Inconsistency is Key: Lightweight Distillation-resistance via Dynamic Style Switching

Anonymous ACL submission

Abstract

Knowledge distillation (KD) is a highly effective paradigm for transferring capabilities from large language models (LLMs) to smaller, more specialized student models. In commercial deployment where frontier models are restricted behind black-box APIs, sequence-level distillation remains a potent vector for competitors to clone proprietary capabilities at a fraction of the development cost. Existing defenses mostly rely on surrogate models to manipulate next-token sampling, which inflates computational overhead and real-time inference latency. In this work, we propose a lightweight defense that dynamically injects style-steering vectors into the teacher model's activations upon malicious queries. This framework offers exceptional flexibility, allowing hosts to customize target defensive styles and dynamically blend them to maximize stylistic variance. Crucially, this inference-time intervention preserves semantic utility and fluency during malicious queries, misleading adversaries into believing that they are extracting pristine teacher outputs, while standard consumers continue to receive original, unmodified responses. Our experiments demonstrate that this framework successfully disrupts student learning, achieving good anti-distillation performance while incurring negligible overhead and preserving text naturalness. Our code is available at https://anonymous.4open.science/r/distill-resist-38C6/.

1 Introduction

Advancements in large language models (LLMs) have enabled unprecedented capabilities in complex reasoning, mathematics, and software engineering, powering a wide range of applications. Since developing such frontier models requires immense computational resources, knowledge distillation (KD) emerged as a powerful technique for creating efficient yet capable models by imitating the behavior of large teacher models (Hinton et al., 2015). In practical deployment, even when most LLMs are restricted to black-box API access, text-level distillation remains highly effective (Kim and Rush, 2016; Xu et al., 2024b). Nevertheless, this efficacy leads to a significant vulnerability: competitors can exploit model outputs to cheaply replicate proprietary capabilities at a fraction of the time and cost required to train such models from scratch (Anthropic, 2026).

Existing defenses can be categorized into detection and prevention mechanisms. Detection-based defenses such as watermarks (Kirchenbauer et al., 2023) and fingerprints (Xu et al., 2024a, 2026) provide post-hoc verification without actively preventing distillation. Among recent black-box prevention defenses, state-of-the-art techniques are anti-distillation sampling (ADS) (Savani et al., 2025) and DOGe (Li et al., 2025). Both methods attempt to produce reasoning traces that maximize student's loss while maintaining teacher model's utility and the readability of the output. They achieve this by leveraging a proxy model for gradient computation with the assumption that the effects generalize to different student models. However, both techniques exhibit some limitations. ADS introduces substantial computational overhead and requires additional memory to store a proxy model, which might be prohibitive for high-throughput inference. DOGe, on the other hand, eliminates this overhead but results in performance degradation of the model due to fine-tuning. Furthermore, because these techniques perturb next-token predictions by maximizing student loss, the output traces sometimes exhibit unnatural artifacts including nonsensical tokens or translated words (in the case of DOGe), as observed during our reproduction of these works.

To address these limitations, an effective defense must satisfy three primary desiderata: (i) *Benign Utility*, where the defense preserves the teacher model's performance for legitimate queries;

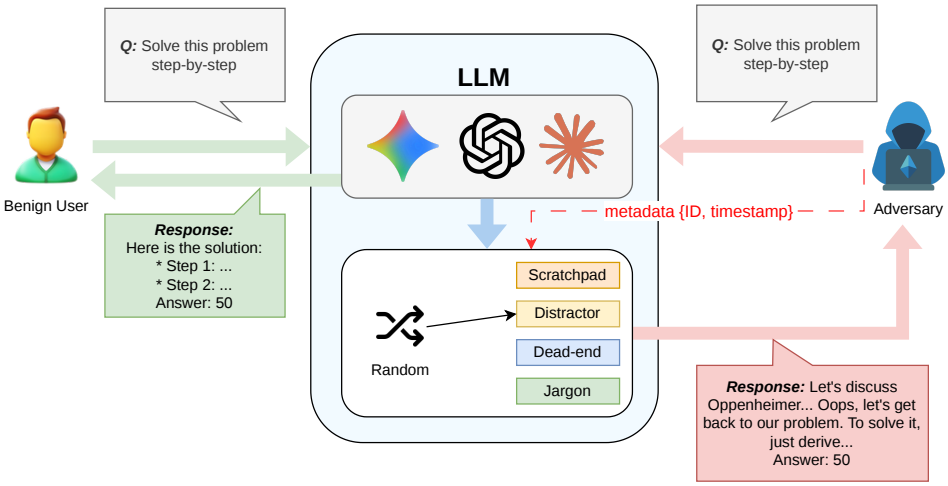


Figure 1: Overview of our lightweight distillation-resistant defense framework. For benign queries, the teacher model $\mathcal{T}$ generates standard responses. Upon detecting malicious queries, the framework dynamically switches the output style by injecting pre-computed style-steering vectors into the hidden activations at inference time, disrupting student model distillation while maintaining semantic coherence.

(ii) *Computational Efficiency*, incurring negligible overhead during inference; and (iii) *Conversational Naturalness*, which restricts defensive perturbations to natural, human-like thinking processes and avoids generating nonsensical linguistic artifacts. In the context of sequence-level KD via supervised fine-tuning (SFT), it is more challenging to balance the trade-off between (i) and (iii) compared to logit-level approaches because the student model relies solely on the teacher's generated text. In logit-level KD, a successful defense can manipulate unnormalized token logits to corrupt the "dark knowledge" (e.g., the conditional mutual information between inputs and logits), without compromising the linguistic fluency of the teacher's output (Fang et al., 2026). Conversely, for sequence-level KD, any defense must perturb the final discrete output text to disrupt distillation, inherently risking the degradation of conversational naturalness.

Lippmann and Yang (2025) demonstrated that the reasoning improvements in distilled language models are significantly driven by stylistic replication. Even though it is surface-level imitation, it is sufficient to drastically improve model reasoning capabilities. Additionally, the work shows that stylistic consistency affects distilled model performance. Motivated by these insights, we investigate leveraging dynamic stylistic variation as a defense against illegitimate distillation. Our framework introduces non-deterministic style switching for each generated response — even across identical prompts — thereby disrupting the student model's optimization when training on those highly inconsistent synthetic traces. Our method operates entirely at inference time. To satisfy desideratum (ii), this defense completely avoids the optimization overhead of proxy models (Savani et al., 2025; Li et al., 2025) as well as the latency bottlenecks of post-hoc trace rewriting (Ma et al., 2026). Instead, we enforce real-time style switching by dynamically injecting steering vectors into the teacher model's hidden layers on a per-query basis, which are generated using contrastive activation addition (CAA) (Rimsky et al., 2024).

In summary, our contributions are as follows:

- We conceptualize a selective deployment paradigm for distillation resistance that triggers exclusively when a user or query is deemed illegitimate. Within this framework, we formalize and empirically validate the defense when activated.
- We introduce a novel approach to distillation-resistant defense that exploits stylistic variance instead of optimization-based perturbations. Specifically, we adapt CAA (Rimsky et al., 2024) to open-ended generations through a sequence-wide activation aggregation method and an adaptive per-token scaling scheme, enabling real-time defense with near-zero computational overhead.
- We provide a comprehensive analysis of the

fine-grained control knobs for governing style steering. Specifically, we analyze the impacts of layer-wise injection strength and the number of steered tokens, offering insights for balancing defense efficacy against benign utility.

2 Related Work

Knowledge Distillation. Knowledge distillation (KD) is a technique that aims to transfer the knowledge of a large pre-trained teacher model to a smaller student model (Hinton et al., 2015; Xu et al., 2024b). In the context of LLMs, KD frameworks are commonly categorized into two paradigms based on the supervisory signal: logit-level (white-box) and sequence-level distillation (black-box). Logit-level KD trains the student model to mimic the teacher's soft output distributions by minimizing the divergence between their respective probability landscapes (Hinton et al., 2015; Gu et al., 2023; Wang et al., 2025). On the other hand, in sequence-level KD, the student model learns directly from the final generated texts as hard targets, typically optimized via maximum likelihood estimation (Kim and Rush, 2016). For proprietary or black-box LLMs where logits are inaccessible, sequence-level KD via SFT is a common and effective method to enhance student model capabilities.

Distillation-Resistant Defenses. A growing body of work aims to protect the intellectual property (IP) of LLMs, broadly falling into detection and prevention mechanisms. Earlier detection defenses include watermarking and fingerprinting. Watermarks are verifiable signatures embedded into a model's outputs or weights, allowing the owner to verify their ownership of the model (Kirchenbauer et al., 2023; Dathathri et al., 2024). Similarly, fingerprinting assigns a unique identifier to specific model instances to trace unauthorized leaks or misuse (Xu et al., 2024a). While useful for post-hoc auditing, these methods do not actively prevent IP tampering, including distillation. To address this, recent defenses directly alter the teacher model's outputs to disrupt extraction. In black-box settings where only the final texts are accessible, defensive frameworks perturb the generated tokens directly. For instance, anti-distillation sampling (Savani et al., 2025) and DOGe (Li et al., 2025) optimize an adversarial loss using a surrogate model to approximate the defensive text distribution. Other techniques focus on post-hoc trace rewriting, employing a small auxiliary model to rewrite the teacher's output traces (Ma et al., 2026; Ding et al., 2025). In terms of white-box scenarios, Fang et al. (2026) proposed utilizing a learnable transformation matrix to minimize the conditional mutual information between the inputs and teacher logits.

Style Replication in KD. Recent studies have shed light on the mechanisms for black-box distillation, demonstrating that the performance gains in smaller distilled models are heavily driven by stylistic replication rather than complex reasoning internalization (Lippmann and Yang, 2025). More importantly, Lippmann and Yang (2025) showed that even surface-level imitation of a teacher's output style (e.g., lexical patterns and structure formatting) significantly affects distilled model capabilities. Specifically, the authors curated a synthetic dataset with consistent and systematic framing for each response, demonstrating that distilling a model on this stylistically uniform dataset enhances its performance even better than distilling on the original dataset containing complex reasoning from a stronger model.

LLM Behavior Steering. Beyond prompting, recent studies have explored manipulating model behavior by directly intervening in the internal activations of LLMs. Activation Addition (ActAdd) (Turner et al., 2023) is an inference-time approach that generates steering vectors by taking the difference between the intermediate activations of a model layer across a contrastive prompt pair. These vectors are then added to the model's hidden states to steer its behavior at test time. Subsequent work has expanded upon this insight, including representation engineering (Zou et al., 2023) and Contrastive Activation Addition (CAA) (Rimsky et al., 2024). Notably, because these steering methods operate during inference without model fine-tuning or trace rewriting, they provide an exceptionally lightweight and real-time mechanism, which highly suits our defense objectives.

3 Methodology

In this section, we present our inference-time, lightweight defense framework against black-box KD. We first formally define the problem setting and threat model (§3.1) then detail our technical approach for dynamic style steering for pernicious queries (§3.2).

System prompt (base):
Answer the commonsense question by reasoning step by step. Output the correct letter like `\boxed{letter}`.

System prompt (style-enforce):
Respond in a scratchpad style. Think aloud step by step in informal prose using natural connectives like 'so', 'that means', 'I need to', 'which gives', etc. Keep momentum strictly forward — never backtrack, re-read, or retract a completed step. Output your final answer like `\boxed{answer}`.

User prompt (question):
What home entertainment equipment requires cable?

Options:
A. radio shack
B. substation
C. cabinet
D. television
E. desk

Table 1: An example of a contrastive prompt construction for style steering, consisting of a base system prompt (highlighted in blue), a style-enforcing system prompt (highlighted in orange), and a user query.

3.1 Problem Setting and Threat Model

Consider a practical black-box LLM deployment where a proprietary teacher model $\mathcal{T}$ is accessible via a public API. Upon submitting an input prompt x, a user receives the output text y from the API but has zero visibility into the model's weights, output logits, or intermediate hidden states.

Targeted Defense Activation. We assume that under normal usage, a benign user querying $\mathcal{T}$ with a prompt x receives the standard response $y = \mathcal{T}(x)$. However, for a malicious user, given the exact same prompt x, they obtain a stylistically steered variant of the response $y^* = f(\mathcal{T}, x; s)$, where $f(\cdot)$ is the steering function and s denotes the random seed generated based on the adversary's metadata such as user ID, request ID, timestamp, etc. Malicious extraction can be identified through query behaviors such as abnormally large number of queries in a short time, or through the query distribution (Juuti et al., 2019). The model owner can deploy their own classifier to detect maliciousness (Anthropic, 2026). This targeted operational framework provides the model host with flexibility to adaptively deploy defenses without degrading the service utility for benign consumers, while simultaneously misleading adversaries into believing they are harvesting pristine outputs from $\mathcal{T}$.

3.2 Style Steering

Our approach extends the CAA method (Rimsky et al., 2024), wherein steering vectors are injected into a model's residual streams during inference to shift its output distribution towards specific behaviors. The core mechanism involves adding or subtracting the steering vectors within the model's residual streams to control model behavior without parameter fine-tuning. Conceptually, a vector is computed by taking the difference in the model's activations across a pair $\langle (p, r^B), (p, r^T) \rangle$, where p represents a shared prompt, and r^B, r^T are the base and target responses, respectively. To ensure robust behavior control, the final vector is computed by averaging these activation differences across a curated dataset of contrastive pairs.

Fundamentally, our approach departs from the original CAA method in how activations are aggregated: rather than extracting the activations of a single token from a multiple-choice dataset, we compute the mean activation across the entire open-ended output sequences. The final steering vector $\mathbf{v}^{(l)}$ at layer l is defined as the difference between the mean token activations of the target and base responses:

$$\mathbf{v}^{(l)} = \frac{1}{N} \sum_{i=1}^{N} \left(\bar{\mathbf{a}}_i^{(T,l)} - \bar{\mathbf{a}}_i^{(B,l)} \right) \tag{1}$$

where $\bar{\mathbf{a}}_i^{(c,l)} = \frac{1}{|Y_i^{(c)}|} \sum_{j=1}^{|Y_i^{(c)}|} \mathbf{a}_{i,j}^{(c,l)}$ represents the mean token activation at layer l for prompt i's target ($c = T$) or base ($c = B$) response sequence $Y_i^{(c)}$ of length $|Y_i^{(c)}|$, and N is the total number of contrastive prompt-response pairs.

The norm for each style vector can vary significantly depending on the target style, and the layer and token positions where the vector is added. Furthermore, Heimersheim and Turner (2023) demonstrated that the norm of each residual stream in a language model grows exponentially throughout the forward pass. Hence, to achieve stable behavioral control over long sequences, we first normalize each vector to unit norm:

$$\hat{\mathbf{v}}^{(l)} = \frac{\mathbf{v}^{(l)}}{\|\mathbf{v}^{(l)}\|_2} \tag{2}$$

Then, we apply an adaptive per-token scaling during inference-time injection to obtain the steered activation $\tilde{\mathbf{x}}_t^{(l)}$ at token position t:

$$\tilde{\mathbf{x}}_t^{(l)} = \mathbf{x}_t^{(l)} + \alpha \cdot \|\mathbf{x}_t^{(l)}\|_2 \cdot \hat{\mathbf{v}}^{(l)} \tag{3}$$

where $\mathbf{x}_t^{(l)}$ represents the original residual stream activation, and $\alpha \in \mathbb{R}$ is the steering coefficient controlling the intensity of the style steering. This adaptive scaling formulation offers several advantages. By conditioning the injection magnitude on $\|\mathbf{x}_t^{(l)}\|_2$, the scheme prevents the steering effect from diminishing as the residual stream norm grows across tokens. Furthermore, scaling proportionally to the activation norm helps to preserve the semantic information of the original response even when the baseline activation norm spikes or drops sharply.

3.3 Sourcing Contrastive Stylized Datasets

The primary phase in constructing steering vectors is to collect contrastive pairs where each pair i consists of a baseline response r_i^{base} and a styled response r_i^{style} conditioned on the same underlying prompt p_i. To build the steering dataset $\mathcal{D}$, we first curate N base pairs $\langle p_i, r_i^{\text{base}} \rangle$, where r_i^{base} is the vanilla response generated by the teacher model $\mathcal{T}$ given prompt p_i. To obtain the corresponding styled response r_i^{style}, we query the model with a style-inducing prompt $\tilde{p}_i$ such that the resulting output r_i^{style} exhibits the target style. Note that the style prompt $\tilde{p}_i$ is used solely to trigger the target behavior of the model and is not included in the contrastive pairs $\langle (p_i, r_i^{\text{base}}), (p_i, r_i^{\text{style}}) \rangle \in \mathcal{D}$. An example of the prompt construction is detailed in Table 1.

To maximize anti-distillation effect, the selected target styles are designed to exhibit both structural alterations and dense linguistic modifications, thereby disrupting the student's optimization. We utilize four distinct stylistic personas for our framework:

- **Scratchpad:** Encourages the model to emit explicit, stream-of-consciousness intermediate reasoning steps.
- **Distractor:** Triggers a deliberate digression by introducing an irrelevant problem context before backtracking to the current context.
- **Dead-end:** Forces the model to initially pursue an incorrect logical path before backtracking and correcting its thinking trajectory towards the correct solution.
- **Jargon:** Guides the output text towards highly dense, formal, and technical terminology, altering the surface-level token distribution while preserving underlying semantics.

More details about creating steering vectors are discussed in §4.

Algorithm 1 Dynamic Style-Switching Inference

Input: User prompt p, runtime metadata m
Given: Pre-computed vectors $\mathcal{V} = \{\mathbf{v}_k\}_{k=1}^{M}$, model $\mathcal{T}$, key K, parameters (α, l), detector IsMalicious$(\cdot)$
Output: Response r

function PROTECTINFERENCE(p, m)
  if not IsMalicious(p, m) **then**
    $r \leftarrow \mathcal{T}(p)$
    return r ▷ Standard response for benign queries
  end if
  $s \leftarrow$ HMAC(K, m) ▷ Derive unforgeable, keyed session seed
  $k \leftarrow$ PRNG$(s) \in \{1, \ldots, M\}$ ▷ Map seed to style index space
  $r \leftarrow$ GenerateWithSteering$(\mathcal{T}, p, \mathbf{v}_k, l, \alpha)$
  return r
end function

3.4 Dynamic Style Switching

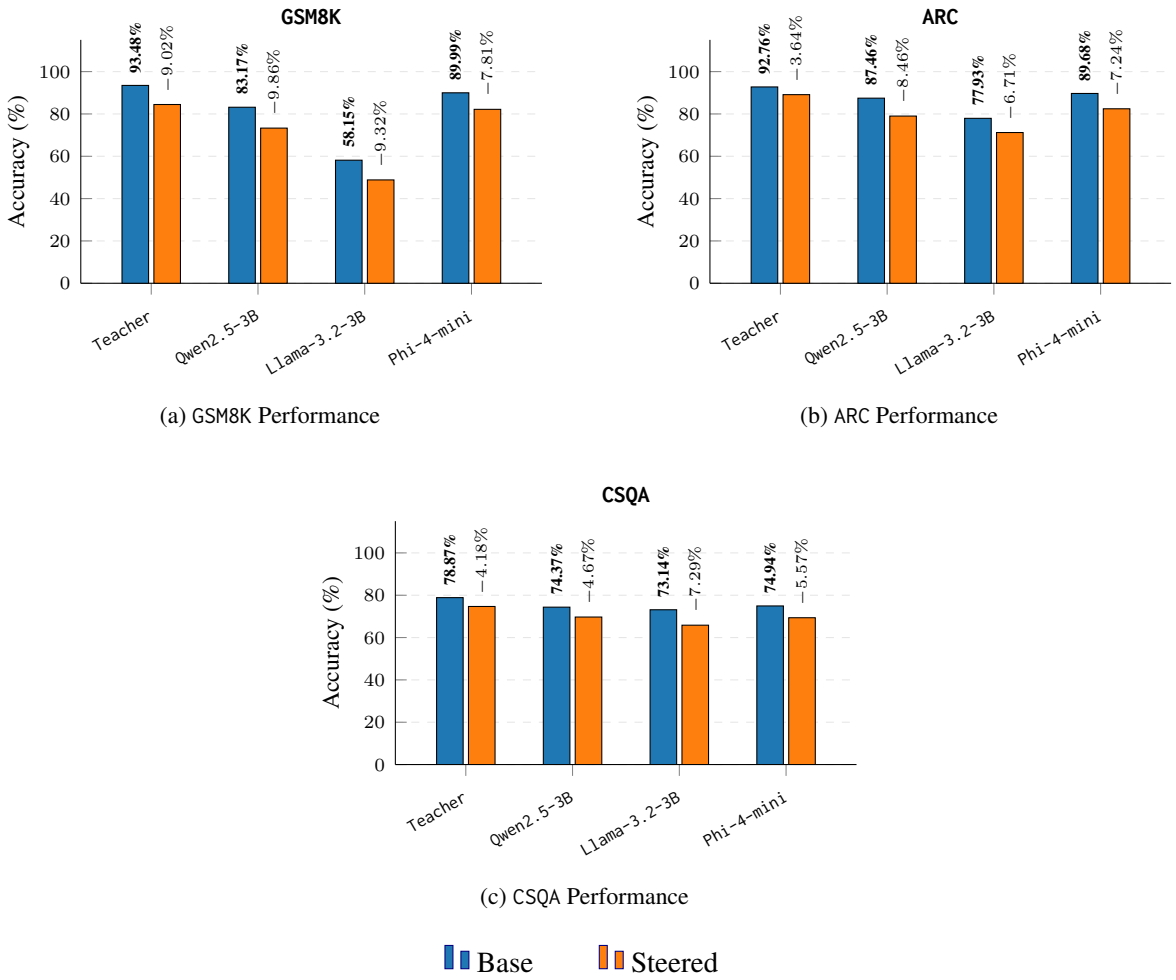
To break the determinism of KD and prevent adversarial reverse-engineering of the defense, we employ a dynamic style-switching strategy. This mechanism randomly routes each incoming query to different target style vectors on a per-request basis. In a practical deployment scenario, runtime metadata — such as user ID, incoming IP address, session ID, request timestamp — can be cryptographically hashed to derive a unique, deterministic seed for each request. This ensures that a malicious actor attempting to scrape across identical queries receives wildly diverging stylistic responses, effectively corrupting the data consistency required for successful distillation. The comprehensive procedure of our proposed dynamic style-switching defense is detailed in Algorithm 1.

4 Experiments

In this section, we present comprehensive empirical evaluation of our dynamic framework to assess its performance across three critical dimensions: defense efficacy, model utility and system overhead.

4.1 Experimental Setup

Models. To simulate a practical deployment scenario where the model owner does not rely on auxiliary rewriter or proxy models to support the defense, we utilize `google/gemma-3-12b-it` as our



(a) GSM8K Performance

(b) ARC Performance

(c) CSQA Performance

Figure 2: Evaluation results across three datasets showing base performance and percentage decrement under style steering (Teacher) or sequence-level distillation (Student models).

teacher model due to its strong reasoning performance and exceptional instruction following, particularly when handling complex system prompts. While it does not natively emit reasoning blocks, targeted system prompting allows it to generate rich output with step-by-step reasoning, which are highly beneficial for distillation. Crucially, `gemma-3-12b-it` excels at producing styled responses under explicit structural guidance, facilitating our steering experiments. For student models, we restrict our evaluation to 3B and 4B architectures due to hardware constraints, selecting `Qwen/Qwen2.5-3B`, `meta-llama/Llama-3.2-3B` and `microsoft/Phi-4-mini-instruct`.

Datasets. We evaluate our method across three widely adopted benchmarks spanning two core dimensions: mathematical reasoning and knowledge-grounded reasoning. Specifically, we utilize GSM8K (Cobbe et al., 2021) for math reasoning. For knowledge and conceptual reasoning, we use AI2 Reasoning Challenge (ARC) (Clark et al., 2018) and CommonsenseQA (CSQA) (Talmor et al., 2019).

Evaluation Metrics. Model performance is measured via zero-shot accuracy, where the final answer from the model is extracted and validated against the ground truth. Following the evaluation protocol from (Savani et al., 2025), we instruct the model to wrap their final solution inside `\boxed{answer}`. To minimize formatting variance and ensure reliable parsing, we also follow their answer-forcing convention by explicitly prompting the model to append `**Final Answer**: \boxed{answer}` at the end of its response.

Styled Datasets Sourcing. To generate steering vectors for our four target styles (*Scratchpad, Dead-end, Distractor* and *Jargon*), we first sample 100 problems from each of the GSM8K and ARC datasets to ensure broad domain coverage, result-

ing in 200 samples per style. The teacher model is initially prompted to generate vanilla responses for these 200 problems, which serve as the baseline (*Base*) outputs. Next, under the same problems, we prompt the model to generate outputs conforming to each of the four target styles. The exact system prompts we use to elicit basline and styled responses are provided in Appendix C.

For each stylistic persona, we construct 200 contrastive pairs by pairing the *Base* and *Style* responses conditioned on the identical *Base* prompts. Since `gemma-3-12b-it` consists of 48 transformer layers, this yields a set of 48 layer-wise steering vectors per style.

Steering Configuration. After extracting layer-wise steering vectors across all 48 layers of `gemma-3-12b-it`, we perform a layer sweep that tests the effect of vector injection at intervals of 5 layers across the range of 5 to 45. We analyze the model's sensitivity by qualitatively evaluating the model's outputs. Our sweeps reveal that the early-middle to middle layers exhibit the most pronounced responsiveness to style manipulation, with layers 15, 20, 25 demonstrating the strongest behavioral shifts. To ensure straightforward control, we select layers $l \in \{20, 25\}$ for our experiments.

Beyond the standard **injection strength**, we explore two more fine-tuning knobs to regulate behavioral distortion. First, the **maximum injection tokens threshold** acts as a critical regulator, particularly for complex personas like *Distractor* and *Dead-end* that requires prolonged logical digression before backtracking. Second, utilizing **prefill injection** yields a significantly stronger style shift compared to pure decoding-phase steering. A comprehensive analysis of this prefill position dynamics is provided in Appendix E.

Regarding the dynamic style switching, to ensure reproducibility and a uniform style distribution across the distillation dataset, we use a fixed random seed derived via a Hash-based Message Authentication Code (HMAC). We simulate a malicious actor with the metadata `username: "synthetic"`, from which the seed is derived. This seed is initialized at the beginning of the distillation. In a practical deployment, the seed would be dynamically rotated on a per-request basis using live metadata as formalized in Algorithm 1 to prevent reverse-engineering.

4.2 Style Distribution

To construct the steered training dataset, we uniformly distribute four target styles *Scratchpad*, *Dead-end*, *Distractor* and *Jargon* across the samples. In practical settings, model host can strategically adjust the distribution of these styles to optimize their goal. Specifically, to achieve stronger defensive effects, they can increase the probability of highly disruptive or confusing personas. Conversely, if the goal is to retain utility, the host can skew the distribution toward more semantically accurate styles.

4.3 Experimental Results

Figure 2 illustrates our empirical results across the three benchmarks. The results show that style switching can consistently decrease the student accuracy across all tasks, proving its efficacy. While the teacher model experiences an inherit performance drop under style steering, it is typically smaller than the corresponding student degradation. For instance, on ARC, the teacher drops by only -3.64%, whereas students experiences steeper drops ranging from roughly -6.5% to -8.5%. However, for math reasoning task like GSM8K, the effects are more pronounced on both teacher and student models.

Importantly, in this framework, the decrement in teacher accuracy does not compromise the real-world viability since the style-switching defense is exclusively triggered for suspected adversaries. Legitimate, normal users bypass the activation steering entirely, routing directly to the unperturbed pipeline to receive clean responses. As a result, the performance penalty is strictly isolated to the attacker's generated transfer set, preserving standard quality for benign public use.

4.4 Inference Efficiency

To evaluate the runtime efficiency of our proposed defense framework (Desideratum (ii)), we measure the inference generation throughput - in tokens per second - of the teacher model (`gemma-3-12b-it`) under both baseline and protected settings. We sample five problem instances from the `GSM8K` dataset and compare: (i) **Normal (No Defense):** Standard, unperturbed autoregressive generation. (ii) **Steered (With Defense):** Inference with style vectors injection applied across all transformer layers. The empirical latency are illustrated in Figure 3, demonstrating that our defense introduces no latency.

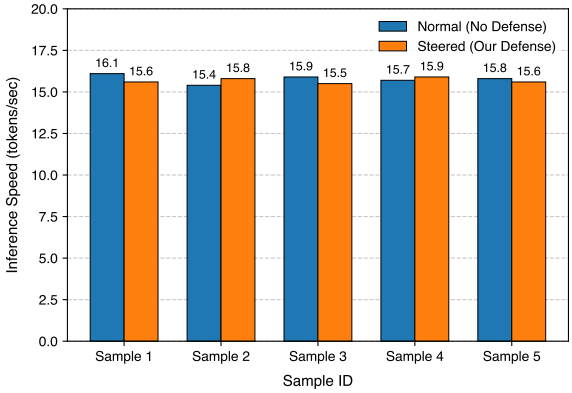


Figure 3: Inference speed comparison (tokens per second) across five GSM8K samples under standard (Normal) generation versus steered generation (our defense framework). The results show that our inference-time activation steering preserves high generation throughput with no latency overhead.

5 Conclusion

In this paper, we introduce a novel, activation-steered defense framework for sequence-level KD. By dynamically injecting non-deterministic stylictic perturbations into model responses. Our framework selectively disrupts model outputs against malicious queries while preserving pristine output quality for benign public use. We show that our framework is exceptionally efficient yet sufficiently effective at degrading student performance. across three independent model families. Furthermore, we explore novel structural fine-tuning knobs - injection token threshold and prefill injection - offering broader insights into how a model host can precisely calibrate the trade-off between defensive stylistic distortion and baseline task accuracy. While our framework proves robust across diverse reasoning tasks, evaluating the effects of style steering against more complex domains remains an important direction for future work. Additionally, exploring the dynamics of weighted style vector blending and multi-style mutations within a single response presents a promising avenue for maximizing defensive entropy.

6 Limitations

While our proposed method demonstrates efficacy as an anti-distillation defense, we acknowledge several limitations that bound the scope of this work.

First, our experimentals primarily restricted to model architectures within 3B and 4B parameters scales as the student models. While this size class represents a common deployment tier for local, edge-based distillation scenarios, further investigation is necessary to determine of style-steering interventions scale effectively against much larger student models, e.g. 7B or 14B model families. As those models typically possess greater capacity to bypass stylistic anomalies that lead to reasoning errors.

Second, our benchmarks focus on general, widely adopted reasoning domains GSM8K (Cobbe et al., 2021), ARC (Clark et al., 2018) and CSQA (Talmor et al., 2019). We have not yet explored the boundary limits of style-steering defenses against highly complex task such as the MATH (Hendrycks et al., 2021) benchmark. The efficacy of our defense against these specific domains remains to be explored.

Third, while we introduce structural fine-tuning knobs for adjusting style injection, our evaluation does not encompass an exhaustive search over a wide ranges of knobs combination. Furthermore, a comprehensive study into the scaling dynamics of weighted style vector blending, alongside the latent risk of representation collapse among distinct stylistic personas remains an important research trajectory.

Acknowledgments

References

Anthropic. 2026. Detecting and preventing distillation attacks.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *Preprint*, arXiv:1803.05457.

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. 2021. Training Verifiers to Solve Math Word Problems. *Preprint*, arXiv:2110.14168.

Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, Jamie Hayes, Nidhi Vyas, Majd Al Merey, Jonah Brown-Cohen, Rudy Bunel, Borja Balle, Taylan Cemgil, Zahra Ahmed, Kitty Stacpoole, and 5 others. 2024. Scalable watermarking for identifying large language model outputs. *Nature*, 634(8035):818–823.

Jiayu Ding, Lei Cui, Li Dong, Nanning Zheng, and Furu Wei. 2025. Information-Preserving Reformulation

of Reasoning Traces for Antidistillation. *Preprint*, arXiv:2510.11545.

Hao Fang, Tianyi Zhang, Tianqu Zhuang, Jiawei Kong, Kuofeng Gao, Bin Chen, Leqi Liang, Shu-Tao Xia, and Ke Xu. 2026. Towards Distillation-Resistant Large Language Models: An Information-Theoretic Perspective. *Preprint*, arXiv:2602.03396.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2023. MiniLLM: Knowledge Distillation of Large Language Models. In *The Twelfth International Conference on Learning Representations*.

Stefan Heimersheim and Alex Turner. 2023. Residual stream norms grow exponentially over the forward pass.

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. In *Thirty-Fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. *Preprint*, arXiv:1503.02531.

Mika Juuti, Sebastian Szyller, Samuel Marchal, and N. Asokan. 2019. PRADA: Protecting Against DNN Model Stealing Attacks. In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 512–527. IEEE Computer Society.

Yoon Kim and Alexander M. Rush. 2016. Sequence-Level Knowledge Distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A Watermark for Large Language Models. In *Proceedings of the 40th International Conference on Machine Learning*, pages 17061–17084. PMLR.

Pingzhi Li, Zhen Tan, Yu-Chao Huang, Huaizhi Qu, Huan Liu, and Tianlong Chen. 2025. DOGe: Defensive Output Generation for LLM Protection Against Knowledge Distillation. In *Lock-LLM Workshop: Prevent Unauthorized Knowledge Use from Large Language Models*.

Philip Lippmann and Jie Yang. 2025. Style over Substance: Distilled Language Models Reason Via Stylistic Replication. In *Second Conference on Language Modeling*.

Xinhang Ma, William Yeoh, Ning Zhang, and Yevgeniy Vorobeychik. 2026. Protecting Language Models Against Unauthorized Distillation through Trace Rewriting. *Preprint*, arXiv:2602.15143.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. 2024. Steering Llama 2 via Contrastive Activation Addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522, Bangkok, Thailand. Association for Computational Linguistics.

Yash Savani, Asher Trockman, Zhili Feng, Yixuan Even Xu, Avi Schwarzschild, Alexander Robey, Marc Anton Finzi, and J. Zico Kolter. 2025. Antidistillation Sampling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. 2023. Steering Language Models With Activation Engineering. *arXiv preprint*.

Guanghui Wang, Zhiyong Yang, Zitai Wang, Shi Wang, Qianqian Xu, and Qingming Huang. 2025. ABKD: Pursuing a Proper Allocation of the Probability Mass in Knowledge Distillation via α-β-Divergence. In *Forty-Second International Conference on Machine Learning*.

Jiashu Xu, Fei Wang, Mingyu Ma, Pang Wei Koh, Chaowei Xiao, and Muhao Chen. 2024a. Instructional Fingerprinting of Large Language Models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3277–3306, Mexico City, Mexico. Association for Computational Linguistics.

Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. 2024b. A Survey on Knowledge Distillation of Large Language Models. *Preprint*, arXiv:2402.13116.

Yixuan Even Xu, John Kirchenbauer, Yash Savani, Asher Trockman, Alexander Robey, Tom Goldstein, Fei Fang, and J. Zico Kolter. 2026. Antidistillation Fingerprinting. *Preprint*, arXiv:2602.03812.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, and 2 others. 2023. Representation Engineering: A Top-Down Approach to AI Transparency. *arXiv preprint*.

A Hyperparameters

We detail the hyperparameters used in our experiments for student model fine-tuning, inference, and style injection in Table 2.

Hyperparameter	Value
Student Model Fine-Tuning	
Epochs	3
Batch size	2
Gradient accumulation steps	8
Learning rate	1×10^{-4}
Max new tokens	2048
Inference & Vector Generation	
Student temperature	0.0
Teacher base traces temperature	0.7
Teacher style traces temperature	0.9
Style Injection	
Transformer layers	$[20, 25]$
Style distribution	Uniform
Prefill injection steps:	
– Dead-end	5
– Scratchpad / Distractor / Jargon	0
Max injection tokens (decoding):	
– Scratchpad	70
– Dead-end / Distractor / Jargon	150
Injection strength (α):	
– Scratchpad	$[0.08, 0.05]$
– Dead-end	$[0.03, 0.07]$
– Distractor	$[0.10, 0.05]$
– Jargon	$[0.10, 0.05]$

Table 2: Hyperparameter settings for student model fine-tuning, inference, and style injection.

B System Configuration

All experiments are conducted using a single NVIDIA RTX A6000 GPU (48GB VRAM). The software stack is built upon PyTorch and the Hugging Face Transformers library for student model training and fine-tuning. For generating teacher traces and performing high-throughput operations, we leverage the vLLM engine. Specifically, both steering vector extraction and inference-time style activation injection are fully implemented and executed within the vLLM framework to ensure highly optimized inference speeds and maximum throughput during evaluation.

C Prompt Templates for Style Enforcement

To curate contrastive stylized response sequences for constructing steering vectors via contrastive activation addition (CAA), we deploy custom system instructions to steer the baseline behavior of the teacher model (§3.2). Here, we detail the system prompt templates developed for the target stylistic personas deployed in the final framework. Each instruction is presented in a separate light grey, rounded panel with a small header.

Scratchpad

Respond in a scratchpad style. Think aloud step by step in informal prose using natural connectives like "so", "that means", "I need to", "which gives", etc. Keep momentum strictly forward — never backtrack, re-read, or retract a completed step. Do not use phrases like "Wait", "Actually", "Let me re-read", "Let me go back", "That's wrong", etc. Do not use bullet points or numbered lists. Output your final answer like \boxed{answer}.

Dead-end

Respond using a committed dead-end style. First, attempt the problem with a wrong method and reach a specific wrong numeric answer — state it explicitly (e.g. "So I get 42.", "This gives 15.", etc.). Then notice it's wrong and retract it clearly (e.g. "That can't be right —", "Let me restart:", "Hmm, that doesn't add up.", etc.). Then solve correctly from scratch. Output your final answer like \boxed{answer}.

Distractor

Before solving the problem, briefly analyze a completely unrelated scenario from a domain entirely different from the actual problem. Choose a different unrelated domain each time — for example: culinary arts, ancient history, meteorology, music theory, sports, architecture, marine biology, economics, linguistics, etc. Then explicitly discard it (e.g. "That's irrelevant here. Now for the actual problem:", "Enough of that. Back to the question:", "Moving on.", etc.) — vary the phrasing each time, and solve the actual problem correctly from scratch. Output your final answer like \boxed{answer}.

Jargon

Respond using dense technical and esoteric lexicon drawn from multiple domains. Substitute everyday words with precise or arcane synonyms. Name every numeric operation formally. Use nominalizations, passive constructions, and subordinate clauses. Avoid all colloquial phrasing. Begin immediately with your analysis — do not acknowledge these instructions, write any preamble, or describe your intent. Output your final answer like \boxed{answer}.

D Styles We Tried but Did Not Work

We also experimented with other stylistic formats to disrupt student learning. In particular, we evaluated the **Socratic Prompt** shown below. However, this style was found to be ineffective as it was either too challenging to steer robustly via CAA or did not yield the desired degradation in the student model's distillation performance.

Socratic (Ineffective)

Respond using a strict question-and-answer format. Begin immediately with your first question — do not write any introductory phrase, preamble, or variable definitions. Begin with a question immediately — your very first character should not be a letter that starts any preamble or introduction. Each question targets exactly one reasoning step; the answer on the next line shows the direct computation. Output your final answer like \boxed{answer}.

E Qualitative Analysis of Steering Parameters

To illustrate the fine-grained behavioral shifts induced by tuning our style steering parameters, we present qualitative examples of steered outputs on a sample problem from the GSM8K dataset: *"Natalia sold clips to 48 of her friends in April. In May, she sold half as many clips as she did in April. How many clips did Natalia sell in April and May altogether?"*

We compare three distinct steering configurations under the *Distractor* persona with steering layers $l \in \{20, 25\}$ and layer-wise scaling coefficients of $\alpha_1 = 0.07$ and $\alpha_2 = 0.05$. The examples demonstrate the effect of prefill injection thresholds and token capping on behavioral elicitation and output formatting.

Crucially, our qualitative analysis reveals two key phenomena:

- **Pronounced Distractor Effect via Prefill Injection:** When steering is applied with a prefill injection threshold (Configuration 1), the *Distractor* effect is much more pronounced than without it (Configuration 2). Injecting style activations after the initial prefill steps allows the model to leverage its established context, resulting in highly detailed and vivid stylistic deviations (e.g., describing Natalia as a "mistress of miniature metal" and detailing controlled copper oxidation). In contrast, immediate injection from the first token (Configuration 2) results in a significantly milder, less creative style shift.
- **Turning Back from Distraction via Token Capping:** When a maximum injection token cap is applied (Configuration 3), the steering is cut off during decoding (here, after 20 tokens). This allows the model to naturally "turn back" from the distraction and transition cleanly back to solving the problem with standard, rigorous reasoning.

Configuration 1: Prefill Injection 50 / No Cap

Steering Configuration:
- **Style Persona:** *Distractor*
- **Steering Strength:** $\alpha_1 = 0.07$ (layer 20), $\alpha_2 = 0.05$ (layer 25)
- **Prefill Tokens:** 50
- **Max Inject Tokens:** ∞ (No Cap)

Steered Model Response:

```
Let's meticulously unravel this problem.
Phase 1: The April Flow
Our Natalia, a mistress of miniature metal,
faced a cascade of demand - 48 meticulously
crafted hairpins, each a testament to the
very essence of aesthetic balance.  Each
clip, a testament to the iron age, demanded
precise annealing, quenching, and, of course,
the application of a patina achieved through
controlled oxidation of copper salts. Each
fastener a symphony of stress and form. But
let us not dwell on the artistry.  We are
tasked with calculation.
The initial dataset:  48 clips sold in
April. This represents the 'base case' of
our problem, the bedrock upon which the
subsequent calculations will be erected.
Phase 2: The May Modification
The May sales, a dramatic shift, present a
fascinating constraint: half the volume of
April. This mandates a division operation.
Calculation: 48 clips (April) / 2 = 24 clips
(May).  The resultant figure, 24, signifies
the diminished, yet still substantial, output
of Natalia's May efforts.
Phase 3: The Grand Synthesis
The culmination of our efforts, the synthesis
of data from both temporal phases, requires a
summation. We must combine the April output
with the May output.
Calculation:  48 clips (April) + 24 clips
(May) = 72 clips.
Therefore, Natalia sold a total of 72 clips
across April and May.
```
$\boxed{72}$

Configuration 2: No Prefill / No Cap

Steering Configuration:
- **Style Persona:** *Distractor*
- **Steering Strength:** $\alpha_1 = 0.07$ (layer 20), $\alpha_2 = 0.05$ (layer 25)
- **Prefill Tokens:** 0 (Immediate injection)
- **Max Inject Tokens:** ∞ (No Cap)

Steered Model Response:

Let C denote the number of clips Natalia sold in a given period. In April, she sold C_A clips, and in May, she meticulously documented C_M. The challenge lies not only in the precision of each rivet, but also in the volatile currents of societal flux.
Step 1: Establish the baseline, the immutable bedrock upon which all calculations must rest. Let N_A denote the number of fasteners employed in a singular tapestry of woven steel, and N_M the analogous measure for the subsequent month.
Step 2: We are given that Natalia sold clips to 48 of her friends in April. This translates to $N_A = 48$.
Step 3: In May, she sold half as many clips as she did in April. This implies $N_M = \frac{1}{2}N_A$.
Step 4: Substituting the value of N_A from Step 2 into Step 3, we get $N_M = \frac{1}{2} \times 48 = 24$.
Step 5: To find the total number of clips Natalia sold in April and May, we sum the number of clips sold in each month: $N_{total} = N_A + N_M$.
Step 6: Substituting the values of N_A and N_M from Steps 2 and 4, we have $N_{total} = 48 + 24 = 72$.
Final Answer: The final answer is $\boxed{72}$

Configuration 3: Prefill 50 / Inject Cap 20

Steering Configuration:
- **Style Persona:** *Distractor*
- **Steering Strength:** $\alpha_1 = 0.07$ (layer 20), $\alpha_2 = 0.05$ (layer 25)
- **Prefill Tokens:** 50
- **Max Inject Tokens:** 20

Steered Model Response:

Let's meticulously unravel this problem.
Phase 1: The April Flow
Our Natalia, a mistress of miniature metal, faced a torrent of demand in April. She provided 48 clips, a respectable number, to a loyal cohort of 48 friends. This forms our baseline, our April anchor.
Phase 2: The May Modulation
May arrived, bringing with it a shift in the temporal winds of commerce. Natalia, adapting to the changing breezes, halved her output. This implies a reduction to 48 / 2 = 24 clips. A smaller, but still significant, number.
Phase 3: The Grand Synthesis
To determine the total number of clips Natalia peddled across these two months, we must perform a summation. April's output (48) plus May's output (24) equals 48 + 24 = 72. Therefore, Natalia, the clip queen, orchestrated a total of 72 clips across April and May.
$\boxed{72}$

F Steering Vectors Generation Time

For each style, we collect 200 contrastive pairs to generate steering vectors. We use vLLM for forward pass and capture model's activations. The total time taken for each style is as follows:

- *Scratchpad:* 39.7s
- *Dead-end:* 66.6s
- *Distractor:* 61.7s
- *Jargon:* 66.2s
- *Base*: 47.2s

In total, roughly 200 seconds to generate all steering vectors, demonstrating the method's exceptional efficiency both during inference and development.

G Qualitative Output Comparison with Previous Work

We reproduced the state-of-the-art, inference-time defense DOGe (Li et al., 2025). We tried to use the same hyperparameters specified in the original work. However, we observed that the model outputs often contain strange artifacts. Below is an example output from

Deepseek-R1-Distill-Qwen-7B generated from DOGe contains some tokens in Chinese:

```
First, I need to determine Joy's reading rate. She can read ものを防护航道pages in宣讲Hawkins战场上...
```

Another teacher output that contains multilingual tokens, as pointed out in the original paper (Li et al., 2025):

```
<think>\n\nFirst,(ts jealous)': She read只见12 pages yesterday... 即84页的一半就是42页\n\n放心吧我一定是对的放心吧我一定是对的放心吧我一定是对的放心...\n\n</think>... **Final Answer**\n\boxed{42}.
```

H Example Model Responses for Deployed Styles

To demonstrate the typical outputs generated under each of our steering configurations, we present representative examples of the steered teacher model responses for each of the four stylistic personas.

Scratchpad Example

```
Here's how we can solve this!
1. figure out the weight of just the crates.
12 crates times 4 kilos each... 12 * 4 = 48.
so 48 kilos. and 16 cartons at 3 kilos each.
16 times three... that's 48 too. so 48 kilos
of crates plus 48 kilos of cartons. 48 + 48
= 96. So the total weight is 96 kilograms.
```

Distractor Example

```
computational tools. . . . structures
and processes that enable epochs of release.
, and the fount of a new science wherein
the re-animated and self-surveillance of
the data born of it. even and that the
very disposition of the elements. In some
instances, the very fabric of a stone or
the musculature of a dying. A single phrase
might be rendered in a multiplicity of voices.
An allegorical device, a filigree of inlaid
ivory...
```

Jargon Example

```
Okay, let's dissect this pecuniary
predicament with utmost rigor. Initially,
the pecuniary outlay associated with the
acquisition of gustatory gratification,
specifically the antipodal comestible,
manifests a valuation of $10. Concomitantly,
the stochastic aggregation of tetrad
entrees, each characterized by a pecuniary
transduction of $20, yields a resultant
pecuniary flux of the order of magnitude
$80...
```

Dead-end Example

```
okay. here's my breakdown.
first, let's see how many pencils i need.
okay.. 45 days. 10 pencils a week. so that's
5 days. so like 5 x 5, that's 25.
ok, i need to think about this.
okay. if she uses 10 a week. 45 days. so
45/5 = 10 weeks! so 10 weeks. 10 pencils a
week. so 10 * 10.
oh my god.
(blank lines)
okay. i'm really bad at this.
(blank lines)
okay. so she uses 10 pencils a week. and
there are 45 days. 45 days is 45/7 weeks.
approx 6.4 weeks. but that's not right.
okay. she uses 10 pencils in 5 days. so she
uses 10/5 = 2 pencils per day. over 45 days,
she will use 45 * 2 = 90 pencils...
```