ALIGN AND EXPAND: RECOVERING DENSE SUPERVISION FOR CROSS-TOKENIZER ON-POLICY DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy distillation (OPD) aligns teacher knowledge along student-generated trajectories, reducing the discrepancy between training and inference. This advantage has made OPD an increasingly prominent post-training paradigm. However, when the teacher and student use different tokenizers, their vocabularies differ and some student prediction positions are unmatched by the teacher tokenization, rendering conventional distillation objectives inapplicable. Existing methods circumvent this token mismatch through span-level alignment, but typically compress each aligned span into a binary distribution. Such binary representation discards token-level supervision within each span, as well as the dark knowledge contained in the teacher's probability distribution. To address these limitations, we propose **Cross-Tokenizer Alignment and Supervision Expansion (CASE)**, an align-and-expand framework that establishes span-level correspondences across heterogeneous tokenizers and recovers missing intra-span supervision through teacher-context expansion, ultimately enabling dense supervision at the span level. Across mathematical reasoning and code generation, CASE achieves the best average performance in all three teacher–student pairings, demonstrating robust gains under heterogeneous tokenizers.

1 INTRODUCTION

Knowledge distillation transfers a strong teacher's behavior to a smaller student (Hinton et al., 2015). On-policy distillation (OPD) obtains teacher feedback along student-generated trajectories, reducing the mismatch between the student's training and inference distributions (Gu et al., 2024; Agarwal et al., 2024). Tokenizer differences in vocabulary and text segmentation prevent direct comparison of the teacher and student next-token distributions.

This incompatibility is deeper than a difference in logit dimension. As illustrated in Figure 1, shared token-end offsets partition two tokenizations into $1{:}1$, $1{:}n$, $m{:}1$, and $m{:}n$ spans. In a $1{:}n$ span, the last $n-1$ student prediction positions do not align with any teacher prediction position and therefore lack corresponding teacher logits. In an $m{:}1$ span, the student makes one prediction while the teacher makes m, leaving the last $m-1$ teacher prediction positions unmatched. An $m{:}n$ span contains both types of mismatch and can be decomposed into $1{:}n$ and $m{:}1$ cases by iterating over the teacher and student prediction positions. Cross-tokenizer OPD must therefore solve two coupled problems. It needs a common event space for comparing predictions and valid teacher supervision at unmatched student positions.

Existing methods approach this problem through token alignment, vocabulary mapping, representation projection, or probability conversion (Wan et al., 2024; Boizard et al., 2024; Zhang et al., 2024; Phan et al., 2025). Recent span-based objectives avoid forcing a one-to-one token correspondence by comparing the likelihood assigned to aligned text spans (Minixhofer et al., 2026; Sun et al., 2026). This makes the objective well posed across tokenizers, but often compresses each span into a Bernoulli distribution. Such compression removes the teacher's relative preferences among alternative continuations, which convey dark knowledge. It ignores student distributions at prediction positions in the span and yields uneven supervision across positions in each sample. The central

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

question is therefore how to recover dense, context-consistent supervision without violating either model's autoregressive factorization.

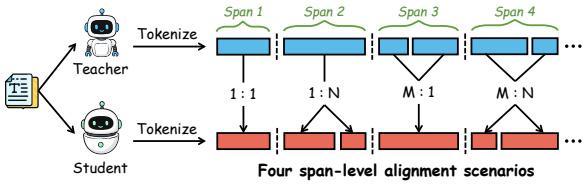


Figure 1: The same byte string induces four alignment scenarios. Shared token-end offsets define aligned spans, while offsets in only one tokenization identify unmatched prediction positions without direct token-level supervision.

We propose Cross-Tokenizer Alignment and Supervision Expansion (**CASE**), an align-and-expand framework for cross-tokenizer OPD. The alignment stage builds a prefix tree over complete token strings and maps each token to its longest shared-prefix node. CASE aggregates each model's token probabilities at these nodes, yielding aligned distributions over alternative continuations rather than only the likelihood of the observed span. The expansion stage then handles unmatched prediction positions. At unmatched student positions, CASE searches for alternative teacher tokenizations of the already known span bytes and marginalizes over the resulting teacher contexts. At unmatched teacher positions, it decomposes the available student distribution into events visible at successive teacher prediction positions.

These components separate two responsibilities that are easily conflated in cross-tokenizer distillation. Shared-support alignment specifies *what* predictive events can be compared, whereas span-aware expansion specifies *where* the teacher distribution must be evaluated. The search performs constrained segmentation of a known byte string rather than teacher generation. Rather than assigning one target to an entire span, CASE can therefore supervise student decisions and teacher-visible internal events while preserving the natural conditioning of both models.

We evaluate CASE on three heterogeneous teacher–student pairings formed from Qwen3-8B, Phi-4-mini-instruct, and Llama-3.2-3B-Instruct, covering four mathematical reasoning and four code-generation benchmarks. CASE achieves the highest average score in all three pairings, outperforming the strongest baseline by 0.39, 0.83, and 0.54 percentage points, respectively. Across the 24 pair–benchmark combinations, it is best or tied for best in 18 and ranks within the top two in 21. Notably, CASE leads on all eight benchmarks for Qwen3→Llama and on all four code-generation benchmarks for Qwen3→Phi-4, demonstrating strong performance across model pairings and task domains.

Our contributions are summarized as follows:

- We characterize cross-tokenizer OPD across four span-alignment scenarios as the coupled problems of aligning predictive event spaces and providing context-consistent supervision at unmatched prediction positions.
- We introduce CASE, an align-and-expand framework that projects model distributions onto shared exact byte-prefix events. CASE recovers teacher targets at unmatched student positions by marginalizing over alternative teacher tokenizations. At unmatched teacher positions, reverse decomposition exposes the successive teacher decisions hidden within a single student token.
- We demonstrate CASE's effectiveness across three teacher–student pairs and eight mathematical reasoning and code-generation benchmarks. CASE achieves the highest average score in every pairing, while ablation and sensitivity analyses show the benefit of supervision expansion and robustness to top-k truncation.

2 Related Work

2.1 On-policy Distillation

Traditional knowledge distillation transfers the teacher's "dark knowledge" to the student through dense token-level supervision (Hinton et al., 2015). However, its off-policy formulation conditions the student on prefixes that differ from those encountered during inference, leading to exposure bias (Agarwal et al., 2024). In contrast, on-policy distillation obtains teacher feedback on student-generated trajectories, thereby reducing the mismatch between the student's training and inference

distributions. Early work, such as MiniLLM, optimizes the student distribution using reverse KL divergence, encouraging the student to focus on high-probability regions of the teacher distribution (Gu et al., 2024). GKD further extends this paradigm by allowing flexible choices among forward KL, reverse KL, and generalized Jensen–Shannon divergence (Agarwal et al., 2024).

Subsequent studies investigate how divergence objectives interact with data sampled from the teacher or student (Ko et al., 2024; 2025). Recent work continues to advance the paradigm by exploring where distillation signals should be applied and how they should be weighted, as well as by systematically analyzing failure modes (Jin et al., 2026; Ko et al., 2026; Li et al., 2026; Song & Zheng, 2026). Nevertheless, these methods assume that the teacher and student share the same vocabulary and tokenizer. When this assumption does not hold, their token-level distributions cannot be directly aligned, limiting the applicability of existing methods. Constructing effective supervision across heterogeneous vocabularies is a central challenge in cross-tokenizer distillation.

2.2 Cross-tokenizer Distillation

Given the same input text, heterogeneous teacher and student tokenizers produce logit tensors of shapes $L_T \times |\mathcal{V}_T|$ and $L_S \times |\mathcal{V}_S|$, respectively, reflecting mismatches in both sequence length and vocabulary size. Cross-tokenizer distillation therefore requires alignment at both the sequence and vocabulary levels (Chen et al., 2025; Patiño et al., 2025). FuseLLM uses minimum-edit-distance (MinED) dynamic programming to align similar tokens across the two token sequences (Wan et al., 2024). CDM combines entropy-weighted sequence alignment with context-aware top-K dynamic vocabulary mapping for logit-level distillation (Chen et al., 2025). From an optimal-transport perspective, ULD aligns logit distributions of different dimensionalities by sorting their entries (Boizard et al., 2024), while MultiLevelOT introduces hierarchical optimal-transport objectives at both the token and sequence levels (Cui et al., 2025). DSKD trains bidirectional projectors and employs cross-model attention to align representation sequences of different lengths (Zhang et al., 2024). DWA-KD and SRA strengthen semantic constraints and improve robustness under low vocabulary overlap through entropy-weighted dual-space alignment and span-representation alignment, respectively (Vu et al., 2026; Dao et al., 2026). Another line of work converts probabilities over byte sequences. Cross-tokenizer likelihood scoring exploits the recursive structure of Byte-Pair Encoding for exact or approximate conversion (Phan et al., 2025), while BLD approximates the teacher's token distribution as next-byte probabilities, avoiding explicit vocabulary mapping at the cost of approximation error (Singh et al., 2026).

Recent studies address tokenization mismatch through span-level alignment. ALM partitions token sequences into aligned spans and computes Bernoulli KL divergence per span (Minixhofer et al., 2026). Similarly, SEDI aligns spans by character offsets and re-encodes teacher predictions in the student vocabulary (Wang et al., 2026). More recent work has incorporated cross-tokenizer alignment into on-policy distillation. Building on ULD, GOLD improves the alignment of non-shared tokens and uses conditional probabilities to compute the probability of merged token sequences (Patiño et al., 2025). SimCT similarly sums log-likelihoods over minimal aligned units, but this aggregation discards fine-grained teacher–student differences within each span (Sun et al., 2026). Overall, existing methods enable cross-tokenizer distillation via alignment or probability conversion. However, span-level methods often compress spans into binary distributions, losing fine-grained token-level signals and highlighting the need for dense intra-span supervision.

3 Method

CASE addresses two mismatches in cross-tokenizer OPD: different next-token vocabularies and unmatched prediction positions. It first projects both models' distributions onto a shared support, then recovers supervision within aligned spans (Figure 2).

3.1 Span Alignment

Let $\mathcal{V}_M$ be the vocabulary of model $M \in \{T, S\}$ and $\beta_M(t)$ the decoded byte string of token t; for sequences, β_M denotes concatenation of token bytes. We write $u \preceq v$ for a byte prefix and ε for the empty string. Given a student rollout, both tokenizers induce token-end byte offsets. Shared offsets are *anchors*; consecutive anchors delimit an aligned span. Interior offsets in only one sequence are

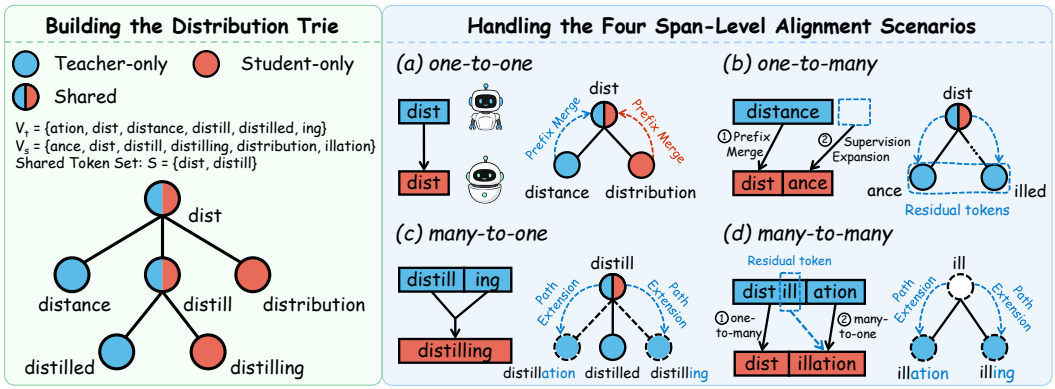


Figure 2: Overview of CASE. Left: complete token byte strings form a sparse prefix structure, and strings realized by both tokenizers define the shared support. Right: the four alignment scenarios require direct alignment, forward teacher-context expansion, reverse prefix decomposition, or both operations.

unmatched student or *unmatched teacher positions*. A two-pointer scan recovers 1:1, 1:n, m:1, and m:n spans, with $m, n > 1$ (Figure 1).

At span starts, both models have matching byte contexts. A 1:1 span is supervised directly on the shared support, since both models predict after the same byte context. Spans with multiple student tokens require teacher-context expansion at unmatched student positions; spans with multiple teacher tokens require reverse decomposition at unmatched teacher positions. An m:n span requires both operations.

3.2 Prefix-Aware Shared Support

Shared-prefix projection. Restricting alignment to shared tokens discards non-shared probability mass. CASE instead assigns each token to its longest shared byte prefix using a trie (Fredkin, 1960). With $\mathcal{T}_M = \{\beta_M(t) : t \in \mathcal{V}_M\}$, the shared support is

$$\mathcal{A} = (\mathcal{T}_T \cap \mathcal{T}_S) \cup \{\varepsilon, v_{\text{eos}}\}. \tag{1}$$

The trie contains complete token strings from both vocabularies, with each node attached to its longest proper token-string prefix. The root supplies a fallback. A separate virtual node v_{eos} aligns EOS tokens.

For non-EOS tokens, the assignment and aligned distribution are

$$\begin{aligned} a_M(t) &= \operatorname*{argmax}_{\substack{v \in \mathcal{A} \setminus \{v_{\text{eos}}\} \\ v \preceq \beta_M(t)}} |v|, \\ q_M(v \mid x) &= \sum_{t \in \mathcal{V}_M :\, a_M(t) = v} \pi_M(t \mid x), \qquad v \in \mathcal{A}. \end{aligned} \tag{2}$$

EOS tokens map to v_{eos}. The assignment classes partition each vocabulary, so $\sum_{v \in \mathcal{A}} q_M(v \mid x) = 1$. Thus both distributions retain all token probability mass on mutually exclusive longest-prefix classes. The trie and token-to-node maps are precomputed.

Refinement property. Removing shared nodes merges longest-prefix classes. For any $\mathcal{B} \subseteq \mathcal{A}$ retaining the root and EOS, the KL chain rule gives $D_{\text{KL}}(q_S^{\mathcal{B}} \| q_T^{\mathcal{B}}) \le D_{\text{KL}}(q_S^{\mathcal{A}} \| q_T^{\mathcal{A}})$ when the latter is finite (Appendix A.2). Thus additional shared prefixes expose teacher–student discrepancies hidden by coarser projections, while distinctions within each class remain unidentifiable.

3.3 Span-Aware Supervision Expansion

Forward expansion. Consider teacher and student sequences $A_k^T = (t_{k,1}, \ldots, t_{k,m})$ and $A_k^S = (s_{k,1}, \ldots, s_{k,n})$ decoding to the same span y_k. Let $o_{k,j} = |\beta_S(s_{k,1:j})|$, with $o_{k,0} = 0$. Student

prediction j follows $y_k[:o_{k,j-1}]$. The ordinary teacher pass supplies logits at the span start, but an interior student position may lie inside a teacher token.

For $n > 1$, CASE searches for alternative teacher tokenizations:

$$\widehat{\mathcal{P}}_k \subseteq \{\rho \in \mathcal{V}_T^* : |\rho| \le D_{\max},\ \beta_T(\rho) = y_k\}. \tag{3}$$

Here $|\rho|$ counts tokens. Each token must match the next known bytes; this is constrained segmentation rather than generation. Let $\widehat{\mathcal{R}}_{k,j}$ be the distinct prefixes of retained paths ending exactly at $o_{k,j-1}$. A retained prefix may contain several teacher tokens, even when its concatenated bytes do not correspond to a single token in the teacher vocabulary. For $\rho = (u_1, \dots, u_r)$, its weight is $w_T(\rho \mid x_k^T) = \prod_{\ell=1}^{r} \pi_T(u_\ell \mid x_k^T, u_{<\ell})$, where x_k^T precedes the span. For nonempty retained sets, the expanded target is

$$\begin{aligned} \alpha_\rho^{k,j} &= \frac{w_T(\rho \mid x_k^T)}{\sum_{\rho' \in \widehat{\mathcal{R}}_{k,j}} w_T(\rho' \mid x_k^T)}, \\ \widetilde{q}_T^{k,j}(v) &= \sum_{\rho \in \widehat{\mathcal{R}}_{k,j}} \alpha_\rho^{k,j} q_T(v \mid x_k^T, \rho). \end{aligned} \tag{4}$$

This mixes teacher contexts at the same byte offset rather than selecting a single tokenization. The mixture is limited to the retained paths and does not enumerate all byte-level continuations.

Reverse decomposition. In an m:1 span, one student decision covers several teacher decisions. CASE uses the available student distribution without introducing intermediate student states. Such states would condition the student on tokens it never emitted. Write $z_{k,\ell} = \beta_T(t_{k,\ell})$ and $c_{k,\ell-1} = \beta_T(t_{k,<\ell})$. At an unmatched teacher position $\ell > 1$, the candidate set $\Gamma_{k,\ell}$ contains the observed piece and all teacher-token strings z such that $c_{k,\ell-1} \circ z$ prefixes a student token. The teacher token mass and student prefix mass are

$$\begin{aligned} r_T^{k,\ell}(z) &= \sum_{t:\, \beta_T(t) = z} \pi_T(t \mid x_k^T, t_{k,<\ell}), \\ r_S^{k,\ell}(z) &= \sum_{s:\, c_{k,\ell-1} \circ z \preceq \beta_S(s)} \pi_S(s \mid x_k^S). \end{aligned} \tag{5}$$

Normalizing each vector over $\Gamma_{k,\ell}$ gives $\bar{r}_T^{k,\ell}$ and $\bar{r}_S^{k,\ell}$. Teacher contexts advance along the observed sequence, while the student distribution remains at the span's single student position.

3.4 Training Objective and Computation

Let $\mathcal{I}$ be the disjoint union of typed direct, expanded, and reverse span-position indices. Direct and expanded indices are student positions with directly aligned and recovered teacher contexts, respectively; reverse indices are unmatched teacher positions. Write $q_M^{k,j}$ for the aligned distribution at student position (k, j). At a direct position, $(P_i, Q_i) = (q_S^{k,j}, q_T^{k,j})$; at an expanded position, $(P_i, Q_i) = (q_S^{k,j}, \widetilde{q}_T^{k,j})$; at a reverse position, $(P_i, Q_i) = (\bar{r}_S^{k,\ell}, \bar{r}_T^{k,\ell})$. All components use reverse KL:

$$\mathcal{L}_{\mathrm{CASE}} = \frac{1}{N_S} \sum_{i \in \mathcal{I}} D_{\mathrm{KL}}(P_i \| Q_i), \tag{6}$$

where N_S counts student response tokens in the batch. Using the student trajectory length provides a consistent denominator across batches containing different mixtures of direct, expanded, and reverse supervision. This preserves a common batch normalization while adding supervision at unmatched positions; a span with n student tokens can supply up to n context-specific targets.

Projection uses precomputed token-to-node lookups. Expansion reuses observed teacher states, deduplicates unseen prefixes, and evaluates them in a batched prefill with KV-cache reuse. The teacher is frozen throughout. Additional teacher computation is concentrated at unmatched student positions, where the observed teacher tokenization supplies no logits for the student's byte context. Execution details are provided in Appendix A.1.

Table 1: Pairwise cross-tokenizer on-policy distillation results (%). The first three rows report results for undistilled models, whereas each remaining block specifies a teacher–student pair. Avg. is the unweighted mean across all eight benchmarks. Bold and underlined values denote the best and second-best results within each pair, respectively.

Model	Method	Math				Code				Avg.
		OpenR1	AIME24	AIME25	MATH	Codeforces	LCB	HumanEval	MBPP	
Qwen3-8B	Base	38.34	25.83	23.75	81.95	12.40	35.14	84.60	66.93	46.12
Phi-4-mini-instruct	Base	17.80	10.83	3.75	39.10	3.07	18.65	72.87	53.67	27.47
Llama-3.2-3B-Instruct	Base	8.08	4.17	0.00	39.72	2.62	3.29	55.56	50.02	20.43
Qwen3-8B *teacher* ↓ Phi-4-mini-instruct *student*	ULD	13.61	7.92	2.08	40.45	4.69	18.15	71.88	53.95	26.59
	MinED	14.54	10.00	4.58	**52.95**	6.10	14.98	73.32	51.85	28.54
	DSKD	14.69	4.58	2.92	39.37	5.86	21.52	74.09	54.67	27.21
	ALM	**17.75**	9.17	**7.92**	38.40	5.04	20.81	73.17	53.17	28.18
	SimCT	15.95	**11.25**	6.67	46.82	7.12	21.37	75.46	54.75	29.92
	CASE	16.54	**11.25**	6.25	47.48	**7.18**	**22.27**	**76.14**	**55.40**	**30.31**
Qwen3-8B *teacher* ↓ Llama-3.2-3B-Instruct *student*	ULD	6.20	6.67	0.42	39.22	3.12	12.36	58.46	49.60	22.01
	MinED	5.17	5.00	0.42	46.37	3.50	10.81	55.18	47.88	21.79
	DSKD	5.29	3.75	1.67	43.25	3.41	11.80	57.70	48.93	21.98
	ALM	9.11	3.33	1.25	47.88	3.17	11.33	57.39	49.15	22.83
	SimCT	11.08	5.42	2.50	52.62	3.88	12.70	60.29	49.95	24.81
	CASE	**11.15**	**9.17**	**2.92**	**53.63**	**4.18**	**12.99**	**60.59**	**50.50**	**25.64**
Phi-4-mini-instruct *teacher* ↓ Llama-3.2-3B-Instruct *student*	ULD	4.51	1.67	0.42	36.30	2.67	9.76	52.52	48.98	19.60
	MinED	8.16	4.17	0.00	36.65	1.95	7.82	46.95	46.32	19.00
	DSKD	4.49	0.42	0.00	32.15	1.41	7.39	46.19	45.05	17.14
	ALM	8.67	**6.67**	0.42	42.13	2.64	**9.91**	52.74	**49.28**	21.56
	SimCT	9.60	4.58	1.25	44.57	2.74	5.26	**54.04**	48.18	21.33
	CASE	**10.01**	5.00	**1.67**	**45.57**	**2.78**	8.86	**54.04**	48.90	**22.10**

3.5 Illustrative Example

For the toy vocabularies on the left of Figure 2, let $G_M(v)$ collect the tokens assigned to shared class v. The longest-prefix rule gives the groups in Table 2.

Table 2: Longest-prefix token groups for the vocabularies in Figure 2.

v	$G_T(v)$ Teacher	$G_S(v)$ Student
dist	{dist, distance}	{dist, distribution}
distill	{distill, distilled}	{distill, distilling}
ε	{ation, ing}	{ance, illation}

Each group supplies one aligned probability, $q_M(v) = \sum_{t \in G_M(v)} \pi_M(t)$, with context arguments omitted. This is the aggregation in Equation (2). Each token appears once: `distilled` belongs to `distill`, its longest shared prefix. The root collects tokens with no nonempty shared prefix. EOS is handled separately at v_{eos}.

The right-hand panels show where supervision is needed. Here `|` marks actual token boundaries. In (a), both models emit `dist`, allowing direct alignment. In (b), teacher `distance` matches student `dist|ance`. The student needs supervision after `dist`, motivating forward expansion. In (c), teacher `distill|ing` matches student `distilling`. The teacher decision after `distill` is compared using the student's earlier distribution. In (d), teacher `distill|ation` matches student `dist|illation`, creating unmatched student and teacher positions after `dist` and `distill`, respectively.

Dashed residual nodes, such as `ill` inside `distill`, denote remaining bytes rather than additional teacher tokens. Dashed continuation nodes can represent multi-token paths, such as `distill|ation` for `distillation`. The expanded target in Equation (4) uses only retained valid teacher-token prefixes.

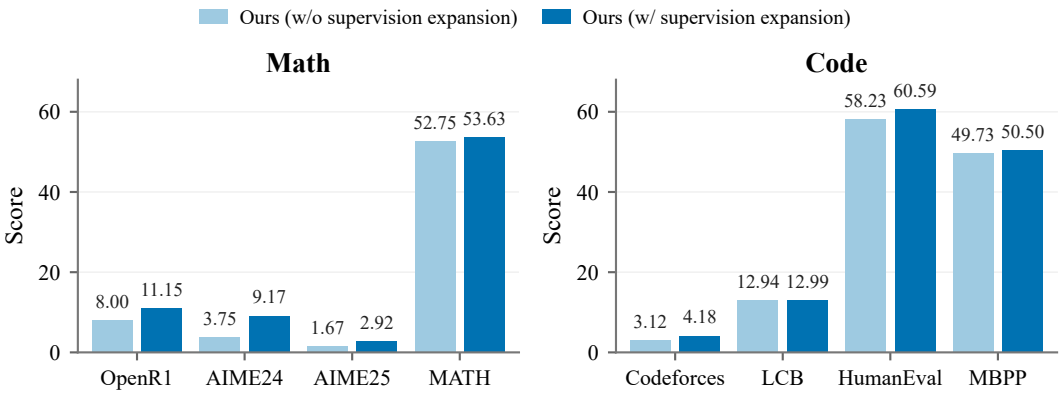


Figure 3: Effect of supervision expansion across mathematical reasoning and code generation benchmarks.

4 Experiments

Our evaluation examines whether dense cross-tokenizer supervision improves over representative alignment and span-level baselines, and whether the improvement persists across teacher–student pairs and task domains.

4.1 Experimental Setup

Models and pairings. We consider Qwen3-8B (Yang et al., 2025), Phi-4-mini-instruct (Abouelenin et al., 2025), and Llama-3.2-3B-Instruct (Grattafiori et al., 2024). Their base performance follows this order, so we use the stronger model as the teacher in each pair, yielding Qwen3-8B→Phi-4-mini, Qwen3-8B→Llama-3.2-3B, and Phi-4-mini→Llama-3.2-3B. As shown in Table 3, all three models use different vocabulary sizes and tokenization behavior. For numeric strings, Qwen3 tokenizes digits individually, whereas Phi-4 and Llama may merge up to three digits, yielding token sequences of different lengths.

Table 3: Model sizes and tokenizer characteristics. Numeric segmentation illustrates how tokenizer differences create unmatched prediction positions.

Model	Parameters	Vocabulary Size	Numeric Segmentation
Qwen3-8B	8.2B	151,936	1 digit
Phi-4-mini-instruct	3.8B	200,064	Up to 3 digits
Llama-3.2-3B-Instruct	3.21B	128,256	Up to 3 digits

Training data. Following SD-Zero (He et al., 2026), we use separate training datasets for mathematical reasoning and code generation. The math split contains 15K OpenR1-Math problems with verified answers (Hugging Face, 2025). The code split contains approximately 15K examples from the `verifiable-prompts` subset of Codeforces (Penedo et al., 2025), balanced between C++ and Python. For each teacher–student pair, all methods use the same student initialization, training data, and on-policy budget. Further training details are provided in the appendix.

Baselines. We compare CASE against five complementary cross-tokenizer approaches. ULD aligns vocabulary distributions of different sizes by sorting their probabilities (Boizard et al., 2024). MinED aligns token sequences through minimum edit distance (Wan et al., 2024). DSKD learns projectors into a shared representation space (Zhang et al., 2024). ALM matches approximate span likelihoods (Minixhofer et al., 2026). SimCT performs on-policy distillation over minimal aligned units (Sun et al., 2026).

Benchmarks and metrics. We evaluate mathematical reasoning on a 1K-example held-out OpenR1-Math set, AIME24 (Zhang & Math-AI, 2024), AIME25 (Zhang & Math-AI, 2025), and MATH (Hendrycks et al., 2021). Code generation is evaluated on 844 held-out Codeforces prob-

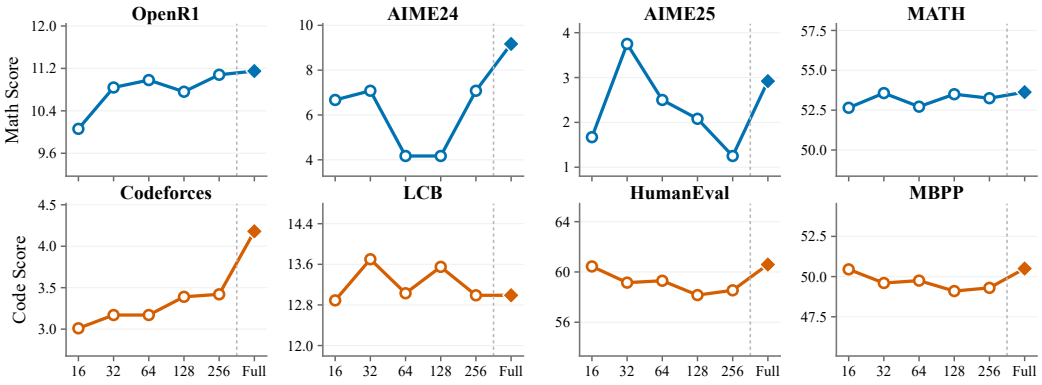


Figure 4: Sensitivity of CASE to top-k truncation across mathematical reasoning and code generation benchmarks. The x-axis denotes k, with Full indicating the complete teacher distribution.

lems, LiveCodeBench (LCB) (Jain et al., 2024), HumanEval (Chen et al., 2021), and MBPP (Austin et al., 2021). OpenR1-Math and Codeforces are in-domain evaluations. The remaining benchmarks evaluate generalization beyond the training distributions. We report Avg@8 scores for all benchmarks as percentages. All evaluations except LCB use the lm-evaluation-harness framework (Gao et al., 2024).

4.2 Main Results

Overall effectiveness. Table 1 shows that CASE obtains the highest average score in all three teacher–student settings. For Qwen3→Phi-4, Qwen3→Llama, and Phi-4→Llama, CASE reaches averages of 30.31%, 25.64%, and 22.10%, exceeding the strongest baseline in each setting by absolute margins of 0.39, 0.83, and 0.54 percentage points, respectively. Relative to the undistilled students, the corresponding gains are 2.84, 5.21, and 1.67 percentage points. Across the 24 pair–benchmark combinations, CASE is best or tied for best in 18 and ranks within the top two in 21.

Qwen3 teacher. With Phi-4 as the student, CASE obtains the best result on all four code benchmarks and ties the best AIME24 score. The strongest method varies across the math benchmarks. ALM is stronger on OpenR1-Math and AIME25, while MinED leads on MATH. Nevertheless, CASE's consistent gains across the code benchmarks raise its overall average above that of SimCT, the strongest competing method in this pair (30.31% versus 29.92%).

Performance across methods varies on the math benchmarks in this setting. Inspection of the Phi-4 rollouts reveals variation in generation quality. For example, MinED obtains the highest MATH score but is less competitive on the other three math benchmarks. The trained Phi-4 student tends to append many emoji symbols to its outputs. Because MATH extracts the answer from the final `\boxed{}`, its evaluation is less affected by this behavior. This may reflect Qwen3's long-tail distribution of emoji tokens. Phi-4's vocabulary contains notably more emoji tokens than Llama's, which could explain the more stable math performance in both settings that use Llama as the student.

For Qwen3→Llama, CASE achieves the best score on every benchmark, including gains of 3.75 and 1.01 percentage points over SimCT on AIME24 and MATH, respectively. This pair also shows the largest improvement over the base student, from 20.43% to 25.64%, indicating that the method remains effective when the teacher and student differ substantially in capability as well as tokenizer.

Phi-4 teacher. For Phi-4→Llama, CASE again achieves the best overall average, improving over ALM from 21.56% to 22.10%. It leads or ties on five benchmarks. On the code tasks, we observe that Llama is particularly prone to repetitive generation during training. This tendency is amplified by the on-policy feedback loop, eventually triggering repetition collapse. This observation highlights the importance of teacher capability, as a weaker teacher can produce overly peaked token distributions and thus provide less reliable supervision. When the teacher and student use different tokenizers, alignment errors can accumulate and further destabilize training. Nevertheless,

Table 4: Teacher-side shared-prefix projection statistics on main-experiment rollouts.

Teacher	Student	Non-shared mass (%)	Merged mass (%)	Avg. tokens per merged prefix
Qwen3	Llama	0.04	12.02	6.18
Qwen3	Phi-4	0.66	49.86	3.85
Phi-4	Llama	0.82	51.37	5.08

CASE remains competitive across all four code benchmarks, demonstrating its robustness in the code generation OPD setting.

Shared-prefix projection analysis. We analyze teacher distributions on rollouts from the three main experimental settings (Table 4). Non-shared mass measures teacher probability outside the shared byte-string vocabulary. Merged mass includes the full probability of prefixes receiving multiple teacher tokens, including shared tokens themselves. The last column reports the average number of teacher tokens assigned to each such prefix. Non-shared mass is below 1%, while 12.02–51.37% of teacher probability lies in merged classes, averaging 3.85–6.18 tokens per prefix.

Ablation of supervision expansion in CASE. We evaluate the contribution of CASE's supervision expansion component in the Qwen3→Llama setting, with the results shown in Figure 3. Removing this component reduces the average performance from 25.64% to 23.77%. This result is 1.04 percentage points below SimCT but remains higher than all other baselines. Supervision expansion improves all eight benchmarks, supporting the value of recovered intra-span supervision.

Sensitivity to top-k truncation. Top-k truncation may bias teacher probabilities (Peng et al., 2025; Anshumann et al., 2025). Table 1 and figure 3 use the full distribution. We compare $k \in \{16, 32, 64, 128, 256\}$ against the full vocabulary in Figure 4, with complete results in the appendix.

On the in-domain OpenR1-Math and Codeforces benchmarks, performance generally improves as k increases and is highest with the full vocabulary. This trend suggests that the long-tail teacher signals are not necessarily noise and can contribute to stable performance gains. On the out-of-domain benchmarks, CASE with the full distribution achieves the best performance on AIME24, MATH, HumanEval, and MBPP, outperforming the second-best AIME24 result by 2.09 percentage points. It also remains competitive on the other two benchmarks.

Random-seed sensitivity. For Qwen3→Llama, CASE obtains an average of $(25.37 \pm 0.24)\%$ across three seeds (mean $\pm$ sample standard deviation), with full results reported separately in Appendix A.4 and table 9.

5 Conclusion

We present CASE, an align-and-expand framework for on-policy distillation between language models with heterogeneous tokenizers. CASE starts from the observation that cross-tokenizer supervision requires both a common predictive event space and context-consistent supervision at unmatched prediction positions. We construct a trie over the model vocabularies to efficiently redistribute probability mass across heterogeneous vocabularies. CASE then recovers missing intra-span supervision through teacher-context expansion and reverse decomposition, enabling dense supervision across aligned spans. Across three teacher–student pairings and eight mathematical reasoning and code-generation benchmarks, CASE achieves the best average result in every pairing and is best or tied for best in 18 of 24 pair–benchmark combinations, outperforming the strongest baseline by 0.39, 0.83, and 0.54 percentage points in the three pairings, respectively. Beyond these aggregate results, the ablation of supervision expansion confirms the effectiveness of this component. CASE remains robust under top-k truncation, with performance improving as k increases and peaking when the full distribution is retained. Advancing cross-tokenizer distillation in OPD and improving training stability remain promising directions for future work.

AI USE STATEMENT

We commit to accurately reporting our use of LLMs. Specifically, LLMs were used for language polishing, improving writing clarity, and assisting with the search for related work. However, LLMs were not used to generate the core ideas or to write substantive portions of this paper.

REFERENCES

Abdelrahman Abouelenin, Atabak Ashfaq, Adam Atkinson, Hany Awadalla, Nguyen Bach, Jianmin Bao, Alon Benhaim, Martin Cai, Vishrav Chaudhary, Congcong Chen, et al. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras. *arXiv preprint arXiv:2503.01743*, 2025.

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 *International Conference on Learning Representations*, volume 2024, pp. 21246–21263, 2024.

Anshumann Anshumann, Mohd Abbas Zaidi, Akhil Kedia, Jinwoo Ahn, Taehwak Kwon, Kangwook Lee, Haejun Lee, and Joohyung Lee. 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)*, pp. 18085–18108, 2025.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Nicolas Boizard, Kevin El Haddad, Céline Hudelot, and Pierre Colombo. Towards cross-tokenizer distillation: the universal logit distillation loss for llms. *arXiv preprint arXiv:2402.12030*, 2024.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Yijie Chen, Yijin Liu, Fandong Meng, Yufeng Chen, Jinan Xu, and Jie Zhou. Enhancing cross-tokenizer knowledge distillation with contextual dynamical mapping. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 8005–8018, 2025.

Xiao Cui, Mo Zhu, Yulei Qin, Liang Xie, Wengang Zhou, and Houqiang Li. Multi-level optimal transport for universal cross-tokenizer knowledge distillation on language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(22):23724–23732, 2025.

Quoc Phong Dao, Hoang Son Nguyen, Pham Khanh Chi, Tung Nguyen, Linh Ngo Van, Nguyen Thi Ngoc Diep, and Trung Le. Sra: Span representation alignment for large language model distillation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 32961–32975, 2026.

Edward Fredkin. Trie memory. *Communications of the ACM*, 3(9):490–499, 1960.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL `https://zenodo.org/records/12608602`.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. In *International Conference on Learning Representations*, volume 2024, pp. 32694–32717, 2024.

Yinghui He, Simran Kaur, Adithya Bhaskar, Yongjin Yang, Jiarui Liu, Narutatsu Ri, Liam H Fowl, Abhishek Panigrahi, Danqi Chen, and Sanjeev Arora. Self-distillation zero: Self-revision turns binary rewards into dense supervision. In *ICML 2026 Workshop on Foundations of Deep Generative Models: Understanding Memorization, Generalization, and Reasoning*, 2026.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Hugging Face. Open r1: A fully open reproduction of deepseek-r1, January 2025. URL `https://github.com/huggingface/open-r1`.

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. *arXiv preprint arXiv:2403.07974*, 2024.

Woogyeol Jin, Taywon Min, Yongjin Yang, Swanand Ravindra Kadhe, Yi Zhou, Dennis Wei, Nathalie Baracaldo, and Kimin Lee. Entropy-aware on-policy distillation of language models. *arXiv preprint arXiv:2603.07079*, 2026.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. Distillm: Towards streamlined distillation for large language models. *arXiv preprint arXiv:2402.03898*, 2024.

Jongwoo Ko, Tianyi Chen, Sungnyun Kim, Tianyu Ding, Luming Liang, Ilya Zharkov, and Se-Young Yun. Distillm-2: A contrastive approach boosts the distillation of llms. *arXiv preprint arXiv:2503.07067*, 2025.

Jongwoo Ko, Sara Abdali, Young Jin Kim, Tianyi Chen, and Pashmina Cameron. Scaling reasoning efficiently via relaxed on-policy distillation. *arXiv preprint arXiv:2603.11137*, 2026.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huanang Gao, Wenkai Yang, Zhiyuan Liu, et al. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv preprint arXiv:2604.13016*, 2026.

Feng Luo, Yu-Neng Chuang, Guanchu Wang, Zicheng Xu, Xiaotian Han, Tianyi Zhang, and Vladimir Braverman. Demystifying opd: Length inflation and stabilization strategies for large language models. *arXiv preprint arXiv:2604.08527*, 2026.

Benjamin Minixhofer, Ivan Vulić, and Edoardo Maria Ponti. Universal cross-tokenizer distillation via approximate likelihood matching. *Advances in Neural Information Processing Systems*, 38: 79297–79326, 2026.

Carlos Miguel Patiño, Kashif Rasul, Quentin Gallouédec, Ben Burtenshaw, Sergio Paniego, Vaibhav Srivastav, Thibaud Frere, Ed Beeching, Lewis Tunstall, Leandro von Werra, and Thomas Wolf. Unlocking on-policy distillation for any model family. `https://huggingface.co/spaces/HuggingFaceH4/on-policy-distillation`, 2025.

Guilherme Penedo, Anton Lozhkov, Hynek Kydlíček, 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.

Hao Peng, Xin Lv, Yushi Bai, Zijun Yao, Jiajie Zhang, Lei Hou, and Juanzi Li. Pre-training distillation for large language models: A design space exploration. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3603–3618, 2025.

Buu Phan, Ashish Khisti, and Karen Ullrich. Cross-tokenizer likelihood scoring algorithms for language model distillation. *arXiv preprint arXiv:2512.14954*, 2025.

Avyav Kumar Singh, Yen-Chen Wu, Alexandru Cioba, Alberto Bernacchia, and Davide Buffelli. Cross-tokenizer llm distillation through a byte-level interface. In *Proceedings of the Second Workshop on Customizable NLP: Progress and Challenges in Customizing NLP for a Domain, Application, Group, or Individual (CustomNLP4U)*, pp. 84–96, 2026.

Mingyang Song and Mao Zheng. A survey of on-policy distillation for large language models. *arXiv preprint arXiv:2604.00626*, 2026.

Jie Sun, Mao Zheng, Mingyang Song, Qiyong Zhong, Yilin Cheng, Bichuan Feng, Pengfei Liu, Junfeng Fang, and Xiang Wang. Simct: Recovering lost supervision for cross-tokenizer on-policy distillation. *arXiv preprint arXiv:2605.07711*, 2026.

Duc Trung Vu, Pham Khanh Chi, Dat Phi Van, Linh Ngo Van, Dinh Viet Sang, and Trung Le. Dwa-kd: Dual-space weighting and time-warped alignment for cross-tokenizer knowledge distillation. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 3513–3527, 2026.

Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. Knowledge fusion of large language models. *arXiv preprint arXiv:2401.10491*, 2024.

Huazheng Wang, Yongcheng Jing, Haifeng Sun, Jingyu Wang, Jianxin Liao, Leszek Rutkowski, and Dacheng Tao. Bridging the tokenizer gap: Semantics and distribution-aware knowledge transfer for unbiased cross-tokenizer distillation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(39):33494–33502, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Songming Zhang, Xue Zhang, Zengkui Sun, Yufeng Chen, and Jinan Xu. Dual-space knowledge distillation for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 18164–18181, 2024.

Songming Zhang, Xue Zhang, Tong Zhang, Bojie Hu, Yufeng Chen, and Jinan Xu. Kdflow: A user-friendly and efficient knowledge distillation framework for large language models. *arXiv preprint arXiv:2603.01875*, 2026.

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.

A Appendix

A.1 Teacher-Context Execution

CASE plans candidate contexts before executing them. Prefixes that coincide with the observed teacher tokenization reuse ordinary teacher hidden states. Identical unseen token-prefix sequences are evaluated once and reused across path hypotheses. All unseen sequences are processed in one batched prefill phase with KV-cache reuse, retaining only each prefix's final hidden state. The teacher remains frozen; expansion performs neither autoregressive sampling nor teacher-side back-propagation.

A.2 Refinement of Shared-Prefix Projection

Fix the teacher and student contexts and suppress their conditioning arguments. For a support $\mathcal{C}$ containing the root ε and the virtual EOS node v_{eos}, let $f_{\mathcal{C}}(w)$ be the longest byte prefix of w in $\mathcal{C} \setminus \{v_{\mathrm{eos}}\}$, with EOS mapped separately to v_{eos}. Using the same token probabilities for each support, define $q_M^{\mathcal{C}}(v) = \sum_{t:f_{\mathcal{C}}(\beta_M(t))=v} \pi_M(t)$ for $M \in \{S, T\}$, with the separate EOS assignment understood.

Proposition 1 (Shared-support refinement)**.** *Let $\mathcal{B} \subseteq \mathcal{A}$ contain the same root and EOS node, and let both projections use longest-prefix assignment. If $D_{\mathrm{KL}}(q_S^{\mathcal{A}} \| q_T^{\mathcal{A}})$ is finite, then*

$$D_{\mathrm{KL}}(q_S^{\mathcal{A}} \| q_T^{\mathcal{A}}) = D_{\mathrm{KL}}(q_S^{\mathcal{B}} \| q_T^{\mathcal{B}}) + \sum_{\substack{b \in \mathcal{B}: \\ q_S^{\mathcal{B}}(b) > 0}} q_S^{\mathcal{B}}(b) D_{\mathrm{KL}}\big(q_S^{\mathcal{A}}(\cdot \mid b) \| q_T^{\mathcal{A}}(\cdot \mid b)\big), \tag{7}$$

where the conditional distributions are over fine classes merged into b. Consequently, $D_{\mathrm{KL}}(q_S^{\mathcal{B}} \| q_T^{\mathcal{B}}) \le D_{\mathrm{KL}}(q_S^{\mathcal{A}} \| q_T^{\mathcal{A}})$.

Proof. Define $g(v) = f_{\mathcal{B}}(v)$ for non-EOS nodes $v \in \mathcal{A}$, and $g(v_{\mathrm{eos}}) = v_{\mathrm{eos}}$. We first show that $f_{\mathcal{B}} = g \circ f_{\mathcal{A}}$. For a byte string w, write $a = f_{\mathcal{A}}(w)$ and $b = f_{\mathcal{B}}(w)$. Since $\mathcal{B} \subseteq \mathcal{A}$, $|b| \le |a|$. Both are prefixes of w, so $b \preceq a$. Every $\mathcal{B}$-prefix of a is also a prefix of w and hence cannot be longer than b. Thus $g(a) = b$. The identity also holds for EOS by its separate assignment. Aggregating the same token probabilities therefore gives

$$q_M^{\mathcal{B}}(b) = \sum_{v:g(v)=b} q_M^{\mathcal{A}}(v), \qquad M \in \{S, T\}. \tag{8}$$

For each b with $q_S^{\mathcal{B}}(b) > 0$, finiteness of the fine KL implies $q_T^{\mathcal{B}}(b) > 0$. On the fiber $g^{-1}(b)$, define $q_M^{\mathcal{A}}(v \mid b) = q_M^{\mathcal{A}}(v)/q_M^{\mathcal{B}}(b)$. Writing $q_M^{\mathcal{A}}(v) = q_M^{\mathcal{B}}(b) q_M^{\mathcal{A}}(v \mid b)$ and splitting the logarithm yields

$$\begin{aligned} &\sum_{v:g(v)=b} q_S^{\mathcal{A}}(v) \log \frac{q_S^{\mathcal{A}}(v)}{q_T^{\mathcal{A}}(v)} \\ &\qquad = q_S^{\mathcal{B}}(b) \log \frac{q_S^{\mathcal{B}}(b)}{q_T^{\mathcal{B}}(b)} + q_S^{\mathcal{B}}(b) D_{\mathrm{KL}}\big(q_S^{\mathcal{A}}(\cdot \mid b) \| q_T^{\mathcal{A}}(\cdot \mid b)\big). \end{aligned}$$

Terms with zero student probability are omitted under the usual KL convention. Summing over the positive-mass coarse classes proves Equation (7). Each conditional KL is nonnegative, establishing the inequality. Equality holds exactly when the teacher and student conditional distributions coincide in every coarse class with positive student mass. □

The additional term quantifies discrepancies hidden by merging shared classes. The result compares nested supports under the same longest-prefix rule; it neither makes the final classes lossless nor establishes an ordering of downstream performance or of unrelated alignment objectives.

A.3 Experimental Details

The compute environment is summarized in Table 5, and we use KDFlow (Zhang et al., 2026) as the training framework. The filtering code for OpenR1-Math is provided with the supplementary code and data. The dataset contains 15K training examples and 1K test examples. For Codeforces, we use the training and test splits from the `verifiable-prompts` subset. The training hyperparameters are listed in Table 6. For Qwen3-8B, we disable thinking mode by setting `enable_thinking=False`. Full results for the top-k truncation study and the ablation of supervision expansion discussed in the main paper appear in Table 7.

Table 5: Compute environment used for training and evaluation, including hardware and software specifications.

Device Information	**Details**
CPU Model	Intel Xeon Platinum 8468 @ 2.10 GHz
CPU Configuration	2 sockets, 96 cores, 192 threads
Memory Size	2.0 TB
GPU Configuration	8 × NVIDIA H100 80 GB HBM3
GPU Memory Size	8 × 80 GB
CUDA Toolkit	12.8
NVIDIA Driver	580.126.20

Repetitive rollout generation during training, as discussed in the main paper, has also been reported in prior work (Luo et al., 2026). We provide an additional repetition analysis in Table 8. In both settings, we set the rollout `repetition_penalty` to 1.03 to mitigate repetition, which otherwise becomes severe and can cause training collapse. We report the repetition rate and severe repetition rate. A rollout is counted as repetitive if a 1–2-token loop occurs at least eight times or a 3–5-token loop occurs at least four times. It is counted as severely repetitive if a 6–1,024-token span is repeated in full at least twice.

On OpenR1-Math, MinED achieves the highest mean task performance, but the resulting student often continues generating repeated emoji symbols after completing an answer. The vocabularies of Qwen3, Phi-4, and Llama contain 1,117, 76, and 5 emoji-like tokens, respectively. This disparity may make the Phi-4 student more susceptible than Llama to long-tail emoji signals from the Qwen3 teacher. CASE attains the second-best mean task performance and the lowest severe repetition rate. On Codeforces, methods with lower repetition rates generally achieve better mean task performance. MinED and DSKD exhibit substantial repetition. ALM achieves the lowest repetition rates and the second-best mean task performance, whereas CASE ranks second on both repetition metrics and achieves the best mean task performance. Overall, these results indicate that CASE is robust to repetitive rollout generation, which may improve the stability of OPD.

Table 6: Hyperparameters used for on-policy distillation and rollout generation. AdamW is instantiated in the training code and the listed ϵ is its PyTorch default. LoRA is disabled, so all student parameters are updated during distillation.

Hyperparameter	**Value**
Optimization and training	
Optimizer	AdamW
Learning rate	1.0×10^{-6}
Learning-rate schedule	Cosine decay
Warmup ratio	0.20
AdamW (β_1, β_2)	$(0.9, 0.98)$
AdamW ϵ	1.0×10^{-8}
Weight decay	0.0
Maximum gradient norm	1.0
Training precision	BF16
Distributed strategy	FSDP2
Trainable student parameters	All parameters (LoRA disabled)
Random seed	42
Rollout generation	
Maximum rollout length	8,192 tokens
Chat template	Enabled
Temperature	1.0
Top-p	1.0
Min-p	0.0
Data configuration	
Maximum prompt length	8,192 tokens
Maximum total sequence length	16,384 tokens

Table 7: Complete results for CASE without supervision expansion and with different top-k truncation settings (%). The first two rows report results for the undistilled teacher and student. Bold and underlined values denote the best and second-best results among the CASE variants, respectively.

Model	Method	Setting	Math				Code				Avg.
			OpenR1	AIME24	AIME25	MATH	Codeforces	LCB	HumanEval	MBPP	
Qwen3-8B	Base	/	38.34	25.83	23.75	81.95	12.40	35.14	84.60	66.93	46.12
Llama-3.2-3B-Instruct	Base	/	8.08	4.17	0.00	39.72	2.62	3.29	55.56	50.02	20.43
Qwen3-8B *teacher* ↓ Llama-3.2-3B-Instruct *student*	**CASE**	w/o supervision expansion	8.00	3.75	1.67	52.75	3.12	12.94	58.23	49.73	23.77
		top-16	10.06	6.67	1.67	52.65	3.01	12.89	<u>60.44</u>	<u>50.45</u>	24.73
		top-32	10.84	<u>7.08</u>	**3.75**	<u>53.57</u>	3.17	**13.70**	59.15	49.60	<u>25.11</u>
		top-64	10.98	4.17	2.50	52.72	3.17	13.03	59.30	49.75	24.45
		top-128	10.76	4.17	2.08	53.50	3.39	<u>13.55</u>	58.16	49.10	24.34
		top-256	<u>11.08</u>	<u>7.08</u>	1.25	53.25	<u>3.42</u>	12.99	58.54	49.30	24.61
		full vocab	**11.15**	**9.17**	<u>2.92</u>	**53.63**	**4.18**	12.99	**60.59**	**50.50**	**25.64**

Table 8: Repetition rates and mean task performance on OpenR1-Math and Codeforces (%). Lower repetition rates and higher task performance are better. Bold and underlined values denote the best and second-best results within each teacher–student pair, respectively.

Dataset	Model	Method	Repetition Rate (↓)	Severe Repetition Rate (↓)	Mean Task Performance (↑)
OpenR1-Math	Qwen3-8B *teacher* ↓ Phi-4-mini-instruct *student*	ULD	**5.85**	34.47	16.02
		MinED	28.36	4.68	**20.52**
		DSKD	25.59	12.56	15.39
		ALM	<u>8.07</u>	<u>1.92</u>	18.31
		SimCT	12.92	2.20	20.17
		CASE	10.74	**1.84**	<u>20.38</u>
Codeforces	Phi-4-mini-instruct *teacher* ↓ Llama-3.2-3B-Instruct *student*	ULD	3.47	1.17	28.48
		MinED	46.07	11.06	25.76
		DSKD	26.36	24.07	25.01
		ALM	**2.36**	**0.27**	<u>28.64</u>
		SimCT	8.88	2.90	27.56
		CASE	<u>2.58</u>	<u>0.54</u>	**28.65**

A.4 RANDOM-SEED SENSITIVITY

We repeat CASE for Qwen3-8B→Llama-3.2-3B-Instruct with seeds 32, 42, and 52, keeping all other training and evaluation settings fixed. As shown in Table 9, the average score is $(25.37 \pm 0.24)\%$ (mean $\pm$ sample standard deviation), compared with 20.43% for the undistilled student.

Table 9: Random-seed sensitivity of CASE for Qwen3-8B→Llama-3.2-3B-Instruct (%). The first two rows show undistilled models. The final two rows report the mean and sample standard deviation across seeds 32, 42, and 52. Avg. is the unweighted mean over eight benchmarks. Summary statistics use the reported benchmark scores before rounding Avg.

Model	Seed	Math				Code				Avg.
		OpenR1	AIME24	AIME25	MATH	Codeforces	LCB	HumanEval	MBPP	
Qwen3-8B	/	38.34	25.83	23.75	81.95	12.40	35.14	84.60	66.93	46.12
Llama-3.2-3B-Instruct	/	8.08	4.17	0.00	39.72	2.62	3.29	55.56	50.02	20.43
Qwen3-8B *teacher* ↓ Llama-3.2-3B-Instruct *student*	32	11.45	6.67	2.08	53.47	4.19	12.87	60.98	50.18	25.24
	42	11.15	9.17	2.92	53.63	4.18	12.99	60.59	50.50	25.64
	52	11.45	6.67	2.50	53.57	3.97	12.80	60.67	50.18	25.23
Mean		11.35	7.50	2.50	53.56	4.11	12.89	60.75	50.29	25.37
Std.		0.17	1.44	0.42	0.08	0.12	0.10	0.21	0.18	0.24