AVSD: Adaptive-View Self-Distillation by Balancing Consensus and Teacher-Specific Privileged Signals

Duy Nguyen[1] **Hanqi Xiao**[1] **Archiki Prasad**[1] **Zaid Khan**[1] **Anirban Das**[2]
Shi-Xiong Zhang[2] **Sambit Sahu**[2] **Hyunji Lee**[1] **Elias Stengel-Eskin**[3] **Mohit Bansal**[1]

[1]UNC Chapel Hill [2]Capital One [3]The University of Texas at Austin

Abstract

Self-distillation enables language models to learn on-policy from their own trajectories by using the same model as both student and teacher, with the teacher being conditioned on privileged information unavailable to the student. Such information can come in different types or views, such as solutions, demonstrations, feedback, or final answers. This setup provides dense token-level feedback without relying on a separate external model, but creates a fundamental *asymmetry*: the teacher may rely on view-specific information that the student cannot access at inference time. Moreover, the best type of privileged information is often task-dependent, making it difficult to choose a single teacher view. In this work, we address both these challenges jointly by introducing AVSD (**A**daptive-**V**iew **S**elf-**D**istillation), a novel method of self-distillation with multiple privileged-information views, which reconstructs token-level supervision by separating stable cross-view consensus from view-specific residual signals. AVSD identifies the consensus signal shared across views, which provides a reliable update direction, and then selectively adds the view-specific residual signal to adjust the update magnitude when it both aligns with the consensus direction and remains proportionate to the consensus signal. Experiments on math competition benchmarks (AIME24, AIME25, and HMMT25) show that AVSD consistently outperforms both single-view self-distillation baselines and GRPO, achieving average Avg@8 gains of 3.1% and 2.2% over the strongest baselines on Qwen3-8B and Qwen3-4B, respectively. Moreover, on code-generation benchmarks (Codeforces, LiveCodeBench v6) using Qwen3-8B, AVSD outperforms the single-view self-distillation baseline by 2.4% on average.[1]

1 Introduction

Training large language models (LLMs) to solve difficult reasoning tasks, such as competition-level math problems, requires converting available training supervision into useful learning signals. Reinforcement learning with verifiable rewards (RLVR) [Shao et al., 2024, DeepSeek-AI, 2025] has become the dominant paradigm for LLM post-training in verifiable tasks such as math and code, but its supervision is typically sparse and outcome level, leading to expensive sampling and limited learning signal when the model fails to produce a successful final solution [Tao et al., 2025]. Distillation offers a complementary source of dense token-level guidance [Hinton et al., 2015], but standard offline distillation suffers from train-test mismatch because the student is trained on trajectories outside its own distribution [Xu et al., 2024, Agarwal et al., 2024]. On-policy distillation addresses this off-policy issue by training on student-generated rollouts while using teacher probabilities to provide local learning signals [Agarwal et al., 2024, Gu et al., 2024]. However, these methods still rely on access to a more capable teacher model, motivating self-distillation methods, which remove the need for an external teacher by employing the student model as its own teacher, supplying a learning signal

[1]Code: https://github.com/duykhuongnguyen/AVSD.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

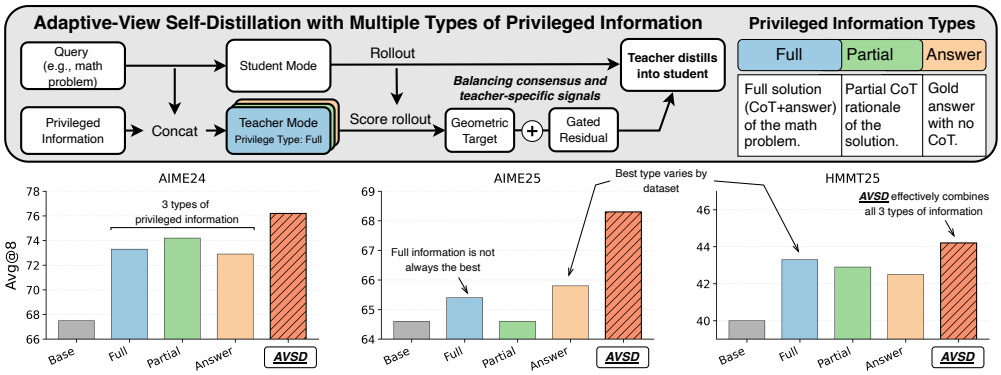

Figure 1: Self-distillation leverages access to privileged ground truth information as a teacher to distill into the student. However, performance across datasets on Qwen3-4B varies substantially depending on the type of privileged ground truth information the teacher has access to (full solution, partial reasoning, or only final answer). We show that no single type is uniformly optimal, and that AVSD consistently achieves the best performance across datasets by effectively combining multiple types of privileged information.

by giving the teacher privileged information that the student cannot access. For example, the student model might learn from a teacher with access to different augmented context views, such as solutions, demonstrations, or feedback [Zhao et al., 2026, Shenfeld et al., 2026].

Current self-distillation methods share a strong and often implicit assumption: once a privileged context type is chosen, it remains fixed throughout training [Shenfeld et al., 2026, Penaloza et al., 2026]. However, in practice, a variety of privileged information types are possible, with no clear way of determining *a priori* which will be optimal. For example, for a math problem, one may have access to a verified full solution, a final answer, a partial rationale, or a concise hint (see Fig. 1 bottom-center, showing differences when using full vs partial vs answer only privileged information). Each type comes with its own limitations, e.g., providing a full solution with chain of thought may lock the teacher into a preexisting thought pattern that does not align with the student's, while providing only the final answer may be more flexible but less informative. These different types of information can result in meaningful performance differences: Penaloza et al. [2026] show that different forms of privileged information vary in information density, task utility, and induced student-teacher distribution shift, and that the best choice depends on the task and model regime. Yang et al. [2026] show that distribution matching under information asymmetry contains an irreducible mutual-information gap, so a student trained to imitate a privileged teacher is pressured to encode view-specific correlations it cannot observe at test time. Kim et al. [2026] show that the most informative privileged form discards important signals that help models identify reasoning mistakes, and degrading out-of-domain mathematical reasoning, even when training traces are correct. These observations suggest that choosing a single privileged view has important consequences for performance and generalization, and that no single view is guaranteed to provide the best learning signals across tasks. This raises the central question of our work: **can we leverage multiple privileged views simultaneously to construct better token-level on-policy learning signals for self-distillation than any single privileged teacher?**

In this work, we propose **A**daptive-**V**iew **S**elf-**D**istillation (AVSD), an on-policy self-distillation framework that uses multiple types of privileged information rather than committing to a single teacher context. The key intuition is: (1) if different privileged views induce similar token-level updates, then the signal is likely stable and task-relevant. (2) In contrast, if a token is strongly favored by only one view, that signal may still be useful because it may capture complementary information not emphasized by other views, but it is riskier because it may depend on information that the student cannot access at inference time. AVSD formalizes this intuition by separating the teacher signal of multiple views into a shared part and an extra view-specific part. We refer to the shared part as the *consensus* signal: it captures tokens that are consistently supported across privileged views and provides a reliable update direction. We refer to the extra view-specific part as the *residual*: it captures additional support that appears when one or a few views strongly favor a token beyond what is shared by all views. To expose these two parts, AVSD uses two pooled targets: the geometric consensus

target and the arithmetic marginal target. The *geometric consensus target* emphasizes tokens that are supported across views, while the *arithmetic marginal target* preserves tokens that receive strong support from at least one view. Rather than distilling from either target directly, AVSD starts from the consensus signal and uses a gate to add the residual only when the different views agree on the promote-or-suppress direction, and the residual magnitude remains proportionate to the consensus magnitude. This allows AVSD to benefit from complementary information in different privileged views while preventing any single view from dominating the consensus token-level learning signal.

We validate AVSD on math and code reasoning benchmarks. Empirically, using three automatically-constructed privileged views for each math problem (full solution, partial solution, and final answer), we show that AVSD outperforms the standard single-view self-distillation baseline [Zhao et al., 2026, Shenfeld et al., 2026] and GRPO [Shao et al., 2024] on math competition benchmarks (AIME24 [Zhang and Math-AI, 2024], AIME25 [Zhang and Math-AI, 2025], and HMMT25 [Balunovic et al., 2025]). On Qwen3-4B [Yang et al., 2025], AVSD improves Avg@8 by 5.5% over the base model and 2.2% over the strongest baseline. The gains are consistent at larger scale: on Qwen3-8B, AVSD achieves the best average score, improving over the strongest baseline by 3.1%. We also observe similar improvements on DeepSeek-R1-Distill-Qwen-7B [DeepSeek-AI, 2025], where AVSD improves over the self-distillation baseline by 2.3% on average. Additionally, on coding benchmarks (Codeforces [Penedo et al., 2025], LiveCodeBench v6 [Jain et al., 2025]), AVSD improves the single-view self-distillation baseline by 2.4% on Qwen3-8B. Ablations show that consensus-only and arithmetic-only variants both underperform AVSD, and our token-level analysis further confirms that AVSD provides a better learning signal by jointly preserving cross-view agreement and selectively adding gated residual support.

2 AVSD: Adaptive-View Self-Distillation

In this section, we present AVSD (Adaptive-View Self-Distillation). In Section 2.1, we first formulate on-policy self-distillation through the token-level reverse-KL advantage, then extend it to the multi-view setting where each privileged view induces a different teacher distribution.

At a high level, AVSD builds a teacher signal by separating consensus from view-specific signal. If several views agree that a token should be promoted or suppressed, this provides a reliable anchor for the student update. In Section 2.2, we capture this shared signal with a *geometric consensus target*, which emphasizes tokens supported across views. In contrast, a token favored by only one or a few views may capture complementary task-relevant information, but may also reflect privileged artifacts. We expose this extra signal with an *arithmetic marginal target*, and define the *residual* as its difference from the geometric consensus target. Finally, in Section 2.3, we propose a gate to reconstruct the teacher signal by selectively adding this residual when the views agree on the update direction and the residual magnitude remains proportionate to the consensus magnitude.

2.1 Problem Setup

Let $\mathcal{D}$ be a training set. For notation simplicity, we write $(x, r) \sim \mathcal{D}$ for a generic example, where x is a prompt and r is the training-time privileged information. The student observes only x, while the teacher conditions on r during training. During training, we sample an on-policy rollout

$$y = (y_1, \ldots, y_T) \sim P_\theta^S(\cdot \mid x).$$

We denote the prefix at position t as $h_t = (x, y_{<t})$. The student next-token distribution over vocabulary $\mathcal{V}$ is

$$p_t(v) := P_\theta^S(Y_t = v \mid h_t), \;\; v \in \mathcal{V},$$

where v denotes a token type in the vocabulary. Standard single-view self-distillation chooses one view r and uses the same model conditioned on that view as the teacher. At the same prefix, denote

$$q_t(v) := \mathrm{sg}\big[P_\theta^T(Y_t = v \mid h_t, r)\big], \;\; v \in \mathcal{V},$$

where P_θ^T denotes the same model evaluated in the privileged teacher context and $\mathrm{sg}[\cdot]$ is the stop gradient operator. The single-view reverse-KL [Gu et al., 2024, Agarwal et al., 2024] objective is

$$\mathcal{L}_{\mathrm{sv}}(\theta) = \mathbb{E}_{(x,r)\sim\mathcal{D}}\mathbb{E}_{y\sim P_\theta^S(\cdot|x)}\left[\frac{1}{|y|}\sum_{t=1}^{|y|} D_{\mathrm{KL}}(p_t \,\|\, q_t)\right].$$

The negative local gradient of the reverse-KL term can be written in policy-gradient form as

$$-\nabla_\theta D_{\mathrm{KL}}(p_t \,\|\, q_t) = \mathbb{E}_{v\sim p_t}\left[A_t(v)\nabla_\theta \log p_t(v)\right],$$

where

$$A_t(v) := \log q_t(v) - \log p_t(v). \tag{1}$$

$A_t(v)$ in Eq. (1) is the token-level distillation advantage. If $A_t(v) > 0$, the teacher assigns token type v higher probability than the student, so gradient descent on the reverse-KL objective promotes v; if $A_t(v) < 0$, the update suppresses v. The full derivation of this policy-gradient form from the reverse-KL objective is given in Appendix B.1. We use $A_t(v)$ to describe the sign and relative magnitude of the token-level updates. In the single-view setting, this corresponds to choosing one privileged information type throughout training, such as always conditioning the teacher on a full solution in Fig. 1.

In the multi-view setting, we construct M privileged views $\mathcal{V}(r) = \{r^{(1)}, \ldots, r^{(M)}\}$, where $r^{(m)} = T_m(r)$ is the transformation that preserves task-relevant information while changing which privileged view is revealed to the teacher. For example, for a math problem, views may be a full solution, final answer, or partial rationale. Each view induces a teacher distribution at the same student prefix, which shows how different privileged information types would promote or suppress the same generated token (examples in Fig. 2 and Fig. 4):

$$q_t^{(m)}(v) := \mathrm{sg}\left[P_\theta^T(Y_t = v \mid h_t, r^{(m)})\right], \qquad m = 1, \ldots, M,$$

The corresponding per-view distillation advantage is $\Delta_t^{(m)}(v) := \log q_t^{(m)}(v) - \log p_t(v)$. A key question then becomes: **Given a teacher family $\{q_t^{(m)}\}_{m=1}^M$, how should we weigh and combine distributions from teachers to determine the token-level signal that the student receives?**

2.2 Consensus and Residual Signal

Different privileged views can agree or disagree at the same student prefix on the *signed token-level update* induced by each teacher view. For a token v, the teacher family provides a stable signal when the per-view advantages $\Delta_t^{(m)}(v)$ have consistent signs and comparable magnitudes: multiple views either agree that v should be promoted or agree that it should be suppressed. This *consensus* is informative because all teachers are conditioned on the same student-visible prefix h_t and differ only in what training-time privileged information they have access to. Therefore, an update that persists across several privileged views is less likely to depend on a particular view-specific artifact and more likely to reflect a *consensus* signal that is compatible with the information available from h_t. At the same time, not all useful information needs to be shared by every view: a token may receive strong support from only a subset of views because those views capture reasoning paths complementary to the consensus signal (Fig. 2 Token 2). However, the same subset-specific support may also reflect privileged artifacts that the student cannot access at test time (Fig. 2 Token 1) To resolve this, we use two pooled targets: one target captures tokens that are consistently supported across views, while the other captures tokens that are strongly supported by at least one view. Their gap defines the residual signal, which AVSD later gates before adding it to the consensus update.

Geometric Consensus Target. The first pooled target captures intersection support: a token receives large probability only if it is supported across the teacher family. We define the unnormalized geometric score and its normalized distribution as

$$\widetilde{q}_t^G(v) := \left(\prod_{m=1}^M q_t^{(m)}(v)\right)^{1/M} = \exp\left(\frac{1}{M}\sum_{m=1}^M \log q_t^{(m)}(v)\right), \quad q_t^G(v) = \frac{\widetilde{q}_t^G(v)}{\sum_{u\in\mathcal{V}} \widetilde{q}_t^G(u)}.$$

We call q_t^G the geometric consensus target as it is the normalized geometric mean of the view-conditioned teachers. The geometric mean downweights tokens that are assigned low probability by any view, and therefore emphasizes tokens that are jointly compatible with the teacher family. Thus, this target can be conservative (Fig. 2 top-center). Equivalently, q_t^G is the distribution that minimizes average reverse KL to the teacher. The full derivation is in Appendix B.

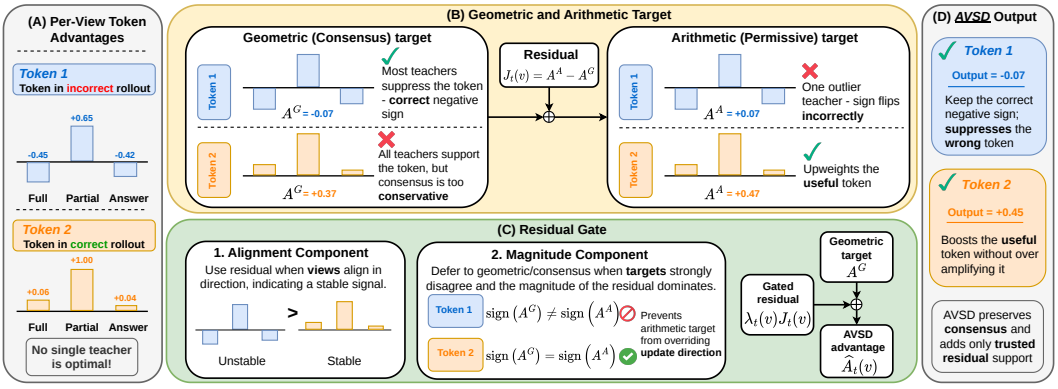

Figure 2: (A) Bar graphs show teacher advantages on student tokens. (B) The geometric target demands consensus between teachers (conservative), while the arithmetic target allows individual teachers to contribute more strongly (permissive). The residual is their difference. (C) A two-component gate controls the residual: the alignment component suppresses unstable signals when views conflict, and the magnitude component prevents the residual from flipping the update direction established by consensus. (D) AVSD combines both: it preserves the consensus direction on risky tokens and boosts useful view-specific signals on tokens with high cross-view alignment.

For token-level updates, we use the unnormalized geometric score, since the normalization contributes only a token-independent constant

$$A_t^G(v) := \log \widetilde{q}_t^G(v) - \log p_t(v) = \frac{1}{M} \sum_{m=1}^{M} \Delta_t^{(m)}(v).$$

Thus, $A_t^G(v)$ is the average signed update induced by the teacher family.

Arithmetic Marginal Target. The second pooled target captures union support: a token can receive high probability if it is strongly supported by any view (Fig. 2 top-center). We define

$$q_t^A(v) := \frac{1}{M} \sum_{m=1}^{M} q_t^{(m)}(v).$$

This is the arithmetic mean of the view-conditioned teachers. Equivalently, if a privileged view index is sampled uniformly at random, q_t^A is the marginal next-token distribution obtained after integrating out the view. It is also the solution to the average forward-KL aggregation problem that finds the single pooled target that minimizes the average forward-KL from the view-conditioned teachers (see Appendix B.2 for details). The arithmetic advantage is

$$A_t^A(v) := \log q_t^A(v) - \log p_t(v).$$

The arithmetic target preserves complementary support that appears in only one or a few views, which can be valuable when different views reveal different valid reasoning paths. However, this permissiveness also makes it sensitive to view-specific artifacts. For example, a full-solution view may strongly promote a token because it has access to a reference derivation, while an answer-only view provides no support for the same token. Therefore, directly distilling from the arithmetic target would lead to an update based on a noisy signal.

Cross-View Residual. Because q_t^G is conservative and q_t^A is permissive, their gap isolates the support retained by the permissive marginal but discarded by the strict consensus:

$$A_t^A(v) = A_t^G(v) + J_t(v), \qquad J_t(v) := \log q_t^A(v) - \log \widetilde{q}_t^G(v) \geq 0.$$

We call $J_t(v)$ the *cross-view residual* (Fig. 2 top-center). It is non-negative by the arithmetic-geometric mean inequality, since $q_t^A(v) \geq \widetilde{q}_t^G(v)$ for every token v. Intuitively, $J_t(v)$ measures how much probability mass is present in the arithmetic marginal but absent from strict geometric consensus. A large residual indicates that q_t^A preserves support from only a subset of teachers, while

a zero residual indicates exact token-level consensus across all teachers. The residual alone does not determine whether the extra support is reliable, because the same large residual can arise from useful complementary evidence or view-specific artifacts. Therefore, we use the consensus as the update direction and use the residual to adjust the magnitude of the update.

2.3 Gated Reconstruction of Token-Level Supervision via Consensus and Residual Signal

Residual Gate. The decomposition above suggests a simple principle: always keep view consensus, and add the cross-view residual only when it is reliable. We define the reconstructed advantage:

$$\widehat{A}_t(v) := A_t^G(v) + \lambda_t(v) J_t(v), \qquad \lambda_t(v) \in [0, 1].$$

The gate $\lambda_t(v)$ is designed around two requirements, illustrated by Token 1 and Token 2 in Fig. 2. For Token 1, one view strongly promotes the token, but the other views suppress it. In this case, the arithmetic target would incorrectly flip the update direction, so the gate should defer to the consensus signal. For Token 2, all views promote the token, but one view provides much stronger evidence than the others. In this case, the residual can be useful, because it strengthens an update direction that the views already agree on. However, it should still be bounded so that the extra view-specific signal does not dominate the consensus. These two cases motivate two components of the gate described below.

First, the residual should not be trusted when teacher views heavily contradict each other in terms of update direction. For example, if some views promote a token while others suppress it, the residual is unreliable (Fig. 2 Alignment Component). We define the alignment component by the ratio between the magnitude of the averaged signed advantage and the average magnitude of the per-view advantages:

$$C_t(v) := \frac{|A_t^G(v)|}{\frac{1}{M}\sum_{m=1}^{M} |\Delta_t^{(m)}(v)| + \epsilon} = \frac{\frac{1}{M}|\sum_{m=1}^{M} \Delta_t^{(m)}(v)|}{\frac{1}{M}\sum_{m=1}^{M} |\Delta_t^{(m)}(v)| + \epsilon} \in [0, 1],$$

$C_t(v)$ is high when the per-view advantages have the same sign or when unsupported views are nearly neutral, and low when positive and negative view-specific updates cancel. This allows AVSD to distinguish complementary support from conflict-driven support.

Second, the residual should be proportionate to the consensus signal (Fig. 2 Magnitude Component). If $J_t(v)$ is much larger than $|A_t^G(v)|$, then the arithmetic residual can dominate or even reverse the consensus update. For example, if $A_t^G(v) < 0$ but $J_t(v) \gg |A_t^G(v)|$, the arithmetic target would promote a token that the teacher family suppresses on average. To prevent this, we define:

$$R_t(v) = \frac{|A_t^G(v)|}{|A_t^G(v)| + J_t(v) + \epsilon}.$$

The final gate is

$$\lambda_t(v) = C_t(v) R_t(v) = C_t(v) \frac{|A_t^G(v)|}{|A_t^G(v)| + J_t(v) + \epsilon}, \tag{2}$$

This gate acts as an adaptive regularizer on the residual signal. It admits residual support when the *consensus is coherent* and the *residual is proportionate*, but suppresses it when the views conflict, when the consensus is weak, or when the residual is too large. Moreover, expanding the gate in Eq. (2) gives $\lambda_t(v) J_t(v) \le |A_t^G(v)|$, since $C_t(v) \le 1$ and $J_t(v) \le |A_t^G(v)| + J_t(v) + \epsilon$ (the full derivation is provided in Appendix B.3); the residual can adjust the update magnitude but cannot reverse the update direction induced by the consensus. Our gate also connects to the failure mode identified by Yang et al. [2026]: directly matching a teacher conditioned on information unavailable to the student can induce an irreducible mutual-information gap and privileged-information leakage. AVSD addresses this risk in the multi-view setting by preventing subset-supported residuals from freely determining the update direction. Fig. 2 illustrates how the gate addresses failure modes of the two pooled targets. For Token 1, one view promotes the token while the others suppress it, so AVSD keeps the consensus direction and prevents the residual from flipping the update. For Token 2, the views agree on the positive direction, so the residual can strengthen the update.

Reconstructed Target. The reconstructed target corresponding to $\widehat{A}_t$ is obtained by reweighting the geometric consensus target with the gated residual and renormalizing:

$$q_t^\star(v) = \frac{\widetilde{q}_t^G(v) \exp(\lambda_t(v) J_t(v))}{\sum_{u \in \mathcal{V}} \widetilde{q}_t^G(u) \exp(\lambda_t(u) J_t(u))}.$$

This interpolates tokenwise between two targets. If $\lambda_t(v) = 0$, the method reduces to the geometric consensus target. If $\lambda_t(v) = 1$ for all v, then the method recovers the arithmetic marginal target.

Training Objective. We train the student by on-policy reverse-KL distillation to the reconstructed target. The distillation loss is

$$\mathcal{L}_{\mathrm{AVSD}}(\theta) = \mathbb{E}_{(x,r)\sim\mathcal{D}}\mathbb{E}_{y\sim P_\theta^S(\cdot|x)}\left[\sum_{t=1}^{|y|} D_{\mathrm{KL}}\left(p_\theta(\cdot \mid h_t) \,\|\, \mathrm{sg}[q_t^\star(\cdot)]\right)\right], \tag{3}$$

Gradients flow only through the student distribution $p_\theta(\cdot \mid h_t)$, while the teacher distributions and the reconstructed target are treated as fixed supervision at the current prefix. Equivalently, using the log-derivative form of the reverse-KL objective, the sampled-token update is the reconstructed advantage $\widehat{A}_t(y_t) = A_t^G(y_t) + \lambda_t(y_t)J_t(y_t)$. Compared with standard single-view self-distillation, AVSD only replaces a single teacher-context evaluation with M view-conditioned teacher evaluations at the same student prefix. It does not require additional student rollouts: all views share the same generated trajectory and can be evaluated in parallel or batched. Since on-policy rollout generation is sequential and typically dominates the training cost, while the additional view-conditioned teacher evaluations are prefill-only forward passes, AVSD only adds modest overhead in practice (details in Table 2). We provide the full training algorithm in Appendix B.4.

3 Experiments

In this section, we evaluate AVSD on math and code reasoning benchmarks across three base models. We first present the experimental setup, including datasets, baselines, and evaluation, and then present the main empirical results.

3.1 Experimental Setup

Models. We evaluate across multiple model families and scales, including Qwen3-4B and Qwen3-8B [Yang et al., 2025], and DeepSeek-R1-Distill-Qwen-7B [DeepSeek-AI, 2025].

Datasets. We use the following math and code datasets for training and evaluation:

- **Math datasets:** Following Zhao et al. [2026], we use the mathematical reasoning subset of OpenThoughts [Guha et al., 2026] as the training data. For the multi-view setup, we use three views that are directly available or easily derived from the dataset: the full solution, the final answer, and a partial solution obtained by retaining the initial reasoning steps of the full solution. We evaluate on competition-level mathematics benchmarks including AIME 2024 [Zhang and Math-AI, 2024], AIME 2025 [Zhang and Math-AI, 2025], and HMMT 2025 [Balunovic et al., 2025].
- **Code datasets:** We train the models on Codeforces [Penedo et al., 2025], sampling 5K examples from the Python subset. For each problem, we construct three privileged views: a reference implementation, an algorithmic hint derived from the reference solution, and execution feedback obtained by running the student rollout on available test cases, which includes the generated program and the verifier's pass/fail results. For evaluation, we hold out 100 Codeforces problems as an in-domain test set and further evaluate on LiveCodeBench v6 [Jain et al., 2025].

The examples of math and code views are provided in Appendix E. Similar to [He et al., 2026], we report Avg@8 throughout the experiments. We provide the detailed training and evaluation hyperparameters in Appendix C.

Baselines. We compare against three methods trained on the same dataset: (1) SFT, standard supervised fine-tuning, which can be seen as off-policy distillation from a static dataset of the reasoning traces, (2) GRPO [Shao et al., 2024], group relative policy optimization with binary outcome rewards verified against ground-truth answers, (3) OPSD, the single-view on-policy self-distillation baseline [Zhao et al., 2026, Shenfeld et al., 2026]. For this baseline, we use the same

Table 1: Avg@8 accuracy on math and code benchmarks across three base models. The best result within each model block is in **bold**, and the second best is underlined.

Method	Math			Code		Avg.
	AIME24	AIME25	HMMT25	CF	LCB	
Qwen3-4B	67.5	64.6	40.0	59.8	43.2	55.0
+ SFT	56.3	52.1	29.6	59.1	40.7	47.6
+ GRPO	70.6	65.7	41.8	61.2	43.9	56.6
+ OPSD	73.3	65.4	43.3	63.6	45.2	58.2
+ **AVSD (Ours)**	**76.2**	**68.3**	**44.2**	**65.6**	**45.4**	**59.9**
Qwen3-8B	71.2	65.4	41.2	60.9	47.6	57.3
+ SFT	70.8	63.5	41.5	61.2	43.2	56.0
+ GRPO	72.5	67.1	43.0	62.8	49.9	59.1
+ OPSD	74.2	65.8	42.5	63.2	50.3	59.2
+ **AVSD (Ours)**	**75.4**	**69.6**	**47.1**	**65.8**	**52.4**	**62.1**
DeepSeek-R1-Distill-Qwen-7B	48.3	37.9	21.7	44.8	35.7	37.7
+ SFT	48.5	36.2	20.5	41.6	35.2	36.4
+ GRPO	51.2	37.8	23.1	46.2	38.0	39.3
+ OPSD	53.3	37.5	22.6	43.6	37.6	38.9
+ **AVSD (Ours)**	**55.8**	**39.2**	**25.4**	**47.4**	**39.1**	**41.4**

reverse-KL distillation objective as AVSD, but condition the teacher on a single privileged view: the full solution for math and the reference implementation for code, following the original OPSD setting [Zhao et al., 2026].

3.2 Results: AVSD Improves Math and Code Reasoning over GRPO and OPSD

Table 1 shows that AVSD consistently improves both math and code reasoning across model families. On the math benchmarks, AVSD achieves the best average Avg@8 for all three base models. For Qwen3-4B, AVSD improves over the base model by 5.5% and outperforms the strongest baseline, OPSD, by 2.2% on average. The gains are consistent at larger scale: on Qwen3-8B, AVSD improves over the base model by 4.8% and over the strongest baseline, GRPO, by 3.1%. We observe similar improvements on DeepSeek-R1-Distill-Qwen-7B, where AVSD improves over the strongest self-distillation baseline by 2.3% on average.

The benefits also transfer to code generation. On Qwen3-8B, AVSD improves the average code Avg@8 on Codeforces and LiveCodeBench over the base model and OPSD by 4.9% and 2.4%, respectively. Similar trends hold for Qwen3-4B and DeepSeek-R1-Distill-Qwen-7B, where AVSD achieves the best code average among the compared methods. Overall, these results demonstrate that AVSD provides consistent gains over GRPO and single-view on-policy self-distillation baselines across both math and code benchmarks.

4 Analysis

In this section, we analyze AVSD and examine why it improves over single-view self-distillation. We first ablate the consensus and arithmetic targets to isolate the role of gated reconstruction, and then examine token-level credit allocation to test whether AVSD provides more reliable learning signals on student-generated rollouts. We provide additional results and analyses in Appendix D.

4.1 Ablating Consensus and Arithmetic Targets

We evaluate the performance of AVSD when using either the consensus target or the arithmetic marginal target alone. The consensus-only variant uses A_t^G as the learning signal and discards the arithmetic-over-consensus residual. The arithmetic-only variant distills directly from q_t^A. This comparison isolates the role of our gated reconstruction. Fig. 3 shows that neither standalone aggregation is uniformly optimal: the consensus target performs strongly on AIME24 and AIME25 but is slightly worse than OPSD on HMMT25, while the arithmetic target improves on AIME24 and

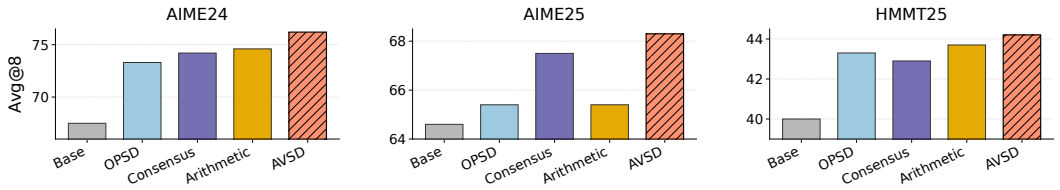

Figure 3: Comparison of single-view OPSD, consensus-only distillation, arithmetic-only distillation, and the full AVSD on Qwen3-4B. AVSD achieves the best Avg@8 across all datasets.

HMMT25 but gives limited gains on AIME25. In contrast, AVSD achieves the best performance across all benchmarks and improves over both standalone multi-view targets. Averaged across the three benchmarks, AVSD outperforms the consensus-only and arithmetic-only variants by 1.4% and 1.7%, respectively. This demonstrates that combining the consensus signal with a selectively gated residual is necessary: the consensus target alone can be overly conservative, while the arithmetic target alone can overuse view-specific support. These results support our central claim that reliable multi-view self-distillation requires reconstructing a token-level target from both stable cross-view agreement and carefully filtered residual information.

4.2 Token-Level Credit Allocation

We further analyze whether the token-level learning signal produced by AVSD assigns credit in the desired direction. The sampled-token advantage determines whether the probability of a generated token is increased or decreased. For an incorrect rollout, generated tokens should generally receive negative credit, indicating that the method suppresses erroneous solutions. To evaluate this behavior, we collect student rollouts whose final answers are incorrect and compute the sampled-token advantage at every generated token. For OPSD and AVSD, we select the top 20 tokens with the largest absolute advantage magnitude, corresponding to the tokens where the distillation signal has the strongest effect on training, and measure the fraction of these tokens whose credit has the incorrect (positive) sign. For GRPO, which does not provide dense token-level supervision, we instead report the fraction of cases with no learning signal (e.g., while rollouts should have either positive or negative signs depending on correctness, by construction, most GRPO methods give no signal when no sampled rollout is correct). The heatmap in Fig. 4 visualizes this behavior at the token level for the top 5 tokens in one rollout: each column is a generated token from the incorrect rollout, and each cell shows whether a method increases or decreases that token's probability. Since the rollout is incorrect, the desired behavior is to assign negative credit to the high-impact tokens, i.e., the generated tokens with the largest absolute advantage magnitudes and therefore the strongest effect on the training loss. Single-view baselines produce mixed signs across tokens, suggesting that their feedback depends strongly on the chosen privileged view, whereas AVSD assigns more consistently negative credit to the key tokens while avoiding the near-zero signal of GRPO. Quantitatively, GRPO has no learning signal in 29.6% of all examples, OPSD assigns incorrect-sign credit to 12.3% of high-impact tokens, whereas AVSD reduces this rate to only 2.9% (see Fig. 8, Appendix D). These results show that AVSD provides a substantially more reliable token-level learning signal than both sparse-reward GRPO and single-view self-distillation.



Figure 4: Example from a single *incorrect rollout*: we show the top-5 generated tokens with largest absolute token-credit magnitude from that rollout. Positive values indicate that the token is encouraged, while negative values indicate that the token is discouraged. For GRPO, the sampled group for the same problem contains no correct rollout, so the outcome-level reward provides no learning signal. **Single-view self-distillation teachers assign inconsistent or misleading signs across tokens, whereas AVSD provides a more coherent credit signal on key tokens.** Full problem and generation are in Appendix D.

5 Related Work

Learning from Privileged Information. Learning from privileged information studies how training-time side information unavailable at inference can improve a model [Lopez-Paz et al., 2016]. In LLM post-training, such information can be reference solutions, final answers, demonstrations, reward models, or environment feedback. Prior work uses these signals in several ways. RLVR methods provide sparse verifiable rewards for math and code [Shao et al., 2024, DeepSeek-AI, 2025], while self-improvement and bootstrapping methods construct additional supervision from model-generated data, verified reasoning traces, or reward models [Zelikman et al., 2022, Wang et al., 2023, Nguyen et al., 2025]. Other approaches use privileged information to improve exploration or curriculum construction: Qu et al. [2026] leverages oracle solution prefixes for on-policy exploration; Yan et al. [2026] learns under off-policy guidance from privileged rollouts; Chen et al. [2026a], Liao et al. [2026], Wang et al. [2026] transform privileged trajectories into abstract hints. Synthetic-curriculum methods such as Sundaram et al. [2026], Chen et al. [2026b] generate easier or more informative training instances to bridge low-signal regimes. These methods demonstrate that privileged or derived information can substantially improve learning when direct reward signal is sparse. However, they generally focus on how to obtain better trajectories, hints, or curricula, rather than how to construct a token-level learning signal when multiple privileged views are simultaneously available. Moreover, the choice of privileged format remains consequential: different forms of privileged information vary in information density, task utility, and induced student–teacher distribution shift [Penaloza et al., 2026, Li et al., 2026b]. Our work is complementary to these approaches: rather than proposing a new source of privileged information or exploration, AVSD provides a principled way to aggregate multiple privileged views into a better learning signal by separating cross-view consensus from view-specific residual support.

On-Policy Self-Distillation and Teacher Aggregation. Self-distillation methods remove the external teacher needed in on-policy distillation by using the same model under richer conditioning. Zhao et al. [2026] condition on reference math solution, Shenfeld et al. [2026], Ye et al. [2026] distill demonstration-conditioned behavior, and reinforcement learning via self-distillation uses feedback, successful rollouts, or reward-conditioned self-revisions as privileged context [Hübotter et al., 2026, Song et al., 2026, He et al., 2026]. Penaloza et al. [2026] further study teacher-student transfer when the teacher has privileged information and the student must act without it. These methods show that the distillation objectives can serve as a useful token-level advantage. Yang et al. [2026] identify a key failure of direct privileged-teacher matching under information asymmetry and anchors the update direction to verifiable rewards while using self-distillation mainly to adjust token-level update strength. Related hybrid RL-distillation methods combine policy optimization with distillation losses [Ding, 2026, Li et al., 2026a, Xu et al., 2025], and ensemble distillation methods aggregate teachers by averaging logits or votes. However, these methods do not characterize which parts of the teacher signal are stable across views versus specific to a privileged context. In contrast, AVSD derives the aggregation structure from the reverse-KL learning signal itself. Thus, our method complements RLSD and other hybrid RL-distillation works: rather than relying on the binary rewards (which are only available in verifiable settings) to determine the update direction for a single privileged teacher in self-distillation, AVSD reconstructs the token-level learning signal from the geometry of a multi-view privileged teacher family.

6 Conclusion

In this work, we introduced AVSD, a multi-view framework for privileged self-distillation. Instead of distilling from a single privileged teacher, AVSD treats different views of the same training-time artifact as a teacher family and reconstructs a better token-level learning signal from their shared structure. This allows the student to benefit from complementary privileged information while reducing the influence of conflicting or unreliable view-specific signals. Across math and code reasoning benchmarks and multiple model families, AVSD consistently outperforms single-view self-distillation and GRPO. Our analysis further shows AVSD's performance generally improves as the number of views increases and it is robust to a single misleading view. that More broadly, our results suggest that moving beyond fixed single-view privileged supervision and explicitly modeling agreement and disagreement across multiple teacher views provides a robust and general approach for constructing dense on-policy learning signals.

Acknowledgements

This work was supported by NSF-CAREER Award 1846185, NSF AI Engage Institute DRL2112635, Capital One Research Award, Apple PhD Fellowship, and NDSEG PhD Fellowship. The views contained in this article are those of the authors and not of the funding agency.

References

Ali E Abbas. A kullback-leibler view of linear and log-linear pools. *Decision Analysis*, 6(1):25–37, 2009.

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=3zKtaqxLhW.

Mislav Balunovic, Jasper Dekoninck, Ivo Petrov, Nikola Jovanovic, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions, February 2025. URL https://matharena.ai/.

Justin Chen, Xiangyu Peng, Prafulla Kumar Choubey, Kung-Hsiang Huang, Jiaxin Zhang, Mohit Bansal, and Chien-Sheng Wu. Nudging the boundaries of LLM reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=hfNnQHkTtv.

Justin Chih-Yao Chen, Archiki Prasad, Zaid Khan, Joykirat Singh, Runchu Tian, Elias Stengel-Eskin, and Mohit Bansal. Cog-drift: Exploration on adaptively reformulated instances enables learning from hard reasoning problems, 2026b. URL https://arxiv.org/abs/2604.04767.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Ken Ding. Hdpo: Hybrid distillation policy optimization via privileged self-distillation, 2026. URL https://arxiv.org/abs/2603.23871.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=5h0qf7IBZZ.

Etash Kumar Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Rea Sprague, Ashima Suvarna, Benjamin Feuer, Leon Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, Wanjia Zhao, John Yang, Shreyas Pimpalgaonkar, Kartik sharma, Charlie Cheng-Jie Ji, Yichuan Deng, Sarah M Pratt, Vivek Ramanujan, Jon Saad-Falcon, Stutee Acharya, Jeffrey Li, Achal Dave, Alon Albalak, Kushal Arora, Blake Wulfe, Chinmay Hegde, Greg Durrett, Sewoong Oh, Mohit Bansal, Saadia Gabriel, Aditya Grover, Kai-Wei Chang, Vaishaal Shankar, Aaron Gokaslan, Mike A Merrill, Tatsunori Hashimoto, Yejin Choi, Jenia Jitsev, Reinhard Heckel, Maheswaran Sathiamoorthy, Alex Dimakis, and Ludwig Schmidt. Openthoughts: Data recipes for reasoning models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=7xjoTuaNmN.

Yinghui He, Simran Kaur, Adithya Bhaskar, Yongjin Yang, Jiarui Liu, Narutatsu Ri, Liam Fowl, Abhishek Panigrahi, Danqi Chen, and Sanjeev Arora. Self-distillation zero: Self-revision turns binary rewards into dense supervision, 2026. URL https://arxiv.org/abs/2604.12002.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. URL https://arxiv.org/abs/1503.02531.

Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, and Andreas Krause. Reinforcement learning via self-distillation, 2026. URL https://arxiv.org/abs/2601.20802.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=chfJJYC3iL.

Jeonghye Kim, Xufang Luo, Minbeom Kim, Sangmook Lee, Dohyung Kim, Jiwon Jeon, Dongsheng Li, and Yuqing Yang. Why does self-distillation (sometimes) degrade the reasoning capability of llms?, 2026. URL https://arxiv.org/abs/2603.24472.

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

Gengsheng Li, Tianyu Yang, Junfeng Fang, Mingyang Song, Mao Zheng, Haiyun Guo, Dan Zhang, Jinqiao Wang, and Tat-Seng Chua. Unifying group-relative and self-distillation policy optimization via sample routing, 2026a. URL https://arxiv.org/abs/2604.02288.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huan ang Gao, Wenkai Yang, Zhiyuan Liu, and Ning Ding. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe, 2026b. URL https://arxiv.org/abs/2604.13016.

Baohao Liao, Hanze Dong, Xinxing Xu, Christof Monz, and Jiang Bian. Self-hinting language models enhance reinforcement learning, 2026. URL https://arxiv.org/abs/2602.03143.

David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. Unifying distillation and privileged information, 2016. URL https://arxiv.org/abs/1511.03643.

Duy Nguyen, Archiki Prasad, Elias Stengel-Eskin, and Mohit Bansal. LASer: Learning to adaptively select reward models with multi-arm bandits. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=tSpWkTFASC.

Emiliano Penaloza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. Privileged information distillation for language models, 2026. URL https://arxiv.org/abs/2602.04942.

Guilherme Penedo, Anton Lozhkov, Hynek Kydlícek, Loubna Ben Allal, Edward Beeching, Agustín Piqueres Lajarín, Quentin Gallouédec, Nathan Habib, Lewis Tunstall, and Leandro von Werra. Codeforces. https://huggingface.co/datasets/open-r1/codeforces, 2025.

Yuxiao Qu, Amrith Setlur, Virginia Smith, Ruslan Salakhutdinov, and Aviral Kumar. Pope: Learning to reason on hard problems via privileged on-policy exploration, 2026. URL https://arxiv.org/abs/2601.18779.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning, 2026. URL https://arxiv.org/abs/2601.19897.

Yuda Song, Lili Chen, Fahim Tajwar, Remi Munos, Deepak Pathak, J. Andrew Bagnell, Aarti Singh, and Andrea Zanette. Expanding the capabilities of reinforcement learning via text feedback, 2026. URL https://arxiv.org/abs/2602.02482.

Shobhita Sundaram, John Quan, Ariel Kwiatkowski, Kartik Ahuja, Yann Ollivier, and Julia Kempe. Teaching models to teach themselves: Reasoning at the edge of learnability, 2026. URL https://arxiv.org/abs/2601.18778.

Leitian Tao, Ilia Kulikov, Swarnadeep Saha, Tianlu Wang, Jing Xu, Sharon Li, Jason E Weston, and Ping Yu. Hybrid reinforcement: When reward is sparse, it's better to be dense, 2025. URL https://arxiv.org/abs/2510.07242.

Hao Wang, Guozhi Wang, Han Xiao, Yufeng Zhou, Yue Pan, Jichao Wang, Ke Xu, Yafei Wen, Xiaohu Ruan, Xiaoxin Chen, and Honggang Qi. Skill-sd: Skill-conditioned self-distillation for multi-turn llm agents, 2026. URL https://arxiv.org/abs/2604.10674.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2023. URL https://aclanthology.org/2023.acl-long.754/.

Hongling Xu, Qi Zhu, Heyuan Deng, Jinpeng Li, Lu Hou, Yasheng Wang, Lifeng Shang, Ruifeng Xu, and Fei Mi. Kdrl: Post-training reasoning llms via unified knowledge distillation and reinforcement learning, 2025. URL https://arxiv.org/abs/2506.02208.

Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. A survey on knowledge distillation of large language models, 2024. URL https://arxiv.org/abs/2402.13116.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. Learning to reason under off-policy guidance. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=vO8LLoNWWk.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. Self-distilled rlvr, 2026. URL https://arxiv.org/abs/2604.03128.

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. On-policy context distillation for language models, 2026. URL https://arxiv.org/abs/2602.12275.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. STar: Bootstrapping reasoning with reasoning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=_3ELRdg2sgI.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2024, 2024.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2025, 2025.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models, 2026. URL https://arxiv.org/abs/2601.18734.

A Limitations and Broader Impact

Limitations. Our work focuses primarily on reasoning tasks such as math and code. Although the formulation of AVSD is general, validating it in broader settings, such as tool-use agents, is a valuable direction for future work. The method also depends on the quality of the constructed views, as well as the underlying data quality. For example, in math reasoning, noisy annotated solutions may lead to less reliable teacher signals. In addition, AVSD is instantiated with reverse-KL on-policy self-distillation. While the reconstructed target can in principle be used with other distribution-matching losses, our advantage interpretation and bounded-residual guarantee require further study for other objectives, as discussed in Appendix B.5.

Broader Impact. This work aims to improve how language models learn from information that is available during training, such as solutions, demonstrations, or feedback. By making better use of this information, AVSD may help improve model performance on reasoning tasks such as math and code. Better models could support education, coding assistance, scientific problem solving, and other applications. However, stronger reasoning models can also be misused, and improvements in post-training methods should be accompanied by careful evaluation and deployment safeguards. In addition, training-time information may contain biases and artifacts from the data collection process. Although AVSD is designed to reduce reliance on any single privileged source, it does not remove the need for high-quality data and evaluation for unintended behavior.

B Additional Details for AVSD

In this section, we provide additional details for our method. In Appendix B.1, we derive the reverse-KL sampled-token advantage used in on-policy self-distillation. In Appendix B.2, we derive the arithmetic and geometric pooled targets and show how they correspond to natural aggregation objectives over the teacher family. In Appendix B.3, we provide further details on the gated reconstruction mechanism and the residual-boundedness property used in the main method. In Appendix B.4, we provide the detailed training algorithm for AVSD. In Appendix B.5, we discuss the applicability to other distillation losses.

B.1 Reverse-KL Token Advantage

Fix a student prefix h_t and a stop-gradient teacher distribution q_t. Let $p_t(v) = p_\theta(Y_t = v \mid h_t)$. The local reverse-KL loss is

$$\ell_t(\theta) = D_{\mathrm{KL}}(p_t \| q_t) = \sum_{v \in \mathcal{V}} p_t(v) \left(\log p_t(v) - \log q_t(v)\right).$$

Taking the gradient with respect to the student parameters,

$$\begin{aligned}\nabla_\theta \ell_t &= \sum_v \nabla_\theta p_t(v) \left(\log p_t(v) - \log q_t(v) + 1\right) \\ &= \mathbb{E}_{v \sim p_t}\left[\left(\log p_t(v) - \log q_t(v) + 1\right) \nabla_\theta \log p_t(v)\right].\end{aligned}$$

The constant 1 is a baseline because $\mathbb{E}_{v \sim p_t}[\nabla_\theta \log p_t(v)] = 0$. Therefore,

$$\nabla_\theta \ell_t = \mathbb{E}_{v \sim p_t}\left[\left(\log p_t(v) - \log q_t(v)\right) \nabla_\theta \log p_t(v)\right].$$

Equivalently, minimizing reverse KL is a policy-gradient update with distillation advantage

$$A_t(v) = \log q_t(v) - \log p_t(v),$$

since gradient descent on ℓ_t moves in the direction

$$\mathbb{E}_{v \sim p_t}\left[A_t(v) \nabla_\theta \log p_t(v)\right].$$

This is the advantage used in the main text to interpret whether a token type is promoted or suppressed. The sampled-token estimator replaces the full expectation over $v \sim p_t$ by the generated token y_t at the visited prefix.

B.2 Derivations of Two Pooled Targets

Fix a prefix h_t and write $q_m = q_t^{(m)}$.

Geometric Target. The geometric target is

$$q_t^G = \arg\min_q \frac{1}{M} \sum_{m=1}^{M} D_{\mathrm{KL}}(q \parallel q_m).$$

Expanding gives

$$\frac{1}{M} \sum_m D_{\mathrm{KL}}(q \parallel q_m) = \sum_v q(v) \log q(v) - \sum_v q(v) \frac{1}{M} \sum_m \log q_m(v).$$

Let

$$\tilde{q}_G(v) = \exp\left(\frac{1}{M} \sum_m \log q_m(v) \right).$$

The Lagrangian is

$$\mathcal{L}(q, \nu) = \sum_v q(v) \log q(v) - \sum_v q(v) \log \tilde{q}_G(v) + \nu \left(\sum_v q(v) - 1 \right).$$

Taking derivatives gives

$$\log q(v) + 1 - \log \tilde{q}_G(v) + \nu = 0,$$

so

$$q(v) \propto \tilde{q}_G(v).$$

Therefore,

$$q_t^G(v) = \frac{\tilde{q}_t^G(v)}{\sum_u \tilde{q}_t^G(u)}.$$

Arithmetic Target. The arithmetic target is

$$q_t^A = \arg\min_q \frac{1}{M} \sum_{m=1}^{M} D_{\mathrm{KL}}(q_m \parallel q).$$

Expanding the terms that depend on q,

$$\frac{1}{M} \sum_m D_{\mathrm{KL}}(q_m \parallel q) = -\sum_v \left(\frac{1}{M} \sum_m q_m(v) \right) \log q(v) + \text{const.}$$

The minimizer of cross-entropy with target distribution

$$\frac{1}{M} \sum_m q_m$$

is exactly

$$q_t^A(v) = \frac{1}{M} \sum_m q_t^{(m)}(v).$$

The geometric and arithmetic targets also connect to classical opinion pooling: linear pools correspond to arithmetic aggregation, while log-linear pools and products of experts combine distributions through normalized products [Abbas, 2009]. The same aggregation perspective naturally extends to non-uniform view weights. If each view is assigned a prefix-level reliability weight $w_t^{(m)} \geq 0$ with $\sum_m w_t^{(m)} = 1$, then the geometric and arithmetic pooled targets become the corresponding weighted KL barycenters by replacing the uniform factor $1/M$ with $w_t^{(m)}$ in the objectives above. This yields the same interpretation: the geometric target captures weighted cross-view consensus, while the arithmetic target captures weighted view-marginal support. In our main experiments, we use uniform weights to keep the method parameter-free and avoid introducing additional view-reliability hyperparameters.

Algorithm 1 Multi-View Self-Distillation with Gated Reconstruction. In our experiments, we find that $M = 3$ works well.

Require: Student model p_θ, dataset $\mathcal{D} = \{(x, r)\}$, view transformations $\{T_m\}_{m=1}^{M}$, small constant $\epsilon > 0$.

for each training iteration **do**
  Sample a minibatch $\{(x_i, r_i)\} \sim \mathcal{D}$.
  for each example (x, r) **do**
    Construct privileged views $r^{(m)} = T_m(r)$ for $m = 1, \dots, M$.
    Sample an on-policy rollout $y \sim p_\theta(\cdot \mid x)$.
    for each prefix $h_t = (x, y_{<t})$ **do**
      Compute student distribution $p_t(\cdot) = p_\theta(\cdot \mid h_t)$.
      for each view $m = 1, \dots, M$ **do**
        Compute teacher distribution

$$q_t^{(m)}(\cdot) = \mathrm{sg}\left[p_\theta(\cdot \mid h_t, r^{(m)})\right].$$

      end for
      Compute

$$q_t^A(v) = \frac{1}{M}\sum_m q_t^{(m)}(v), \qquad \widetilde{q}_t^G(v) = \left(\prod_m q_t^{(m)}(v)\right)^{1/M}.$$

      Compute residual

$$J_t(v) = \log q_t^A(v) - \log \widetilde{q}_t^G(v).$$

      Compute per-view advantages

$$\Delta_t^{(m)}(v) = \log q_t^{(m)}(v) - \log p_t(v), \qquad A_t^G(v) = \frac{1}{M}\sum_m \Delta_t^{(m)}(v).$$

      Compute gate

$$C_t(v) = \frac{|A_t^G(v)|}{\frac{1}{M}\sum_m |\Delta_t^{(m)}(v)| + \epsilon}, \qquad \lambda_t(v) = C_t(v)\frac{|A_t^G(v)|}{|A_t^G(v)| + J_t(v) + \epsilon}.$$

      Reconstruct teacher

$$q_t^\star(v) = \frac{\widetilde{q}_t^G(v)\exp(\lambda_t(v)J_t(v))}{\sum_{u\in\mathcal{V}} \widetilde{q}_t^G(u)\exp(\lambda_t(u)J_t(u))}.$$

      Accumulate loss

$$\mathcal{L}_{\mathrm{AVSD}} \leftarrow \mathcal{L}_{\mathrm{AVSD}} + D_{\mathrm{KL}}\left(p_t(\cdot) \,\|\, \mathrm{sg}[q_t^\star(\cdot)]\right).$$

    end for
  end for
  Update θ by minimizing $\mathcal{L}_{\mathrm{AVSD}}(\theta)$.
end for

B.3 Bounded Residual Property

For the gate

$$\lambda_t(v) = C_t(v)\frac{|A_t^G(v)|}{|A_t^G(v)| + J_t(v) + \epsilon},$$

where $C_t(v) \in [0, 1]$, we have

$$0 \le \lambda_t(v)J_t(v) = C_t(v)\frac{|A_t^G(v)|J_t(v)}{|A_t^G(v)| + J_t(v) + \epsilon} \le |A_t^G(v)|.$$

Table 2: Runtime overhead of AVSD compared with single-view self-distillation. We report average wall-clock training time per gradient update and relative overhead using Qwen3-4B under the same batch size and hardware setup. AVSD uses the same student rollouts as OPSD and only adds view-conditioned teacher evaluations.

Method	Time / step (s)	Relative time
OPSD	41.5	1.00×
AVSD	46.7	1.13×

Therefore, the reconstructed advantage

$$\widehat{A}_t(v) = A_t^G(v) + \lambda_t(v) J_t(v)$$

preserves the direction of the consensus update: if $A_t^G(v) > 0$, then $\widehat{A}_t(v) \geq 0$; if $A_t^G(v) < 0$, then $\widehat{A}_t(v) \leq 0$. Thus, the residual can strengthen a positive consensus update or soften a negative consensus update, but cannot flip the consensus-induced direction.

B.4 Detailed Training Algorithm

Algorithm 1 summarizes the full training procedure. The method preserves the on-policy nature of self-distillation: all teacher distributions are evaluated on prefixes generated by the current student. Compared to single-view self-distillation, the only additional cost is evaluating multiple privileged views for the same prefix. In practice, these view-conditioned teacher forward passes can be batched. We report the run time comparison with single-view OPSD in Table 2. AVSD increases training time only modestly, from 41.5s to 46.7s per step, corresponding to a 1.13× relative cost, since all views share the same student rollout and only add batched teacher evaluations.

B.5 Applicability to other Distillation Losses

Although we derive AVSD with reverse KL, the teacher reconstruction itself is not tied to a particular divergence. The multi-view aggregation step constructs a reconstructed target $q_t^\star$ from the teacher family $\{q_t^{(m)}\}_{m=1}^M$, and this target can in principle be used with other distribution-matching losses, such as forward KL,

$$D_{\mathrm{KL}}(q_t^\star \,\|\, p_\theta(\cdot \mid h_t)),$$

or symmetric objectives such as Jensen–Shannon divergence. However, the policy-gradient interpretation in our method is specific to the reverse-KL objective: the sampled-token update is driven by the reconstructed advantage

$$\widehat{A}_t(y_t) = A_t^G(y_t) + \lambda_t(y_t) J_t(y_t),$$

which is naturally evaluated on student-generated tokens and therefore remains on-policy. In contrast, forward KL places expectation under the reconstructed teacher $q_t^\star$, requiring full-vocabulary supervision or teacher-sampled tokens, and does not yield the same sampled-token advantage interpretation. As a result, while the reconstructed teacher $q_t^\star$ can be paired with other distillation losses, our main design and guarantees are most directly aligned with reverse-KL on-policy self-distillation. Exploring whether the same multi-view reconstruction improves forward-KL or symmetric distillation objectives is an interesting direction for future work.

C Implementation Details of Experiments

Training and Evaluation Configuration. We provide the detailed training hyperparameters for different methods in Table 3. Following prior work [Zhao et al., 2026, Li et al., 2026a], we disable the Qwen3 thinking mode during training. Prior work has also observed that Qwen models with thinking mode enabled tend to produce long, exploratory chains of thought, which can be harder to learn from [Kim et al., 2026]. Therefore, we leave this as an open problem for future work.

For evaluation, we use vLLM [Kwon et al., 2023] with the following settings for all models: temperature $= 0.6$, top-$p = 0.95$, top-$k = 20$, min-$p = 0$, and max tokens 38912.

Table 3: Training hyperparameters for GRPO and self-distillation methods.

Category	Parameter	GRPO	OPSD/AVSD
Training	Learning Rate	5×10^{-6}	5×10^{-6}
	Effective Batch Size	16	16
	Optimizer	AdamW	AdamW
	Gradient Clip Norm	0.1	0.1
	Training Steps	500	200
LoRA	LoRA Rank (r)	64	64
	LoRA Alpha (α)	128	128
	LoRA Target Modules	q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj	
Generation	Number of Rollouts	8	1
	Sampling Temperature	1.0	0.7
	Inference Engine	vLLM	vLLM

GPUs. Most of the experiments are run on 4 A6000 GPUs. Some experiments such as training for Qwen3-8B and GRPO baselines are run on 4 A100 GPUs.

D Additional Experiments

D.1 Privileged Information Analysis

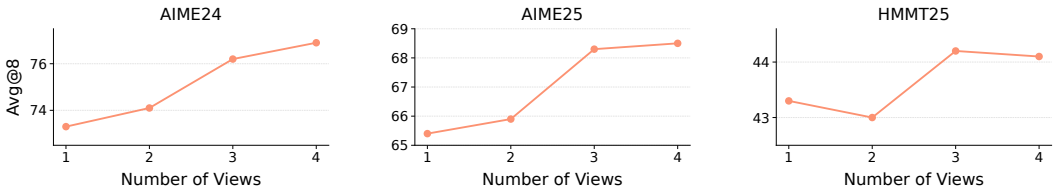

Figure 5: Scaling the number of privileged math views for AVSD on Qwen3-4B.

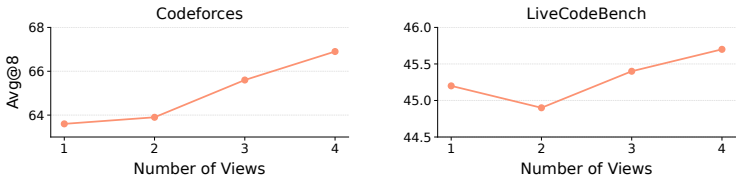

Figure 6: Scaling the number of privileged code views for AVSD on Qwen3-4B. Final performance generally improves as more complementary privileged views are added.

Scaling the Number of Views. We study how performance changes as we increase the number of privileged views used by AVSD. For math, we start from a single full-solution view and incrementally add a partial-solution view, the final-answer view used in our main experiments, and a fourth view that conditions the teacher on the model's own attempt together with the reference solution. As shown in Fig. 5, performance generally improves as the view family becomes more informative and diverse. On Qwen3-4B, moving from one to three math views improves Avg@8 by 2.9% on AIME24 and 2.8% on AIME25. Adding a fourth math view provides only a small additional gain on AIME24 and AIME25 (+0.7% and +0.2%), while remaining comparable on HMMT25, suggesting diminishing returns once the main sources of privileged information are covered.

We observe a similar trend on code, where we start from a single reference-implementation view and incrementally add complementary code views. Fig. 6 shows that moving from one to three views improves Codeforces by 2.0%. Unlike the math setting, the fourth code view continues to provide a

Table 4: Effect of privileged-view choice in code self-distillation. We report Avg@8 on Codeforces and LiveCodeBench with Qwen3-8B.

Method	View(s)	Codeforces	LiveCodeBench	Avg.
Base		60.9	47.6	54.3
OPSD	code	<u>63.2</u>	50.3	56.8
OPSD	hint	61.3	48.2	54.8
OPSD	feedback	62.8	<u>50.9</u>	<u>56.9</u>
AVSD (Ours)	all	**65.8**	**52.4**	**59.1**

Table 5: Robustness of AVSD to a misleading privileged view and to highly correlated views on Qwen3-4B.

Setting	AIME24	AIME25	HMMT25	Avg.
AVSD (misleading answer view)	75.4	67.9	**44.2**	62.5
AVSD (90%/95%/100% views)	74.6	65.4	43.3	61.1
AVSD (complementary views)	**76.2**	**68.3**	**44.2**	**62.9**

clearer benefit, with gains of 3.3% and 0.5% over the single-view setting, respectively. Overall, these results suggest that AVSD benefits from richer teacher families across both math and code: multiple views expose more stable cross-view signal for gated reconstruction, while additional views can still help when they provide complementary information.

Effect of Privileged-View Choice. In Fig. 1, we show that for math reasoning no single privileged view is uniformly optimal across benchmarks. We further examine whether the same phenomenon holds in the code domain on Qwen3-8B. As shown in Table 4, different code views lead to different strengths: the reference-implementation view performs best among single-view OPSD variants on Codeforces, while the feedback view performs best on LiveCodeBench and achieves the strongest single-view average. This confirms that the optimal privileged view is task-dependent in code as well. In contrast, AVSD combines all views and achieves the best performance on both benchmarks, improving over the strongest single-view OPSD baseline by 2.6% on Codeforces and 1.5% on LiveCodeBench. These results further support our motivation that multi-view self-distillation can construct a stronger and more robust learning signal than committing to any single privileged view.

Robustness to Misleading and Highly Correlated Views. We next test whether AVSD remains robust when the teacher family contains either unreliable or redundant privileged information under two different settings. For the *misleading-view* setting, we corrupt the answer-only view by replacing its correct final answer with a randomly sampled, different final answer from another training example, while leaving the full- and partial-solution views unchanged. For the *highly correlated-view* setting, we replace the complementary partial-solution and answer-only views with prefixes retaining 90% and 95% of the full solution, yielding three closely related views containing 90%, 95%, and 100% of the same solution.

Table 5 shows that AVSD is robust to misleading views. Corrupting the answer-only view reduces the average score by only 0.4% and leaves HMMT25 performance unchanged. This suggests that when one view conflicts with the remaining informative views, the residual gate can limit the influence of unreliable view-specific information. In contrast, replacing the complementary views with highly correlated solution prefixes reduces average performance by 1.8%. This result highlights that AVSD benefits most from genuinely complementary privileged views: highly redundant views provide less additional information and therefore reduce the benefit of multi-view aggregation.

D.2 AVSD Analysis

Gate Analysis. We analyze how often AVSD uses the residual pathway during training. Specifically, we report the percentage of token positions for which the gate is open, indicating that the method adds residual support from the arithmetic target beyond the geometric consensus signal. As shown in Table 6, the gate-open rate varies across models but remains consistently between 21% and 27%.

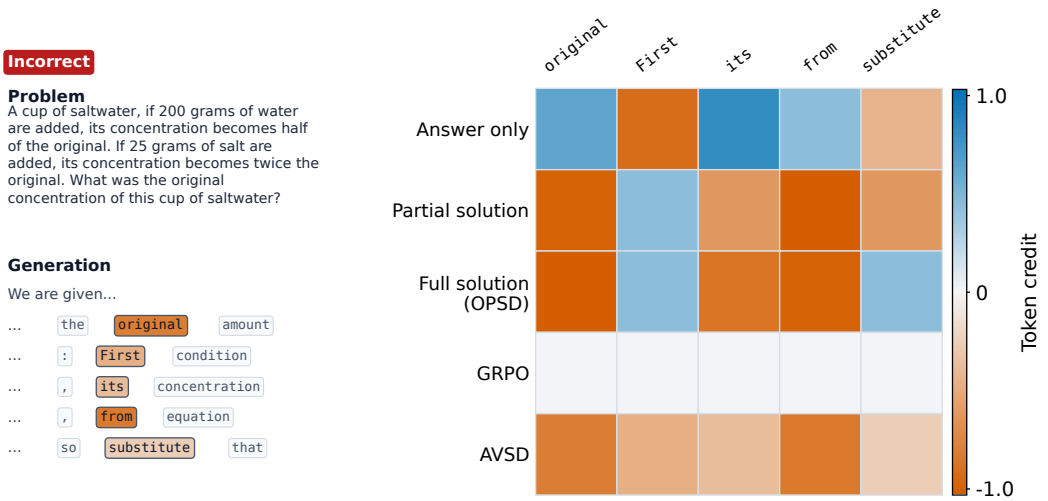

Figure 7: The full problem with generation trace for Fig. 4.

Model	Gate Open (%)
Qwen3-4B	24.6
Qwen3-8B	21.1
DeepSeek-R1-Distill-Qwen-7B	26.6

Table 6: Gate-open rate during training. We report the percentage of token positions for which the residual gate is open.

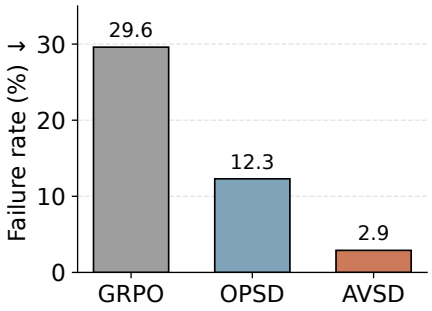

Figure 8: Quantitative reliability of token-level learning signals. For OPSD and AVSD, we report the percentage of top generated tokens with the largest absolute credit magnitude whose assigned credit has the incorrect sign. For GRPO, we report the percentage of cases with no learning signal, which occurs when no sampled rollout is correct. Lower is better.

This suggests that AVSD does not simply collapse to either the consensus or arithmetic target: it primarily relies on stable cross-view consensus, while selectively incorporating residual information for tokens where the multi-view signal is reliable.

Token-Level Credit Allocation. We provide supplementary token-level credit results for the main paper in Fig. 7 and Fig. 8. Fig. 7 expands the qualitative example from Fig. 4 by showing the full problem, the generated trace, and the token credits assigned by each method, showing the benefits of AVSD over baselines. Fig. 8 quantifies the reliability of the learning signal: GRPO provides no token-level signal in 29.6% of incorrect-rollout cases, OPSD assigns incorrect-sign credit to 12.3% of high-impact tokens, while AVSD reduces this rate to 2.9%.

D.3 Additional Baselines and Results

Multi-Seed Results. We further evaluate AVSD and OPSD, the strongest baseline in the Qwen3-4B setting, using five independent training seeds while keeping the training data, hyperparameters, and evaluation configuration fixed. Table 7 shows that AVSD consistently outperforms OPSD across all

Table 7: Multi-seed Avg@8 results on Qwen3-4B over five independent training seeds. We report mean ± standard deviation.

Method	AIME24	AIME25	HMMT25
OPSD	72.92 ± 0.66	65.50 ± 0.56	42.76 ± 0.76
AVSD (Ours)	$\mathbf{75.76 \pm 0.57}$	$\mathbf{68.04 \pm 0.58}$	$\mathbf{44.20 \pm 0.49}$

three benchmarks, with mean improvements of 2.84, 2.54, and 1.44% on AIME24, AIME25, and HMMT25, respectively. Moreover, AVSD outperforms OPSD in every paired seed across all three benchmarks, indicating that its gains are consistent across training runs rather than arising from a favorable single seed.

Confidence-based View-selection Baseline. We additionally compare against a parameter-free confidence-based view-selection baseline. At each generated token y_t, this baseline selects the view-conditioned teacher assigning the highest probability to that token,

$$m_t^\star = \arg\max_m q_t^{(m)}(y_t),$$

and performs standard single-view reverse-KL distillation using the selected teacher $q_t^{(m_t^\star)}$. AVSD outperforms confidence-based selection by 2.7% on average across AIME24, AIME25, HMMT25. This suggests that simply selecting the teacher assigning the highest probability to each generated token is insufficient: high confidence may reflect view-specific privileged information or overconfidence rather than a reliable learning signal. AVSD instead preserves cross-view consensus and only incorporates view-specific support when it is aligned with and proportionate to the consensus.

E Examples of Multi-View Privileged Information

Example of Multi-View Privileged Information in Math

Question. Real numbers a, b, c, d are such that $a + b = \frac{9}{c-d}$ and $c + d = \frac{25}{a-b}$. What is the minimum value that the quantity $a^2 + b^2 + c^2 + d^2$ can take?

Full Solution View Given the equations involving real numbers a, b, c, and d:

$$a + b = \frac{9}{c - d}$$
$$c + d = \frac{25}{a - b}$$

We aim to find the minimum value of $a^2 + b^2 + c^2 + d^2$.
1. **Multiply both given equations by their respective denominators**:

$$(a + b)(c - d) = 9$$

$$(c + d)(a - b) = 25$$

2. **Expand and rearrange the products**:

$$ac - ad + bc - bd = 9$$

$$ac - bc + ad - bd = 25$$

3. **Add the two equations to eliminate terms**:

$$2(ac - bd) = 34 \implies ac - bd = 17$$

4. **Establish a lower bound for $a^2 + b^2 + c^2 + d^2$**:
Since $a^2 + c^2 \geq 2ac$ (which follows from the AM-GM inequality $(a - c)^2 \geq 0$) and $b^2 + d^2 \geq -2bd$ (which follows from $(b + d)^2 \geq 0$):

$$a^2 + c^2 \geq 2ac$$

$$b^2 + d^2 \geq -2bd$$

Adding these inequalities yields:

$$a^2 + b^2 + c^2 + d^2 \geq 2(ac - bd)$$

Given $ac - bd = 17$:

$$a^2 + b^2 + c^2 + d^2 \geq 2 \times 17 = 34$$

5. **Check if the equality can be achieved**:
Equality in the inequalities $a^2 + c^2 \geq 2ac$ and $b^2 + d^2 \geq -2bd$ is achieved when $a = c$ and $b = -d$.
Substitute $a = c$ and $b = -d$ into the original equations. The first equation becomes:

$$a + b = \frac{9}{c - d} \implies a - d = \frac{9}{a - d} \implies (a - d)^2 = 9.$$

The second equation becomes:

$$c + d = \frac{25}{a - b} \implies a + d = \frac{25}{a + d} \implies (a + d)^2 = 25.$$

Taking positive roots gives $a - d = 3$ and $a + d = 5$, so $a = 4$ and $d = 1$. Hence $c = a = 4$ and $b = -d = -1$. Substituting back:

$$a^2 + b^2 + c^2 + d^2 = 4^2 + (-1)^2 + 4^2 + 1^2 = 16 + 1 + 16 + 1 = 34.$$

This confirms the minimum value is achievable.
Conclusion

$$\boxed{34}$$

Partial Solution View Given the equations involving real numbers a, b, c, and d:

$$a + b = \frac{9}{c - d}$$
$$c + d = \frac{25}{a - b}$$

We aim to find the minimum value of $a^2 + b^2 + c^2 + d^2$.
1. **Multiply both given equations by their respective denominators**:

$$(a + b)(c - d) = 9$$

$$(c + d)(a - b) = 25$$

2. **Expand and rearrange the products**:

$$ac - ad + bc - bd = 9$$

$$ac - bc + ad - bd = 25$$

Final Answer View

$$\boxed{34}$$

Example of Multi-View Privileged Information in Math

Question. In a company, employees have a combined monthly salary of \$10,000. A kind manager proposes to triple the salaries of those earning up to \$500, and to increase the salaries

of others by \$1,000, resulting in a total salary of \$24,000. A strict manager proposes to reduce the salaries of those earning more than \$500 to \$500, while leaving others' salaries unchanged. What will the total salary be in this case?

Full Solution View

1. Let us denote by x the total number of employees whose monthly salary is less than or equal to 500 dollars.
2. Let y be the total number of employees whose monthly salary is more than 500 dollars.
3. The firm's total monthly salary expenditure before any changes is 10,000 dollars. This can be written as:

$$500x + \sum_{i=1}^{y} s_i = 10,000$$

where s_i represents the individual salaries of the employees earning more than 500 dollars.
4. The kind manager proposes the following changes: - Tripling the salaries of employees earning 500 dollars or less. - Increasing the salaries of employees earning more than 500 dollars by 1000 dollars each.
The new total salary expenditure proposed by the kind manager is 24,000 dollars. This can be expressed as:

$$3 \cdot 500x + \sum_{i=1}^{y} (s_i + 1000) = 24000$$

5. Simplifying the equation:

$$1500x + \sum_{i=1}^{y} s_i + 1000y = 24000$$

6. Since we know that $\sum_{i=1}^{y} s_i = 10000 - 500x$ from the original total monthly salary equation, substituting this we get:

$$1500x + (10000 - 500x) + 1000y = 24000$$

7. Simplifying further, we obtain:

$$1000x + 1000y = 14000 \implies x + y = 14$$

8. The cruel manager proposes the following changes: - Reducing the salaries of employees earning more than 500 dollars to 500 dollars each. - Keeping the salaries of employees earning 500 dollars or less unchanged.
The new total salary can be calculated as:

$$500x + 500y = 500(x + y)$$

9. From step 7, we know that $x + y = 14$, so substituting this into the equation for the total salary proposed by the cruel manager:

$$500 \cdot 14 = 7000$$

10. Conclusion: The new total salary expenditure after the cruel manager's proposed changes will be:

$$\boxed{7000}$$

Partial Solution View

1. Let us denote by x the total number of employees whose monthly salary is less than or equal to 500 dollars.
2. Let y be the total number of employees whose monthly salary is more than 500 dollars.
3. The firm's total monthly salary expenditure before any changes is 10,000 dollars. This can be written as:

$$500x + \sum_{i=1}^{y} s_i = 10,000$$

where s_i represents the individual salaries of the employees earning more than 500 dollars.

Final Answer View

$$\boxed{7000}$$

Example of Multi-View Privileged Information in Code

Question.
Given n sticks with lengths a_i, find the maximum number of regular equal-sided polygons you can construct. Each side uses one stick, and no stick can be reused.

Reference Solution View

```
import sys
from collections import Counter

t = int(sys.stdin.readline())
for _ in range(t):
    n = int(sys.stdin.readline())
    a = list(map(int, sys.stdin.readline().split()))
    cnt = Counter(a)
    total = 0
    for v in cnt.values():
        total += v // 3
    print(total)
```

Hint View
Greedily form triangles from sticks of the same length. For S sticks of one length, you can create $\lfloor S/3 \rfloor$ polygons. Sum this over all stick lengths.

Feedback View
Wrong student rollout:

```
import sys
input=sys.stdin.readline
t=int(input())
for _ in range(t):
    n=int(input())
    a=list(map(int,input().split()))
    print(0)
```

Feedback: The student's solution failed on a test case.
Failing input:

```
4
1
1
2
1 1
6
2 2 3 3 3 3
9
4 2 2 2 2 4 2 4 4
```

Expected output:

```
0
0
1
2
```

Student output:

```
0
0
0
0
```

Example of Multi-View Privileged Information in Code

Question.
$S(x)$ is the sum of digits of x. Count integers x with $1 \le x \le n$ such that

$$S(x+1) < S(x).$$

Reference Solution View.

```
t = int(input())
for _ in range(t):
    n = int(input())
    print((n + 1) // 10)
```

Hint View.
A number is interesting exactly when it ends in 9. Count numbers $\le n$ whose last digit is 9, which is $\lfloor (n+1)/10 \rfloor$.

Feedback View.
Wrong student rollout:

```
t=int(input())
for _ in range(t):
    n=int(input())
    print(n//10)
```

Feedback: The student's solution failed on a test case.
Failing input:

```
5
1
9
10
34
880055535
```

Expected output:

```
0
1
1
3
88005553
```

Student output:

```
0
0
1
3
88005553
```

F Prompts

Teacher Prompt

```
Problem: {problem}

Here is training-time reference information for this problem. It
may contain a final answer, a hint, a partial solution, or a full
solution to the problem.

=== Privileged Reference ({view_type}) Begin ===
```

```
{privileged information}
=== Privileged Reference ({view_type}) End ===

Use the reference only as hidden guidance.  Use it only to understand,
verify, or correct your solution.
Do not mention, quote, copy, cite, or refer to the privileged
reference or its existence.
Now solve the original problem independently using your own
reasoning.
Please reason step by step, and put your final answer within
\boxed{}.
```

Student Prompt

```
Problem:  {problem}

Please reason step by step, and put your final answer within
\boxed{}.
```

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main contribution as multi-view self-distillation with gated reconstruction, and our claims are supported by the theoretical decomposition, experiments, and analyses in the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We include a Limitations section in Appendix A of the paper.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: We state the main results and intuition in the method section, with additional detailed derivations in the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We describe the training data, models, baselines, evaluation benchmarks, and other implementation details in Section 3.1 and Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We use publicly available datasets and models, and provide code in the supplementary material.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: We describe the training data, models, baselines, evaluation benchmarks, and other implementation details in Section 3.1 and Appendix C.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We report Avg@8 for the evaluation throughout the paper, and include multiple training seed error bars in Appendix D.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the GPUs setup in Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and believe our work conforms to its requirements.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We include a Broader Impact section in Appendix A.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not include a discussion of safeguards because we do not release any models or datasets as part of this work.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We explicitly credit the use of datasets and models in the appendix and references section of the paper.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We do not release any new assets in this paper.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: We do not have any crowdsourcing experiments or research with human subjects in this paper.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human subjects are involved in this research.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We do not use LLM for the core method development.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.