HIDING IN PLAIN SIGHT: DETECTABILITY-AWARE ANTIDISTILLATION OF REASONING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Distillation via reasoning traces exposes frontier models to adversarial third parties, who can bypass their guardrails and misappropriate their capabilities. Existing antidistillation defenses perturb the teacher's sampling distribution to hinder downstream student learning, but these perturbations also degrade the trace quality even when task accuracy is preserved, often leaving visible artifacts such as syntactic incoherencies and repetition chains. Beyond eroding user trust, such traces are statistically distinguishable, allowing adversaries to filter or resample them and weaken the defense. An effective defense must therefore limit how *visibly* it degrades the teacher. To address this gap, we formulate antidistillation as a max-min game whose constraint set explicitly encodes defense detectability. Satisfying both axes— degrading the student while remaining undetectable—may require more clever perturbations, akin to sparse but targeted adversarial attacks in robustness literature. We apply the same analogy to reasoning steps, targeting *thought anchors*: sentences identified in prior interpretability work as strongly influencing a model's output. We propose Thought Anchor RemovalS (TARS), which identifies and removes thought anchors, whose content is semantically redundant, so that the reasoning trace remains legible to the user. Modeling these removals as burst-deletions, we then provide formal guarantees on its detectability. Finally, we empirically demonstrate the efficacy of TARS in consistently hindering downstream distillation across teacher model scales (DeepSeek R1 Distill Qwen 7B, Gemma 3 12B), reasoning benchmarks (MMLU Pro, MATH, DeepMath), and student architectures (Llama 3.2 1B, Llama 3.2 3B, Gemma 3 1B, Qwen 3 1.7B) while preserving teacher performance.

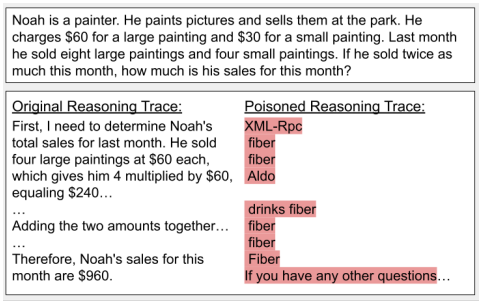
1 INTRODUCTION

Reasoning in large language models (LLMs) has driven substantial performance gains on multi-step tasks such as mathematics, coding, and strategic planning (OpenAI, 2024; Qwen, 2024; Anthropic, 2026b). Beyond accuracy, the traces themselves offer practical insight into how answers are produced, strengthening trust in model outputs. However, releasing traces carries a real risk: although knowledge distillation from reasoning traces was originally proposed as a route to more efficient training (Hsieh et al., 2023; Mukherjee et al., 2023), distilled models have since been shown to match, and at times surpass, their teachers at a fraction of the cost (Guo et al., 2025), raising pressing concerns about intellectual property for frontier laboratories competing at the edge of learnability (Anthropic, 2026a). In fact, these concerns now sit at the forefront of U.S. national policy, with recent legislative proposals explicitly targeting the extraction of capabilities from closed-source AI models (119th Congress, 2026; Kratsios, 2026), which essentially circumvent export controls on compute. Furthermore, from a safety perspective, distilled models often lose the safety alignment trained into the teacher model (Jahan & Sun, 2025; Shi et al., 2026), thus creating a direct pathway from publicly exposed reasoning traces to unmoderated and dangerous models.

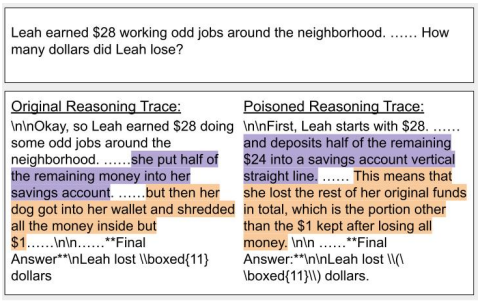
Defending against misuse therefore requires traces that remain useful to users while being ineffective for training a student, a goal known as *antidistillation*. Summarization is a commonly deployed preventive measure, but compressing reasoning traces can introduce hallucinations (Tang et al., 2024; Song et al., 2024) and obscure key logical steps, eroding user trust. Pioneering work in this area (Savani et al., 2025; Li et al., 2025) instead alters the teacher's output distribution, which requires either access to proxy students during trace generation or fine-tuning the teacher itself. Doing so,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

however, leaves visible artifacts in the output traces. Specifically, Antidistillation Sampling (ADS) (Savani et al., 2025) often makes poisoned traces grammatically incoherent, inserts nonsensical tokens, and falls into loops of repeated tokens, while Defensive Output Generation (DOGe) (Li et al., 2025) renders traces difficult for users to follow by abruptly changing topic or repeating itself in different words (Figure 1). Such *detectability*, beyond eroding user trust, makes poisoned traces statistically distinguishable from clean ones. As a result, both methods are vulnerable to simple countermeasures such as filtering, resampling, or prompting a smaller LLM to flag poisoned traces, making them easy to bypass.



(a) ADS example student output on a prompt. Grammatical incoherency (such as "fiber fiber Aldo drinks fiber...") throughout the poisoned trace (highlighted in red).



(b) DOGe example student output on a prompt. Poisoned trace looks plausible, but rewritten parts (such as "savings account vertical straight line") are hard to understand.

Figure 1: Original and Poisoned Reasoning traces by ADS and DOGe show the shortcomings of existing defense methods on samples from the GSM8K (Cobbe et al., 2021) dataset. These methods are all detectable or obscure the trace to the degree that the trace may be unhelpful for an actual user.

To address current limitations, we formulate antidistillation in a game-theoretic framework which explicitly allows for the enforcement of detectability constraints, drawing on its connection to data poisoning. We then draw on recent work in LLM interpretability, which has shown that specific sentences, known as *Thought Anchors* (Bogdan et al., 2025), in reasoning traces are disproportionately important to the model output and reasoning flow to provide a method which accounts for detectability. We argue that poisoning Thought Anchors can disrupt the downstream student's ability to distill the teacher's reasoning. Empirically, we show that students trained on poisoned traces exhibit degraded distillation performance relative to those trained on unpoisoned traces. We further characterize the detectability of our method both theoretically and empirically, and compare it with ADS using appropriate detectability metrics, demonstrating competitive performance. Our contributions are as follows:

1. **Theoretical Formulation:** We formulate antidistillation as a max-min optimization problem, mapping data poisoning frameworks to LLM distillation setting to ground future research.
2. **Thought Anchor RemovalS (TARS) Method:** We introduce a method that implements targetted poisoning using thought anchors. Across multiple datasets and choice of teacher-student model pairs, we comprehensively show that it degrades downstream distillation while retaining coherence, unlike competing methods.
3. **Detectability Bounds:** We provide rigorous bounds on the detectability of the TARS defense by modeling the method as performing k burst deletions.

2 Related Work

Antidistilation: Antidistillation techniques aim to prevent adversarial model distillation attacks by modifying a teacher's outputs to be poor training data for the student. Savani et al. (2025) proposed ADS, which perturbs the next-token probability distribution of the teacher model by encouraging the generation of tokens to degrade a student model, via a proxy model. Ma et al. (2026) proposed a prompting approach, which instructs the LLM to generate less distillation-friendly reasoning traces, and a gradient-based approach, which perturbs tokens post-generation to maximize test loss on a proxy model. DOGe (Li et al., 2025) finetunes the final layer of the teacher model with a loss to

prevent student learning. Ding et al. (2025) finetunes an LLM to remove self-talk tokens from the reasoning trace. There is also concurrent work, such as SGRE (Li et al., 2026), which edits the structure and verbalization of completed traces to preserve answer accuracy and naturalness, LADS (Diao et al., 2026) correlates responses across distillation accounts, making semantically similar prompts across accounts have shared randomness while allowing individual users to see no disruption, and (Allouah et al., 2026), who formulate a minimax game with an adaptive student and derive a product-of-experts decoding defense. Complementary evaluations find that defense efficacy varies by task (Jiang, 2026) and by the attacker's query budget, data budget, and API access (Libon et al., 2026). Building on this literature, we explicitly incorporate detectability into a game-theoretic formulation and target sparse, influential portions of reasoning traces to impede distillation while maintaining trace coherence.

Reasoning in LLMs: Chain-of-thought (CoT) reasoning, in which models are trained to generate intermediate reasoning steps, emerged in recent years as a technique to improve the performance of models on complex multi-step tasks (Wei et al., 2022). The Tree-of-thought (ToT) technique generalized and improved CoT by enabling LLMs to consider different reasoning paths (Yao et al., 2023). Reinforcement learning post-training with human-annotated reasoning data was used as an approach to distill reasoning capabilities models. Reinforcement learning without human-labeled data was then proposed (OpenAI, 2024; Qwen, 2024). Recently, models have been trained to learn reasoning by distilling knowledge from a large reasoning model, such as the R1-distill series (Guo et al., 2025). Interpretability of Reasoning models has recently been studied, such as work which identifies how parts of a reasoning trace at the sentence and token-level are unequally important to answer generation (Wang et al., 2025; Bogdan et al., 2025).

Data Poisoning: A useful motivation for a framework for antidistillation comes from data poisoning. Data poisoning is an integrity attack on the training phase of machine learning models which manipulates the training data distribution to harm the model (Fan et al., 2022). The adversary is additionally constrained to perturbations that remain close to legitimate data, so that the attack is not trivially detectable. Early work considered attacks on support vector machines and neural networks (Biggio et al., 2012; Shafahi et al., 2018). These attacks have also been shown to be effective against LLMs (Carlini et al., 2024; Souly et al., 2025). Further work has studied targeted, less-detectable poisoning methods (Su et al., 2017; Wiyatno & Xu, 2018). Our work shows that antidistillation can be formulated as a generalization of the classical game-theoretic data poisoning framework.

3 FORMULATION

To motivate our proposed formulation for antidistillation, we first look at the classical data poisoning formulation. Herein, an attacker attempts to find a set of data points sufficiently "close" to real data points that maximally harm the performance of a given statistical model. This format can be viewed as a Stackelberg game, where the attacker commits to a poisoned dataset $\tilde{D}$, after which the defender attempts to find the optimal model parameters from it. Observe that the best action for the defender in this game is:

$$\theta^*(\tilde{D}) := \arg\min_{\theta \in \Theta} \tilde{\mathcal{L}}(\theta; \tilde{D} \cup D),$$

while the optimal action for the attacker to take is:

$$\tilde{D}^* := \arg\max_{\tilde{D} \in \mathcal{B}} \mathcal{L}(\theta^*(\tilde{D})).$$

Therefore, the game can be viewed as the following bi-level optimization problem:

$$\tilde{D} = \arg\max_{\tilde{D} \in \mathcal{B}} \mathcal{L}(\arg\min_{\theta} \tilde{\mathcal{L}}(\theta; \tilde{D} \cup D)), \tag{1}$$

where $\tilde{D}$ is the poisoned dataset, $\mathcal{B}$ is the set of admissible poisoned datasets, D is a clean dataset also given for training, $\mathcal{L}$ is the loss function to evaluate the optimal model θ, and $\tilde{\mathcal{L}}$ is the loss function for evaluating the model on the poisoned data (Fan et al., 2022). The solution, $(\tilde{D}^*, \theta^*(\tilde{D}^*))$, to equation 1 constitutes the Stackelberg equilibrium for this data poisoning game. Classically, the admissible set $\mathcal{B}$ may include constraints like

$$\mathcal{B} = \mathcal{B}_\epsilon^p = \{(x + \delta, y) \mid \|\delta\|_p \leq \epsilon\}, \tag{2}$$

or

$$\mathcal{B} = \mathcal{B}_\tau = \{(\tilde{x}, y) | S(x) < \tau\}, \tag{3}$$

where $S(\cdot)$ is some known detector that the attacker assumes the defender will use and thus wants to prevent from detecting the poisoning. These constraints are explicitly created to prevent detection of the poisoning as, particularly in the case where poisoned data is simply appended to clean data, if a defender can detect the poisoning they can simply avoid training on that data.

While this framework models something very similar to antidistillation, we argue that there are 2 key changes that need to be made. First, in antidistillation, the model provider will not provide clean traces for training so modeling the student's available data as $D \cup \tilde{D}$ it not a good assumption. Secondly, this framework only considers a student who does empirical risk minimization (ERM) on one specific model class over the data but in practice distillers will do comprehensive post-processing on the received data and the model provider may not know the specific model class being used by the student. As such, we modify Equation 1 to

$$\tilde{D}_T^* = \arg\max_{\tilde{D}_T \in \mathcal{B}} \inf_{\mathcal{A} \in \mathcal{F}} \mathbb{E}_{x \sim p_{\text{data}}} \left[\mathcal{L}\left(\mathcal{A}\left(\tilde{D}_T \right); x \right) \right], \tag{4}$$

where $\mathcal{B}$ is the set of admissible poisoned datasets which can also be dependent on the teacher model (e.g. to allow for retraining), $\mathcal{F}$ is the set of admissible set of training algorithms $\mathcal{A} : \mathcal{X}^n \to \mathcal{H}$ which take the poisoned data and output a hypothesis, p_{data} is the true data distribution, and $\mathcal{L}$ is the population loss. This framework encompasses the data poisoning framework by taking $\mathcal{A}$ to be ERM and by taking $\mathcal{B} = \{(D_c \cup \tilde{D}) : \tilde{D} \in \mathcal{B}_{\text{dp}}\}$.

Similar to how detectability constraints are enforced on $\mathcal{B}$ in data poisoning, one can take enforce analogous constraints in the case of antidistillation. For example, we specifically consider the case of $\mathcal{B}$ being a KL-divergence ball around the original teacher distribution (analogous to Equation 2) but one can also consider datasets which satisfy some given detector such as a perplexity or accuracy detector (analogous to Equation 3).

To connect this formulation to existing works in literature, we show in Appendix A how this framework subsumes ADS (Savani et al., 2025) and DOGe (Li et al., 2025). Crucially, both of these methods make implicit assumptions on student model architecture by way of a suitable proxy, while being easy to detect. Therefore, the main takeaway here is that *any good antidistillation defense in practice should limit detectability while also be downstream student-agnostic to a realistic degree, as the model provider often has no knowledge of the student's model/training procedure.*

4 METHOD

4.1 UNDERSTANDING UNITS OF REASONING STRUCTURE

A major limitation of prior works such as ADS and DOGe, especially in the context of practical detectability, is the need to perturb the teacher's output distribution at every token position. This is analogous to dense ℓ_2/ℓ_∞ attacks on images (Moosavi-Dezfooli et al., 2016; Madry et al., 2019) in the adversarial robustness literature, and the main reason behind noticeable artifacts as shown in Fig. 1. Extending the same analogy, a well-established finding is that sparse, targeted perturbations can often be equally as effective while being harder to detect by humans. Specifically, one-pixel attacks (Su et al., 2017) and Jacobian-based Saliency Map Approach (JSMA) (Wiyatno & Xu, 2018) demonstrate that perturbing a small number of carefully chosen input dimensions suffices to fool classifiers, often with greater success than uniform perturbation. Motivated by this analogy, we question whether it is possible to transfer this principle to LLM reasoning.

As pixels for a natural image, tokens serve as the fundamental units of language data. A natural question to then ask is if there are specific tokens in a trace that are critical to the reasoning process. Prior work has shown that particular tokens can exert disproportionate influence on planning and downstream outputs (Anthropic, 2025), or induce behaviors such as backtracking and correction (Muennighoff et al., 2025). However, systematic token-level interventions are hard because such signals can be noisy (Min et al., 2026), making them difficult to identify and reliably target and may also induce noticeable artifacts. Instead, a recent line of work treats sentences as the basic units of analysis. Bogdan et al. (2025) identifies a small subset of sentences, termed *thought anchors*, that exert disproportionate counterfactual influence on the remainder of a reasoning trace and its final

response. Besides planning sentences, such anchors are often uncertainty-management (e.g., "Wait, I made a mistake"), or self-reflection (e.g., "Maybe I can calculate this in decimal") statements, while many active-computation steps simply execute operations determined by the preceding reasoning. Concurrent work (Kapoor et al., 2026) similarly finds that errors at particular steps in multi-step reasoning can cascade through the remainder of a solution, while perturbations to routine steps have much less effect. Together, they suggest that sentence-level steps provide a more structured and reliable intervention target than individual tokens and thought anchors serve as our point of inquiry.

4.2 Identifying Thought Anchors

We implement thought anchors through their attention-based *broadcasting* behavior, described in Bogdan et al. (2025). A key characteristic is that a small set of *receiver heads* stand out: rather than spreading attention over the entire trace, they consistently attend back to a handful of sentences, often ones that appeared much earlier in the trace. We identify these heads by measuring how concentrated the attention each head directs at earlier sentences is, using its kurtosis. A head that repeatedly singles out a few sentences has a sharply peaked distribution and high kurtosis, whereas a head that attends broadly has low kurtosis. Each sentence is assigned a *broadcasting score*: the attention it receives through these receiver heads from sentences several steps later in the trace. High-scoring sentences are treated as thought anchors. We provide details of receiver-head calibration and sentence scoring in Algorithm 2 of Appendix B.

Importantly, structural influence and semantic role are distinct. The broadcasting score identifies which sentences are influential, while a separate reasoning-type classification distinguishes planning, uncertainty management, self-checking, active computation, fact retrieval, and other steps. We find that many influential sentences are planning or self-reflective in nature: they shape how reasoning proceeds without directly adding problem-specific computations or facts. This makes them particularly attractive targets, as intervening on them can potentially disrupt the reasoning structure available to a student while largely preserving the substantive content and final answer of the trace. An illustrative example is presented in Fig. 2, where we used Gemini 3 as a judge to label anchors.

Sentence The thought anchor sentence	Label The designated category for the thought anchor.	Importance Whether the anchor is key for the final answer.
"Alright, let me try to figure out this problem."	Problem setup	✗
"So, we have a matrix that looks like a rotation matrix."	Problem setup	✗
"It is: $\begin{bmatrix}\cos 170^\circ & -\sin 170^\circ \\ \sin 170^\circ & \cos 170^\circ\end{bmatrix}$ And we need to when we raise this"	Fact retrieval	✓
"Hmm, okay."	Uncertainty Management	✗
"So, this matrix is a rotation matrix."	Active computation	✓

(a) Thought Anchor labeling example in MATH.

Sentence The thought anchor sentence	Label The designated category for the thought anchor.	Importance Whether the anchor is key for the final answer.
"Okay, so I need to figure out the answer..."	Problem setup	✗
"The question is a fill-in-the-blank based on a passage..."	Problem setup	✗
"Let me read through the passage again..."	Plan generation	✗
"Putin and Jiang Zemin signed a joint declaration calling for a "multipolar world"..."	Fact retrieval	✓
"Putin emphasized that Russia wants to "work hard" with the new generation... China praised their treaty.."	Active computation	✓

(b) Thought Anchor labeling example in MMLU.

Figure 2: Illustration of Thought Anchors and sentence types of reasoning traces generated by Deepseek R1 Distil Qwen 7B. We use the same taxonomy for determining thought anchors as Bogdan et al. (2025) and ask Gemini 3 to label the thought anchors according to this scheme.

4.3 A Proof of Concept: Thought Anchors Transfer to Student

While thought anchors may influence the teacher's own reasoning, that does not imply that a student distilled from the teacher's traces inherits similar reasoning mechanisms. Therefore, as a first sanity check, we generate reasoning traces on MATH and MMLU with a teacher model (DeepSeek-R1-Distill-Qwen-7B) and fine-tune a student model (Llama-3.2-3B) on them. If the student inherits the teacher's anchors, the two models should concentrate attention on the same sentences of the trace.

To do this, we isolate the thought anchor receiver heads for each model (see Algorithm 2 for details) and dataset separately. Both models read 128 teacher traces, and both score every sentence by the attention it receives through its own receiver heads. Once each trace is scored, we measure the similarity of how the two models rank traces sentences (Spearman ρ) and how much their five highest-scoring sentences overlap (top-5 Jaccard similarity), averaged over traces. We refer to these measures collectively as *anchor agreement*. The experimental details and parameters are in Appendix C

Table 1: Anchor agreement between the teacher (DeepSeek-R1-Distill-Qwen-7B) and the student (Llama-3.2-3B) before and after distillation, averaged over 128 traces. Spearman ρ measures how similarly the two models rank all sentences, and top-5 Jaccard measures the overlap between their five highest-scored sentences. Δ denotes the relative change after distillation.

	MATH		MMLU	
Student	Spearman ρ	Top-5 Jaccard	Spearman ρ	Top-5 Jaccard
Undistilled	0.70	0.33	0.67	0.23
Distilled	0.78	0.41	0.71	0.31
Δ	11%	24%	6%	35%

To isolate the effect of distillation, we compare the teacher and student scores before and after fine-tuning. As Table 1 shows, distillation increases anchor agreement on both metrics, indicating that the student inherits the teacher's thought anchors. Combined with prior work establishing their importance for reasoning (Bogdan et al., 2025), this suggests that disrupting thought anchors in the teacher's traces may disrupt the reasoning a student learns from them, forming the basis of TARS.

4.4 TARS: Thought Anchor Removals

Having established that thought anchors transfer from teacher to student, we now introduce TARS[1], which detects and deletes thought anchors from reasoning traces as a defense against distillation. TARS removes thought anchors in order of reasoning category and then broadcasting score, deleting first the uncertainty-management, planning, and self-checking steps, which shape how the reasoning proceeds without adding problem-specific content. Receiver heads are calibrated once per teacher, after which each trace requires a single teacher forward pass to score. Unlike ADS, TARS edits traces after generation rather than altering the teacher's sampling, and requires neither a proxy student nor gradient computation during generation. The teacher's final answers are therefore unchanged, and the deletion budget k trades off distillation resistance against detectability, which can be swept at no additional cost. TARS is formalized in Algorithm 1, and the thought anchor detection routine in Algorithm 2 (Appendix B), together with all hyperparameters.

Algorithm 1 TARS removes up to k sentences, ordered by reasoning category and score, deleting steps a student would most learn from while keeping the problem setup and final answer intact.

Require: response y, sentences $(s_1, \ldots, s_n)$, scores score, budgets $\mathcal{K}$

$c_j \leftarrow \textsc{Classify}(s_j)$ for $j = 1, \ldots, n$

$$\text{tier}(c) \leftarrow \begin{cases} 1 & c = \texttt{uncertainty_management} \\ 2 & c = \texttt{plan_generation} \\ 3 & c = \texttt{self_checking} \\ 4 & \text{otherwise} \end{cases}$$

$P \leftarrow \{\texttt{problem_setup}, \texttt{final_answer}\}$

$Q \leftarrow \{\, j : c_j \notin P \,\}$ sorted by $(\text{tier}(c_j) \uparrow,\ \text{score}[j] \downarrow)$

for each $k \in \mathcal{K}$ **do**

  $y'_k \leftarrow y$ with sentences $Q[1..\min(k, |Q|)]$ removed

end for

return $\{\, y'_k : k \in \mathcal{K} \,\}$

5 Experiments

We now evaluate the efficacy of TARS in disrupting downstream distillation by characterizing how the student downstream accuracy changes when distilled using traces from a teacher model with an increasing thought anchor removal budget: $k \in \{5, 10, \ldots, 30\}$. We consider two teacher models:

[1] ...an antidistillation method, not the robot (for the optimal experience we suggest listening to Hans Zimmer)

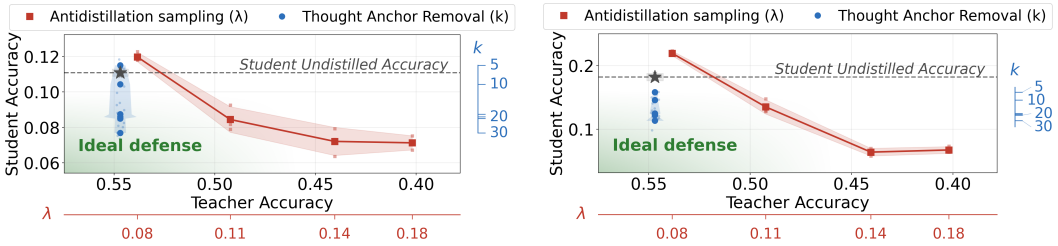


Figure 3: Student versus teacher accuracy for ADS (red) and TARS (blue) on MATH, with DeepSeek-R1-Distill-Qwen-7B as the teacher. The students used were Llama 3.2 3B (left) and Gemma 3 1B (right); Llama 3.2 1B is provided in Appendix D.2.

DeepSeek-R1-Distill-Qwen-7B (consistent with prior work) and Gemma 3 12B. For DeepSeek-R1-Distill-Qwen-7B, we use Llama-3.2-3B, Llama-3.2-1B, and Gemma 3 1B as student models, with MATH (Hendrycks et al., 2021) and MMLU Pro (Wang et al., 2024) as datasets. For Gemma 3 12B, we use the stronger Qwen 3 1.7B as the student, and DeepMath (He et al., 2026) as the dataset.

Then, to ground our results in the broader antidistillation landscape, we compare TARS to the main prior work, ADS. We plot the teacher accuracy versus student accuracy for varied intensity of attacks (varying k for TARS and λ for ADS). We do not directly compare to DOGe due to the key issue: as part of the defense, DOGe requires adversarially retraining the final layer in the teacher model on a set of benchmarks. In practice, fine-tuning on a specific subset without having a larger, diverse mix of general data leads to catastrophic forgetting (Kotha et al., 2024; Luo et al., 2025; Wu et al., 2026), making it practically infeasible. This is without considering the post-training and guardrails that modern models require, or the visible signs of model degradation as in Fig. 1.

To compare to ADS, we jointly consider detectability and student accuracy. This framing rewards a method that suppresses student distillation and remains hard to detect. We measure detectability by evaluating Perplexity (Jelinek et al., 1977; Radford et al., 2018), TF-IDF (Salton & Buckley, 1988), Semantic Similarity, token change length, teacher accuracy, and an LLM Judge, measured relative to baseline. Further details on the distillation and detectability setup are in Appendices E and H.

5.1 Results

Figures 3 and 4 show student distillation degradation for varying maximum sentence removal budgets across the experimental suite described in Section 5. In the plots, the ideal antidistillation method lies in the bottom-left region, where student accuracy is significantly degraded with minimal teacher degradation. The inherent teacher-student accuracy trade-off in ADS does not exist in TARS. There is a positive relationship between the number of poisoned thought anchor sentences and the accuracy drop between baseline and poisoned distillation. We further verify that these accuracy results are due to thought anchors (and not just any sentence) by including an additional ablation experiment where we compare TARS to random sentence removal in Appendix G.1. We provide the additional results with the remaining student/dataset pairs in Appendix D.

In Tables 2-3, we provide a detectability comparison of TARS and ADS. The perplexity is lower in ADS compared to TARS, which is expected since ADS perturbs the distribution per token, whereas TARS removes entire blocks of tokens. On the other hand, the metrics closer to human perception, such as TF-IDF and Semantic L2 distances for TARS are much stronger, indicating that TARS is better at semantically representing the original traces. The change in tokens with respect to baseline traces is similar and highly dependent on the method's hyperparameter selection across ADS and TARS. We also use an LLM Judge to compute the conference, fluency, and completeness of poisoned traces, with details provided in Appendix E.1.

Each poisoned arm's traces are compared to the unpoisoned baseline. **PPL** is the poisoned-to-baseline perplexity ratio under a Llama-3-8B-Instruct detector; **TF-IDF** and **Sem. L2** are the TF-IDF and semantic-embedding (L_2) distances to baseline traces; Δ**Tok.** is the mean token-count difference between the poisoned trace and baseline; **Overall** is the LLM judge (Qwen3 8B) score rating based on

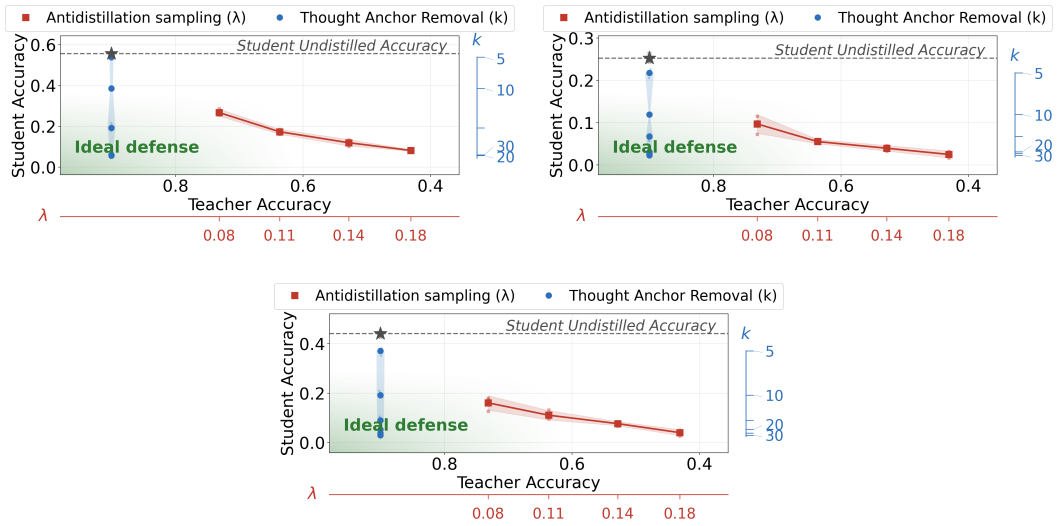


Figure 4: Student versus teacher accuracy for ADS and TARS on GSM8K, with DeepSeek-R1-Distill-Qwen-7B as the teacher. The students used were Llama 3.2 3B (top-left), Llama 3.2 1B (top-right), and Gemma 3 1B (bottom).

Table 2: Detectability for ADS and Thought Anchor Removal (TARS) on MATH, with DeepSeek-R1-Distill-Qwen-7B as teacher. ADS uses Qwen2.5-3B as its proxy model.

Method	Param.	PPL↓	TF-IDF↓	Sem. L2↓	ΔTok.↓	Overall↑	TA↑
Baseline	—	1.000	—	—	—	4.62	54.7
ADS	$\lambda = 0.08$	1.016	0.230	0.525	315	4.44	53.8
ADS	$\lambda = 0.11$	1.066	0.396	0.683	384	4.01	49.2
ADS	$\lambda = 0.14$	1.057	0.574	0.800	428	3.57	44.0
ADS	$\lambda = 0.18$	1.023	0.650	0.870	437	2.97	40.2
TARS	$k = 5$	1.236	0.025	0.255	142	4.34	54.7
TARS	$k = 10$	1.455	0.126	0.425	289	4.05	54.7
TARS	$k = 15$	1.697	0.202	0.535	411	3.32	54.7
TARS	$k = 20$	1.986	0.253	0.614	515	3.00	54.7
TARS	$k = 25$	2.311	0.287	0.671	603	2.93	54.7
TARS	$k = 30$	2.750	0.328	0.717	690	2.56	54.7

coherence, fluency, and thought completeness (exact prompt stated in Appendix E.1). **TA** is teacher accuracy (%). Arrows indicate whether lower (↓) or higher (↑) values are favorable to the defense.

In Appendix G.3, we extend our empirical testing for detectability by providing trace statistics for both TARS and ADS outputs. These statistics include *mean trace length* (MTL), number of *repetition tokens*, and percentage of trace that is *degenerate* (has no semantic content). We find that while TARS traces have a lower MTL than clean outputs, ADS has significantly higher MTL (**~1.17**×). Furthermore, ADS on average has about **~600** repetition tokens per trace and **~59%** of the trace is degenerate. These issues with ADS make it trivially detectable by cheap post-processing schemes.

We further supplement this detectability analysis with an analysis of the robustness of TARS vs ADS to different attacks. The first attack analyzed is a simple re-querying scheme where the attacker re-queries the teacher 4 times and chooses the longest response. This is intended to be a countermeasure to TARS which typically has shorter responses due to sentence deletions. The second attack analyzed was a more defense-specific attack which applies the above attack to TARS and a re-querying + repetition filter attack to ADS to combat the degenerate repetition that often happens to ADS traces. Due to space limitations, the details of the experiment and results are provided in Appendix G.2 but we largely see that TARS is robust to both attacks and, in fact, applying the attack can further harm the student while ADS shows clear student performance improvements from both attacks.

Table 3: Detectability for ADS and Thought Anchor Removal (TARS) on GSM8K, with DeepSeek-R1-Distill-Qwen-7B as teacher. ADS uses Qwen2.5-3B as its proxy model.

Method	Param.	PPL↓	TF-IDF↓	Sem. L2↓	ΔTok.↓	Overall↑	TA↑
Baseline	—	1.000	—	—	—	4.89	88.9
ADS	$\lambda = 0.08$	0.846	0.533	0.731	995	3.04	38.6
ADS	$\lambda = 0.11$	0.820	0.713	0.883	1289	2.10	26.5
ADS	$\lambda = 0.14$	0.836	0.760	0.935	1309	1.62	17.5
ADS	$\lambda = 0.18$	0.914	0.840	1.036	1344	1.35	11.4
TARS	$k = 5$	1.648	0.038	0.400	116	4.68	88.9
TARS	$k = 10$	2.525	0.313	0.740	256	3.12	88.9
TARS	$k = 15$	3.262	0.728	1.133	379	1.73	88.9
TARS	$k = 20$	4.478	0.894	1.288	422	1.16	88.9
TARS	$k = 25$	5.775	0.940	1.327	434	1.09	88.9
TARS	$k = 30$	6.775	0.943	1.332	437	1.06	88.9

5.2 Theoretical Analysis of Detectability

We also theoretically analyze TARS's detectability by modeling it as burst deletions over the reasoning trace. Under this model, detectability grows linearly with the number of deleted sentences. If the deletion mask is instead drawn from a distribution conditioned on the input prompt, the relationship stays linear but gains an additive term: the mutual information between the mask and the input. The latter result further motivates TARS as the specific thought anchors targeted (uncertainty-management, planning, and self-checking) intuitively have minimal correlation with the input prompt but are seemingly important for computation.

Proposition 5.1 (Informal)**.** *Assume the LLM acts as an m-order Markov Process and the probability of reaching any token from over all m-token histories is greater than 0. Then, the detectability of a sequence with k random, burst-deletions against a true LLM-generated sequence is $\mathcal{O}(km)$.*

Corollary 5.2 (Informal)**.** *Under the same assumptions as Proposition 5.1, if the deletion mask S is chosen from some distribution $q(\cdot|x)$ then the bound becomes $\mathcal{O}(km) + I(S; X)$.*

The formal statements and proofs of Proposition 5.1 and Corollary 5.2 are provided in Appendix I. The m-order Markov assumption is an approximation of LLM behavior (Basu et al., 2023). Papers such as Liu et al. (2024) and Xiao et al. (2024) support that the effective context window is not actually the full context window length. For the lower-bounded probability assumption, this can be fulfilled by a top-k selection method as is typically done in practice.

Proposition 5.1 and Corollary 5.2 show that for token deletions, the detectability follows a linear relationship with the number of deletions. This supports our use of k as a hyperparameter to determine the trade-off between detectability and student degradation.

6 Conclusion & Discussion

We cast antidistillation as a max-min game, explicitly integrating detectability into the constraint set to address the semantic and syntactic vulnerabilities ignored by prior methods. Drawing on mechanistic interpretability, we isolate thought anchors as the ideal targets for these sparse interventions and validate this approach with TARS. Empirically, TARS illustrates the trade-off between student degradation and defense detectability that our framework anticipates. Ultimately, our combined theoretical and practical findings represent a significant step toward robust antidistillation mechanisms that resist counter-attacks without compromising the integrity of the teacher's reasoning.

Several directions remain open. TARS is highly dependent on reasoning patterns, therefore, it will fail in situations where certain thought anchor types are not used. The max-min game-theoretic formulation also opens the door for a rigorous derivation of achievability limits for antidistillation defense mechanisms. Finally, studying specific methods for antidistillation detection can introduce further theoretically-motivated metrics to analyze the detectability performance of defense methods.

Reproducibility statement

We report all implementation details for all experiments including models, datasets, compute used, training hyperparameters, and seeds in the Appendix. For the theory, fully rigorous theorem statements and proofs are provided in the Appendix. Finally, we will also open source all of our code upon paper publication.

References

119th Congress. H.r. 8283 (ih) - Deterring American AI Model Theft Act of 2026. Technical report, U.S. House of Representatives, April 2026. URL https://www.congress.gov/bill/119th-congress/house-bill/8283. Introduced in House (IH).

Youssef Allouah, Mahdi Haghifam, Sanmi Koyejo, and Reza Shokri. The distillation game: Adaptive attacks & efficient defenses, 2026. URL https://arxiv.org/abs/2605.22737.

Anthropic. On the biology of a large language model. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/biology.html.

Anthropic. Detecting and preventing distillation attacks. https://www.anthropic.com/news/detecting-and-preventing-distillation-attacks, February 2026a.

Anthropic. Claude Opus 4.6 system card. Technical report, Anthropic, February 2026b. URL https://www-cdn.anthropic.com/14e4fb01875d2a69f646fa5e574dea2b1c0ff7b5.pdf. Accessed: 2026-04-11.

Sourya Basu, Moulik Choraria, and Lav R. Varshney. Transformers are universal predictors. In *ICML 2023 Workshop Neural Compression: From Information Theory to Applications*, 2023. URL https://openreview.net/forum?id=WMHuneCelF.

Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, pp. 1467–1474, Madison, WI, USA, 2012. Omnipress. ISBN 9781450312851.

Paul C. Bogdan, Uzay Macar, Neel Nanda, and Arthur Conmy. Thought anchors: Which llm reasoning steps matter?, 2025. URL https://arxiv.org/abs/2506.19143.

Nicholas Carlini, Matthew Jagielski, Christopher A. Choquette-Choo, Daniel Paleka, Will Pearce, Hyrum Anderson, Andreas Terzis, Kurt Thomas, and Florian Tramer. Poisoning Web-Scale Training Datasets is Practical . In *2024 IEEE Symposium on Security and Privacy (SP)*, pp. 407–425, Los Alamitos, CA, USA, May 2024. IEEE Computer Society. doi: 10.1109/SP54263.2024.00179. URL https://doi.ieeecomputersociety.org/10.1109/SP54263.2024.00179.

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

Zibo Diao, Jingchu Gai, Xinyue Ai, Zhang Zhang, Zhenyu He, and Di He. Lossless anti-distillation sampling, 2026. URL https://arxiv.org/abs/2605.18829.

Jiayu Ding, Lei Cui, Li Dong, Nanning Zheng, and Furu Wei. Information-preserving reformulation of reasoning traces for antidistillation, 2025. URL https://arxiv.org/abs/2510.11545.

Jiaxin Fan, Qi Yan, Mohan Li, Guanqun Qu, and Yang Xiao. A survey on data poisoning attacks and defenses. In *2022 7th IEEE International Conference on Data Science in Cyberspace (DSC)*, pp. 48–55, 2022. doi: 10.1109/DSC55868.2022.00014.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645 (8081):633–638, September 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://dx.doi.org/10.1038/s41586-025-09422-z.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=kHB5Te5IWm.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=7Bywt2mQsCe.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! Outperforming larger language models with less training data and smaller model sizes. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 8003–8017, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.507. URL https://aclanthology.org/2023.findings-acl.507/.

Sohely Jahan and Ruimin Sun. Black-box behavioral distillation breaks safety alignment in medical llms, 2025. URL https://arxiv.org/abs/2512.09403.

F. Jelinek, R. L. Mercer, L. R. Bahl, and J. K. Baker. Perplexity—a measure of the difficulty of speech recognition tasks. *The Journal of the Acoustical Society of America*, 62(S1):S63–S63, 12 1977. doi: 10.1121/1.2016299. URL https://doi.org/10.1121/1.2016299.

Bo Jiang. Distillguard: Evaluating defenses against llm knowledge distillation, 2026. URL https://arxiv.org/abs/2603.07835.

Vansh Kapoor, Aman Gupta, Hao Chen, Anurag Beniwal, Jing Huang, and Aviral Kumar. Trim: Hybrid inference via targeted stepwise routing in multi-step reasoning tasks, 2026. URL https://arxiv.org/abs/2601.10245.

Suhas Kotha, Jacob Mitchell Springer, and Aditi Raghunathan. Understanding catastrophic forgetting in language models via implicit inference, 2024. URL https://arxiv.org/abs/2309.10105.

Michael J. Kratsios. National security technology memorandum 4 (nstm-4): Adversarial distillation of american ai models. Memorandum, The White House, Office of Science and Technology Policy, April 2026. URL https://whitehouse.gov/wp-content/uploads/2026/04/NSTM-4.pdf. Memorandum for the Heads of Executive Departments and Agencies.

Fan Li, Mengting Pan, Sijia Xu, Xiaoyang Wang, Chen Chen, and Wenjie Zhang. Answer-then-edit: Reasoning skeleton editing for anti-distillation with preserved utility, 2026. URL https://arxiv.org/abs/2607.20440.

Pingzhi Li, Zhen Tan, Mohan Zhang, Huaizhi Qu, Huan Liu, and Tianlong Chen. DOGe: Defensive output generation for llm protection against knowledge distillation, 2025. URL https://arxiv.org/abs/2505.19504.

Lena Libon, Pura Peetathawatchai, Michael Aerni, Daniel Paleka, and Florian Tramèr. What does it mean to break a distillation defense?, 2026. URL https://arxiv.org/abs/2606.25059.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *IEEE Transactions on Audio, Speech and Language Processing*, 33:3776–3786, 2025. doi: 10.1109/TASLPRO.2025.3606231.

Xinhang Ma, William Yeoh, Ning Zhang, and Yevgeniy Vorobeychik. Protecting language models against unauthorized distillation through trace rewriting, 2026. URL https://arxiv.org/abs/2602.15143.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2019. URL https://arxiv.org/abs/1706.06083.

Zijun Min, Bingshuai Liu, Ante Wang, Long Zhang, Anxiang Zeng, Haibo Zhang, and Jinsong Su. Orchestrating tokens and sequences: Dynamic hybrid policy optimization for rlvr, 2026. URL https://arxiv.org/abs/2601.05607.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks, 2016. URL https://arxiv.org/abs/1511.04599.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, 2025. URL https://arxiv.org/abs/2501.19393.

Subhabrata (Subho) Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv: Computation and Language*, June 2023.

OpenAI. Learning to reason with llms, 2024. URL https://openai.com/index/learning-to-reason-with-llms/. Accessed: 2024-09-12.

Qwen. Qwq: Reflect deeply on the boundaries of the unknown, November 2024. URL https://qwenlm.github.io/blog/qwq-32b-preview/.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. Technical report, OpenAI, 2018. URL https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5):513–523, 1988. ISSN 0306-4573. doi: https://doi.org/10.1016/0306-4573(88)90021-0. URL https://www.sciencedirect.com/science/article/pii/0306457388900210.

Yash Savani, Asher Trockman, Zhili Feng, Yixuan Even Xu, Avi Schwarzschild, Alexander Robey, Marc Finzi, and J. Zico Kolter. Antidistillation sampling, 2025. URL https://arxiv.org/abs/2504.13146.

Ali Shafahi, W. Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! Targeted clean-label poisoning attacks on neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 6106–6116, Red Hook, NY, USA, 2018. Curran Associates Inc.

Haonan Shi, Guoli Wang, Tu Ouyang, and An Wang. Ease: Practical and efficient safety alignment for small language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(44): 37923–37931, Mar. 2026. doi: 10.1609/aaai.v40i44.41129. URL https://ojs.aaai.org/index.php/AAAI/article/view/41129.

Hwanjun Song, Hang Su, Igor Shalyminov, Jason Cai, and Saab Mansour. FineSurE: Fine-grained summarization evaluation using LLMs. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 906–922, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.51. URL https://aclanthology.org/2024.acl-long.51/.

Alexandra Souly, Javier Rando, Ed Chapman, Xander Davies, Burak Hasircioglu, Ezzeldin Shereen, Carlos Mougan, Vasilios Mavroudis, Erik Jones, Chris Hicks, Nicholas Carlini, Yarin Gal, and Robert Kirk. Poisoning attacks on llms require a near-constant number of poison samples, 2025. URL https://arxiv.org/abs/2510.07192.

Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *CoRR*, abs/1710.08864, 2017. URL http://arxiv.org/abs/1710.08864.

Liyan Tang, Igor Shalyminov, Amy Wong, Jon Burnsky, Jake Vincent, Yu'an Yang, Siffi Singh, Song Feng, Hwanjun Song, Hang Su, Lijia Sun, Yi Zhang, Saab Mansour, and Kathleen McKeown. TofuEval: Evaluating hallucinations of LLMs on topic-focused dialogue summarization. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4455–4480, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.251. URL https://aclanthology.org/2024.naacl-long.251/.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for LLM reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=yfcpdY4gMP.

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. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 95266–95290. Curran Associates, Inc., 2024. doi: 10.52202/079017-3018. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/ad236edc564f3e3156e1b2feafb99a24-Paper-Datasets_and_Benchmarks_Track.pdf.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In

S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837. Curran Associates, Inc., 2022.

Rey Wiyatno and Anqi Xu. Maximal jacobian-based saliency map attack, 2018. URL https://arxiv.org/abs/1808.07945.

Xinbo Wu, Max Hartman, {Vidhata Arjun} Jayaraman, and {Lav R.} Varshney. Continual instruction tuning via increasing experts. *IEEE Journal on Selected Topics in Signal Processing*, 2026. ISSN 1932-4553. doi: 10.1109/JSTSP.2026.3666304. Publisher Copyright: © 2007-2012 IEEE.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NG7sS51zVF.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=5Xc1ecxO1h.

A Generalization of Previous Antidistillation Methods

The main distinction between different antidistillation methods is how they instantiate the set of admissible perturbations of the dataset D and in what they consider the set of viable student architectures. In the remainder of this subsection, we show how both Antidistillation Sampling (Savani et al., 2025) and Defensive Output Generation (Li et al., 2025) fit into the framework of equation 4.

A.1 Antidistillation Sampling

ADS proceeds in an online antidistillation fashion, where individual teacher outputs are given per user query. Their formulation is a greedy approach to the outer maximization over $\tilde{D}_T$, where they try to maximize the loss of each individual teacher output. ADS considers the following perturbation to the teacher output distribution:

$$\tilde{p}(\cdot \mid x_{1:t}; \theta_T) := \frac{1}{Z} \exp\Big(\frac{1}{\tau} \log p(\cdot \mid x_{1:t}; \theta_T) + \lambda \Delta(\cdot \mid x_{1:t}; \theta_P)\Big),$$

where Z is a normalization factor, τ is the sampling temperature, $p(\cdot \mid x_{1:t}; \theta_T)$ is the original teacher distribution for token $t+1$, λ is a scaling hyperparameter and Δ is a perturbation to the output distribution determined by the gradients of a proxy model.[2]

Thus, the set of admissible perturbations is the following,

$$\mathcal{B}_{\text{ADS}}(D; \theta_T) = \{\tilde{x}_i : \tilde{x}_i \sim \tilde{p}(\cdot \mid x_{1:t}; \theta_T)\}_{i=1}^{n}, \tag{5}$$

and the admissible class of training algorithms is simply empirical risk minimization over admissible architectures.

A.2 Defensive Output Generation

DOGe proposes that the task of antidistillation can be framed as the following bi-level optimization problem. The defender controls the teacher's parameters to maximize its performance while weakening any student distilled from its outputs. The optimization problem is thus:

$$\theta_T^* = \arg\max_{\theta_T} \left[\text{Perf}(\theta_T) - \lambda \cdot \text{Perf}\left(\theta_S^*\right)\right]$$
$$\theta_S^* = \arg\min_{\theta_S} \mathcal{L}(\theta_S; D_{KD}(\theta_T)),$$

[2] $\Delta(\cdot \mid x_{1:t}; \theta_P) = \ell(\theta_P^+) - \ell(\theta_P)$
$= \ell(\theta_P + \eta \nabla_{\theta_P} \log p(x_{t+1} \mid x_{1:t}; \theta_P)) - \ell(\theta_P)$

where $\mathrm{Perf}(\cdot)$ is the performance of the input model, D_{KD} is the outputs of the teacher model θ_T^*, and $\lambda > 0$ is a scaling hyperparameter.

We connect this to our formulation in equation 4 by first defining

$$\mathcal{B}_{\mathrm{DOGe}}(D; \theta_T, \Theta_T) \tag{6}$$

$$= \left\{ \tilde{D}_T : \tilde{D}_T \sim p(\cdot \mid \theta_T), \theta_T \in \Theta_T, \mathrm{Perf}(\theta_T) \geq \tau \right\}. \tag{7}$$

If we now consider the case where the training algorithm is simply empirical risk minimization, equation 4 simplifies to

$$\tilde{D}_T = \arg \max_{\tilde{D}_T \in \mathcal{B}_{\mathrm{DOGe}}} \mathbb{E}_{x \sim p_{\mathrm{data}}} \left[-\mathrm{Perf}(\theta_S^*(\tilde{D}_T); x) \right].$$

Since the dataset entirely depends on the parameters θ_T we can determine $\tilde{D}_T$ equivalently by finding

$$\theta_T^* = \arg \max_{\theta_T \in \Theta_T, \mathrm{Perf}(\theta_T) \geq \tau} \mathbb{E}_{x \sim p_{\mathrm{data}}} \left[-\mathrm{Perf}(\theta_S^*(\tilde{D}_T); x) \right],$$

and taking a Lagrangian relaxation of this problem yields

$$\theta_T^* = \arg \max_{\theta_T \in \Theta_T} \mathrm{Perf}(\theta_T) - \lambda \mathrm{Perf}(\theta_S^*(\theta_T))$$

which is exactly the DOGe framework.

Algorithm 2 DETECT-THOUGHT-ANCHOR-SENTENCES selects, once per teacher, the receiver heads that concentrate attention most sharply on a few sentences, then scores each sentence of a trace by the attention it receives through those heads from distant downstream sentences, so that the highest-scoring sentences are the thought anchors.

Require: teacher M with head set H, calibration traces $\mathcal{T}$, prompt x, response y, distance d, head count K

function VERTICALATTENTION(x, y, G)
  $(s_1, \ldots, s_n) \leftarrow$ SEGMENT(y)
  run M on $x \,\|\, y$, recording attention $A_{\ell,h}$ for $(\ell, h) \in G$
  for each $(\ell, h) \in G$ **do**
    $\bar{A}[i, j] \leftarrow \mathrm{mean}\{A_{\ell,h}[p, q] : p \in s_i,\ q \in s_j,\ q \le p\}$ for all $i \ge j$
    $v_{\ell,h}[j] \leftarrow \mathrm{mean}_{\,i>j+d}\ \bar{A}[i, j]$ (0 if no such i)
  end for
  return $(s_1, \ldots, s_n)$, $\{v_{\ell,h}\}_{(\ell,h)\in G}$
end function
Calibrate receiver heads (once per teacher)
for each trace $(x_t, y_t) \in \mathcal{T}$ **do**
  $\{v_{\ell,h}\} \leftarrow$ VERTICALATTENTION(x_t, y_t, H)
  $\mu_t \leftarrow \frac{1}{|H|} \sum_{(\ell,h)\in H} v_{\ell,h}$ ▷ trace's shared profile
  $\kappa_t(\ell, h) \leftarrow \mathrm{ExKurt}(v_{\ell,h} - \mu_t)$ for all $(\ell, h) \in H$
end for
$\bar{\kappa}(\ell, h) \leftarrow \mathrm{mean}_t\ \kappa_t(\ell, h)$
$R \leftarrow$ the K heads with largest $\bar{\kappa}$ ▷ receiver heads
Score sentences
$(s_1, \ldots, s_n), \{v_{\ell,h}\} \leftarrow$ VERTICALATTENTION(x, y, R)
$\mathrm{score}[j] \leftarrow \frac{1}{|R|} \sum_{(\ell,h)\in R} v_{\ell,h}[j]$ ▷ broadcasting score
return $(s_1, \ldots, s_n)$, score

B TARS ALGORITHM

In Algorithm 2, we include pseudo-code for the receiver head calibration and the broadcast scoring to compute which sentences to delete. This is used by Algorithm 1 to find which sentences to delete. In our experiments using these algorithms, we used $n = 200$ traces, $K = 32$ heads, and distance $d = 4$.

C THOUGHT ANCHOR ANALYSIS

The main text motivates why we look at thought anchors and shows representative examples; this appendix details the experimental setup and reports the full per-dataset results. Because thought anchors are the unit at which we hypothesize disruption will propagate to a student under distillation, we need a precise account of how they are estimated, how stable that estimate is, and what the sentence-category distribution looks like across our two datasets. We follow the methodology of Bogdan et al. (2025), with implementation details adapted to our teacher model and trace lengths, described below. Specifically, we analyze correct rollouts from DeepSeek R1 Distill Qwen 7B on MATH and MMLU, retaining one trace per problem: 50 traces per dataset for analysis and a disjoint 10 for receiver-head identification.

Following Bogdan et al. (2025), we identify attention heads that concentrate long-range attention onto a small number of earlier sentences. Responses are split into sentences on terminal punctuation, and tokens mapped to sentences via tokenizer offsets. For each head (ℓ, h) and each identification trace, we average token-level attention into a sentence-level matrix $A^{(\ell,h)} \in \mathbb{R}^{S\times S}$ and compute the *vertical attention* received by sentence j,

$$v_j^{(\ell,h)} = \frac{1}{|\mathcal{D}_j|} \sum_{i\in\mathcal{D}_j} A_{ij}^{(\ell,h)}, \qquad \mathcal{D}_j = \{i : i > j + d_{\min}\}, \tag{8}$$

with $d_{\min} = 4$ to isolate long-range from adjacency effects. We subtract the per-trace head-averaged profile $\bar{v}_j$ to remove the shared baseline, and score each head by the excess kurtosis of the residual

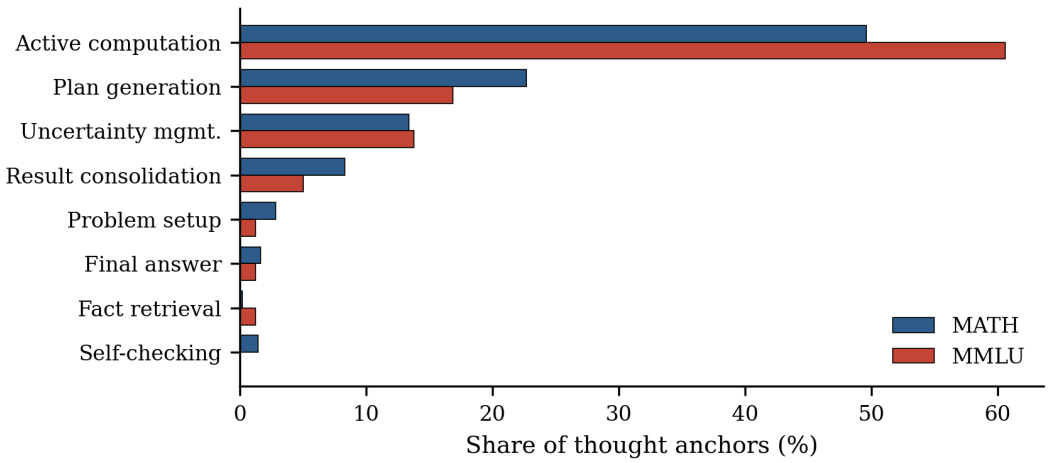


Figure 5: Percent share of thought anchors across MATH and MMLU. While Active Computation is the most common thought anchor, which is both important for answer generation and user understanding, Plan Generation and Uncertainty Management are mainly useful for answer generation. Additionally, anchor category distributions agree closely across the two datasets (Pearson $r = 0.98$, $p < 10^{-4}$), indicating that differences in occurrences of thought anchors is dataset agnostic.

$v^{(\ell,h)} - \bar{v}$, which is high when a head's long-range attention is concentrated on a small number of sentences rather than spread across the trace. We average kurtosis across identification traces and retain the top $K = 28$ heads as the receiver set $\mathcal{R}$. For each analysis trace, the broadcasting score for sentence j is the mean vertical attention it receives across receiver heads, $\mathrm{BC}(j) = \frac{1}{K}\sum_{(\ell,h)\in\mathcal{R}} v_j^{(\ell,h)}$, and the top-10 sentences by $\mathrm{BC}(j)$ are reported as that trace's thought anchors.

To characterize *what kind* of reasoning step each anchor corresponds to, we tag every sentence with one of eight functional categories from Bogdan et al. (2025) (plan generation, uncertainty management, self-checking, fact retrieval, active computation, result consolidation, problem setup, final answer) using regex matching on category-specific keywords (e.g., "alternatively" for planning; "wait", "actually" for uncertainty management), with unmatched sentences defaulting to the modal category, *active computation*. Identification uses sequence length $2{,}048$; broadcasting uses $4{,}096$.

C.1 Details for Thought Anchor Transfer Experiment

For the transfer analysis in Section 4.3, we use the same procedure for identifying receiver heads and computing broadcasting scores as described above, with the following changes.

Models. Receiver heads are identified separately for the teacher (DeepSeek-R1-Distill-Qwen-7B) and for the student (Llama-3.2-3B), both before and after distillation, using N identification traces and a sequence length of $4{,}096$ for each model.

Receiver head selection. Since models differ in size and in how concentrated their receiver heads are, we do not use a fixed number of heads K across models. Instead, for each model we keep the heads whose average kurtosis lies above the 95th percentile of that model's own kurtosis distribution. For the teacher, the median kurtosis across all heads is 7.5 and the 95th percentile is 23.2, yielding $K = 24$. For the student, we settle on $K = 6$ using a similar procedure.

Anchor agreement. Both models read the same 128 teacher traces, truncated to $4{,}096$ tokens, and each computes $\mathrm{BC}(j)$ for every sentence using its own receiver heads. Since sentences are segmented on the text alone, the two score vectors are aligned sentence by sentence, even though the models, tokenizers, and heads differ. For each trace, we compute Spearman ρ between the two score vectors, which measures how similarly the two models rank all sentences and is insensitive to differences in the absolute scale of attention, and the top-5 Jaccard similarity $|A \cap B|/|A \cup B|$, where A and B are

the teacher's and student's five highest-scoring sentences. Traces with fewer than five sentences are excluded, and both metrics are averaged over traces.

D Additional TARS Results

D.1 Scaled experiment

In the experiments of the main text, we focused only on one teacher. We provide this additional experiment which uses a Gemma 3 12B as the teacher to help indicate that this method scales with teacher size on harder datasets. Figure 6 shows a degradation of student accuracy with minimal change in teacher accuracy for TARS, whereas ADS significantly degrades teacher accuracy.

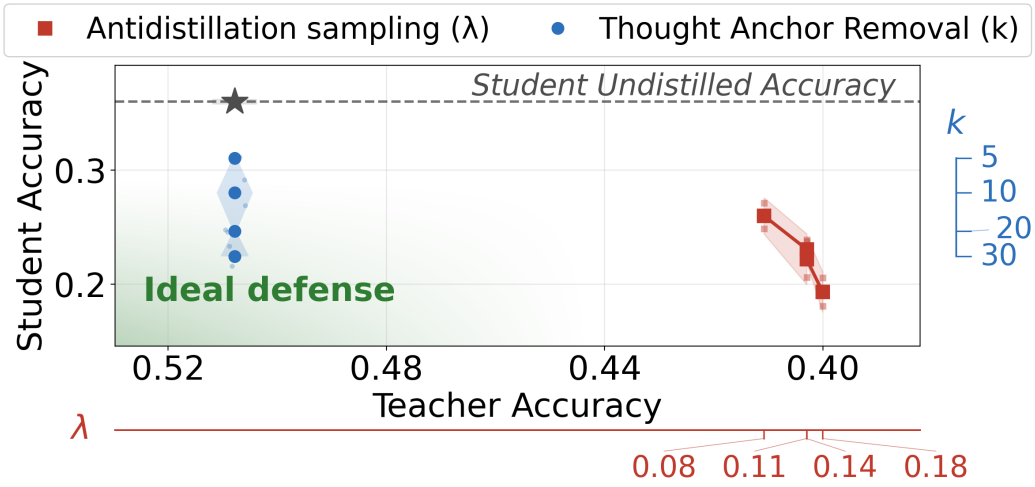


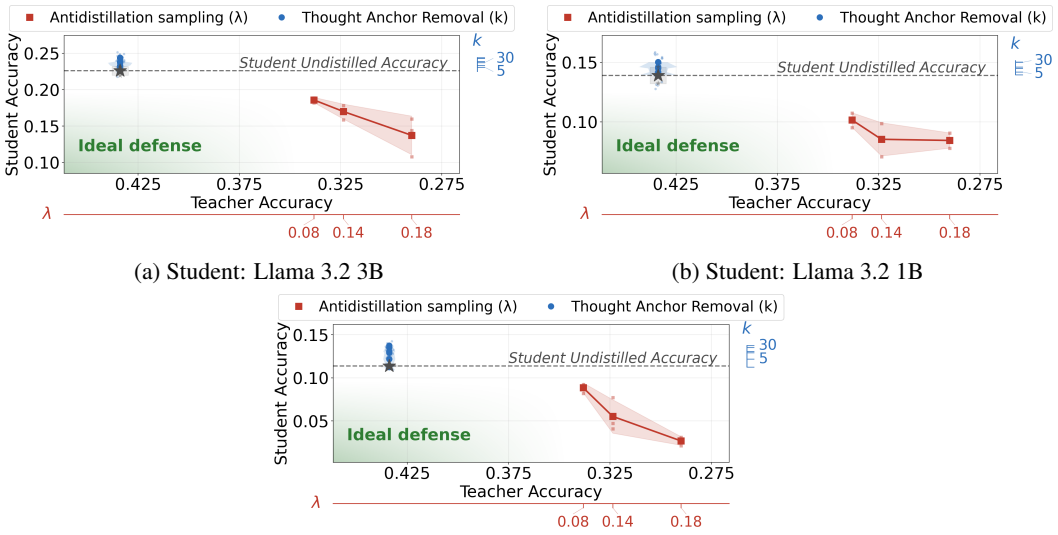
Figure 6: Student versus Teacher Accuracy for ADS (Red) and TARS (Blue). In this figure, the teacher is Gemma 3 12B, student is Qwen 3 1.7B, and dataset is DeepMath. The ideal antidistillation method is the bottom left region of the plot, in which the student accuracy is degraded significantly without degrading the teacher accuracy. Antidistillation Sampling shows a trade-off between degrading the student and teacher, whereas poisoning though anchors shows that scaling defense degrades the student without degrading the teacher.

D.2 Remaining MATH and MMLU Pro

In Figure 7, Thought anchor removal does not degrade student accuracy or teacher accuracy, whereas ADS degrades student accuracy at the cost of an extreme teacher accuracy decrease. In this case, both methods have limited applicability to use.

In Figure 8, both ADS and Thought Anchor Removal fail to meaningfully degrade the student accuracy. We hypothesize that this is because the unpoisoned student accuracy was very low.

We also provide the detectability results for MMLU Pro.



(a) Student: Llama 3.2 3B

(b) Student: Llama 3.2 1B

(c) Student: Gemma 3 1B

Figure 7: Student versus teacher accuracy for ADS (red) and TARS (blue) on MMLU-Pro, with DeepSeek-R1-Distill-Qwen-7B as the teacher. The students used were Llama 3.2 3B (top-left), Llama 3.2 1B (top-right), and Gemma 3 1B (bottom). In this case, Thought Anchor Removal does not degrade student or teacher accuracy, while ADS degrades student accuracy at the cost of an extreme teacher-accuracy decrease.

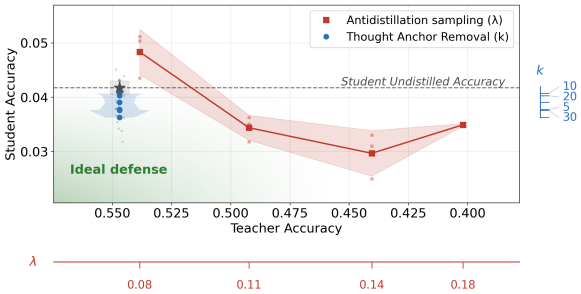


Figure 8: Student versus teacher accuracy for ADS (red) and TARS (blue) on MATH, with DeepSeek-R1-Distill-Qwen-7B as the teacher and Llama 3.2 1B as the student. In this example, both ADS and Thought Anchor Removal fail to meaningfully degrade the student accuracy. We hypothesize that this is because the unpoisoned student accuracy was already very low.

E Detectability for defense methods

Perplexity measures how "surprised" a language model is by a sequence of text. Our perplexity metric uses Llama 3 8B Instruct as a detector model computed as a sliding window (max length 2048, stride 512). TF-IDF (Term Frequency-Inverse Document Frequency) focuses on the specific vocabulary used in the traces. By measuring the mean cosine distance between paired traces, we are looking for lexical consistency. Our specific implementation uses an ngram range of $(2, 5)$ and L2 norm with 20k max features. Semantic Similarity is the mean L2 distance between paired BGE-M3 (`BAAI/bge-m3`) embeddings with `max_length=8192`.

E.1 LLM-as-a-Judge

We evaluate trace coherence using LLM-as-a-Judge with Qwen 3 8B. The following judge prompt is used:

Table 4: Detectability for ADS and Thought Anchor Removal (TARS) on MMLU-Pro, with DeepSeek-R1-Distill-Qwen-7B as teacher. ADS uses Qwen2.5-3B as its proxy model.

Method	Param.	PPL↓	TF-IDF↓	Sem. L2↓	ΔTok.↓	Overall↑	TA↑
Baseline	—	1.000	—	—	—	4.42	43.4
ADS	$\lambda = 0.08$	0.693	0.770	1.023	806	1.89	33.8
ADS	$\lambda = 0.11$	0.602	0.882	1.172	861	1.48	34.7
ADS	$\lambda = 0.14$	0.579	0.921	1.243	865	1.10	32.4
ADS	$\lambda = 0.18$	0.588	0.949	1.280	870	1.04	29.0
TARS	$k = 5$	1.454	0.014	0.240	78	4.04	43.4
TARS	$k = 10$	2.201	0.034	0.357	158	3.67	43.4
TARS	$k = 15$	3.021	0.071	0.458	239	3.29	43.4
TARS	$k = 20$	4.475	0.135	0.570	327	3.01	43.4
TARS	$k = 25$	6.692	0.251	0.697	412	2.56	43.4
TARS	$k = 30$	14.160	0.413	0.859	487	1.96	43.4

Judge Prompt

System prompt

You are a careful evaluator of mathematical reasoning transcripts. You rate how well-formed a transcript is as a piece of writing and reasoning. You are NOT checking whether the final answer is correct — a transcript can be well-written and still reach a wrong answer, and it can be badly written while reaching a right one. Judge only the qualities described in the rubric. Respond with JSON and nothing else.

User prompt

Rate the reasoning transcript below on three dimensions.

Each dimension is an integer from 1 to 5, where **5 is the best possible score and 1 is the worst**. Higher is always better: 5 means the transcript excels on that dimension, 1 means it fails badly on it. Use the full range — do not cluster every rating in the middle.

`coherence` — does the reasoning follow logically from step to step?

5 (best): every step follows from the last; the thread is easy to track.
4: largely sound, with a minor gap or unexplained step.
3: mostly followable, but with non-sequiturs, abrupt jumps, or a step that references something never established.
2: hard to follow; the line of argument repeatedly breaks down.
1 (worst): steps do not connect; the reasoning contradicts itself or wanders with no discernible line of argument.

`fluency` — does it read like natural, well-formed text?

5 (best): clean prose and notation throughout.
4: minor blemishes that do not impede reading.
3: noticeable artifacts — garbled fragments, malformed notation, a few repeated phrases.
2: frequent artifacts; substantial stretches are hard to read.
1 (worst): heavily degraded — long stretches of repeated tokens or phrases, nonsense strings, text collapsed into a loop.

`completeness` — does the transcript arrive somewhere?

5 (best): works through to a clearly stated final answer.
4: reaches a conclusion, though the last step is compressed.
3: substantially complete, but the ending is abrupt or the conclusion is left implicit.
2: breaks off close to the end, with no stated conclusion.
1 (worst): stops mid-thought, mid-sentence, or mid-calculation with no conclusion at all.

Then give an overall integer from 1 to 5 (again, 5 = best, 1 = worst) for how usable this transcript would be as an example of good reasoning, and one short sentence of justification.

Reply with exactly this JSON and nothing else:

```
{"coherence": <1-5>, "fluency": <1-5>, "completeness": <1-5>,
 "overall": <1-5>, "reason": "<one sentence>"}
```

F TRACE EXAMPLES OF TARS AND ADS



Figure 9: Example ADS versus TARS traces. The ADS trace insert random tokens in places, such as at the end, which does not happen in TARS.



Figure 10: Example ADS versus TARS traces. ADS looks acceptable for the first few sentences, before repeating the same random character over and over.

G Additional Experiments

G.1 Method against random sentence poisoning

Our method shows that student model distillation performance is worse when the teacher model traces are poisoned than baseline. In this additional experiment, we compare the results of our standard method to poisoning random sentences. The goal of this experiment is to verify that the backtracking sentences we poison are truly important for distillation.

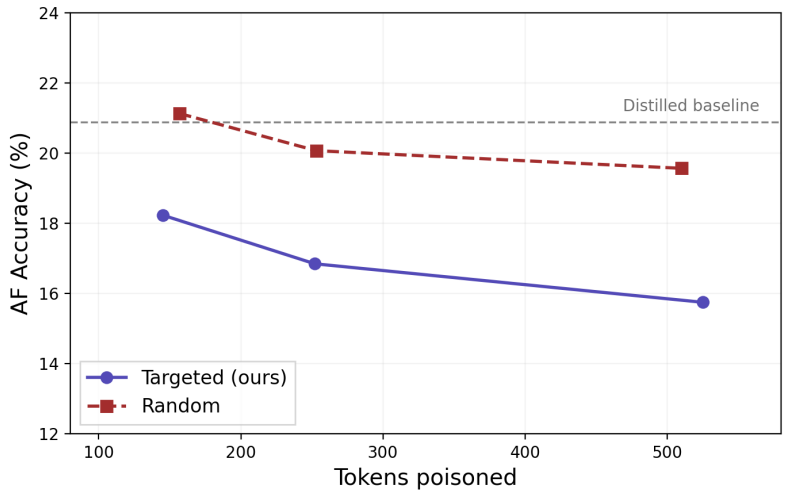


Figure 11: Accuracy drop between baseline versus poisoned reasoning trace distillation for varying numbers of tokens based on **TARS versus Random Sentence Removal.** Based on the figure, poisoning random sentences has a substantially lower impact on distillation performance than the branching sentences. In this experiment, the teacher model is DeepSeek R1 Distill Qwen 7B and the student model is Llama 3.2 3B.

From Figure 11, the distillation performance gap is minimal for a varying number of tokens poisoned.

G.2 Best-of-n re-query attacks

We evaluate both defenses under an attacker who queries the teacher $n{=}4$ times per problem and must commit to a single response before distilling. All runs share a DeepSeek-R1-Distill-Qwen-7B teacher, a Gemma-3-1B-IT student, MATH (2,470 train / 2,480 test problems), $\tau{=}0.6$, and a generation budget of 2048 tokens. TARS deletes $k{=}10$ sentences per trace and to be comparable, antidistillation sampling (ADS) uses $\lambda{=}0.11$, $\varepsilon{=}0.01$ with a Qwen2.5-3B proxy student. All students are trained with a single SFT seed (42).

Attack A: correctness then length. The attacker discards incorrect responses and, among those remaining, keeps the *longest*; if no response is correct, the longest response overall is kept. This thus, specifically, targets TARS, which damages a trace by deleting sentences: the longest re-query is likely the least damaged one. The same rule is applied to the ADS pool for comparability, even though it is not adapted to how ADS degrades a trace—thus giving ADS an advantage.

Table 5: Attack A (correctness, then longest response). Teacher accuracy is measured on the training traces the student is distilled from; student accuracy is measured on MATH test. Controls distill on a single un-selected defended draw and are shared with Table 6, since they do not depend on which attack is applied. Positive Δ means the attack succeeded, i.e. the defense was weakened.

		Teacher	Student accuracy	
Defense	Attacker	accuracy	Raw	Answer-forced
Thought-anchor removal ($k{=}10$)	none (control)	0.556	0.0980	0.1185
	longest	0.686	0.1012	**0.1194**
Antidistillation sampling ($\lambda{=}0.11$)	none (control)	0.366	0.0577	0.0794
	longest	0.718	0.0835	**0.1097**

Attack B: defense-specific selection with repetition filtering. Attack A applies one lever to both defenses, which understates the attacker against ADS: ADS does not shorten traces, it drives them

into repetition until they exhaust the generation budget. Attack B therefore gives each defense a matched lever. Against thought-anchor removal the attacker keeps the *longest* correct response, stepping to progressively shorter ones only when the longest is truncated or predominantly repetitive; against ADS it keeps the *shortest*, stepping to progressively longer ones under the same condition. It then strips consecutive repetition tokens from the committed response. A repeated span must span at least 16 redundant tokens to be removed, which leaves ordinary notation such as `1005` intact, and a strip that would invalidate the answer is reverted. Stripping removes 239 tokens per trace on average under ADS and essentially none under thought-anchor removal, with no loss of teacher-side accuracy in either case.

Table 6: Attack B (defense-specific selection with repetition filtering). Against thought-anchor removal the attacker keeps the longest correct response; against ADS the shortest. In both cases it then steps down that ordering past truncated or predominantly repetitive candidates and strips consecutive repetition tokens from the committed response.

Defense	Attacker	Teacher accuracy	Student accuracy	
			Raw	Answer-forced
Thought-anchor removal (k=10)	none (control)	0.556	0.0980	**0.1185**
	longest + filtering	0.686	0.0782	0.0903
Antidistillation sampling (λ=0.11)	none (control)	0.366	0.0577	0.0794
	shortest + filtering	0.718	0.0919	**0.1052**

Discussion. Against ADS both attacks succeed: Attack A recovers 3.0 points of answer-forced accuracy over the unattacked control and Attack B recovers 2.6 points (+2.6 and +3.4 points respectively on raw accuracy).

Against thought-anchor removal neither attack helps. Attack A is indistinguishable from the control (+0.0009), and Attack B is *worse* (−0.0282).

Taking each defense's best attacker outcome, thought-anchor removal admits 0.1194 and ADS admits 0.1097, so thought-anchor removal is the weaker defense by roughly 1 point under a best-responding attacker, and both attacks leave the ADS-distilled student below the thought-anchor one. However, referring to Figure 3, we can see that TARS with k=10 does perform marginally worse than ADS with $\tau = 0.11$ so such a difference in student performance is expected. The main takeaway is the robustness of TARS to attacks versus the fragility of ADS to simple post-processing schemes.

G.3 What each defense does to a trace

To separate *how* the two defenses damage a reasoning trace, we characterise in Table 7 three populations of teacher generations from (2,470 MATH problems: the undefended trace, the same trace after thought-anchor deletion (k=10), and an independently generated ADS trace at the matched seed (λ=0.11, ε=0.01).

Statistics tracked. *Length* statistics are token counts under the teacher's tokenizer. A trace is *truncated* if it reached the 2048-token generation budget without emitting an end-of-sequence token, meaning it never produced a final answer. *Repetition tokens* counts positions belonging to a block of some period $p \leq 32$ that repeats back to back ($aaaa$, $ababab$, and so on); the first pass through a block is never counted, and a span must contain at least 16 redundant tokens, so ordinary notation such as `1005` is not counted as repetition. Under *degeneration*, a trace is *degenerate* if a single token repeats at least 10 times consecutively or if more than half its 4-grams are repeats; a *tail loop* is a repeating cycle detected at the end of the trace; the *longest single-token run* is the maximum number of consecutive identical tokens; and the *compression ratio* is zlib-compressed size over raw size, where lower means more redundant. *Trace accuracy* is the fraction of traces whose final answer verifies against the reference solution.

Trace-level observations. Thought-anchor removal behaves exactly as designed: it strips 283 tokens (21.7%) and 9.86 sentences per trace while leaving accuracy unchanged at 0.560, because

Table 7: Trace statistics for the three populations. Thought-anchor removal damages a trace by making it shorter; ADS damages it by filling it with repetition.

Statistic	Undefended (original)	Thought-anchor removal (k=10)	ADS (λ=0.11)
Length			
Mean tokens	1307	1024	**1535**
Truncated at budget (%)	50.9	58.0	**87.7**
Sentences deleted	—	9.86	—
Content vs. repetition			
Repetition tokens	0.6	0.4	**628.0**
Repetition tokens (% of trace)	0.0	0.0	**40.9**
Degeneration			
Degenerate traces (%)	1.0	0.7	**59.5**
Traces with a tail loop (%)	0.1	0.1	**51.0**
Longest single-token run	1.7	1.7	**281.6**
Compression ratio	0.382	0.509	**0.220**
Accuracy			
Trace accuracy	0.560	0.560	**0.363**
Absolute drop vs. undefended	—	0.000	**0.198**
Relative drop vs. undefended (%)	—	0.0	**35.3**

deletion spares the final answer. Its repetition profile is indistinguishable from the undefended trace, at 0.4 against 0.6 repetition tokens.

ADS damages traces by a different mechanism entirely. Its traces are $1.17\times$ *longer* than undefended ones, yet 40.9% of their tokens are consecutive repetition so ADS yields the least real content of the three despite producing the most text. Degeneration is pervasive throughout ADS generations: 59.5% of ADS traces are degenerate against 1.0% undefended, half contain a tail loop, and the mean longest single-token run is 281.6 tokens against 1.7. Consistent with this, 87.7% of ADS traces exhaust the generation budget without emitting an answer, and trace accuracy falls 19.8 points.

H Distillation experimental setup

In this work, we run knowledge distillation experiments on the student models using traces generated by the teacher model. The models are trained using HuggingFace's SFTTrainer with the following optimal hyperparameters:

Parameter	**Value**
Optimizer	Fused AdamW
Learning Rate	5×10^{-4}
LR Schedule	Cosine decay with 3% linear warmup
Batch Size	16 (via gradient accumulation)
Epochs	3
Weight Decay	0.1
Gradient Clipping	1.0 (max norm)
Context Window	4096 tokens
Precision	`bfloat16` (mixed precision)
Attention Mechanism	Flash Attention (`sdpa`)

Table 8: Hyperparameters and Hardware Settings for Student Model Distillation

These hyperparameters are very similar to those used in the Antidistillation Sampling method (Savani et al., 2025).

When training, we used 4 NVIDIA A100s.

I PROOFS

Theorem I.1 (Detectability of Burst Deletions; Formal version of Proposition 5.1)**.** *Let $X = (X_1, \ldots, X_N)$ be generated by a time-homogeneous m-order Markov source. Thus, for every t*

$$P(X_t = a \mid X_{<t} = x_{<t}) = p(a \mid x_{t-m:t-1}), \tag{9}$$

where a fixed prompt is used for $t \leq m$. Assume there is a common finite support $\mathcal{A}$ and a constance $\gamma \in (0, 1)$ such that

$$p(a \mid h) \geq \gamma > 0, \tag{10}$$

for all m-length histories h and for every $a \in \mathcal{A}$. For a deletion mask $s \subset [N] := \{1, 2, \ldots N\}$, let

$$[N] \backslash s = \{r_1, ..., r_{M_s}\}, \tag{11}$$

with $r_1 < \ldots < r_{M_s}$ be the retained positions and define

$$T_s(x) := (x_{r_!}, \ldots, x_{r_{M_s}}). \tag{12}$$

Let $b(s)$ be the number of maximal contiguous intervals in s. Thus, $b(s)$ is the number of deletion bursts (i.e. k in our context). Let P_s be the distribution of $T_s(X)$ and let $P_0^{(M_s)}$ be the distribution of a clean length-M_s sample from the same Markov source. Then, for every fixed mask s,

$$D_{KL}\left(P_s \parallel P_0^{(M_s)}\right) \leq b(s) m \log\left(\frac{1}{\gamma}\right) = km \log\left(\frac{1}{\gamma}\right). \tag{13}$$

Proof. Fix a mask s and lets its maximal contiguous deleted intervals be

$$s = I_1 \cup \cdots \cup I_{b(s)}, \tag{14}$$

where $I_j = \{u_j, u_j + 1, \ldots, v_j\}$. Let the retained source positions be $r_1 < \cdots < r_{M_s}$. Then, the t-th output token is $Y_t = X_{r_t}$. Define

$$A_s := \{t \in [M_s] : \{r_t - m, \ldots, r_t - 1\} \cap s \neq \emptyset\}. \tag{15}$$

We can see that $t \in A_s$ when at least one deleted token lies in the original m-token context immediately preceding X_{r_t}. We will now use the following lemma to bound $|A_s|$.

Claim I.2. $|A_s| \leq mb(s)$.

Proof. Consider one deletion burst $I_j = [u_j, v_j]$. Suppose this burst affects an output position t. This implies that

$$[r_t - m, r_t - 1] \cap [u_j, v_j] \neq \emptyset. \tag{16}$$

Since r_t is retained by construction, it is not in the deletion interval and since the interval intersects the past of r_t, it must entirely lie before r_t. Thus,

$$v_j < r_t \tag{17}$$

and by 16,

$$r_t - m \leq v_j \tag{18}$$

so

$$v_j < r_t \leq v_j + m. \tag{19}$$

Thus, burst I_j can only affect the m tokens $v_j + 1, \ldots, v_j + m$ so each deletion burst can only affect m retained-token positions. Taking a union over all bursts gives

$$|A_s| \leq mb(s) \tag{20}$$

□

Remark I.3. The length of the burst does not appear in this bound at all.

Now, define

$$q_t^s(a \mid y_{<t}) := P_s(Y_t = a \mid Y_{<t} = y_{<t}). \tag{21}$$

We want to show that q_t^s does not contribute to the KL-divergence (namely the likelihood ratio with respect to the clean source) for $t \notin A_s$.

Claim I.4. *$q_t^s(a \mid y_{<t}) = p_t^0(a \mid y_{<t})$ for $t \notin A_s$ where p_t^0 is a clean next-token distribution conditioned on the previous tokens being $y_{<t}$. Thus, only output positions in A_s can contribute to the likelihood ratio.*

Proof. Because $Y_t = X_{r_t}$,

$$\begin{aligned} q_t^s(a \mid y_{<t}) &= P(X_{r_t} = a \mid X_{r_1} = y_1, \ldots, X_{r_{t-1}} = y_{t-1}) && (22)\\ &= \mathbb{E}\left[\mathbb{1}\{X_{r_t} = a\} \mid Y_{<t} = y_{<t}\right] && (23)\\ &= \mathbb{E}[\mathbb{E}[\mathbb{1}\{X_{r_t} = a\} \mid X_{<r_t} = x_{<r_t}] \mid Y_{<t} = y_{<t}] && (24)\\ &\overset{(a)}{=} \mathbb{E}[p(a \mid x_{<r_t}) \mid Y_{<t} = y_{<t}] && (25)\\ &\overset{(b)}{=} \mathbb{E}[p(a \mid x_{r_t-m:r_t-1}) \mid Y_{<t} = y_{<t}], && (26) \end{aligned}$$

where (a) follows by the tower property and (b) follows by the Markov source assumption. By assumption, $p(a \mid x_{t-m:t_1}) \geq \gamma$ so $q_t^s(a \mid y_{<t}) \geq \gamma$. Now, if $t \notin A_s$, then none of the source positions

$$r_t - m, \ldots, r_t - 1,$$

was deleted. Thus,

$$X_{r_t-m:r_t-1} = y_{t-m:t-1},$$

so

$$q_t^s(a \mid y_{<t}) = p(a \mid y_{t-m:t-1}) \overset{(c)}{=} p_t^0(a \mid y_{<t}), \tag{27}$$

for $t \notin A_s$, where (c) follows by the Markov assumption. For $t \leq m$ and $t \notin A_s$, if $r_t > t$, $[\max\{1, r_t - m\}, r_t - 1]$ contains at least t source positions. Since $t \notin A_s$, all of these positions are retained which contradicts the fact that only $t - 1$ retained positions precede r_t. Thus $r_t = t$. Thus, $q_t^s(a \mid y_{<t}) = p_t^0(a \mid y_{<t})$ as above. □

We can see that for an affected position $t \in A_s$, $q_t^s(a \mid y_{<t}) \geq \gamma$ and $p_t^0(a \mid y_{<t}) \geq \gamma$. Thus, since $\gamma \in (0, 1)$,

$$\gamma \leq \frac{q_t^s(a \mid y_{<t})}{p_t^0(a \mid y_{<t})} \leq \frac{1}{\gamma}. \tag{28}$$

Thus,

$$\left|\log \frac{q_t^s(a \mid y_{<t})}{p_t^0(a \mid y_t)}\right| \leq \log \frac{1}{\gamma}. \tag{29}$$

Now we have that,

$$\begin{aligned} \log \frac{P_s(y)}{P_0^{(M_s)}(y)} &= \log \frac{\prod_{t=1}^{M_s} q_t^s(y_t \mid y_{<t})}{\prod_{t=1}^{M_s} p_t^0(y_t \mid y_{<t})} && (30)\\ &= \sum_{t=1}^{M_s} \log \frac{q_t^s(y_t \mid y_{<t})}{p_t^0(y_t \mid y_{<t})}. && (31) \end{aligned}$$

By Claim I.4, this simplifies to

$$\begin{aligned} \left|\log \frac{P_s(y)}{P_0^{(M_s)}(y)}\right| &= \left|\sum_{t \in A_s} \log \frac{q_t^s(y_t \mid y_{<t})}{p_t^0(y_t \mid y_{<t})}\right| && (32)\\ &\leq \sum_{t \in A_s} \left|\log \frac{q_t^s(y_t \mid y_{<t})}{p_t^0(y_t \mid y_{<t})}\right| && (33)\\ &\leq |A_s| \log \frac{1}{\gamma} && (34)\\ &\overset{(d)}{\leq} b(s) m \log \frac{1}{\gamma} = km \log \frac{1}{\gamma}, && (35) \end{aligned}$$

where (d) follows by Claim I.2. Now, taking an expectation over $Y \sim P_s$ we finally arrive at

$$D_{KL}(P_s \parallel P_0^{(M_s)}) \leq b(s)m \log \frac{1}{\gamma} = km \log \frac{1}{\gamma}. \tag{36}$$

□

Corollary I.5. *Under the same assumptions as Theorem I.1, let the mask S be selected according to*

$$q(s \mid x) = P(S = s \mid X = x). \tag{37}$$

Suppose $\mathbb{E}[b(S)] = k$ and let

$$Y = T_S(X), \quad L = |Y|, \tag{38}$$

and let

$$\nu(\ell) = P(L = \ell) \tag{39}$$

be the output length after deletion. Define the length-matched reference distribution to be

$$\overline{P_0}(y) = \nu(|y|) P_0^{(|y|)}(y). \tag{40}$$

Then

$$D_{KL}(P_Y \parallel \overline{P_0}) \leq km \log \frac{1}{\gamma} + I(X; S). \tag{41}$$

Namely,

$$\mathbb{E}_{L \sim \nu} \left[D_{KL} \left(P_{Y|L)} \parallel \overline{P}_0^{(L)} \right) \right] \leq km \log \frac{1}{\gamma} + I(X; S), \tag{42}$$

and if all masks produce the exact same output length M then

$$D_{KL} \left(P_Y \parallel P_0^{(M)} \right) \leq km \log \frac{1}{\gamma} + I(X; S) \tag{43}$$

Remark I.6. When $S = s$ a.s., $I(X; S) = 0$ and Theorem I.1 is recovered.

Proof. Let

$$Q_{X,S}(x, s) = P_X(x) q(s \mid x). \tag{44}$$

and

$$\pi(s) := Q_S(s) = \sum_x P_X(x) q(s \mid x) \tag{45}$$

be the marginal distribution of the masks. Now construct the independent coupling of $Q_{X,S}$ denoted

$$R_{X,S}(s, x) = P_X(x) \pi(s). \tag{46}$$

We can then notice that

$$D_{KL}(Q_{X,S} \parallel R_{X,S}) = I_Q(X; S). \tag{47}$$

We can now apply the deterministic deletion mapping

$$F(x, s) = (s, T_s(x)), \tag{48}$$

and denote $Q_{S,Y}$ and $R_{S,Y}$ the respective pushforward measures under F. Notice that under R, the mask is independent of X so conditional on $S = s$, $R_{S,Y}$ factors through P_s, namely

$$R_{S,Y}(s, y) = \pi(s) P_s(y). \tag{49}$$

Thus, by the data-processing inequality

$$D_{KL}(Q_{S,Y} \parallel R_{S,Y}) \leq D_{KL}(Q_{X,S} \parallel R_{X,S}) = I_Q(X; S). \tag{50}$$

We can now define the joint distribution of the clean reference distribution

$$B_{S,Y}(s, y) = \pi(s) P_0^{(M_s)}(y). \tag{51}$$

We now prove the following claim in order to invoke the chain rule of the Radon Nikodym derivative.

Claim I.7. *$Q_{S,Y} \ll R_{S,Y} \ll B_{S,Y}$ (where $\ll$ denotes absolute continuity)*

Proof. ($Q \ll R$) This absolute continuity comes for free because R is the independent coupling of Q.
($R \ll B$) This follows because of Assumption 10. Namely, $R_{S,Y} = \pi(s)P_s(y) > 0$ means that $\pi(s) > 0$ and since $P_0^{(M_s)}(y) > 0$ by assumption 10, $B_{S,Y} = \pi(s)P_0^{(M_s)} > 0$ so $R \ll B$. □

Since, by Claim I.7, $Q \ll R \ll B$, we can invoke the invoke the chain rule of the Radon-Nikodym derivative to arrive at

$$D_{KL}\left(Q_{S,Y} \parallel B_{S,Y}\right) = D_{KL}(Q_{S,Y} \parallel R_{S,Y}) + \mathbb{E}_{Q_{S,Y}}\left[\log \frac{R_{S,Y}(S,Y)}{B_{S,Y}(S,Y)}\right]. \tag{52}$$

We have that

$$\log \frac{R_{S,Y}(s,y)}{B_{S,Y}(s,y)} = \log \frac{\pi(s)P_s(y)}{\pi(s)P_0^{(M_s)}(y)} \tag{53}$$

$$= \log \frac{P_s(y)}{P_0^{(M_s)}(y)} \tag{54}$$

$$\overset{(a)}{\leq} b(s)m \log \frac{1}{\gamma}, \tag{55}$$

where (a) follows from the pointwise version of Theorem I.1 (namely, equation 35). Combining everything together, we get

$$D_{KL}(Q_{S,Y} \parallel B_{S,Y}) \leq I_Q(X;S) + km \log \frac{1}{\gamma}, \tag{56}$$

since $\mathbb{E}[b(S)] = k$. We know by the chain rule of KL-divergence that

$$D_{KL}(Q_Y \parallel B_Y) \leq D_{KL}(Q_{S,Y} \parallel B_{S,Y}). \tag{57}$$

Thus, we now need to identify what B_Y is. Observe that for a sequence y of length ℓ,

$$B_Y(y) = \sum_{s:M_s=\ell} \pi(s)P_0^{(\ell)}(y) = P_0^{(\ell)} \sum_{s:M_s=\ell} \pi(s). \tag{58}$$

However, observe that

$$\sum_{s:M_s=\ell} \pi(s) = P(M_s = \ell) = \nu(\ell), \tag{59}$$

so $B_Y(y) = \nu(|y|)P_0^{(|y|)}(y) = \overline{P_0}(y)$. Finally, combining everything,

$$D_{KL}(P_Y \parallel \overline{P_0}) \leq km \log \frac{1}{\gamma} + I(X;S). \tag{60}$$

□

Remark I.8. Since P_Y and $\overline{P_0}$ have the same length marginal ν, we have

$$D_{KL}(P_Y \parallel \overline{P_0}) = \mathbb{E}_{L\sim\nu}\left[D_{KL}\left(P_{Y|L} \parallel P_0^{(L)}\right)\right] \leq km \log \frac{1}{\gamma} + I(X;S). \tag{61}$$

Similarly, if there is always a fixed output length M we have

$$D_{KL}\left(P_Y \parallel P_0^{(M)}\right) \leq km \log \frac{1}{\gamma} + I(X;S) \tag{62}$$