Beyond Top-k: Mass-Aware Support Sampling for On-Policy Distillation

Anonymous ACL submission

Abstract

On-policy distillation reduces train–inference mismatch by supervising student-generated trajectories, but full-vocabulary token-level distribution matching remains expensive for large language models. A common approximation truncates reverse-KL distillation to the student's top-k tokens, which preserves efficiency but drops tail-token supervision and ignores support-level mass mismatch between the student and teacher. We analyze these approximation gaps and propose Stochastic Support Sampling with Teacher Mass Reweighting, which samples fixed-size supports from the student distribution and uses teacher support mass as a reliability weight for the conditional distillation loss. Under the same support budget, our method outperforms the strongest baseline by $+1.4$ hard-math average points on Qwen3-8B→Qwen3-1.7B-Base, and by $+1.1$ math average points and $+2.8$ code average points on DeepSeek-R1-Distill-Llama-8B→Meta-Llama-3.1-8B-Base.

1 Introduction

Knowledge distillation (KD) (Hinton et al., 2015) is a widely adopted model compression technique that transfers the knowledge of a large teacher model into a lightweight student model, enabling efficient deployment without significant performance degradation. A central recent paradigm for distilling autoregressive language models is **on-policy distillation (OPD)**, introduced by GKD (Agarwal et al., 2024), which reduces train–inference mismatch by letting the teacher supervise trajectories sampled from the student itself rather than only fixed data or teacher-generated prefixes.

Token-level distribution matching under OPD is typically instantiated in two forms. The first uses a REINFORCE-style objective that supervises only the tokens appearing in student rollouts, treating teacher feedback on those on-policy trajectories as the training signal (Gu et al., 2024). The second directly aligns the *full token-level vocabulary distribution* at each decoding step on student-generated prefixes, optimizing a token-wise divergence such as the reverse KL (RKL) divergence between the student and teacher distributions (Agarwal et al., 2024). Compared with sampled-token OPD, distribution-level objectives provide denser local supervision and have been reported to improve optimization stability and downstream performance (Agarwal et al., 2024; Fu et al., 2026); however, materializing the full-vocabulary version incurs memory and computational overhead proportional to the vocabulary size $|V|$, which reaches 151,936 tokens in Qwen3 (Qwen Team, 2025).

To mitigate this cost, a direct approximation is **student top-k truncated RKL**: only the top-k tokens under the student distribution are retained to form the support set, and RKL distillation is performed on the renormalized distributions over this subset (Li et al., 2026). Despite its efficiency, we identify two fundamental problems that cause top-k truncated RKL to diverge from the full-vocabulary RKL objective it aims to approximate.

Tail Token Neglect. By construction, the top-k support is dominated by the student's highest-probability tokens, while low-probability *tail tokens* are systematically excluded from training. During inference, however, non-greedy decoding can still visit tokens outside the truncated support. Because these tokens receive little or no distillation signal at prefixes where they lie outside the top-k support, the student's predictions in this region can remain poorly calibrated relative to the teacher, creating an *exposure-bias*-like mismatch between the truncated training support and the test-time decoding distribution.

Support Mass Discrepancy. Because the support is selected from the student's top-k tokens, it is designed to retain high student probability mass but provides no guarantee of retaining sufficient

teacher probability mass. After renormalization, however, both conditional distributions assign total mass one to the same support. Consequently, the truncated objective may match the teacher's relative preferences inside the support while ignoring how much teacher probability mass lies outside it. This omitted mass information matters because low teacher support mass makes the renormalized conditional teacher distribution a less reliable proxy for the full-vocabulary teacher distribution.

To address these two problems, we propose two complementary methods. First, we replace deterministic top-k truncation with **stochastic support sampling (SSS)**: we draw k tokens *without replacement* from the student distribution to construct the support set at each step. This preserves high expected coverage of head tokens while endowing tail tokens with a non-zero probability of inclusion, enabling progressive tail-token alignment as training proceeds. Second, we introduce **teacher mass reweighting (TMR)**: we weight the conditional RKL loss by a support-level function of the teacher's full-vocabulary mass over the sampled support set. Since the omitted mass term indicates whether the selected support captures a meaningful portion of the teacher distribution, TMR uses teacher support mass as a reliability signal rather than as a direct mass-matching target.

Our main contributions are:

- We provide a systematic analysis of two failure modes of top-k truncated RKL—tail token neglect and support mass discrepancy.
- We propose two complementary components, Stochastic Support Sampling (SSS) and Teacher Mass Reweighting (TMR), to improve tail-token coverage and use teacher support mass as a reliability signal for conditional distillation.
- We show that, under the same support budget, SSS+TMR consistently improves over the strongest baseline, with gains of +1.4 hard-math average points on Qwen3-8B→Qwen3-1.7B-Base and +1.1 math / +2.8 code average points on DeepSeek-R1-Distill-Llama-8B→Meta-Llama-3.1-8B-Base.

2 Motivation

2.1 Preliminaries

Given a context $\mathbf{x}$, let $q(\cdot|\mathbf{x})$ denote the teacher's token distribution, and let $p_\theta(\cdot|\mathbf{x})$ denote the student's distribution over the vocabulary V. We omit the conditioning on $\mathbf{x}$ for brevity. The **full-vocabulary reverse KL** (RKL) objective is:

$$\mathcal{L}_{\text{RKL}} = D_{\text{KL}}(p_\theta \| q) = \sum_{v \in V} p_\theta(v) \log \frac{p_\theta(v)}{q(v)}. \quad (1)$$

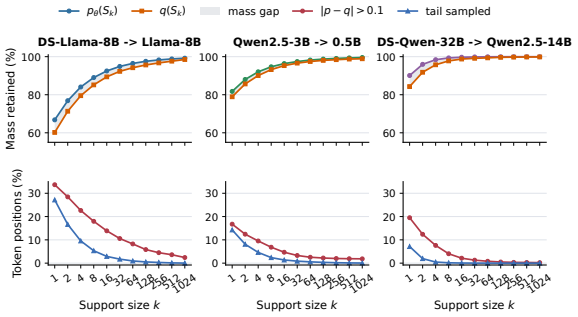

Figure 1: Empirical approximation gap of student top-k supports. Top row: the deterministic student top-k support captures high student mass $p_\theta(S_k)$, but consistently lower teacher mass $q(S_k)$; the shaded region marks the support-mass gap. Bottom row: a non-trivial fraction of positions still exhibit large mass discrepancy, and sampled next tokens can fall outside the deterministic top-k support.

Top-k OPD. For any distribution p over V and any subset $S \subseteq V$, define $p(S) = \sum_{j \in S} p(j)$ as the probability mass assigned to S. Let $S_k \subseteq V$ be the set of k tokens with the highest probabilities under the student distribution p_θ. Top-k OPD does not match the full distributions p_θ and q over V; instead, it matches their *conditional*, or relative, distributions on S_k:

$$\mathcal{L}_{\text{rel}}(S_k) = D_{\text{KL}}\Big(p_\theta(\cdot \mid S_k) \,\Big\|\, q(\cdot \mid S_k)\Big), \quad (2)$$

where, for $i \in S_k$,

$$p_\theta(i \mid S_k) = \frac{p_\theta(i)}{p_\theta(S_k)}, \quad q(i \mid S_k) = \frac{q(i)}{q(S_k)}. \quad (3)$$

Property: Top-k OPD matches conditional distributions rather than absolute probability mass. We provide a proof in Appendix A. For a fixed selected support S_k, the optimum of the truncated-support objective only constrains *relative* probabilities within S_k:

$$\frac{p_\theta(i)}{p_\theta(j)} = \frac{q(i)}{q(j)}, \qquad i, j \in S_k, \quad (4)$$

but it does not explicitly constrain the total probability mass assigned to the support. In particular, the objective does not require $p_\theta(S_k) = q(S_k)$.

2.2 Decomposing the Approximation Gap

To formally characterize the gap between $\mathcal{L}_{\text{rel}}(S_k)$ and $\mathcal{L}_{\text{RKL}}$, we apply the following decomposition, derived in Appendix B. For any partition of V into S and $\bar{S} = V \setminus S$:

$$D_{\text{KL}}(p_\theta \| q) = H(S) + T(S) + M(S), \quad (5)$$

where $H(S)$, $T(S)$, and $M(S)$ denote the head conditional, tail conditional, and head/tail mass terms:

$$\begin{aligned} H(S) &= p_\theta(S)\, D_{\text{KL}}(p_\theta(\cdot \mid S) \,\|\, q(\cdot \mid S))\,, \\ T(S) &= p_\theta(\bar{S})\, D_{\text{KL}}\big(p_\theta(\cdot \mid \bar{S}) \,\|\, q(\cdot \mid \bar{S})\big)\,, \\ M(S) &= D_{\text{KL}}\big(\text{Bern}(p_\theta(S)) \,\|\, \text{Bern}(q(S))\big). \end{aligned}$$

Standard top-k OPD keeps only the normalized conditional divergence inside the head group S_k. By Eq. (4), this matches only the relative distribution within S_k.

Tail Token Neglect. The second term in Eq. (5), $p_\theta(\bar{S}_k)\, D_{\text{KL}}(p_\theta(\cdot|\bar{S}_k) \| q(\cdot|\bar{S}_k))$, is dropped entirely. Tokens outside S_k—the *tail*—receive zero gradient regardless of how miscalibrated they are relative to the teacher. This creates a persistent blind spot at each prefix: tokens excluded from the current support are not corrected by the truncated distillation loss, yet the student may still generate them at inference time.

Support Mass Discrepancy. The term $D_{\text{KL}}(\text{Bern}(p_\theta(S_k)) \| \text{Bern}(q(S_k)))$ is also omitted from the truncated objective. This Bernoulli KL is the only term in the decomposition that compares how much total probability mass the student and teacher assign to S_k; it is minimized only when $p_\theta(S_k) = q(S_k)$ and grows as the two masses separate. We analyze this term in Appendix C. More importantly for truncated conditional distillation, if $q(S_k)$ is small then $q(\cdot \mid S_k)$ is formed by renormalizing a low-mass teacher region into a full distribution, so the conditional comparison can become over-confident about a support the teacher itself considers unimportant. As a result, the objective may match relative probabilities inside S_k without explicitly matching the total probability mass assigned to S_k or reflecting how reliable that support is as a proxy for the teacher's full distribution. We provide the derivation in Appendix D.

2.3 Empirical Evidence of the Approximation Gap

We further measure whether the two omitted terms in Eq. (5) are observable in real teacher–student distributions. For each decoding position, we construct S_k from the student's top-k probabilities and record the student support mass $p_\theta(S_k)$, the teacher support mass $q(S_k)$, the fraction of positions where $|p_\theta(S_k) - q(S_k)| > 0.1$, and the fraction of sampled next tokens that fall outside S_k. We run this diagnostic on AIME24 (Mathematical Association of America, 2024), setting temperature to 1.0 and top-p sampling with $p = 0.8$, covering three representative student scales: around 1B parameters, 7B/8B parameters, and 14B parameters. The corresponding teacher→student pairs are Qwen2.5-3B→Qwen2.5-0.5B (Yang et al., 2024), DeepSeek-R1-Distill-Llama-8B→Meta-Llama-3.1-8B-Base (DeepSeek-AI, 2025; Llama Team, 2024), and DeepSeek-R1-Distill-Qwen-32B→Qwen2.5-14B (DeepSeek-AI, 2025; Yang et al., 2024).

Figure 1 answers two empirical questions. **First, are tokens outside the deterministic student top-k support actually visited?** The bottom row shows that they are, especially when the student distribution is less concentrated or the support budget is small. For example, under DeepSeek-R1-Distill-Llama-8B→Meta-Llama-3.1-8B-Base, 0.94% of sampled tokens fall outside the student top-64 support, and the rate remains non-zero for larger supports. Even a small per-token rate can accumulate over long reasoning trajectories: a tail rate r yields about rT tail tokens in a length-T response. Thus, $r = 0.94\%$ corresponds to roughly 38 tail tokens in 4K tokens and 75 in 8K tokens. Once sampled, these unsupervised tokens enter the prefix and can influence subsequent decoding states. This tail visitation is also present for the other two student scales.

Second, does the student top-k support differ systematically from the teacher distribution? Because the support is selected as the student's top-k, the top row shows that it consistently retains more student mass than teacher mass. For DeepSeek-R1-Distill-Llama-8B→Meta-Llama-3.1-8B-Base, the student top-256 support contains 98.3% of the student mass but only 96.7% of the teacher mass, leaving 3.3% of teacher probability outside the truncated objective. The bottom row further shows that 4.5% of token positions still

have a support-mass mismatch larger than 0.1 at $k = 256$. We observe the same qualitative pattern at the other student scales, although the magnitude varies with model scale and teacher–student alignment. Thus, both tail visitation and support-mass discrepancy appear in realistic settings, with the student-selected top-k support often retaining lower teacher mass than student mass.

Together, these two failure modes motivate our proposed methods, SSS and TMR, which address tail token neglect and support mass discrepancy, respectively. SSS determines which tokens can be observed by truncated distillation, while TMR determines how strongly a sampled support should be trusted once it is observed.

3 Method

We propose **Stochastic Support Sampling with Teacher Mass Reweighting** for efficient OPD distillation. At each decoding position, our method constructs a small support set from the student distribution, computes a conditional RKL loss on this support, and reweights the loss by the teacher probability mass covered by the support. As shown in Figure 2, the pipeline first uses SSS to replace deterministic top-k support selection with stochastic student-support sampling, and then uses TMR to modulate the conditional distillation loss according to the selected support's teacher-mass reliability. This retains the efficiency of top-k truncated RKL while reducing the deterministic blind spots induced by top-k truncation.

3.1 Stochastic Support Sampling

Let $p_\theta(\cdot|\mathbf{x})$ be the student distribution at context $\mathbf{x}$. Instead of selecting the deterministic top-k support $S_k = \mathrm{TopK}(p_\theta)$, **Stochastic Support Sampling** (SSS) samples a support set S of size k without replacement from p_θ:

$$S \sim \mathrm{SampleK}(p_\theta(\cdot|\mathbf{x}), k)\,. \tag{6}$$

Unlike deterministic top-k, SSS gives every token with non-zero student probability a non-zero chance of being selected. Thus, high-probability tokens are still sampled frequently, while tail tokens can enter the support across training steps and receive distillation gradients. The benefit is not per-step deep-tail coverage but the progressive removal of a hard exclusion boundary: near-cutoff tokens accumulate gradient signal over training rather than being permanently excluded by deterministic truncation.

Given a sampled support S, we form the conditional student and teacher distributions:

$$p_\theta(i \mid S, \mathbf{x}) = \frac{p_\theta(i|\mathbf{x})}{p_\theta(S|\mathbf{x})},$$
$$q(i \mid S, \mathbf{x}) = \frac{q(i|\mathbf{x})}{q(S|\mathbf{x})}, \qquad i \in S. \tag{7}$$

The SSS distillation loss is the conditional reverse KL on the sampled support:

$$\mathcal{L}_{\mathrm{SSS}}(\mathbf{x}, S) = D_{\mathrm{KL}}(p_\theta(\cdot \mid S, \mathbf{x}) \| q(\cdot \mid S, \mathbf{x}))\,. \tag{8}$$

This objective preserves the relative-distribution learning behavior of top-k truncated RKL, while breaking the fixed-support constraint imposed by deterministic top-k selection.

3.2 Teacher Mass Reweighting

SSS improves support coverage, but it still optimizes a conditional distribution-matching objective on the sampled support. For a sampled support S, the teacher distribution used by the loss is

$$q(i \mid S, \mathbf{x}) = \frac{q(i \mid \mathbf{x})}{q(S \mid \mathbf{x})}, \qquad i \in S. \tag{9}$$

Therefore, if the teacher assigns only a small full-vocabulary probability mass to S, the conditional teacher distribution is obtained by renormalizing a low-mass region into a full distribution. In this case, conditional RKL can over-emphasize a local comparison that is only weakly supported by the teacher's original distribution.

A natural alternative would be to directly optimize the Bernoulli support-mass term from Eq. (5) so that $p_\theta(S)$ matches $q(S)$. However, this term explicitly reallocates student probability mass between S and its complement $V \setminus S$. As shown in Section 2, the teacher support mass $q(S)$ is systematically lower than the student support mass $p_\theta(S)$, meaning the teacher places relatively more mass on $V \setminus S$. In a truncated-support setting, however, token-level teacher preferences on $V \setminus S$ are not observed, so directly optimizing this mass term can push probability into an unobserved complement without teacher guidance. We therefore use teacher support mass not as a direct matching target, but as a reliability signal for the conditional distillation loss. Let

$$P_T(S, \mathbf{x}) = \sum_{i \in S} q(i|\mathbf{x}). \tag{10}$$

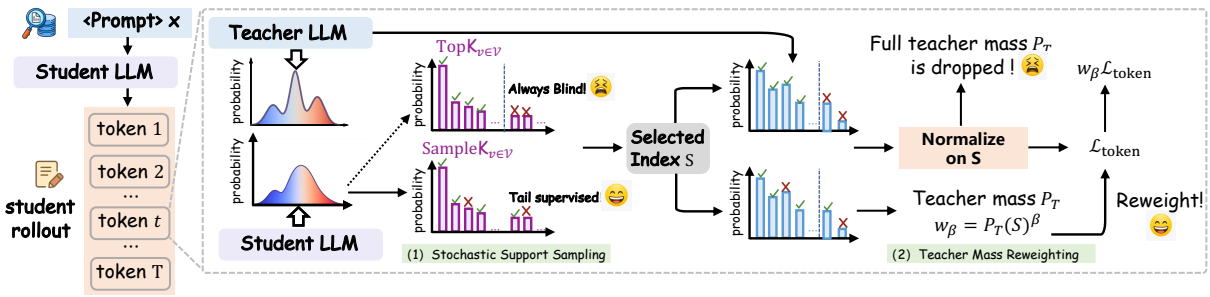

Figure 2: Overview of SSS+TMR. Given a student rollout, SSS samples a stochastic support from the student distribution to expose tail tokens, while TMR uses teacher support mass as a reliability weight for the conditional distillation loss.

We define the weighting function

$$w_\beta(P_T) = P_T^\beta, \qquad \beta \geq 0, \tag{11}$$

where β controls how strongly low-teacher-mass supports are down-weighted. The case $\beta = 0$ recovers the original unweighted conditional OPD objective, $\beta = 1$ gives direct teacher-support-mass weighting, $0 < \beta < 1$ gives a milder discount, and $\beta > 1$ gives a more aggressive discount. The resulting objective is

$$\mathcal{L}_{\text{TMR}} = w_\beta(P_T)\mathcal{L}_{\text{SSS}}(\mathbf{x}, S). \tag{12}$$

The weight modulates the OPD gradient magnitude according to how much teacher probability mass falls on the sampled support. If $P_T \approx 1$, the teacher regards the sampled support as important and the weight is close to one; if $P_T \ll 1$, the support lies in a low-probability teacher region and its conditional OPD gradient is suppressed. This is appropriate because near-zero teacher mass implies $q(\cdot|S)$ is renormalized from a low-probability region, making it an unreliable reference whose gradient direction cannot be trusted. Thus, TMR does not ask the student to move probability mass according to an incompletely observed complement distribution; instead, it trusts conditional distillation only when the sampled support retains sufficient teacher probability mass.

Note that $P_T(S, \mathbf{x})$ is computed by summing the teacher's original full-vocabulary probabilities over the selected support set S, before support-level renormalization. In practice, once the teacher logits or log-probabilities are available from the standard teacher forward pass, we only gather the probabilities of the k selected support tokens and sum them to obtain $P_T(S, \mathbf{x})$ (see Algorithm 1 in Appendix F).

Qwen3-8B → Qwen3-1.7B-Base									
Method	**AIME24**		**AIME25**		**AMC23**		**Minerva Math**		**Avg.**
	Avg@16	**Pass@16**	**Avg@16**	**Pass@16**	**Avg@16**	**Pass@16**	**Avg@4**	**Pass@4**	
Base	11.9	<u>36.7</u>	10.8	26.7	47.0	<u>87.5</u>	24.8	37.1	23.6
Teacher	40.6	55.7	33.9	46.9	85.7	94.0	50.2	58.1	52.6
OPD	10.6	26.7	9.8	26.7	46.6	**90.0**	<u>26.3</u>	38.6	23.3
vOPD	13.1	<u>36.7</u>	10.8	26.7	45.2	<u>87.5</u>	25.3	<u>39.0</u>	23.6
OPD$_{\text{top-}k}$	14.6	<u>36.7</u>	12.1	36.7	46.9	**90.0**	25.1	35.7	24.7
Revisiting OPD	14.8	**43.3**	<u>12.5</u>	**43.3**	<u>48.9</u>	<u>87.5</u>	24.3	36.0	<u>25.1</u>
Ours	**16.5**	**43.3**	**12.7**	<u>40.0</u>	**50.2**	**90.0**	**26.5**	**40.1**	**26.5**

Table 1: Hard math results for Qwen3-8B → Qwen3-1.7B-Base. **Bold**/<u>underline</u>: best/second best among non-teacher methods.

Appendix C provides the full derivation of the Bernoulli support-mass term and a more detailed explanation of why we do not optimize it directly.

3.3 Training Objective

For each generated sequence, let $\mathbf{x}_t$ denote the prefix context before predicting the token at position t. At each decoding position, we sample a support S_t using SSS and apply TMR to the conditional RKL loss. The final distillation objective is

$$\mathcal{L}_{\text{distill}} = \sum_t w_\beta(P_T(S_t, \mathbf{x}_t))\mathcal{L}_{\text{SSS}}(\mathbf{x}_t, S_t),$$
$$S_t \sim \text{SampleK}\left(p_\theta(\cdot|\mathbf{x}_t), k\right). \tag{13}$$

The training procedure is summarized in Appendix F.

4 Experiments

We evaluate SSS+TMR under the same fixed support budget as top-k OPD, so all truncated-support methods use the same number of selected tokens per decoding position. Our experiments are designed to answer four questions: (1) whether SSS+TMR improves downstream math and code performance over existing OPD baselines, (2) how much each component contributes, (3) how sensitive the method is to the support size k and teacher-mass exponent β, and (4) whether the additional

DeepSeek-R1-Distill-Llama-8B → Meta-Llama-3.1-8B-Base															
Method	**GSM8K**		**MATH500**		**ASDiv**		**Math Avg.**	**MBPP**		**HumanEval**		**BigCodeBench**		**Code Avg.**
	Avg@4	**Pass@4**	**Avg@4**	**Pass@4**	**Avg@4**	**Pass@4**		**Avg@4**	**Pass@4**	**Avg@4**	**Pass@4**	**Avg@4**	**Pass@4**	
Base	9.3	30.3	5.1	17.0	15.0	44.9	9.8	15.3	40.1	0.3	1.8	18.0	18.0	11.2
Teacher	80.2	93.5	79.7	90.8	88.5	95.8	82.8	74.1	88.3	75.5	90.2	68.4	88.7	72.7
OPD	3.3	11.6	3.0	9.8	5.3	18.1	3.9	17.4	48.6	2.3	9.1	22.8	55.8	14.2
vOPD	6.4	22.4	3.8	12.4	12.9	39.9	7.7	15.1	43.2	1.7	6.1	16.1	42.5	11.0
$\text{OPD}_{\text{top-}k}$	13.7	38.5	8.8	22.8	23.1	52.6	15.2	<u>29.2</u>	55.3	5.6	15.2	<u>35.6</u>	<u>65.8</u>	<u>23.5</u>
Revisiting OPD	<u>28.0</u>	**68.8**	<u>13.4</u>	<u>33.2</u>	<u>27.3</u>	<u>60.7</u>	<u>22.9</u>	25.4	<u>58.0</u>	**9.5**	**25.7**	27.1	48.0	20.7
Ours	**29.3**	<u>64.8</u>	**14.4**	**34.5**	**28.2**	**61.1**	**24.0**	**30.1**	**61.1**	<u>8.8</u>	<u>23.2</u>	**40.0**	**67.8**	**26.3**

Table 2: Math and code results for DeepSeek-R1-Distill-Llama-8B → Meta-Llama-3.1-8B-Base. **Bold**/<u>underline</u>: best/second best among non-teacher methods.

sampling and reweighting keep training overhead reasonable.

4.1 Experimental Setup

Models. We evaluate SSS+TMR on two teacher–student pairs across math and code settings. Specifically, we use Qwen3-8B as the teacher for Qwen3-1.7B-Base (Qwen Team, 2025) and DeepSeek-R1-Distill-Llama-8B as the teacher for Meta-Llama-3.1-8B-Base (DeepSeek-AI, 2025; Llama Team, 2024). For math, we evaluate both teacher–student pairs; for code, we additionally evaluate DeepSeek-R1-Distill-Llama-8B as the teacher for Meta-Llama-3.1-8B-Base. Each teacher–student pair shares the same tokenizer, which allows token-level OPD losses to be computed directly without vocabulary projection. The two pairs are chosen to cover a within-family (Qwen3-8B, Qwen3-1.7B-Base) and a cross-family same-vocabulary (DeepSeek-R1-Distill-Llama-8B, Meta-Llama-3.1-8B-Base) setting, spanning different model scales and architectural families.

Datasets. For training, we use DAPO-Math-17K (Yu et al., 2025) for the math domain and sample 17K examples from EpiCoder-func-380K (Wang et al., 2025) for the code domain. For evaluation, we use two groups of math benchmarks. For stronger base models such as Qwen3-1.7B-Base, we evaluate on harder math benchmarks, including AIME24 (Mathematical Association of America, 2024), AIME25 (Mathematical Association of America, 2025), AMC23 (Mathematical Association of America, 2023), and Minerva Math (Lewkowycz et al., 2022). For weaker base models such as Meta-Llama-3.1-8B-Base, we evaluate on simpler math benchmarks, including GSM8K (Cobbe et al., 2021), MATH500 (Lightman et al., 2024), and ASDiv (Miao et al., 2020). For code, we evaluate on MBPP (Austin et al., 2021), HumanEval (Chen et al., 2021), and BigCodeBench (Zhuo et al., 2025) to test whether our method also brings gains beyond the math domain. Detailed descriptions of all datasets are provided in Appendix H.

Baselines. We compare against the following baselines. **Base** denotes the initial student model before distillation. **Teacher** denotes the corresponding teacher model. **OPD** computes the distillation loss on sampled student tokens. **vOPD** uses a control-variate baseline for on-policy distillation (Oh et al., 2026). **Revisiting OPD** denotes the Fu et al. (2026) recipe that replaces sampled-token matching with teacher-top-k local support matching. $\textbf{OPD}_{\textbf{top-}k}$ computes the RKL loss on the student's top-k support. **Ours** is SSS+TMR with $\beta = 1$. For a fair comparison, all truncated-support methods use the same support size $k = 256$.

Detailed training and evaluation settings are provided in Appendix E, Table 5.

4.2 Main Results

Tables 1 and 2 summarize the main downstream results. Under the same support size k, Ours consistently achieves the best average scores across both settings, outperforming both $\text{OPD}_{\text{top-}k}$ (student-top-k support) and Revisiting OPD (teacher-top-k support).

For Qwen3-1.7B-Base, Ours achieves a hard-math average of 26.5, improving over $\text{OPD}_{\text{top-}k}$ (24.7) and Revisiting OPD (25.1). Ours obtains the best average-accuracy score on all four benchmarks. While Revisiting OPD matches Ours on AIME24 Pass@16 (43.3) and achieves higher Pass@16 on AIME25 (43.3 vs. 40.0), its Avg@16 scores are lower. Overall, Ours improves the average single-sample accuracy across the hard-math benchmarks while remaining competitive on best-of-k success rates.

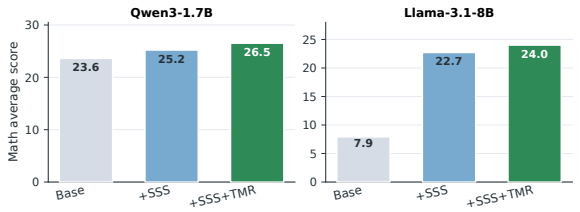

Figure 3: Ablation study on Qwen3-1.7B-Base and Meta-Llama-3.1-8B-Base. Each panel reports the math average score after progressively adding SSS and TMR.

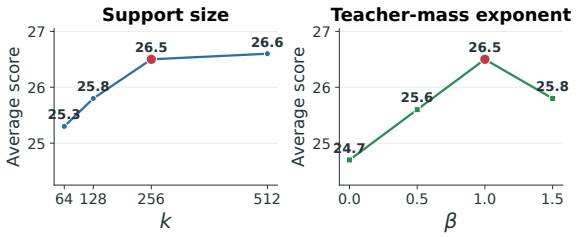

Figure 4: Sensitivity of SSS+TMR to support size k and teacher-mass exponent β. Red markers indicate the default hyperparameters used in our main experiments.

For Meta-Llama-3.1-8B-Base, the gains are larger. Ours improves the math average from 15.2 ($\text{OPD}_{\text{top-}k}$) and 22.9 (Revisiting OPD) to 24.0, and achieves the best code average of 26.3 compared to 23.5 ($\text{OPD}_{\text{top-}k}$) and 20.7 (Revisiting OPD). On math, the largest improvements appear on GSM8K, where Avg@4 increases from 13.7 ($\text{OPD}_{\text{top-}k}$) to 29.3. On code, Revisiting OPD is slightly stronger on HumanEval Avg@4 (9.5 vs. 8.8), but its BigCodeBench score (27.1) falls below $\text{OPD}_{\text{top-}k}$ (35.6) and Ours (40.0), suggesting that its gains are benchmark-specific rather than broadly transferable. Ours achieves the best aggregate results and remains competitive across individual benchmarks. Taken together, these results show that SSS+TMR provides more consistent improvements than either student-top-k or teacher-top-k support construction, and transfers to both math and code domains. We further provide qualitative case studies in Appendix G.

4.3 Ablation Study

We ablate SSS and TMR in Figure 3. For each model, we start from the base student, then add stochastic support sampling, and finally add teacher mass reweighting. This isolates the contribution of support construction from the contribution of support-level reliability weighting. On Qwen3-1.7B-Base, SSS improves the math average from 23.6 to 25.2, and adding TMR further raises it to 26.5. On Meta-Llama-3.1-8B-Base, SSS gives a larger gain, improving the average from 9.8 to 22.7, while TMR further improves the result to 24.0. The consistent improvement after adding SSS indicates that replacing deterministic support selection with stochastic sampling is the main driver of the gains, and the additional improvement from TMR shows that teacher support mass provides a useful reliability signal for the conditional distillation loss.

4.4 Sensitivity Study

We vary the support size k and the teacher-mass exponent β to examine whether SSS+TMR is sensitive to these hyperparameters. Figure 4 shows that the average score is stable across both sweeps. When k varies from 64 to 512, the average score ranges from 25.3 to 26.6. Increasing k generally improves performance, indicating that a larger sampled support provides more reliable conditional distribution matching under the same stochastic construction. We choose $k = 256$ for the main experiments because it reaches 26.5 average score, nearly matching the 26.6 score at $k = 512$ while using only half as many support logits. This makes $k = 256$ a better accuracy–cost trade-off than the largest tested support size. When β varies from 0 to 1.5, the average score ranges from 24.7 to 26.5. The gain from $\beta = 0$ to $\beta = 1.0$ shows that teacher-mass reweighting is beneficial—a conclusion corroborated by the SSS vs. SSS+TMR ablation in Figure 3, ruling out the interpretation that gains stem purely from stochastic gradient noise—whereas an overly large exponent can down-weight too many sampled supports and slightly reduce performance. We therefore use $\beta = 1.0$ as the default setting, while the support-size sweep shows that performance remains stable across the tested support budgets.

4.5 Efficiency

Like $\text{OPD}_{\text{top-}k}$, SSS+TMR uses vocabulary-wide student and teacher distributions to construct the support. However, the fixed k-support budget applies to the differentiable distillation path: after support selection, the conditional RKL loss is materialized and backpropagated only over the $B \times T \times k$ gathered logits, rather than a dense $B \times T \times |V|$ full-vocabulary KL tensor. Therefore, SSS+TMR does not remove the shared vocabulary-wide proposal computation, but it reduces the loss-side memory

Qwen3-8B → Qwen3-1.7B-Base			
Method	**Tokens/s**	**Time/step**	**Rel. cost**
$\text{OPD}_{\text{top-}k}$	225.9	39.2s	1.0×
Ours	218.5	41.7s	1.03×
DS-R1-Distill-Llama-8B → Meta-Llama-3.1-8B-Base			
$\text{OPD}_{\text{top-}k}$	163.4	83.7s	1.0×
Ours	153.4	86.1s	1.07×

Table 3: Training efficiency comparison on 8×NVIDIA A800 80GB GPUs. Tokens/s and time/step are averaged after excluding the first 50 steps. Relative cost is normalized by $\text{OPD}_{\text{top-}k}$ within each model pair.

and backward cost (see Appendix F). Table 3 shows that the overhead of SSS+TMR remains modest relative to deterministic top-k OPD. On Qwen3 8B→1.7B, our method reduces throughput from 225.9 to 218.5 tokens/s and increases time per step from 39.2s to 41.7s, corresponding to a 1.03× relative cost. On Llama-3.1 8B→8B, throughput decreases from 163.4 to 153.4 tokens/s, while time per step increases from 83.7s to 86.1s, giving a 1.07× relative cost. These results indicate that the additional stochastic sampling and teacher-mass computation introduce only a small training overhead while preserving the same k-token support budget as top-k OPD. A detailed breakdown of each computation stage is provided in Table 6.

5 Related Work

Knowledge distillation for language models. Knowledge distillation transfers the behavior of a high-capacity teacher into a smaller student, originally by matching soft output distributions (Hinton et al., 2015), and has also been widely used for language-understanding model compression (Sanh et al., 2019; Jiao et al., 2020). For sequence generation, early work distilled teacher-generated sequences or trained students on fixed supervised data (Kim and Rush, 2016). These off-policy objectives are efficient, but the resulting train–test mismatch between training prefixes and student-generated inference prefixes motivates distillation objectives that expose the student to its own states (Bengio et al., 2015; Agarwal et al., 2024).

On-policy distillation and divergence choices. On-policy distillation addresses this mismatch by computing teacher supervision on student-generated trajectories. GKD is a representative framework that interpolates between on- and off-policy data and supports multiple token-level divergences (Agarwal et al., 2024). MiniLLM motivates reverse KL for LLM distillation from a sequence-level perspective (Gu et al., 2024). However, directly applying full-vocabulary RKL incurs memory and computation proportional to $|V|$ at every student-generated prefix.

Truncated distribution matching. Full-vocabulary token-level distillation can provide dense supervision, but materializing vocabulary-wide logits becomes costly for modern LLM vocabularies, such as the roughly 150K-token vocabulary of Qwen3 and 160K-token vocabulary of Kimi-K2.6 (Qwen Team, 2025; Moonshot AI, 2026). Top-k sparse distillation is a natural fixed-budget approximation for reducing this cost, but sparse logit approximations can introduce biased distribution estimates (Anshumann et al., 2025). Anshumann et al. (2025) target teacher logit compression via importance-sampling-based sparse logit storage; our work is complementary, focusing on training-time distribution matching quality. We identify two problems caused by top-k truncation: tokens outside the selected support receive no direct distillation signal at that prefix, and renormalization on the support ignores the teacher's total probability mass assigned to that support. Li et al. (2026) find that converged OPD concentrates on shared top-k tokens; SSS is consistent with this, as high-probability tokens still dominate the sampled support in expectation while stochastic construction expands effective coverage beyond the hard cutoff.

6 Conclusion

We analyzed two failure modes of top-k OPD as an approximation to full-vocabulary RKL: the truncated objective constrains only conditional distributions on the selected support, leaving tail-token coverage and support-level mass discrepancy unaddressed. To mitigate these issues, we proposed SSS, which samples support tokens without replacement from the student distribution, and TMR, which scales the conditional OPD loss by the teacher mass on the sampled support. Together, these components preserve the fixed support budget of top-k truncated RKL while improving tail coverage and reducing the influence of teacher-low-mass supports.

Limitations

Due to computational constraints, our experiments are limited to student models up to 8B parameters, and to two teacher–student pairs; these pairs represent a within-family (Qwen3-8B→Qwen3-1.7B-Base) and a cross-family same-vocabulary (DeepSeek-R1-Distill-Llama-8B→Meta-Llama-3.1-8B-Base) setting, and broader validation across additional model families and scales is left to future work. We also do not report formal confidence intervals, following standard LLM benchmarking practice; the Avg@k metric partially addresses variance by averaging over k independent rollouts per problem. In addition, our method requires the teacher and student to share the same vocabulary, which limits its applicability to cross-tokenizer distillation settings; supporting cross-vocabulary settings via vocabulary mapping is an avenue for future work. Finally, TMR uses teacher support mass as a reliability signal rather than directly optimizing full-vocabulary mass matching; it therefore mitigates, but does not fully eliminate, the approximation gap to full-vocabulary RKL.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*.

Anshumann, Mohd Abbas Zaidi, Akhil Kedia, Jinwoo Ahn, Taehwak Kwon, Kangwook Lee, Haejun Lee, and Joohyung Lee. 2025. Sparse logit sampling: Accelerating knowledge distillation in LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18085–18108.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie J. Cai, Michael Terry, Quoc V. Le, and Charles Sutton. 2021. Program synthesis with large language models. *CoRR*, abs/2108.07732.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015*, pages 1171–1179.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew M. Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *CoRR*, abs/2107.03374.

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

DeepSeek-AI. 2025. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *CoRR*, abs/2501.12948.

Yuqian Fu, Haohuan Huang, Kaiwen Jiang, Jiacai Liu, Zhuo Jiang, Yuanheng Zhu, and Dongbin Zhao. 2026. Revisiting on-policy distillation: Empirical failure modes and simple fixes. *CoRR*, abs/2603.25562.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. MiniLLM: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay V. Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022*.

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

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Llama Team. 2024. The Llama 3 herd of models. *CoRR*, abs/2407.21783.

Mathematical Association of America. 2023. American Mathematics Competitions 10/12 2023. MAA American Mathematics Competitions.

Mathematical Association of America. 2024. American Invitational Mathematics Examination 2024. MAA Invitational Competitions.

Mathematical Association of America. 2025. American Invitational Mathematics Examination 2025. MAA Invitational Competitions.

Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A diverse corpus for evaluating and developing english math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 975–984.

Moonshot AI. 2026. Kimi-K2.6. Hugging Face model card. 160K vocabulary size.

Minjae Oh, Sangjun Song, Gyubin Choi, Yunho Choi, and Yohan Jo. 2026. KL for a KL: On-policy distillation with control variate baseline. *arXiv preprint arXiv:2605.07865*.

Qwen Team. 2025. Qwen3 technical report. *CoRR*, abs/2505.09388.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Yaoxiang Wang, Haoling Li, Xin Zhang, Jie Wu, Xiao Liu, Wenxiang Hu, Zhongxin Guo, Yangyu Huang, Ying Xin, Yujiu Yang, Jinsong Su, Qi Chen, and Scarlett Li. 2025. EpiCoder: Encompassing diversity and complexity in code generation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024. Qwen2.5 technical report. *CoRR*, abs/2412.15115.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Weinan Dai, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. 2025. DAPO: An open-source LLM reinforcement learning system at scale. *CoRR*, abs/2503.14476.

Terry Yue Zhuo, Minh Chien Vu, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widyasari, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, Simon Brunner, Chen Gong, James Hoang, Armel Randy Zebaze, Xiaoheng Hong, Wen-Ding Li, Jean Kaddour, Ming Xu, Zhihan Zhang, Prateek Yadav, et al. 2025. BigCodeBench: Benchmarking code generation with diverse function calls and complex instructions. In *The Thirteenth International Conference on Learning Representations*.

A Proof of Relative Distribution Learning

We prove that the top-k truncated RKL objective constrains only the conditional distribution on the selected support, not the total probability mass assigned to that support. Let $S_k \subseteq V$ be fixed and assume $p_\theta(S_k) > 0$ and $q(S_k) > 0$. The truncated objective is

$$\mathcal{L}_{\text{rel}}(S_k) = D_{\text{KL}}(p_\theta(\cdot \mid S_k) \| q(\cdot \mid S_k)) \,. \tag{14}$$

Since KL divergence is non-negative and equals zero if and only if the two distributions are identical, its optimum over the conditional distribution satisfies

$$p_\theta(i \mid S_k) = q(i \mid S_k), \qquad \forall i \in S_k. \tag{15}$$

Substituting the definitions of the conditional distributions gives

$$\frac{p_\theta(i)}{p_\theta(S_k)} = \frac{q(i)}{q(S_k)}, \qquad \forall i \in S_k. \tag{16}$$

Therefore, for any $i, j \in S_k$ with positive probability,

$$\frac{p_\theta(i)}{p_\theta(j)} = \frac{q(i)}{q(j)}. \tag{17}$$

Thus, minimizing $\mathcal{L}_{\text{rel}}(S_k)$ matches the relative probabilities among tokens inside S_k.

However, the objective contains only the normalized quantities $p_\theta(i)/p_\theta(S_k)$ and $q(i)/q(S_k)$. The scalar masses $p_\theta(S_k)$ and $q(S_k)$ do not appear as quantities to be matched. Indeed, for any mass $a \in (0, 1]$, the assignment

$$p_\theta(i) = a\, q(i \mid S_k), \qquad i \in S_k \tag{18}$$

induces $p_\theta(\cdot \mid S_k) = q(\cdot \mid S_k)$ and hence achieves zero relative KL on S_k, regardless of whether $a = q(S_k)$. Therefore, the truncated objective alone does not enforce $p_\theta(S_k) = q(S_k)$.

B Proof of KL Decomposition

Let $S \subseteq V$ and $\bar{S} = V \setminus S$, and assume $p_\theta(S), q(S), p_\theta(\bar{S})$, and $q(\bar{S})$ are positive. Starting from the full-vocabulary KL,

$$\begin{aligned} D_{\text{KL}}(p_\theta \| q) = &\sum_{i \in S} p_\theta(i) \log \frac{p_\theta(i)}{q(i)} \\ &+ \sum_{i \in \bar{S}} p_\theta(i) \log \frac{p_\theta(i)}{q(i)}. \end{aligned} \tag{19}$$

For $i \in S$, write $p_\theta(i) = p_\theta(S) p_\theta(i \mid S)$ and $q(i) = q(S) q(i \mid S)$. Then

$$\begin{aligned} A_S &:= \sum_{i \in S} p_\theta(i) \log \frac{p_\theta(i)}{q(i)} \\ &= p_\theta(S) \sum_{i \in S} p_\theta(i \mid S) \log \frac{p_\theta(S) p_\theta(i \mid S)}{q(S) q(i \mid S)} \\ &= p_\theta(S) D_{\text{KL}}(p_\theta(\cdot \mid S) \| q(\cdot \mid S)) \\ &\quad + p_\theta(S) \log \frac{p_\theta(S)}{q(S)}. \end{aligned} \tag{20}$$

Applying the same expansion on $\bar{S}$ gives

$$\begin{aligned} A_{\bar{S}} &:= \sum_{i \in \bar{S}} p_\theta(i) \log \frac{p_\theta(i)}{q(i)} \\ &= p_\theta(\bar{S}) D_{\text{KL}}\big(p_\theta(\cdot \mid \bar{S}) \| q(\cdot \mid \bar{S})\big) \\ &\quad + p_\theta(\bar{S}) \log \frac{p_\theta(\bar{S})}{q(\bar{S})}. \end{aligned} \tag{21}$$

Combining the two expansions yields

$$\begin{aligned} D_{\text{KL}}(p_\theta \| q) = &\, p_\theta(S) D_{\text{KL}}(p_\theta(\cdot \mid S) \| q(\cdot \mid S)) \\ &+ p_\theta(\bar{S}) D_{\text{KL}}\big(p_\theta(\cdot \mid \bar{S}) \| q(\cdot \mid \bar{S})\big) \\ &+ p_\theta(S) \log \frac{p_\theta(S)}{q(S)} \\ &+ p_\theta(\bar{S}) \log \frac{p_\theta(\bar{S})}{q(\bar{S})}. \end{aligned} \tag{22}$$

Since $p_\theta(\bar{S}) = 1 - p_\theta(S)$ and $q(\bar{S}) = 1 - q(S)$, the last two terms are exactly the Bernoulli KL in Eq. (5). This proves the decomposition.

C Properties of the Bernoulli Mass Term

The third term in Eq. (5) is

$$D_{\text{KL}}(\text{Bern}(p_\theta(S)) \| \text{Bern}(q(S))) \,. \tag{23}$$

Let $a = p_\theta(S)$ and $b = q(S)$. Then this term can be written as

$$\phi(a, b) = a \log \frac{a}{b} + (1 - a) \log \frac{1 - a}{1 - b}. \tag{24}$$

This is the KL divergence between the binary event "the next token lies in S" under the student and teacher distributions. Therefore, unlike the conditional term $D_{\text{KL}}(p_\theta(\cdot \mid S) \| q(\cdot \mid S))$, it does not compare the relative probabilities of tokens inside S; it compares only the total mass assigned to S and its complement.

The term has two useful properties. First, by non-negativity of KL divergence,

$$\phi(a, b) \geq 0, \tag{25}$$

with equality if and only if $a = b$. Thus, this term is exactly the penalty that enforces support-mass matching. Second, for fixed teacher mass b, its derivative with respect to the student mass a is

$$\frac{\partial \phi(a,b)}{\partial a} = \log \frac{a(1-b)}{b(1-a)}. \tag{26}$$

The derivative is negative when $a < b$, zero when $a = b$, and positive when $a > b$. Consequently, minimizing this term increases the student support mass when it is below the teacher mass and decreases it when it is above the teacher mass. Dropping this term removes the only explicit force that corrects support-mass discrepancy between $p_\theta(S_k)$ and $q(S_k)$ on the selected support.

Why not directly optimize the Bernoulli KL? Although the Bernoulli mass term is useful for diagnosing the approximation gap, we do not use it as an additional optimization objective. This term performs explicit support-mass matching: it encourages the student total mass $p_\theta(S)$ to match the teacher total mass $q(S)$. For a fixed support S, the gradient of the student support mass is

$$\frac{\partial p_\theta(S)}{\partial z_j} = p_\theta(j)\left(\mathbb{I}[j \in S] - p_\theta(S)\right), \qquad j \in V. \tag{27}$$

Thus, the Bernoulli KL changes the allocation of probability mass between S and its complement $V \setminus S$. When $p_\theta(S) > q(S)$, it suppresses the student mass on S and pushes probability mass to the unobserved complement. However, in a truncated-support OPD setting, the teacher conditional distribution on $V \setminus S$ is not available, so this redistribution is not guided by token-level teacher preferences.

This is especially problematic for SSS, where S is constructed from the student distribution. A low teacher mass $q(S)$ should indicate that the conditional comparison on S is less reliable, not necessarily that the student should directly reduce probability on the sampled support. In other words, the Bernoulli term treats support-mass discrepancy as a direct optimization target, whereas TMR interprets teacher support mass as a confidence signal for whether the conditional RKL on that support should be trusted. Therefore, instead of optimizing the Bernoulli KL as a mass-matching loss, TMR uses the teacher support mass $P_T(S, \mathbf{x})^\beta$ only as a reliability weight. This preserves the conditional RKL direction of OPD while suppressing gradients from low-teacher-mass supports.

Empirical comparison with SSS+Bernoulli. To verify this design choice, we compare SSS+TMR (Ours) against a variant that adds the Bernoulli support-mass term as an auxiliary loss on top of SSS (SSS+Bernoulli) on the Qwen3-8B→Qwen3-1.7B-Base setting. As shown in Table 4, SSS+Bernoulli underperforms Ours on all benchmarks, confirming that directly optimizing the Bernoulli KL is harmful in the truncated-support setting.

Qwen3-8B → Qwen3-1.7B-Base					
Method	**AIME24 Avg@16**	**AIME25 Avg@16**	**AMC23 Avg@16**	**Minerva Math Avg@4**	**Avg.**
SSS+Bernoulli	11.7	12.5	49.8	25.1	24.78
Ours	**16.5**	**12.7**	**50.2**	**26.5**	**26.48**

Table 4: SSS+Bernoulli vs. Ours (SSS+TMR) on Qwen3-8B→Qwen3-1.7B-Base. Best is **bolded**.

D Derivation of Support-Mass Discrepancy

Let $S_k = \mathrm{TopK}(p_\theta)$ be the student's top-k support. Truncated OPD matches the conditional teacher distribution

$$q(i \mid S_k) = \frac{q(i)}{q(S_k)}, \qquad i \in S_k, \tag{28}$$

rather than the absolute teacher probabilities $q(i)$. If $q(S_k) < 1$, this renormalization inflates every teacher probability inside S_k by a factor of $1/q(S_k)$ relative to the teacher's absolute probabilities on those tokens. Thus, the truncated objective treats the teacher mass on S_k as a complete distribution and omits the teacher's remaining mass outside S_k from the conditional comparison.

E Experimental Settings

Table 5 summarizes the training and evaluation hyperparameters used in all experiments.

F Algorithm for SSS+TMR

Algorithm 1 gives the per-token training procedure for SSS+TMR. The support is sampled from the student distribution without replacement, while the teacher mass is used only as a support-level reliability weight.

Hyperparameter	Train	Eval
Framework	verl	–
Maximum output length	4096	8192
Temperature	1.0	1.0
Top-p sampling	1.0	0.8
Batch size	32	–
Learning rate	1×10^{-6}	–
Training steps	500	–
Metric	–	avg@16 / avg@4
Precision	bfloat16	bfloat16

Table 5: Training and evaluation settings. We report avg@16 for AIME24, AIME25, and AMC23, and avg@4 for all other datasets.

Algorithm 1 SSS+TMR Distillation

Require: Training sequence $\mathbf{y}$, teacher q, student p_θ, support size k, mass exponent β

$\mathcal{L} \leftarrow 0$
for each decoding position t **do**
  Build context $\mathbf{x}_t = (\mathbf{x}, y_{<t})$
  Compute student logits $z^\theta(\cdot|\mathbf{x}_t)$ and teacher logits $z^T(\cdot|\mathbf{x}_t)$
  Compute original probabilities $p_\theta(\cdot|\mathbf{x}_t)$, $q(\cdot|\mathbf{x}_t)$ via full-vocab softmax
  Sample $S_t \sim \mathrm{SampleK}(p_\theta(\cdot|\mathbf{x}_t), k)$ without replacement ▷ SSS: `torch.multinomial(replacement=False)` replaces `torch.topk`
  $P_T \leftarrow \sum_{i \in S_t} q(i|\mathbf{x}_t)$ ▷ TMR: sum over k gathered probs, shape $(B, T, K) \to (B, T)$
  $w_\beta \leftarrow P_T^\beta$
  for each token $i \in S_t$ **do** ▷ support-level renormalization
    $\bar{p}_i \leftarrow p_\theta(i|\mathbf{x}_t) / \sum_{j \in S_t} p_\theta(j|\mathbf{x}_t)$
    $\bar{q}_i \leftarrow q(i|\mathbf{x}_t) / \sum_{j \in S_t} q(j|\mathbf{x}_t)$
  end for
  $\mathcal{L}_t \leftarrow D_{\mathrm{KL}}(\bar{p}_{S_t} \| \bar{q}_{S_t})$
  $\mathcal{L} \leftarrow \mathcal{L} + w_\beta \mathcal{L}_t$
end for
Update θ by backpropagating $\mathcal{L}$

F.1 What Does the Fixed Support Budget Save?

Student-top-k OPD and SSS+TMR both require full-vocabulary student and teacher logits/probabilities to construct the support. Therefore, our method is not intended to avoid the vocabulary projection or full-vocabulary normalization step. The fixed-k budget refers to the support on which the distillation loss is formed: after selecting the support, we gather only k teacher/student probabilities, renormalize them on the support, and compute conditional RKL over these k tokens.

Compared with full-vocabulary RKL, this avoids materializing and optimizing a dense token-level KL loss over all vocabulary entries. Compared with student-top-k OPD, SSS+TMR uses the same full-vocabulary distributions but replaces deterministic top-k selection with sampling and adds a scalar teacher-mass weight.

Table 6 summarizes the per-token computation stages after student rollout. Bold entries indicate full-vocabulary computation or full-vocabulary token coverage. The table separates the full-vocabulary forward/normalization shared by truncated-support methods from the support-restricted gather–renormalize–RKL pipeline.

G Case Study

Figure 5 separates the qualitative behavior into two stages. Panels (a) and (b) show two typical weaknesses of the base model. First, the model may perform part of the arithmetic correctly, but the correct answer is buried inside noisy continuation and cannot be reliably extracted. Second, the model sometimes fails to enter a math-solving mode at all, producing empty boxes, copied instructions, webpage fragments, or unrelated worksheet-style text. Deterministic top-k OPD alleviates these base-model failures by inducing more structured reasoning and boxed-answer formats, but panels (c) and

Stage	**Full-vocab RKL**	OPD$_{\text{top-}k}$	SSS+TMR
Student forward	✓	✓	✓
Support construction	$S_t = V$	TopK	SampleK
Student prob. selection	**all** $\lvert V \rvert$	gather k	gather k
Teacher forward	✓	✓	✓
Teacher prob. selection	**all** $\lvert V \rvert$	gather k	gather k
Support renorm.	–	✓	✓
Cond. RKL on support	–	✓	✓
Full-vocab RKL	✓	–	–
Teacher mass reweight	–	–	✓

Table 6: Per-token computation comparison after student rollout. Bold entries indicate full-vocabulary computation or full-vocabulary token coverage. SSS+TMR shares the same support-restricted gather–renormalize–RKL pipeline as OPD$_{\text{top-}k}$, but replaces deterministic top-k support construction with stochastic support sampling and adds teacher mass reweighting.

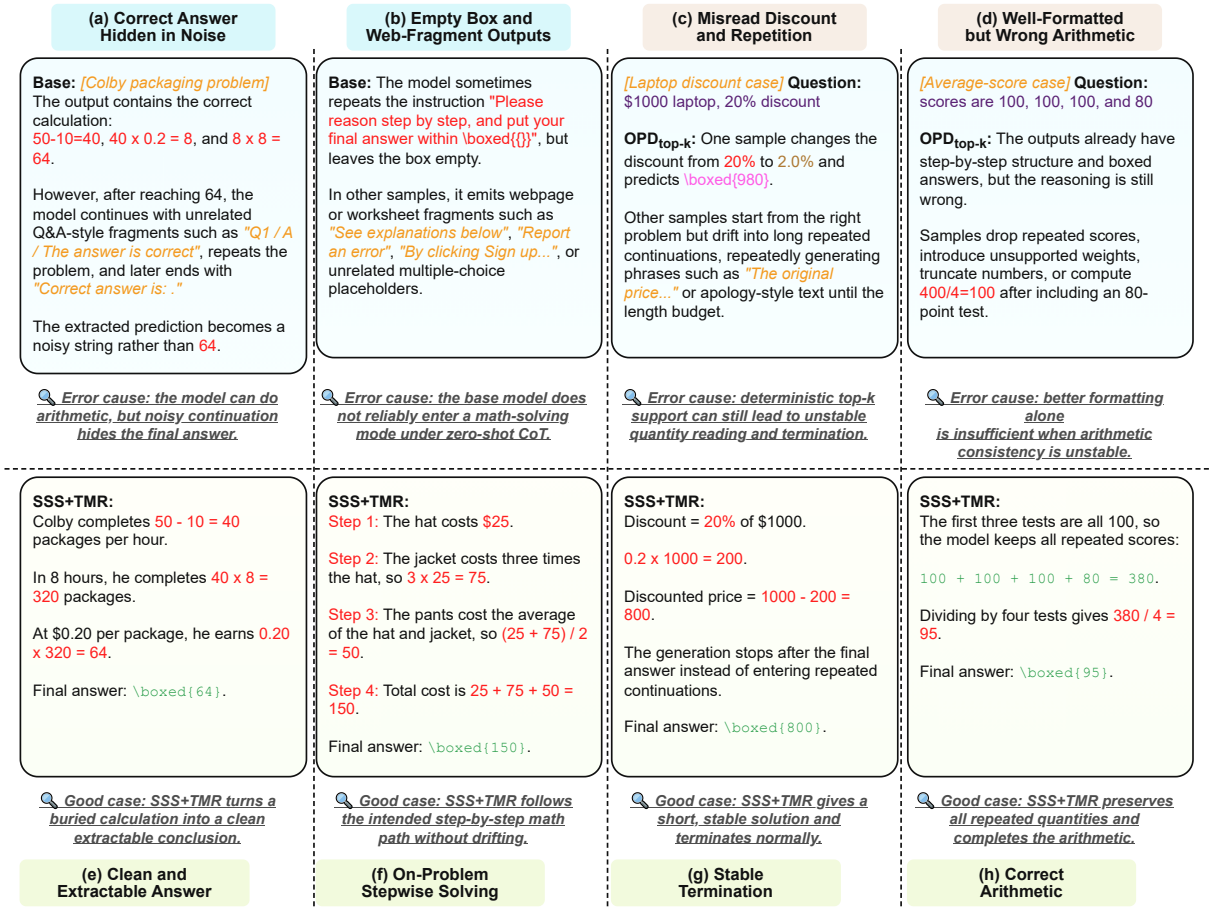

Figure 5: Case study examples illustrating SSS+TMR behavior.

(d) show that new errors remain: the model can misread key quantities, fail to terminate, drop repeated numbers, introduce unsupported assumptions, or produce well-formatted but arithmetically inconsistent solutions.

SSS+TMR improves the behavior beyond this formatting gain. SSS exposes the student to tokens outside the deterministic top-k support, including low-probability but important numeric, operator, and termination tokens. TMR then reduces the effect of sampled supports that receive little teacher probability mass, making the conditional supervision less sensitive to unrepresentative supports. As shown in panels (e)–(h), the resulting outputs are not only extractable and on-problem, but also more stable in termination, quantity preservation, and final arithmetic.

H Dataset Descriptions

AIME24 & AIME25. The American Invitational Mathematics Examination (AIME) is a 15-problem competition exam requiring integer answers in $[0, 999]$. AIME24 and AIME25 refer to the 2024 and 2025 editions respectively.

AMC23. The American Mathematics Competition 10/12 (AMC23) is a multiple-choice exam; we use the 2023 edition.

Minerva Math. Minerva Math (Lewkowycz et al., 2022) is a subset of MATH requiring multi-step scientific reasoning.

GSM8K. GSM8K (Cobbe et al., 2021) is a benchmark of grade-school math word problems.

MATH500. MATH500 (Lightman et al., 2024) is a 500-problem subset of the MATH benchmark spanning five difficulty levels and seven subject areas.

ASDiv. ASDiv (Miao et al., 2020) is a diverse arithmetic word problem dataset covering a wide range of problem types and linguistic patterns.

MBPP. MBPP (Austin et al., 2021) is a Python programming benchmark of entry-level problems.

HumanEval. HumanEval (Chen et al., 2021) is a function-level code generation benchmark with unit-test evaluation.

BigCodeBench. BigCodeBench (Zhuo et al., 2025) is a challenging code benchmark requiring complex library usage and multi-step reasoning.

I Prompt Templates

We use the following chat templates for training and evaluation.

Llama-3.1 Chat Template

```
<|begin_of_text|>
<|start_header_id|>system<|end_header_id|>
Please reason step by step, and put your final answer within \boxed
      {}.
<|eot_id|>
<|start_header_id|>user<|end_header_id|>
{question}<|eot_id|>
<|start_header_id|>assistant<|end_header_id|>
```

Qwen3 Chat Template

```
<|im_start|>system
Please reason step by step, and put your final answer within \boxed
      {}.
<|im_end|>
<|im_start|>user
{question}<|im_end|>
<|im_start|>assistant
<think>
</think>
```