; Luo et al., 2025; Hou et al., 2025). Other mechanisms include LLM probability-based heuristics or post-processing rules for removing reasoning segments, as well as the use of frontier LLMs to generate candidate compact reasoning chains (Kang et al., 2025; Xia et al., 2025; Qiao et al., 2025). Although effective, these approaches provide limited insight into whether the reasoning LLM internally encodes *token-level functional importance* in its reasoning for answer generation.

Compared to prior work, we take a different perspective and ask a more fundamental question: *Do LLMs internally encode the functional importance of reasoning tokens for answer generation?* Rather than proposing another method for generating compact reasoning, we study whether a model's own likelihood and attention patterns can serve as signals for ranking reasoning tokens by their functional importance. This framing treats reasoning compression as a diagnostic problem aimed at revealing internally encoded token-level functional structure in the model's reasoning.

To this end, we introduce *greedy pruning*, a likelihood-preserving deletion procedure inspired by perturbation-based attribution methods (Zeiler and Fergus, 2014; Zintgraf et al., 2017) and greedy decoding. Perturbation-based attribution methods assess the importance of input parts by perturbing or removing them and observing the resulting changes in the model's output, while greedy decoding in LLMs incrementally adds tokens to maxi-

[1] **Code:** github.com/iamjanvijay/greedy-token-pruner; **Demo:** iamjanvijay.github.io/greedy-token-pruner.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 30749–30773

mize likelihood. Building on these ideas, greedy pruning starts from a complete reasoning chain and incrementally removes tokens whose deletion minimally degrades the model's likelihood under a specific objective (described in Section 3.2). This process produces a ranking over reasoning tokens and a set of length-controlled reasoning chains, where pruning ranks reflect functional importance under the model's own distribution. Under this interpretation, greedy pruning serves as a probe for identifying reasoning-token subsequences important for maintaining predictive behavior.

To test whether greedily pruned reasoning chains indeed preserve functionally important tokens, we evaluate them using a teacher–pruner–student distillation framework across multiple reasoning benchmarks, including GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), and the multi-domain benchmark MMLU-Pro (Wang et al., 2024). Quantitatively, we show that students distilled on greedily pruned reasoning chains outperform multiple pruning baselines, including `TokenSkip` (Xia et al., 2025), which relies on token importance labels from frontier models. Qualitatively, we analyze the functional structure of pruned reasoning chains, revealing consistent patterns in which types of tokens are preserved or removed at different pruning stages. Finally, we examine whether pruning ranks can be predicted from attention scores, providing evidence that importance signals revealed by greedy pruning are accessible from the model's own internals.

Overall, our results suggest that LLMs encode a nontrivial token-level functional structure within reasoning, which can be revealed through likelihood-preserving pruning and corroborated by attention-based signals. Beyond reasoning compression, greedy pruning offers a principled tool for probing the internal organization of LLM-generated reasoning.

2 Related Work

Compact Reasoning in LLMs. Recent work has explored reducing the costs of long reasoning chains by compressing model-generated reasoning sequences. At a high level, most approaches discover compact chains through external exploration mechanisms and then train models to prefer shorter candidates. A common strategy relies on stochastic sampling, where LLMs generate multiple reasoning chains using temperature-controlled decoding and are trained via supervised fine-tuning or reinforcement learning to favor shorter yet correct outputs (Liu et al., 2024; Hassid et al., 2025; Aggarwal and Welleck, 2025; Luo et al., 2025; Hou et al., 2025; Yi et al., 2025). Other methods apply LLM probability-based heuristics or post-processing rules to prune reasoning steps (Li et al., 2023; Qiao et al., 2025; Zeng et al., 2025). A third line of work employs frontier LLMs to compress reasoning chains and distills smaller models on the resulting outputs (Kang et al., 2025; Cui et al., 2025; Xia et al., 2025). Despite their differences, prior approaches share a common design choice: compact reasoning chains are discovered through external mechanisms such as stochastic exploration, heuristic rules, or additional supervision. Consequently, they offer limited controllability, provide little characterization of removed reasoning tokens, and shed limited light on whether token-level importance is encoded by the model. In contrast, our approach enables explicit and controllable pruning, directly characterizes removed reasoning tokens, and provides a principled diagnostic tool for probing internal token-level importance.

Attribution Methods for LLM Reasoning. Attribution methods identify parts of the input or model responsible for a prediction by tracing importance signals through the network (Zhao et al., 2024). In neural language models, such signals have been derived from input gradients (Sundararajan et al., 2017; Yin and Neubig, 2022), attention patterns (Abnar and Zuidema, 2020; Hou et al., 2023), representation similarity (Yin and Neubig, 2022; Enguehard, 2023), and perturbation-based deletion analyses (Zeiler and Fergus, 2014; Ribeiro et al., 2016; Zintgraf et al., 2017). Recent work extends attribution to reasoning in LLMs, including Shapley-style analyses of reasoning tokens (Gao, 2023), gradient-based importance scores (Wu et al., 2023), intervention-based faithfulness tests (Lanham et al., 2023), attention-based attribution (Cohen-Wang et al., 2025), and counterfactual resampling of reasoning steps (Bigelow et al., 2025; Bogdan et al., 2025). These methods reveal that reasoning traces exhibit some internal importance structure, but are primarily restricted to analysis and understanding. In contrast, our work adopts a more pragmatic perspective (Nanda et al., 2025), aiming to produce internal, token-level importance rankings for reasoning tokens that can also be used to train models for compact reasoning.

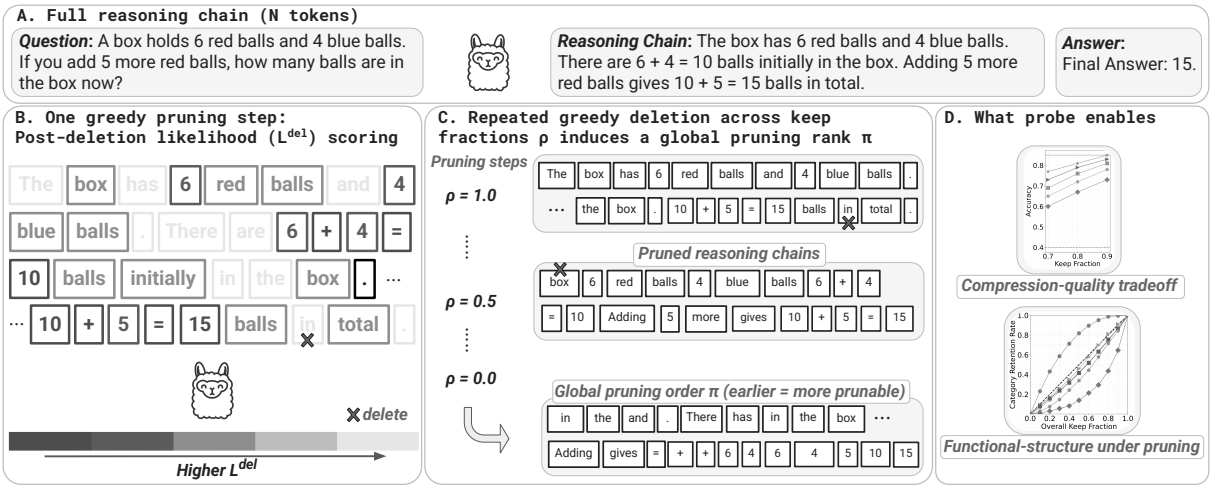


Figure 1: **Greedy pruning as a diagnostic probe. A.** A teacher model generates a full reasoning chain for a given question. **B.** A greedy pruning step scores candidate token deletions by post-deletion likelihood L^{del} and removes the token whose deletion best preserves likelihood. **C.** Iterating this procedure over decreasing keep fractions ρ yields length-controlled chains and induces a pruning order π, where earlier-ranked tokens are safer to prune. **D.** This order enables analysis of compression–quality tradeoffs and the structure of encoded functional importance.

3 Greedy Pruning: Likelihood-Preserving Deletion of Reasoning Tokens

3.1 Problem Setup

Given a question Q, we consider three model roles: (i) a teacher LLM $\mathcal{T}$ that generates a reasoning chain $R = (r_1, \ldots, r_n)$ of n tokens and an answer A; (ii) a pruner LLM $\mathcal{P}$ with parameters $\theta_{\mathcal{P}}$ that evaluates likelihood-based objectives over subsequences of R; and (iii) a student model $\mathcal{S}$ that can be trained on pruned reasoning chains. We refer to this as a teacher–pruner–student distillation framework, which we instantiate in Section 4. We study pruning in the *post-generation* setting, where a complete reasoning chain is first produced and pruning is applied offline. Given a target keep fraction $\rho \in (0, 1]$, let $m = \lceil \rho n \rceil$ denote the retained length, and let R_K denote the subsequence of R indexed by the kept index set $K \subseteq \{1, \ldots, n\}$. Pruning is defined with respect to a likelihood-based objective $\mathcal{L}_{\theta_{\mathcal{P}}}(Q, R_K, A)$ under the pruner model, with the goal of identifying a subset K of size m such that $\mathcal{L}$ is approximately preserved.

3.2 Greedy Pruning

Greedy pruning is a likelihood-preserving deletion procedure, analogous to greedy decoding. While greedy decoding incrementally *adds* tokens to maximize likelihood, greedy pruning starts from a complete reasoning chain and iteratively *removes* the token whose deletion minimally degrades the pruning objective. The resulting deletion order induces a monotonic ranking over reasoning tokens, ordered by their contribution to preserving the pruning objective. Unlike leave-one-out attribution (Zintgraf et al., 2017), which assigns independent importance scores under a fixed input, greedy pruning re-evaluates token importance under a changing context as pruning progresses, capturing token interactions (Section 5.2). We illustrate greedy pruning in Figure 1 and formally describe it in Algorithm 1. While greedy pruning is a myopic approximation and does not guarantee optimal subset selection, our claims focus on the existence and accessibility of internal ranking signals under likelihood-based probes, rather than the optimality of the induced subsets. Greedy pruning is chosen for its simplicity; exploring richer search strategies, such as beam search, is left to future work.

Pruning Objectives. We consider two likelihood-based objectives that induce different notions of token importance with respect to answer generation. The first objective preserves answer likelihood,

$$\mathcal{L}^{\text{ANS}}_{\theta_{\mathcal{P}}}(Q, R_K, A) = \log P_{\theta_{\mathcal{P}}}(A \mid Q, R_K), \quad (1)$$

allowing aggressive pruning of intermediate reasoning tokens provided that the pruner assigns high probability to the correct answer. The second objective preserves both the reasoning and the answer under the model's distribution,

$$\mathcal{L}^{\text{JOINT}}_{\theta_{\mathcal{P}}}(Q, R_K, A) = \log P_{\theta_{\mathcal{P}}}(R_K, A \mid Q), \quad (2)$$

Algorithm 1 Greedy Pruning

Require: Question Q; reasoning tokens $R = (r_1, \ldots, r_n)$; answer A; objective $\mathcal{L}$; keep fraction $\rho \in (0, 1]$; pruner model $\mathcal{P}_{\theta_\mathcal{P}}$

Ensure: Pruning ranks π; K, the set of token indices kept after pruning; pruned reasoning R_K, where R_K denotes the subsequence of R indexed by K, preserving the original order

$K \leftarrow \{1, \ldots, n\}; \quad m \leftarrow \lceil \rho \cdot n \rceil$
Initialize $\pi[i] \leftarrow \infty$ for all $i \in \{1, \ldots, n\}$
$t \leftarrow 1$
while $|K| > m$ **do**
  for all $i \in K$ **do**
    $L_i^{del} \leftarrow \mathcal{L}_{\theta_\mathcal{P}}(Q, R_{K \setminus \{i\}}, A)$
  end for
  $i^\star \leftarrow \arg\max_{i \in K} L_i^{del}$
  $\pi[i^\star] \leftarrow t$
  $K \leftarrow K \setminus \{i^\star\}; \quad t \leftarrow t + 1$
end while
return K, R_K, π

and is therefore more restrictive, favoring pruned reasoning chains that remain consistent with the pruner's reasoning trajectory while supporting answer generation. These objectives induce different notions of likelihood sensitivity over answers and reasoning structure, yielding distinct pruning behaviors under the model's distribution (Section 5.1). All model likelihoods are computed conditioned on the gold token prefix; additional implementation details are provided in Appendix B.4.

Interpretation of Pruning Ranks. Pruning ranks are not intended as faithful or causal explanations of a model's internal computation, nor as human-interpretable reasoning steps. Instead, they capture an interventional notion of likelihood sensitivity to token deletion under the model's distribution. Under this view, a token's pruning rank reflects its functional importance for generation rather than mechanistic faithfulness. To analyze patterns beyond tokenization artifacts, we aggregate tokens into higher-level semantic categories (Section 5.1). Overall, greedy pruning serves as a diagnostic probe for identifying reasoning-token subsequences that preserve predictive behavior. We next instantiate greedy pruning in a teacher–pruner–student framework to evaluate its effect on distillation with compressed reasoning.

4 Distillation Experiments

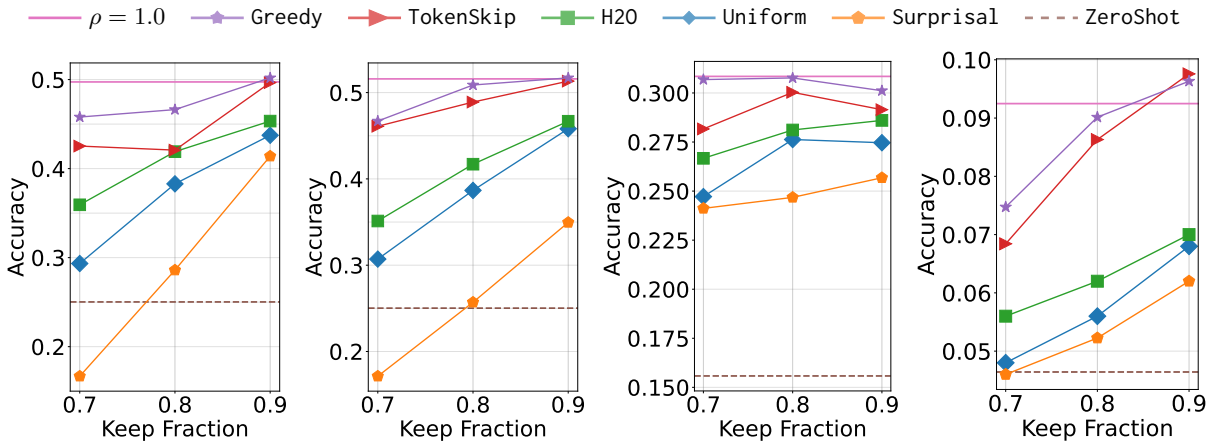
4.1 Experimental Setup

We instantiate the teacher–pruner–student distillation framework (Section 3.1) to evaluate greedy-pruned reasoning against multiple baselines. If pruning preserves functionally important tokens for answer generation, the resulting reasoning chains should support effective student distillation.

Datasets and Models. We conduct experiments on three reasoning-intensive benchmarks: (i) GSM8K (Cobbe et al., 2021), consisting of grade-school arithmetic word problems; (ii) MMLU-Pro (Wang et al., 2024), a multi-domain benchmark spanning diverse subject areas; and (iii) MATH (Hendrycks et al., 2021), containing olympiad-level mathematical reasoning problems. We use four instruction-tuned models spanning multiple families and performance levels: Llama-3.1-8B-Instruct, Qwen-2.5-7B-Instruct, Llama-2-7B-chat-hf, and Mistral-7B-Instruct, referred to as Llama3.1-8B, Qwen2.5-7B, Llama2-7B, and Mistral-7B, respectively. Additional dataset and model details are provided in Appendix B.1.

Teacher–Pruner–Student Instantiation. We first evaluate all models in a zero-shot setting using deterministic greedy decoding (Table 2) and characterize relative performance per dataset. Higher-performing models (Llama3.1-8B, Qwen2.5-7B) serve as teachers, while lower-performing models (Llama2-7B, Mistral-7B) serve as students, reflecting the intuition that stronger models induce more reliable reasoning chains and pruning signals that can be distilled into weaker models via compact reasoning supervision. Reasoning chains are generated from the teacher using rejection sampling with temperature 0.7. The pruner then applies greedy pruning under a likelihood-based objective to produce compressed reasoning sequences at a specified keep fraction. Unless otherwise stated, the pruner equals the teacher and the JOINT objective is used. Students are trained via supervised fine-tuning (SFT) on the pruned reasoning dataset and evaluated on downstream task accuracy. Additional details are provided in Appendix B.2 and B.4.

Baselines. To evaluate greedily pruned reasoning chains, we compare against four token-level importance baselines: `TokenSkip` (Xia et al., 2025), `H2O` (Zhang et al., 2023), `Surprisal` (Li et al., 2023), and `Uniform`. `TokenSkip` prunes reason-



(a) $\mathcal{T}, \mathcal{P}$ = Llama3.1-8B; GSM8K. (b) $\mathcal{T}, \mathcal{P}$ = Qwen2.5-7B; GSM8K. (c) $\mathcal{T}, \mathcal{P}$ = Llama3.1-8B; MMLU-Pro. (d) $\mathcal{T}, \mathcal{P}$ = Qwen2.5-7B; MATH.

Figure 2: **Distillation under reasoning token pruning.** Accuracy of a Llama2-7B student trained on pruned reasoning at varying keep fractions, teacher, pruner, and dataset; dashed lines indicate zero-shot performance. Greedy pruning achieves the strongest performance at matched lengths, indicating preservation of important tokens.

ing tokens using a learned notion of semantic importance, with supervision from GPT-4 (Achiam et al., 2023). `H2O` is a cache-eviction method that ranks tokens by aggregated attention from future steps. `Surprisal` defines importance by token prediction probability, where higher probability implies lower surprisal and importance. `Uniform` is a non-informative baseline assigning equal importance to all tokens. For all comparisons, reasoning chains are identical and pruned to the same keep fraction, with token counts measured using the student tokenizer to account for tokenizer differences. Baseline details are provided in Appendix B.3.

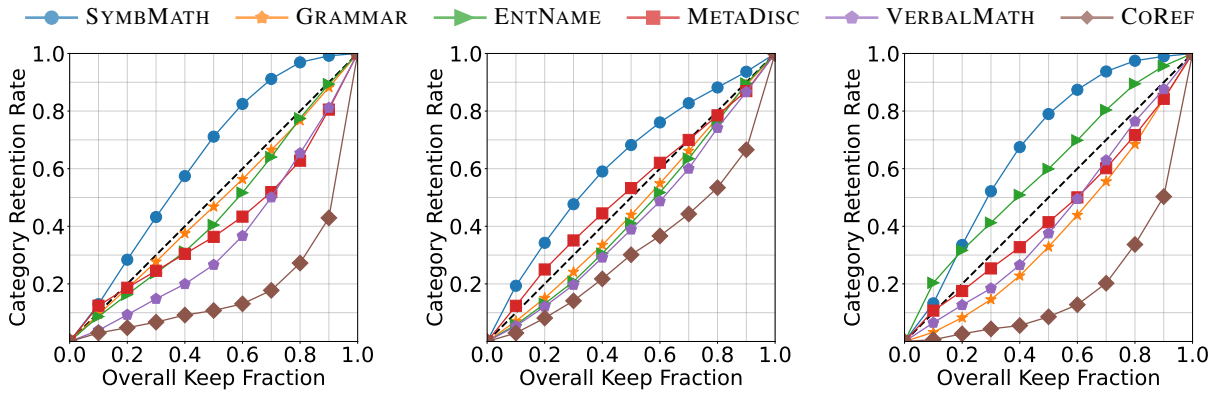
4.2 Results

Figure 2 reports downstream accuracy of student models trained on pruned reasoning traces across keep fractions, teacher–pruner configurations, and datasets. All pruning methods operate on identical teacher-generated reasoning chains, and all students are trained with the same SFT protocol. Under this controlled setup, distillation performance tests whether pruned reasoning retains the information needed for correct answer generation.

Across all settings, greedy pruning yields the strongest student performance among pruning-based methods at matched keep fractions. Accuracy improves smoothly as the keep fraction increases, indicating graceful degradation under compression. On GSM8K, students trained on greedy-pruned reasoning outperform `TokenSkip`, `H2O`, `Surprisal`, and `Uniform` across all keep fractions, for both Llama3.1-8B and Qwen2.5-7B teacher–pruner pairs (Figures 2a, 2b). `Uniform` and `Surprisal` drop sharply at aggressive pruning, while `TokenSkip` improves with higher keep fractions but remains below greedy pruning. Zero-shot student performance is shown for reference and is consistently worse than all pruning-based distillation methods. Similar trends hold on the more challenging MMLU-Pro and MATH benchmarks (Figures 2c, 2d): despite lower absolute accuracy, greedy pruning remains best at matched reasoning lengths. Additional results with Mistral-7B as the student show similar trends (Appendix C). Overall, pruned reasoning supports effective distillation across datasets, models, and keep fractions, consistently outperforming the frontier-model–supervised `TokenSkip` baseline.

We further analyze three key aspects of the method. First, we quantify the computational cost of greedy pruning (Appendix B.5), showing that while the naive implementation scales superlinearly in sequence length, it is a one-time offline preprocessing step and admits practical optimizations. Second, we evaluate the readability and structural integrity of pruned traces (Appendix E), finding that even at 30% token reduction, degradation is primarily surface-level, with semantic coherence and mathematical state largely preserved. Third, we study the behavior of students trained on pruned data (Appendix F), observing that they learn to generate proportionally shorter reasoning traces without meaningful loss in semantics or structure.

Together, these results indicate that greedy pruning removes redundant tokens while preserving



(a) $\mathcal{T}, \mathcal{P}$ = Qwen2.5-7B; $\mathcal{L}^{\text{JOINT}}_{\theta_{\mathcal{P}}}$. (b) $\mathcal{T}, \mathcal{P}$ = Qwen2.5-7B; $\mathcal{L}^{\text{ANS}}_{\theta_{\mathcal{P}}}$. (c) $\mathcal{T}$ = Qwen2.5-7B; $\mathcal{P}$ = Llama2-7B; $\mathcal{L}^{\text{JOINT}}_{\theta_{\mathcal{P}}}$.

Figure 3: **Functional structure under greedy pruning.** Each curve shows the fraction of tokens retained per category at a given keep fraction. Panels vary teacher, pruner, and pruning objective; the dashed line indicates uniform pruning. (a) Pruning preferentially preserves symbolic computation while removing referential, descriptive, and linguistic scaffolding. (b) Excluding reasoning likelihood in pruning objective softens induced functional structure. (c) A weaker pruner preserves symbolic computation but disrupts the balance of non-symbolic structure.

those critical for answer generation, motivating a closer analysis of the induced importance structure, which we study next.

5 Analysis of Pruning Behavior

5.1 Functional Structure Under Pruning

To analyze which reasoning tokens are preferentially preserved or removed under greedy pruning, we annotate each reasoning token by its *functional role in reasoning*, complementing prior reasoning-step–level analyses of reasoning structure (Marjanović et al., 2025) with a token-level functional characterization. This annotation allows us to track how different functional components evolve across pruning stages and to characterize the emergent structure of pruned reasoning chains. We conduct this analysis on 1,000 randomly sampled GSM8K examples, as GSM8K offers particularly interpretable token-level functional categories; extending this analysis to additional datasets is left for future work. Specifically, we define six coarse, interpretable functional categories: SymbMath (explicit equations and mathematical symbols), MetaDisc (reasoning narration and structuring), CoRef (pronouns and referential expressions), EntName (concrete entities), VerbalMath (natural-language arithmetic relations), and Grammar (grammatical fillers). Annotation is performed using `gpt-5-mini` and validated through manual inspection and stability checks; details are provided in Appendix D.1 and D.2.

Core functional structure. Figures 3a and 7a (Appendix D.3) plot the *category retention rate* as a function of the *overall keep fraction* for teacher–pruner settings where both models are Qwen2.5-7B and Llama3.1-8B, respectively, under the Joint objective. Each point at keep fraction k reports the average fraction of tokens within a category that remain after pruning to k, with the dashed diagonal indicating a uniform deletion baseline. Across both settings, greedy pruning exhibits a stable and interpretable functional ordering. SymbMath tokens are strongly over-retained throughout pruning, deviating substantially above the uniform baseline, indicating that explicit symbolic computations are preserved disproportionately even under aggressive compression. In contrast, CoRef tokens are markedly under-retained until late pruning stages, suggesting that explicit referential bookkeeping is among the earliest information removed. VerbalMath tokens also exhibit systematically lower retention than SymbMath, falling below the diagonal across a wide range of keep fractions, while Grammar, MetaDisc, and EntName tokens lie closer to the uniform baseline. Together with the token-frequency plots in Appendix D.3, these findings show that greedy pruning goes beyond category frequency, selectively preserving symbolic computation while removing referential, descriptive, and linguistic scaffolding. We provide analogous functional-structure analyses for baseline methods in Appendix D.3.

Effect of pruning objective. We next analyze how the pruning objective influences the induced

functional structure and its relationship to downstream distillation performance. Fixing a single teacher–pruner–student configuration ($\mathcal{T}, \mathcal{P}$ = Qwen2.5-7B, $\mathcal{S}$ = Llama2-7B) on GSM8K, we compare the JOINT objective to an answer-only (ANS) objective. As shown in Figure 3b, answer-only pruning preserves coarse functional ordering, with SYMBMATH remaining over-retained and COREF pruned early. However, separation across functional categories is weakened: retention curves for VERBALMATH, METADISC, and ENTNAME largely collapse toward the uniform baseline, and even SYMBMATH and COREF move closer to uniform deletion. This attenuation of functional structure aligns with lower student accuracy observed under the ANS objective (Figure 4), indicating that incorporating reasoning likelihood sharpens the structure revealed by greedy pruning and yields reasoning chains more effective for distillation.

Effect of pruner model strength. Finally, we study how pruner model strength affects the functional structure induced by greedy pruning and its downstream impact. Using the same configuration as for pruning objective, we replace the Qwen2.5-7B pruner with a weaker Llama2-7B pruner. Comparing Figures 3a and 3c, the overall ordering of SYMBMATH and COREF tokens remains stable, with symbolic computation preferentially retained and coreference pruned in both settings. In contrast, the treatment of non-symbolic categories shifts: ENTNAME tokens move from being under-retained to over-retained, GRAMMAR tokens are pruned more aggressively, and VERBALMATH moves closer to the uniform deletion baseline. This altered functional structure coincides with a larger drop in student accuracy observed under weaker pruner (Figure 4), suggesting that effective distillation depends not only on preserving symbolic computation but also on maintaining an appropriate balance of non-symbolic structure.

5.2 Dynamics of Pruning Ranks

A natural question raised by greedy pruning is whether it reveals a fixed global ordering of token importance or whether importance is dynamically reshaped as pruning progresses. If importance were static, rankings computed early in pruning should remain predictive. Here, we test whether greedy pruning instead continually re-evaluates token importance as the retained context contracts, a property that could help explain its advantage over

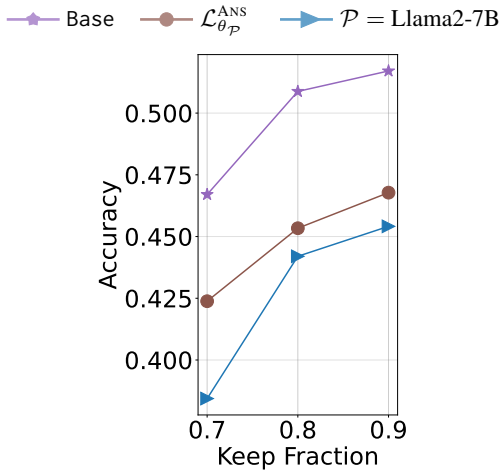


Figure 4: **Effect of pruning objective and pruner strength on distillation.** The base greedy setting uses $\mathcal{T}, \mathcal{P}$ = Qwen2.5-7B and $\mathcal{S}$ = Llama2-7B with the JOINT objective. We ablate pruner strength ($\mathcal{P}$ = Llama2-7B) and pruning objective (answer-only, ANS). Both weaken student performance, with pruner strength having the larger effect.

baselines relying on static importance signals.

Greedy pruning (Algorithm 1) assigns pruning ranks sequentially using post-deletion likelihoods as importance signals. At keep fraction ρ, for the remaining token set K_ρ, the pruner evaluates

$$L_i^{\text{del}} = \mathcal{L}_{\theta_{\mathcal{P}}}(Q, R_{K_\rho \setminus \{i\}}, A)$$

for each $i \in K_\rho$. Tokens with higher L_i^{del} incur smaller likelihood drops and are considered safer to remove earlier, inducing a *local pruning ranking* at each pruning stage. All experiments use the same GSM8K subset as Section 5.1, with Qwen2.5-7B as both teacher and pruner under the JOINT objective.

Figure 5 measures how well local pruning rankings predict subsequent pruning decisions across keep fractions. At each transition from ρ_{prev} to ρ_{curr}, a set of tokens

$$S_{\rho_{\text{curr}}} = K_{\rho_{\text{prev}}} \setminus K_{\rho_{\text{curr}}}$$

is pruned. We evaluate alignment using `Hit@|S|`, defined as the overlap between the top-ranked tokens under the local ranking at ρ_{prev} and the tokens actually pruned at ρ_{curr}.

Dynamic local rankings consistently outperform both a `Frozen=1.0` baseline, which reuses rankings computed at $\rho = 1.0$ to simulate a fixed global ordering, and a random baseline. The widening gap between dynamic and frozen rankings, especially at intermediate keep fractions, shows that

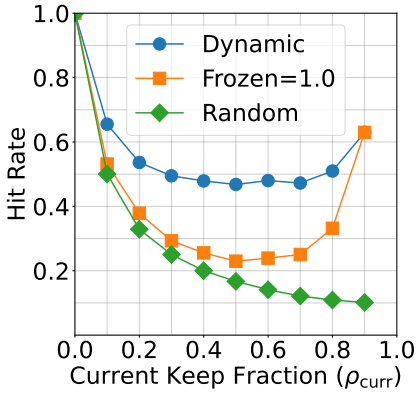


Figure 5: **Dynamics of pruning ranks.** `Hit@|S|` alignment between tokens removed at keep fraction ρ_{curr} and L^{del}-based local ranks at the previous pruning stage. Dynamic rankings ($\rho_{\text{prev}} = \rho_{\text{curr}} + 0.1$) consistently outperform frozen (ρ_{prev}=1.0) and random baselines across keep fractions, indicating that greedy pruning re-evaluates token importance as context contracts.

greedy pruning does not expose a static importance ordering early in pruning. Instead, token importance is continually re-evaluated as reasoning chain contracts, with later-stage local rankings remaining most predictive of imminent pruning decisions. This dynamic behavior highlights greedy pruning as an adaptive process rather than one governed by a fixed global notion of importance.

5.3 Predicting Pruning Ranks from Attention

We test whether the pruning ranks induced by greedy pruning are predictable from attention patterns alone. Inspired by prior work on attribution (Cohen-Wang et al., 2025), we train a two-layer MLP surrogate to predict token-level pruning scores using only attention signals from the teacher model. For each reasoning token, we construct a feature vector of dimension [`#layers` × `#heads`] by aggregating the average attention it receives from subsequent tokens across all future positions at each attention head.

The surrogate is trained to predict the post-deletion likelihood L^{del} associated with removing each token under greedy pruning, using a Pearson correlation objective that emphasizes relative importance rather than absolute scale. This setup allows us to test whether attention patterns encode sufficient information to predict the pruning order of reasoning tokens. Using only 200 GSM8K examples for training with Llama3.1-8B as the teacher, the surrogate achieves a Pearson correlation of 0.88 with true post-deletion likelihoods on a held-out set of 1,000 GSM8K examples. These results indicate that attention patterns encode information strongly predictive of greedy pruning decisions, and suggest a promising direction for incorporating such signals into more efficient pruning or search procedures.

6 Discussion & Future Directions

Greedy pruning can be viewed as a structured deletion operator for compressing reasoning chains under likelihood-based objectives, providing a controlled mechanism for exploring reasoning space in LLMs. Under this perspective, it represents one instance of a broader class of reasoning-space operators. Extending beyond deletion to token insertion or replacement under alternative objectives could enable finer-grained manipulation of reasoning trajectories, offering a more principled alternative to purely sampling-based control.

Beyond compression, greedy pruning also serves as a diagnostic probe of model-internal reasoning. By selectively removing tokens while preserving output likelihood, it reveals which parts of a model's own generated reasoning are functionally necessary for producing correct answers. Our results further show that pruning behavior depends on both the pruning objective and the strength of the pruner, suggesting that the induced importance structure reflects properties of the underlying model. Understanding how this structure varies across models, scales, and training procedures is a promising direction for future work.

Finally, we find that pruning ranks are strongly predictable from attention-based signals, indicating that token-level importance may be partially accessible during generation. This observation connects to recent work on inference-time pruning (Yang et al., 2025; Monea et al., 2025), which discards completed reasoning segments to manage context length. While our work focuses on likelihood-preserving post-hoc pruning as an analysis and distillation tool, extending similar objectives to inference-time settings or incorporating pruning signals into training-time curricula represents a natural next step toward more efficient and controllable reasoning systems.

Together, these directions suggest that pruning-based objectives may provide a unifying interface for both analyzing and controlling reasoning in LLMs.

7 Conclusion

We investigate whether LLMs encode token-level functional importance for answer generation within their reasoning chains. Using greedy pruning, a likelihood-preserving deletion procedure, we show that models can systematically compress reasoning while retaining information critical for answer generation, and that students trained on such pruned reasoning outperform multiple baselines at matched reasoning lengths. Our analyses reveal a stable and interpretable functional structure under pruning, with symbolic computation preferentially preserved and supporting linguistic scaffolding pruned earlier, and further show that pruning behavior depends on the pruning objective and pruner strength. We also demonstrate that pruning ranks are dynamically reshaped as context contracts and are strongly predictable from attention patterns, indicating that models encode internal signals of reasoning-token importance. Together, these findings establish greedy pruning as a principled diagnostic method for exposing token-level importance structure in model-generated reasoning.

Limitations

We study likelihood-based greedy pruning as a tool for exposing token-level importance structure in LLM reasoning. A primary limitation of this approach is its computational cost. Greedy pruning requires repeated likelihood evaluations over candidate token deletions, leading to superlinear scaling in sequence length and making naive implementations expensive for long reasoning chains. In our setting, this cost is incurred as a one-time offline preprocessing step and is amortized over downstream student training. Moreover, our goal is not to provide a universally scalable pruning algorithm but to analyze token-level importance at realistic reasoning lengths (e.g., $\leq$500 tokens), where the method is empirically tractable.

Several directions could further reduce computational costs. First, prefix caching[2] can reuse KV states for shared prefixes across deletion candidates, yielding constant-factor speedups. Second, the stability of local pruning ranks (Section 5.2) suggests that multiple tokens could be removed per iteration, reducing the number of recomputation steps. Third, attention-based or learned surrogate models could approximate token importance without full likelihood recomputation, as supported by the strong correlation between post-deletion likelihoods and attention-derived features (Section 5.3). We provide a detailed analysis of computational costs and scaling behavior in Appendix B.5.

Beyond efficiency, aggressive pruning can degrade the surface-level readability of reasoning traces. While our results show that moderate pruning primarily introduces formatting-level artifacts rather than loss of semantic or mathematical structure (Appendix E), we do not characterize the precise threshold at which reasoning becomes unsuitable for human interpretation. Developing principled criteria for reasoning interpretability under compression remains an open problem.

Additionally, our analysis is restricted to teacher-generated, correctness-filtered reasoning traces. It remains unclear how pruning behaves on incorrect or low-quality reasoning, where likelihood signals may be less aligned with functional importance.

Finally, our experiments are conducted on moderately sized instruction-tuned LLMs and do not extend to very large reasoning-focused models (e.g., DeepSeek-style (Guo et al., 2025b)) due to computational constraints. However, our ablations indicate that pruning quality is strongly governed by pruner strength (Section 5.1), suggesting that larger models may yield more reliable token-importance estimates. As a result, the qualitative trends observed in this work are likely to persist or strengthen at larger scales.

[2] https://docs.vllm.ai/en/stable/design/prefix_caching/

References

Samira Abnar and Willem Zuidema. 2020. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4197, Online. Association for Computational Linguistics.

OpenAI Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, and 260 others. 2023. Gpt-4 technical report.

Pranjal Aggarwal and Sean Welleck. 2025. L1: Controlling how long a reasoning model thinks with reinforcement learning. In *Second Conference on Language Modeling*.

Axolotl maintainers and contributors. 2023. Axolotl: Post-training for ai models.

Eric J Bigelow, Ari Holtzman, Hidenori Tanaka, and Tomer Ullman. 2025. Forking paths in neural text generation. In *The Thirteenth International Conference on Learning Representations*.

Paul C. Bogdan, Uzay Macar, Neel Nanda, and Arthur Conmy. 2025. Thought anchors: Which LLM reasoning steps matter? In *Submitted to The Fourteenth International Conference on Learning Representations*. Under review.

Karl Cobbe, Vineet Kosaraju, Mo Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *ArXiv*, abs/2110.14168.

Benjamin Cohen-Wang, Yung-Sung Chuang, and Aleksander Madry. 2025. Learning to attribute with attention. *ArXiv*, abs/2504.13752.

Yingqian Cui, Pengfei He, Jingying Zeng, Hui Liu, Xianfeng Tang, Zhenwei Dai, Yan Han, Chen Luo, Jing Huang, Zhen Li, Suhang Wang, Yue Xing, Jiliang Tang, and Qi He. 2025. Stepwise perplexity-guided refinement for efficient chain-of-thought reasoning in large language models. *ArXiv*, abs/2502.13260.

Ahmed El-Kishky. 2024. Openai o1 system card. *ArXiv*, abs/2412.16720.

Joseph Enguehard. 2023. Sequential integrated gradients: a simple but effective method for explaining language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7555–7565, Toronto, Canada. Association for Computational Linguistics.

Sicheng Feng, Gongfan Fang, Xinyin Ma, and Xinchao Wang. 2025. Efficient reasoning models: A survey. *Transactions on Machine Learning Research*.

Leo Gao. 2023. Shapley value attribution in chain-of-thought. Alignment Forum.

Daya Guo, Dejian Yang, Haowei Zhang, Jun-Mei Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiaoling Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025a. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645:633 – 638.

Daya Guo, Dong Yang, Hao Zhang, and 1 others. 2025b. Deepseek-r1 incentivizes reasoning in large language models through reinforcement learning. *Nature*, 645:633–638.

Michael Hassid, Gabriele Synnaeve, Yossi Adi, and Roy Schwartz. 2025. Don't overthink it. preferring shorter thinking chains for improved llm reasoning. *ArXiv*, abs/2505.17813.

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

Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. 2025. Thinkprune: Pruning long chain-of-thought of LLMs via reinforcement learning. *Submitted to Transactions on Machine Learning Research*. Under review.

Yifan Hou, Jiaoda Li, Yu Fei, Alessandro Stolfo, Wangchunshu Zhou, Guangtao Zeng, Antoine Bosselut, and Mrinmaya Sachan. 2023. Towards a mechanistic interpretation of multi-step reasoning capabilities of language models. *ArXiv*, abs/2310.14491.

Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. 2025. C3ot: generating shorter chain-of-thought without compromising effectiveness. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson E. Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, John Kernion, Kamil.e Lukovsiut.e, Karina Nguyen, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Robin Larson, Sam McCandlish, Sandipan Kundu, and 11 others. 2023. Measuring faithfulness in chain-of-thought reasoning. *ArXiv*, abs/2307.13702.

Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. 2023. Compressing context to enhance inference efficiency of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6342–6353, Singapore. Association for Computational Linguistics.

Tengxiao Liu, Qipeng Guo, Xiangkun Hu, Cheng Jiayang, Yue Zhang, Xipeng Qiu, and Zheng Zhang. 2024. Can language models learn to skip steps? In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA. Curran Associates Inc.

Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. 2025. O1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning. *ArXiv*, abs/2501.12570.

Sara Vera Marjanović, Arkil Patel, Vaibhav Adlakha, Milad Aghajohari, Parishad BehnamGhader, Mehar Bhatia, Aditi Khandelwal, Austin Kraft, Benno Krojer, Xing Han Lù, Xing Han Lù, Nicholas Meade, Dongchan Shin, Amirhossein Kazemnejad, Gaurav Kamath, Marius Mosbach, Karolina Sta'nczak, Karolina Stańczak, and Siva Reddy. 2025. Deepseek-r1 thoughtology: Let's think about llm reasoning.

Giovanni Monea, Yair Feldman, Shankar Padmanabhan, Kianté Brantley, and Yoav Artzi. 2025. Breadcrumbs reasoning: Memory-efficient reasoning with compression beacons. *ArXiv*, abs/2510.13797.

Neel Nanda, Josh Engels, Arthur Conmy, Senthooran Rajamanoharan, Bilal Chughtai, Callum McDougall, János Krámár, and Lewis Smith. 2025. A pragmatic vision for interpretability. https://www.alignmentforum.org/posts/StENzDcD3kpfGJssR/a-pragmatic-vision-for-interpretability. AI Alignment Forum.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, Dongmei Zhang, Karl Cobbe, Vineet Kosaraju, Mo Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, and 4 others. 2024. Llmlingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In *Annual Meeting of the Association for Computational Linguistics*.

Ziqing Qiao, Yongheng Deng, Jiali Zeng, Dong Wang, Lai Wei, Guanbo Wang, Fandong Meng, Jie Zhou, Ju Ren, and Yaoxue Zhang. 2025. ConCISE: Confidence-guided compression in step-by-step efficient reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 8021–8040, Suzhou, China. Association for Computational Linguistics.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 1135–1144, New York, NY, USA. Association for Computing Machinery.

Claude E. Shannon. 1948. A mathematical theory of communication. *Bell Syst. Tech. J.*, 27:623–656.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3319–3328. JMLR.org.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024. MMLU-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Skyler Wu, Eric Meng Shen, Charumathi Badrinath, Jiaqi W. Ma, and Himabindu Lakkaraju. 2023. Analyzing chain-of-thought prompting in large language models via gradient-based feature attributions. *ArXiv*, abs/2307.13339.

Heming Xia, Chak Tou Leong, Wenjie Wang, Yongqi Li, and Wenjie Li. 2025. TokenSkip: Controllable chain-of-thought compression in LLMs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 3351–3363, Suzhou, China. Association for Computational Linguistics.

Chenxiao Yang, Nathan Srebro, David McAllester, and Zhiyuan Li. 2025. PENCIL: Long thoughts with short memory. In *Forty-second International Conference on Machine Learning*.

Jingyang Yi, Justin Wang, and Sida Li. 2025. Shorter-better: Guiding reasoning models to find optimal inference length for efficient reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Kayo Yin and Graham Neubig. 2022. Interpreting language models with contrastive explanations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 184–198, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Matthew D. Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *Computer Vision – ECCV 2014*, pages 818–833, Cham. Springer International Publishing.

Wenhao Zeng, Yaoning Wang, Chao Hu, Yuling Shi, Chengcheng Wan, Hongyu Zhang, and Xiaodong Gu. 2025. Pruning the unsurprising: Efficient code reasoning via first-token surprisal. *ArXiv*, abs/2508.05988.

Zhenyu (Allen) Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark W. Barrett, Zhangyang Wang, and Beidi Chen. 2023. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *ArXiv*, abs/2306.14048.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2024. Explainability for large language models: A survey. *ACM Trans. Intell. Syst. Technol.*, 15(2).

Luisa M Zintgraf, Taco S Cohen, Tameem Adel, and Max Welling. 2017. Visualizing deep neural network decisions: Prediction difference analysis. In *International Conference on Learning Representations*.

A LLM Usage

Other than being used as part of the experiments conducted in this work, LLMs were used solely as a writing assistance tool in preparing this paper submission. Their role was limited to polishing language, improving clarity, and reducing redundancy. The prompt used for this purpose was similar to "Please revise the writing of this, making sure to remove any grammatical mistakes." All research ideas, experimental designs, analyses, and claims presented in the paper are entirely the original work of the authors. No part of the conceptual, methodological, or empirical contributions relies on or originates from LLM outputs.

B Implementation Details

B.1 Datasets

We conduct experiments on three reasoning-intensive benchmarks: (i) GSM8K (Cobbe et al., 2021), consisting of grade-school arithmetic word problems; (ii) MMLU-Pro (Wang et al., 2024), a multi-domain benchmark spanning diverse domains or subjects; and (iii) MATH (Hendrycks et al., 2021), containing Olympiad-level mathematical reasoning problems. Hugging Face identifiers for original datasets used are summarized in Table 1. As these benchmarks differ in their provided train/dev/test splits, we standardize the evaluation protocol by constructing explicit training, development, and test splits as described below.

- **GSM8K.** The original dataset provides 7.47K training examples and 1.32K test examples. We randomly partition the training split into 6.97K training examples and 0.5K development examples while keeping the original test split intact. The resulting split is released at `greedy-token-pruner/gsm8k`.
- **MMLU-Pro.** The original dataset does not provide a standard training split. We therefore randomly partition the available 12K examples into 10K training, 0.8K development, and 1.23K test examples. The resulting split is released at `greedy-token-pruner/MMLU-Pro`.
- **MATH.** The dataset provides 12K training examples and a 500-example test set. We randomly split the training data into 11.2K training examples and 0.8K development examples and evaluate on the original MATH-500 test set. The resulting split is released at `greedy-token-pruner/openaimath`.

We use these fixed, constructed splits to conduct all the experiments.

Shorthand	HuggingFace Identifier
Llama3.1-8B	`meta-llama/Llama-3.1-8B-Instruct`
Qwen2.5-7B	`Qwen/Qwen2.5-7B-Instruct`
Llama2-7B	`meta-llama/Llama-2-7b-chat-hf`
Mistral-7B	`mistralai/Mistral-7B-Instruct-v0.3`
GSM8K	`openai/gsm8k`
MMLU-Pro	`TIGER-Lab/MMLU-Pro`
MATH	`simplescaling/openaimath`

Table 1: Mapping from shorthand model and dataset names to their corresponding Hugging Face identifiers.

Model	GSM8K	MMLU-Pro	MATH-500
Llama2-7B	25.02	15.58	4.60
Mistral-7B	31.46	30.28	13.40
Qwen2.5-7B	90.98	56.25	65.80
Llama3.1-8B	82.34	45.21	43.60

Table 2: **Zero-shot Performance.** Accuracy (%) of models on the test splits of the datasets.

B.2 Teacher–Pruner–Student Configuration

We instantiate the teacher–pruner–student distillation framework with four instruction-tuned models and the datasets listed in Table 1. To justify model role assignments, we first evaluate each model's zero-shot performance on the corresponding test splits (Table 2). All zero-shot evaluations use deterministic greedy decoding (`temperature=0.0, n=1, top_p=1.0`). Based on these results, we designate Qwen2.5-7B and Llama3.1-8B as teacher models and Llama2-7B and Mistral-7B as student models. This choice follows standard distillation practice: stronger teachers provide higher-quality reasoning supervision, while weaker students ensure meaningful headroom for knowledge and reasoning transfer. Unless otherwise stated, we set the pruner model equal to the teacher to avoid confounding pruning quality with pruner capacity. To isolate the effect of pruner strength (Section 5.1), we additionally instantiate the pruner as Llama2-7B while holding the teacher fixed. We use a shared prompt per dataset to ensure comparability across experiments; the exact system prompts and user prompt templates are provided below.

Prompt: GSM8K

SYSTEM PROMPT

```
You are a helpful assistant. Please solve the
      following problem step by step. At the end,
      output the final **answer only in the JSON
      format**:\n\n{"answer": "[numeric answer
      only]"}
```

USER PROMPT TEMPLATE

```
{{question}}
```

Prompt: MMLU-Pro

SYSTEM PROMPT

```
You are a helpful assistant. Please solve the
      following MCQ problem step by step to select
      the correct answer from the given options. At
      the end, output the final answer only in the
      following format: "Thus, the final answer is:
      [correct letter choice only]"
```

USER PROMPT TEMPLATE

```
{{question}}
```

Prompt: MATH-500

SYSTEM PROMPT

```
You are a helpful assistant expert at solving math
      problems. Please solve the following math
      problem. First, think through the problem step
      by step. At the end, output the final answer
      only in the following format: "Thus, the final
      answer is: \\boxed{[numeric or mathematical
      expression only]}"
```

USER PROMPT TEMPLATE

```
{{question}}
```

B.3 Baselines

We evaluate the quality of greedy-pruned reasoning chains in a teacher–pruner–student distillation framework by comparing them with pruning baselines that induce a token-level importance (or an equivalent pruning rule). We describe each baseline in detail below.

- `TokenSkip` (Xia et al., 2025). `TokenSkip` prunes reasoning chains using a learned notion of token-level semantic importance. Similar to our setup, `TokenSkip` first generates reasoning chains with a teacher model and then prunes them offline. Token-level importance is predicted by a bidirectional Transformer trained for token-importance estimation (Pan et al., 2024), using supervision derived from frontier models (e.g., GPT-4 (Achiam et al., 2023)). For a token at position t, semantic importance is given by the predicted probability of the "important" class,

$$I(t) = P_\phi(y_t = 1 \mid x_{1:T}),$$

where P_ϕ denotes the importance-classification head and $x_{1:T}$ is the full reasoning sequence. Tokens are ranked by $I(t)$, and those with lower predicted importance are pruned first. For a fair comparison, we use the same released semantic-importance model (`llm-lingua-2-xlm-roberta-large`) as in `TokenSkip` to prune reasoning tokens.

- `H2O` (Zhang et al., 2023) is a KV-cache eviction method that identifies "heavy-hitter" tokens based on accumulated attention. Specifically, for a token at position t, `H2O` defines its importance as the cumulative attention it receives from future generation steps,

$$I(t) = \sum_{i=t+1}^{T} \sum_{l,h} A_{i \to t}^{(l,h)},$$

where $A_{i \to t}^{(l,h)}$ denotes the attention weight from token i to token t at layer l and head h. Tokens with low cumulative attention are evicted from the KV cache. We note that `H2O` was originally designed for KV-cache eviction rather than explicit token deletion. Under KV-cache eviction, information associated with a token may still be indirectly accessible via the KV-cache of other tokens, whereas token pruning removes the token entirely from the sequence. In our experiments, we adapt this notion of importance to token pruning by ranking reasoning tokens according to $I(t)$ and deleting the lowest-ranked tokens.

- `Surprisal` (Li et al., 2023). Surprisal (self-information) quantifies the information content of a token under a given probability distribution (Shannon, 1948). For a token at position t, we compute its surprisal under the teacher model using teacher forcing on the full reasoning chain,

$$I(t) = -\log P(x_t \mid x_{<t}).$$

Unlike attention-based importance, surprisal reflects local token predictability rather than task-level or reasoning-specific importance. Tokens are ranked by $I(t)$, and those with lower surprisal (higher probability) are pruned first.

- `Uniform`. As a non-informative baseline, we delete reasoning tokens uniformly at random to match the target keep ratio.

For fair comparison across methods, we define keep ratios using the student tokenizer and ensure that all pruned reasoning chains contain same number of student-tokenizer tokens.

B.4 Training and Inference

Below, we describe the training and inference implementation details for different stages of our setup:

- **Reasoning Chain Generation (Teacher).** We generate reasoning chains via rejection sampling, retaining only responses with correct final answers. For each question, we sample 10 responses using temperature-based decoding (temperature = 0.7, top-p = 1.0) and randomly select one correct response for training; questions with no correct responses are discarded. This procedure yields a sufficient number of high-quality reasoning trajectories per dataset for downstream pruning and distillation. All inference is performed using vLLM.

  We construct four effective training datasets for student models under the following teacher–pruner–dataset configurations: (a) Llama3.1-8B as both teacher and pruner on GSM8K; (b) Qwen2.5-7B as both teacher and pruner on GSM8K; (c) Llama3.1-8B as both teacher and pruner on MMLU-Pro; and (d) Qwen2.5-7B as both teacher and pruner on MATH. After reasoning chain generation, the resulting dataset sizes for (a)–(d) are 6,714, 6,650, 5,500, and 3,800 examples, respectively. We do not use the full training splits listed in Appendix due to the computational cost of pruning and student training, as well as example filtering during rejection sampling.

- **Likelihood Computation (Pruner).** To perform greedy pruning, we represent each question–reasoning–answer instance as a sequence of token IDs and compute post-deletion likelihoods by removing one reasoning token at a time. All likelihoods are computed under teacher forcing.

  We use the vLLM inference engine (Kwon et al., 2023), which supports efficient token-level log-probability evaluation for a fixed input sequence. For both pruning objectives, we compute token-level log-probabilities under teacher forcing and aggregate them to obtain sequence-level likelihoods. Specifically, the JOINT objective is computed as

$$\mathcal{L}^{\text{JOINT}}_{\theta_\mathcal{P}}(Q, R_K, A) = \sum_{t=1}^{|R_K|+|A|} \log P_{\theta_\mathcal{P}}(y_t \mid Q, y_{<t}), \quad (3)$$

  where $\{y_t\}$ denotes the concatenation of the pruned reasoning tokens R_K followed by the answer tokens A. The answer-only objective is computed as

$$\mathcal{L}^{\text{ANS}}_{\theta_\mathcal{P}}(Q, R_K, A) = \sum_{t=1}^{|A|} \log P_{\theta_\mathcal{P}}(a_t \mid Q, R_K, a_{<t}). \quad (4)$$

  During each pruning step, all deletion candidates yield sequences of identical length. As a result, comparing post-deletion likelihoods using either summed or mean log-probabilities induces the same ranking, and we use the summed formulation throughout.

  When pruning a single example, many deletion candidates share long common prefixes, since the input sequences differ only by the removal of a single token. While vLLM provides automatic prefix caching[3] for text generation, this functionality does not currently extend to log-probability computation. Extending prefix caching to likelihood evaluation could substantially reduce the computational cost of greedy pruning.

- **Distillation Training (Student).** All supervised fine-tuning (SFT) experiments are implemented using the AXOLOTL framework (Axolotl maintainers and contributors, 2023). We sweep learning rates in $\{1 \times 10^{-6}, 3 \times 10^{-6}, 5 \times 10^{-6}, 1 \times 10^{-5}, 3 \times 10^{-5}, 5 \times 10^{-5}\}$ using a cosine decay scheduler. Across all Llama2-7B training runs, a learning rate of 3×10^{-5} consistently yields the best performance, whereas Mistral-7B prefers a lower learning rate of 5×10^{-6}. All student models are trained with an effective batch size of 28 for 1,200 gradient steps, and the best checkpoint is selected using validation set performance.

B.5 Computational Cost and Scaling of Greedy Pruning

Greedy pruning incurs a nontrivial computational cost due to repeated likelihood evaluations under token deletions. For a reasoning trace with n tokens and a target keep fraction of ρ, the algorithm evaluates $\mathcal{O}((1-\rho)n)$ pruning steps. At each step, all remaining tokens are considered deletion candidates, resulting in a total of approximately $\mathcal{O}((1-\rho)n^2)$ forward passes. Each candidate evaluation requires

[3] https://docs.vllm.ai/en/stable/features/automatic_prefix_caching/

scoring the full sequence under the pruner model. Under standard transformer implementations, this corresponds to $\mathcal{O}(n^2)$ self-attention costs per forward pass. Thus, the naive worst-case FLOP complexity is $\mathcal{O}((1-\rho)n^4)$, which can become expensive for longer reasoning traces.

In practice, however, candidate evaluations within each pruning step are independent and can be batched. We implement greedy pruning using `vLLM` with batched forward passes, so wall-clock time scales with the number of batches rather than with fully sequential execution. Empirically, in our GSM8K setting (average sequence length ≈ 250 tokens), pruning 8K samples from keep ratio $\rho = 1.0 \rightarrow 0.7$ required approximately 20 hours on 8×H100 GPUs. This cost is incurred once as an offline preprocessing step and is amortized over subsequent student training.

Effect of sequence length. The quadratic dependence on sequence length implies that greedy pruning becomes increasingly expensive for longer generations (e.g., $>$ 1K tokens). While our experiments demonstrate feasibility for moderately long reasoning traces (up to $\sim$500 tokens), scaling to substantially longer outputs would require additional approximations.

Optimization opportunities. As previously mentioned, several optimizations can reduce the practical runtime:

- **Prefix KV-cache reuse.** For deletion candidates that share common prefixes, key–value states can be reused exactly, reducing redundant computation. While this does not change asymptotic complexity, it yields a meaningful constant-factor speedup.

- **Block or local multi-token deletion.** Instead of removing a single token per step, one can remove a small set of high-ranked tokens and recompute importance locally. This reduces the number of pruning iterations at the cost of introducing approximation.

- **Surrogate importance models.** Section 5.3 shows that attention-based features correlate strongly with pruning ranks. Lightweight surrogate models can approximate token importance and reduce the number of exact likelihood evaluations.

Discussion. We emphasize that greedy pruning is designed as an *offline analysis and data construction procedure*, rather than a test-time algorithm. Our goal is to expose token-level functional structure and enable efficient distillation, rather than to provide a universally scalable pruning method for arbitrarily long generations. Improving scalability remains an important direction for future work.

C Additional Distillation Results

We report additional distillation results using Mistral-7B as the student model, shown in Figure 6. Consistent with results using Llama2-7B as the student (Section 4.2), greedy pruning produces pruned reasoning chains that achieve the strongest distillation performance among baseline methods. These findings further support the conclusion that greedy pruning preserves functionally important reasoning tokens and indicate that the observed trends generalize across diverse student models.

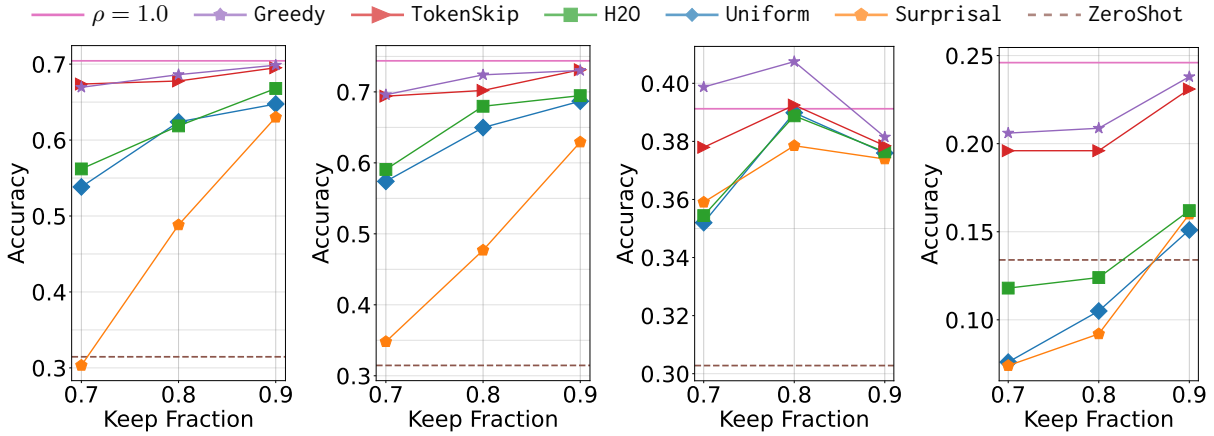
D Functional Structure Under Pruning

D.1 Categorization Scheme

To analyze which tokens of a model's reasoning are preserved or removed under greedy pruning, we annotate each token in the generated reasoning chain according to its *functional role in reasoning*. Broadly, we distinguish among tokens that organize or narrate the reasoning process, tokens that support grammatical fluency or referential bookkeeping, and tokens that encode arithmetic content. Specifically, we assign each token *exactly one* of six categories:

- SYMBMATH. SYMBOLICMATH includes explicit numeric or symbolic computation, including digits, currency symbols, arithmetic operators, equations, fractions, and numeric constants.
- METADISC. METADISCOURSE includes tokens that organize, narrate, or scaffold the reasoning process. Arithmetic verbs (e.g., add, subtract) are labeled METADISCOURSE only when used to narrate a step, not when expressing a mathematical relation. *Examples:* "to find", "we need to", "step", "first", "now", "so", "let's", "calculate", "the final answer is", step nos., list markers.
- COREF. COREFERENCE includes pronouns and referential expressions that refer to previously introduced entities or quantities. *Examples:* "she", "her", "it", "they", "we", "this".
- ENTNAME. ENTITYNAME includes proper names and concrete entities central to the problem, including people, objects, units, and counted nouns. *Examples:* "Natalia", "Julie", "wallet", "book", "pages", "year".
- VERBALMATH. This category includes tokens that describe arithmetic relations or quantities in natural language. Arithmetic verbs (e.g., add, multiply, divide) are labeled VERBALMATH only when describing the operation itself, not when narrating a step. *Examples:* "half", "twice", "total", "remaining", "more", "per", "rate".
- GRAMMAR. This category includes tokens that act as grammatical fillers with minimal standalone content, including articles, prepositions, conjunctions, auxiliary verbs, punctuation, formatting tokens, and whitespace.

All tokens are labeled independently with exactly one category, subject to the constraint that all subword pieces of the same word share the same



(a) $\mathcal{T}, \mathcal{P}$ = Llama3.1-8B; GSM8K. (b) $\mathcal{T}, \mathcal{P}$ = Qwen2.5-7B; GSM8K. (c) $\mathcal{T}, \mathcal{P}$ = Llama3.1-8B; MMLU-Pro. (d) $\mathcal{T}, \mathcal{P}$ = Qwen2.5-7B; MATH.

Figure 6: **Distillation under reasoning token pruning.** Accuracy of a Mistral-7B student trained on pruned reasoning at varying keep fractions, teacher, pruner, and dataset; dashed lines indicate zero-shot performance. Greedy pruning achieves the strongest performance at matched lengths, indicating preservation of important tokens.

category. Furthermore, when a token could fit multiple categories, we resolve ties using a fixed priority order to ensure deterministic annotation (SYMBMATH > METADISC > COREF > ENTNAME > VERBALMATH > GRAMMAR).

D.2 Annotation Process Details

We annotate reasoning tokens with `gpt-5-mini-2025-08-07` under fixed system and user prompts. The system prompt specifies the annotator role and meta-task instructions, while the user prompt defines the annotation task, category definitions, and output constraints; both are included at the end of this section. Although `gpt-5-mini` does not expose sampling settings (e.g., temperature), the labeling procedure is highly stable under a fixed prompt. Rerunning the annotation on the same reasoning traces yields a mean token-level agreement of >0.90 and a median agreement of 0.92 across 1,000 GSM8K examples, with agreement exceeding 0.97 for 90% of examples. Because our goal is to capture coarse structural trends rather than fine-grained linguistic distinctions, we employ lightweight validation rather than full-scale human annotation. We additionally randomly sample 20 reasoning traces (approximately 6,000 tokens) and manually inspect the assigned labels, observing no systematic category collapse or prompt-induced artifacts. Together, these checks indicate that the labeling procedure is sufficiently stable for analyzing relative pruning behavior across functional categories.

D.3 Additional Results

Figure 7 presents additional functional-structure analyses comparing greedy pruning to two baseline pruning criteria, `TokenSkip` and `Surprisal`. Consistent with the main analysis, `TokenSkip` exhibits a markedly different functional structure from greedy pruning. While symbolic content is retained to some extent, SYMBMATH no longer shows strong preferential retention relative to other categories and instead tracks closer to ENTNAME and VERBALMATH. At the same time, GRAMMAR tokens are pruned aggressively across most keep fractions, and COREF tokens are removed almost entirely at early stages. Overall, the retention curves are more tightly clustered, indicating weaker separation across functional roles. This pattern suggests that semantic-importance signals derived from frontier models emphasize surface-level semantic salience rather than the internal computational structure supporting symbolic reasoning.

`Surprisal`-based pruning induces an even less aligned functional structure. High-probability tokens, including SYMBMATH, are pruned early, leading to under-retention of symbolic computation relative to descriptive categories. In contrast, METADISC and VERBALMATH are over-retained, reflecting the tendency of surprisal to preserve less predictable narrative or descriptive tokens rather than computation-critical ones. GRAMMAR and ENTNAME show inconsistent retention behavior across keep fractions, with patterns largely driven by local token predictability rather than reasoning function. As a result, `surprisal` produces a

functional structure that is misaligned with the requirements of reasoning.

In contrast, greedy pruning induces a clearer and more interpretable functional ordering, sharply separating symbolic computation from supporting linguistic scaffolding. These structural differences provide additional context for the weaker distillation performance observed for `TokenSkip` and `Surprisal` relative to greedy pruning.

Figure 8 additionally reports the token-level functional category distribution for reasoning traces generated by Qwen2.5-7B and Llama3.1-8B prior to pruning. These distributions provide context for the functional-structure analyses by showing the baseline prevalence of each category in unpruned reasoning traces.

E Comprehensibility and SFT Suitability of Pruned Reasoning

We evaluate the structural and semantic quality of reasoning traces produced by greedy pruning. While our method optimizes a likelihood-based objective, it is important to verify that the resulting supervision remains suitable for training and interpretable to humans.

E.1 Evaluation Protocol

We evaluate pruned reasoning traces on GSM8K using Llama3.1-8B as the teacher and pruner. For each keep fraction $\rho \in \{0.7, 0.8, 0.9\}$, we sample 1,000 reasoning traces after pruning.

Each trace is evaluated along five dimensions:

- **Corruption**: surface-level degradation of text.
- **Structural integrity**: preservation of a stepwise reasoning structure.
- **Semantic coherence**: logical consistency and readability.
- **Mathematical state preservation**: whether key variables, intermediate quantities, and final answers are present and consistent.
- **Suitability for SFT**: whether the trace is usable as supervision for fine-tuning.

Annotations are generated using an LLM-based evaluator (temperature $= 0$) under a fixed JSON schema to ensure deterministic outputs. To validate annotation quality, we manually inspect 50 randomly sampled traces at $\rho = 0.7$.

E.2 Annotation Rubric

Each dimension is labeled using a discrete scale:

Corruption

- **OK**: fully readable text.
- **Minor**: surface artifacts such as missing punctuation, dropped function words, or formatting noise; meaning remains recoverable.
- **Severe**: fragmented or malformed text that is difficult to parse.

Structural Integrity

- **Intact**: step structure preserved.
- **Mildly broken**: minor step merging or header loss; order remains recoverable.
- **Broken**: reasoning flow is difficult to reconstruct.

Semantic Coherence

- **OK**: logically understandable.

- **Minor**: connective or phrasing gaps, but reasoning remains clear.
- **Severe**: logical flow is disrupted.

Mathematical State Preservation

- **Yes**: key quantities and equations are present and numerically consistent.
- **Borderline**: minor omissions such as units; state remains inferable.
- **No**: critical quantities are missing or inconsistent.

Suitability for SFT

- **Yes**: directly usable supervision.
- **Borderline**: usable with minor cleanup.
- **No**: too degraded for training.

E.3 Results

Table 3 reports results across keep fractions.

Metric	Category	$\rho = 0.7$	$\rho = 0.8$	$\rho = 0.9$
Corruption	OK	3.9%	5.1%	22.7%
	Minor	96.1%	94.9%	77.3%
	Severe	0.0%	0.0%	0.0%
Structural Integrity	Intact	57.3%	78.8%	92.2%
	Mildly broken	42.4%	21.2%	7.8%
	Broken	0.4%	0.0%	0.0%
Semantic Coherence	OK	62.0%	76.1%	92.2%
	Minor	38.0%	23.9%	7.8%
	Severe	0.0%	0.0%	0.0%
Math State Preserved	Yes	94.5%	97.3%	99.6%
	Borderline	5.5%	2.7%	0.4%
	No	0.0%	0.0%	0.0%
SFT Suitability	Yes	67.1%	85.9%	94.1%
	Borderline	32.9%	14.1%	5.9%
	No	0.0%	0.0%	0.0%

Table 3: Evaluation of pruned reasoning traces across multiple quality dimensions and keep fractions ρ.

E.4 Analysis

At $\rho = 0.7$ (approximately 30% token reduction), degradation is primarily superficial. Most traces exhibit minor corruption, such as missing punctuation or dropped function words, but remain semantically coherent and structurally interpretable. Severe corruption is absent across all settings. Structural failures are rare (0.4% at $\rho = 0.7$), and no examples exhibit a loss of mathematical state. In all cases, final answers remain present and numerically consistent. Suitability for SFT remains high even at aggressive pruning levels. At $\rho = 0.7$, 67.1% of traces are directly usable, and the remaining ones are borderline, requiring only minor cleanup. No traces are deemed unusable. Manual inspection confirms that degradation is dominated by formatting artifacts rather than logical errors. Key reasoning steps, intermediate quantities, and final answers are preserved.

Takeaways: These results suggest that greedy pruning preserves the semantic and mathematical integrity of reasoning traces even under substantial compression. While surface fluency degrades at lower keep fractions, this effect is largely orthogonal to the underlying reasoning and can be addressed with lightweight post-processing if desired. Overall, pruned traces remain suitable supervision for distillation and support the use of greedy pruning as a token-level compression method.

Prompt: Corruption Annotation

```
SYSTEM PROMPT

You are a strict annotator, evaluating the
      corruption of a pruned response compared to
      the original response.

USER PROMPT TEMPLATE

You are evaluating whether a pruned response is
      corrupted, using the following rubric.

Your task has TWO STEPS, to be performed and output
      in order:

Step 1 (Reasoning): Compare the PRUNED response to
      the ORIGINAL response step by step, in order.
- For each step, directly compare the pruned and
      full response step (first with first, second
      with second, etc).
- For each pair of steps, write a single line of
      analysis that considers: corruption
      (surface-level preservation, missing words,
      fragments, readability), structural integrity
      (step structure), semantic coherence (clarity
      of reasoning), math state
      (clarity/preservation of quantities), and
      suitability for SFT.
- For each step, clearly state what is intact, what
      is mildly broken, and what is severely
      degraded according to the rubric.
- Do not evaluate answer correctness; focus only on
      surface damage, structure, and coherence.
- Write one line of analysis per step, in order,
      each on its own NEW LINE, precisely covering
      all rubric metrics for the step-by-step
      differences.

Step 2 (Label): On a NEW LINE, output ONLY the final
      JSON (no extra text before or after it), using
      EXACTLY this schema and label space:

{
  "corruption": "ok|minor|severe",
  "structural_integrity":
      "intact|mildly_broken|broken",
  "semantic_coherence": "ok|minor|severe",
  "math_state_preserved": "yes|borderline|no",
  "suitable_for_sft": "yes|borderline|no",
  "notes": ""
}

Rubric (for labeling):
- corruption:
    * ok (readable/coherent),
    * minor (surface artifacts such as missing
      punctuation or truncated fragments but meaning
      recoverable),
    * severe (hard to parse or nonsensical)
- structural_integrity: intact / mildly_broken /
      broken (whether step structure survives)
- semantic_coherence: ok / minor / severe (whether
      the reasoning remains understandable to a
      human reader)
- math_state_preserved: yes (main intermediate
      quantities/equations clearly present and
```

Prompt: Corruption Annotation (continued)

```
      consistent) / borderline (some degradation) /
      no (missing or inconsistent)
- suitable_for_sft: yes (readable and structurally
      usable for SFT), borderline (usable but mildly
      degraded), no (too broken for practical SFT)
- notes: If everything is perfect (corruption="ok"
      AND structural_integrity="intact" AND
      semantic_coherence="ok"), notes MUST be "". If
      suitable_for_sft="no", notes may still be "".
      Otherwise, use <= 15 words describing the
      surface-level defects.

Remember:
- Do NOT evaluate correctness of the answer.
- Assess only readability, structure, and coherence
      of the PRUNED text versus the FULL text.

Question:
{{question}}

Original Response (full):
{{reason}}

Pruned Response:
{{pruned_reason}}

First think step-by-step about each corresponding
      step in the pruned response and the original
      response. Then return the JSON. Return EXACTLY
      this JSON schema:
{
  "corruption": "ok|minor|severe",
  "structural_integrity":
      "intact|mildly_broken|broken",
  "semantic_coherence": "ok|minor|severe",
  "math_state_preserved": "yes|no|unclear",
  "suitable_for_sft": "yes|borderline|no",
  "notes": ""
}
```

F Student Behavior Under Pruned-Trace Distillation

In this section, we analyze how student models behave after training on greedily pruned reasoning traces. In particular, we study whether (i) students learn to generate shorter reasoning chains consistent with the pruned supervision, and (ii) whether this compression leads to degradation in structural integrity or semantic coherence.

F.1 Reasoning Length Adaptation

We first measure the length of reasoning traces generated by student models trained with different keep fractions $\rho \in \{1.0, 0.9, 0.8, 0.7\}$. Table 4 reports the average number of generated reasoning tokens on the GSM8K validation set.

Keep Fraction ρ	Student Output Length
1.0	189 tokens
0.9	161 tokens ($\sim$0.85$\times$ of $\rho = 1.0$)
0.8	148 tokens ($\sim$0.78$\times$ of $\rho = 1.0$)
0.7	134 tokens ($\sim$0.70$\times$ of $\rho = 1.0$)

Table 4: Student reasoning lengths under different pruning levels. Values in parentheses indicate relative length normalized to the $\rho = 1.0$ setting.

We observe that the student model closely matches the length distribution of the pruned training data. In particular, a keep fraction of $\rho = 0.7$ yields approximately 30% shorter reasoning traces at inference time. This indicates that the student does not revert to verbose reasoning, but instead internalizes the compressed reasoning style present in the supervision.

F.2 Structural and Semantic Properties of Student Outputs

We next evaluate whether compressed reasoning leads to degradation in output quality. Following the same protocol used for evaluating pruned training traces, we assess student-generated outputs along four dimensions: (i) structural integrity, (ii) semantic coherence, (iii) preservation of mathematical state, and (iv) overall suitability for SFT-style reasoning.

Across all pruning levels, we find that student outputs largely preserve the structure and semantics of the reasoning process. In particular, even at $\rho = 0.7$, we do not observe cases where the reasoning becomes logically invalid or mathematically inconsistent. Instead, degradation is primarily stylistic, consisting of:

- missing function words (e.g., articles, prepositions),
- reduced punctuation,
- occasional merging of reasoning steps.

These artifacts are consistent with those observed in the pruned training data and suggest that the student faithfully learns the distribution of compressed reasoning traces.

F.3 Interpretability of Compressed Reasoning

Despite the reduction in length, student-generated reasoning remains interpretable. In most cases, key intermediate quantities and computation steps are preserved, allowing the reasoning process to be followed with minimal effort. The primary effect of pruning is the removal of redundant narrative scaffolding rather than core computational content.

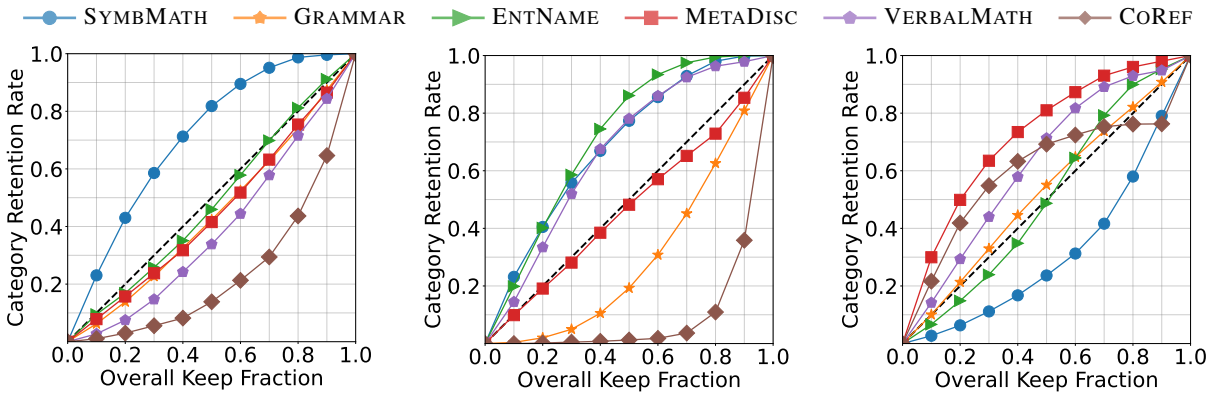
We emphasize that the goal of greedy pruning is not to produce stylistically optimal reasoning, but to preserve functional correctness under compression. If surface fluency is desired, this can be addressed orthogonally (e.g., via a lightweight rewriting or post-processing step), without altering the underlying token importance structure.

F.4 Summary

Overall, compressed-reasoning distillation yields student models that:

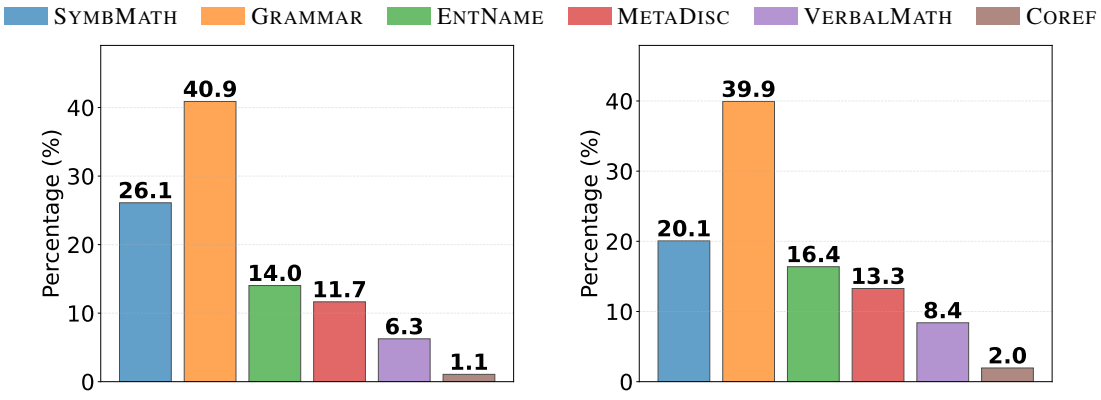
- generate expected shorter reasoning traces,
- preserve semantics and mathematical state,
- exhibit only minor stylistic degradation.

These results suggest that greedy pruning provides a viable mechanism for training token-efficient reasoning models without materially compromising interpretability or correctness.



(a) $\mathcal{T}, \mathcal{P} =$ `Llama3.1-8B`; $\mathcal{L}^{\text{JOINT}}_{\theta_{\mathcal{P}}}$ (b) $\mathcal{T} =$ `Qwen2.5-7B`; `TokenSkip` (c) $\mathcal{T} =$ `Qwen2.5-7B`; `Surprisal`

Figure 7: **Functional structure under different pruning criteria.** Each curve shows the fraction of tokens retained within a functional category at a given keep fraction; the dashed line denotes uniform pruning. (a) Greedy pruning with Llama3.1-8B preserves a clear functional ordering, strongly retaining symbolic computation while pruning referential, descriptive, and grammatical scaffolding. (b) `TokenSkip` exhibits weaker separation across functional categories, with symbolic computation tracking closer to non-symbolic components and early removal of coreference and grammar. (c) `Surprisal`-based pruning under-retains symbolic computation while over-retaining narrative and descriptive categories, yielding a functional structure misaligned with reasoning requirements.



(a) Reasoning generated from `Qwen2.5-7B`. (b) Reasoning generated from `Llama3.1-8B`.

Figure 8: **Token Category Distribution.** Token-level functional category distribution over 1,000 randomly sampled GSM8K reasoning traces generated by (a) Qwen2.5-7B and (b) Llama3.1-8B. Percentages are computed over all reasoning tokens prior to pruning and provide additional context for our functional structure analysis.

Prompt: Functional Role Annotation

SYSTEM PROMPT

```
You are a careful linguistic annotator. Your task is to label tokens using a fixed category set and strict priority
      rules. You must follow the definitions exactly and output valid JSON only. Do not include any other text in your
      response.
```

USER PROMPT TEMPLATE

```
You are annotating tokens from a large language model's chain-of-thought according to their FUNCTIONAL ROLE in
      mathematical reasoning.

Your goal is to assign EXACTLY ONE category per token such that the resulting labels reveal which types of reasoning
      content are present and how they differ in importance.

Tokens may be subword pieces (e.g., "Weng" -> "W", "eng"; "babysitting" -> "babys", "itting"). Label each token
      independently, but ALL subword pieces belonging to the same word MUST receive the SAME category.

You MUST choose exactly one of the following categories:

------------------------------------------------------
CATEGORIES
------------------------------------------------------

1. SYMBOLIC_MATH
   - Explicit numeric or symbolic computation
   - Includes: digits, currency symbols, arithmetic operators, equations, fractions, numeric constants

2. META_DISCOURSE
   - Tokens that organize, narrate, or scaffold the reasoning process
   - Includes instructional or planning language rather than mathematical content
   - Examples: "to find", "we need to", "step", "first", "now", "so", "let's", "calculate", "find", "the final answer
      is", step numbers, list markers
   - Arithmetic verbs (e.g., "add", "subtract") are META_DISCOURSE ONLY when they introduce or narrate a step, not when
      expressing a mathematical relation

3. COREFERENCE
   - Pronouns or references pointing to previously mentioned entities or quantities
   - Includes: she, her, he, it, they, we, him, this
   - "this" is COREFERENCE ONLY when referential, not when used as a grammatical filler

4. ENTITY_NAME
   - Proper names or concrete entities central to the problem
   - Includes: people's names, objects, units, or concrete nouns being counted
   - Examples: "Natalia", "Julie", "wallet", "book", "pages", "year"

5. VERBAL_MATH
   - Natural-language descriptions of arithmetic relationships or quantities
   - Includes: "half", "twice", "total", "remaining", "more", "per", "rate"
   - Arithmetic verbs (e.g., "add", "multiply", "divide") belong here ONLY when describing the operation itself, not
      when narrating steps

6. GRAMMATICAL
   - Grammatical glue with little standalone semantic content
   - Includes: articles, prepositions, conjunctions, auxiliary verbs, punctuation, formatting tokens, whitespace

------------------------------------------------------
IMPORTANT NOTES
------------------------------------------------------

1. Adjectives are NOT a separate category. Assign adjectives based on function:
  - Arithmetic-modifying adjectives -> VERBAL_MATH
  - Discourse or narrative adjectives -> META_DISCOURSE
  - Entity-identifying adjectives -> ENTITY_NAME
  - Otherwise -> FUNCTION

2. ENTITY_NAME vs VERBAL_MATH: When a noun denotes a concrete object being counted (e.g., "pages", "wallet"), label it
      ENTITY_NAME. Mathematical relations involving those nouns are captured by VERBAL_MATH or SYMBOLIC_MATH.

------------------------------------------------------
PRIORITY RULES (STRICT)
------------------------------------------------------

If a token could belong to multiple categories, apply the FIRST matching rule:

1. If part of an explicit numeric or symbolic expression -> SYMBOLIC_MATH
2. Else if it narrates or structures reasoning -> META_DISCOURSE
3. Else if it is referential -> COREFERENCE
4. Else if it names a concrete entity -> ENTITY_NAME
5. Else if it describes arithmetic verbally -> VERBAL_MATH
6. Else -> GRAMMATICAL


------------------------------------------------------
CONSISTENCY CONSTRAINT
------------------------------------------------------

If the same surface word appears multiple times with the same functional role, assign it the SAME category across
      occurrences unless its role clearly changes.
```

Prompt: Functional Role Annotation (continued)

```
------------------------------------------------------
INPUT FORMAT
------------------------------------------------------
Question:
<question in natural language>

Output Reason:
<reason in natural language>

Output Answer:
<answer in natural language>

Reason Tokens:
<reason tokens in JSON format>
[
  {
    "token_position": <int>,
    "token_text": "<string>",
  }
]

------------------------------------------------------
OUTPUT FORMAT
------------------------------------------------------

Return a JSON list. Each entry MUST have:

{
  "token_position": <int>,
  "token_text": "<string>",
  "category": "<one of the 6 categories>",
  "justification": "<6-word explanation>"
}

Do NOT assign multiple categories.
Do NOT invent new categories.
Be consistent across similar tokens.

------------------------------------------------------
ANNOTATE THE FOLLOWING
------------------------------------------------------

Question:
{{question}}

Output Reason:
{{reason}}

Output Answer:
{{answer}}

Output Reason Tokens (JSON Format):
{{reason_tokens}}
```