IOD: An Iterative On-policy Distillation Framework for Self-Improving Language Models

Anonymous ACL submission

Abstract

Knowledge distillation is an effective approach for compressing large language models, but sequence generation distillation suffers from train–test mismatch caused by teacher forcing. On-policy distillation addresses this by training on student-generated trajectories, yet it typically lacks an explicit mechanism for selecting and reusing useful trajectories across training cycles. We propose **Iterative On-policy Distillation (IOD)**, a multi-cycle framework that constructs a teacher-filtered curriculum from student generations. At each cycle, the student generates candidates, a fixed teacher scores them by conditional likelihood, and high-scoring samples are retained, accumulated, and re-filtered for subsequent distillation. This enables quality-controlled reuse of student-generated data without an external reward model. Experiments on summarization and machine translation show that IOD improves over supervised fine-tuning and standard on-policy distillation, with ablations confirming the importance of filtering and accumulation.

1 Introduction

While large language models (LLMs) excel at natural language generation, deploying them remains challenging due to high inference costs and latency (Brown et al., 2020; Chowdhery et al., 2023; Achiam et al., 2023). Knowledge distillation (KD) addresses this by transferring knowledge to smaller models (Bucilua et al., 2006; Hinton et al., 2015; Zhu et al., 2023). However, distilling sequence models introduces challenges beyond standard supervised learning (Kim and Rush, 2016) due to exposure bias—a train-test mismatch between teacher-forced training and autoregressive inference (Bengio et al., 2015; Lamb et al., 2016; Ranzato et al., 2016). On-policy distillation mitigates this by training on student-generated trajectories (Agarwal et al., 2024). However, it typically operates as a single-cycle pipeline, lacking explicit mechanisms to accumulate a refined curriculum as the student improves. Concurrently, iterative self-training frameworks like ReST yield progressive improvements (Gulcehre et al., 2023), but heavily rely on costly models for generation, risking policy mismatch over time.

To bridge this gap, we propose **Iterative On-policy Distillation (IOD)**. IOD shares the intuition of iterative loops and student-centric generation, but differs in that it excludes the teacher from generation and uses the teacher as a fixed evaluator for student-generated candidates, while constructing an explicit data curriculum that can be accumulated and refreshed across cycles. Concretely, IOD follows a multi-cycle *Grow-Evaluate/Filter-Improve* framework. In each cycle, the student generates candidate outputs y on-policy for inputs x (**Grow**); the teacher then evaluates each candidate with its conditional log-likelihood score (**Evaluate**) and filters to retain only a top fraction of high-scoring samples (**Filter**). The student is subsequently updated using the filtered samples (**Improve**), and the process repeats. Across cycles, previously filtered samples are carried forward, combined with newly generated candidates, and re-filtered using the same teacher-based criterion. This filtered accumulation expands the set of teacher-approved on-policy samples available for student training while reducing the risk of accumulating noisy or poorly aligned generations, distinguishing IOD from standard single-cycle on-policy distillation setups.

In experiments on summarization and machine translation tasks, we show that IOD improves over standard on-policy distillation baselines, highlighting the benefits of multi-cycle, teacher-scored curriculum construction. Our contributions are:

1. We propose IOD, a multi-cycle on-policy distillation framework that constructs an explicit curriculum from student-generated trajectories

through teacher-based filtering.

2. We introduce a filtered accumulation strategy that carries teacher-approved student generations across cycles and re-filters the accumulated pool using the same teacher-based criterion, without relying on an external reward model.
3. We validate the approach with controlled experiments and ablations on summarization and machine translation, isolating the effects of filtering, accumulation, and training objectives.

2 Related Work

Our study connects knowledge distillation for language models with self-training and data filtering. The former studies how a smaller student learns from a stronger teacher, while the latter studies how model-generated data can be selected and reused. We focus on their intersection: how to construct a multi-cycle distillation curriculum from student-generated trajectories.

2.1 Knowledge Distillation for Language Models

Knowledge distillation (KD) transfers knowledge from a large teacher model to a smaller student by training the student to imitate the teacher's behavior (Bucilua et al., 2006; Hinton et al., 2015). For language generation, this imitation can be performed through token-level distribution matching or sequence-level distillation, where the student learns from teacher-produced outputs (Kim and Rush, 2016). However, autoregressive generation introduces a mismatch between teacher-forced training and free-running inference: the model is trained on reference prefixes but must condition on its own previous outputs at test time, which can lead to error accumulation (Bengio et al., 2015; Lamb et al., 2016; Ranzato et al., 2016). Recent LLM distillation studies have explored improved objectives and efficient use of generated outputs. MiniLLM emphasizes reverse KL divergence for distilling generative language models (Gu et al., 2024), while DistiLLM studies objective design and adaptive off-policy strategies for efficient distillation (Ko et al., 2024). These works mainly focus on divergence choices or computational efficiency. In contrast, our work focuses on how student-generated data are selected and reused during distillation.

On-policy distillation directly addresses the training–inference mismatch by training on trajectories sampled from the student policy. Generalized Knowledge Distillation (GKD) matches teacher and student next-token distributions under student-generated prefixes, bringing training closer to deployment-time conditions (Agarwal et al., 2024). GKD also allows different divergence choices, including reverse KL and generalized JSD. Building on this premise, we study a complementary question: how student-generated trajectories should be selected, retained, and reused as the student changes over training. Standard on-policy distillation serves as our primary baseline.

2.2 Self-Training and Data Filtering

Self-training is a classical paradigm in which a model uses its own predictions as additional supervision, often by generating pseudo-labels and retraining on selected examples (Scudder, 1965; Yarowsky, 1995). Modern language model training uses related iterative pipelines that generate candidates, score or filter them, and update the model using selected data. The effectiveness of such pipelines depends strongly on the selection criterion, since low-quality generated data can reinforce model errors.

Reinforced Self-Training (ReST) is an RL-motivated instance of iterative generated-data training (Gulcehre et al., 2023). It shows that repeated generation, filtering, and improvement can be effective when a reward or evaluation signal is available. Our work is related to this iterative perspective, but does not directly combine ReST with on-policy distillation. Instead, we take standard on-policy distillation as the primary baseline and study whether student-generated trajectories can be filtered, accumulated, and reused using the existing teacher as a fixed scoring signal.

A broader line of work studies filtering and data selection for synthetic or model-generated text. Self-Instruct filters invalid or similar model-generated instruction data before instruction tuning (Wang et al., 2023), and work on instruction tuning and alignment highlights the value of carefully curated data (Zhou et al., 2023). More generally, data selection is an important design axis for language model training because selection criteria can affect both performance and training cost (Albalak et al., 2024). Prior work also suggests that repeated training on generated data can distort the training distribution when data quality is not controlled (Shumailov et al., 2023). These findings

motivate quality control when reusing generated data, which we study in the teacher–student distillation setting.

3 Methodology

This section presents Iterative On-policy Distillation (IOD), a multi-cycle framework for constructing a teacher-scored curriculum from student-generated trajectories. We first define the problem setup, then provide a theoretical motivation for teacher-based filtering, and finally describe the full IOD procedure.

3.1 Problem Setup

Let π_T denote a fixed teacher model and let $\pi_S^{(c)}$ denote the student model after cycle c, where $\pi_S^{(0)}$ is the initial student obtained after supervised fine-tuning. We consider an original human-labeled training set

$$\mathcal{D}_{\mathrm{H}} = \{(x_i, y_i^{\mathrm{H}})\}_{i=1}^{N}, \tag{1}$$

where x_i is an input and y_i^{H} is the corresponding human reference. Let $\mathcal{X}_{\mathrm{H}} = \{x_i : (x_i, y_i^{\mathrm{H}}) \in \mathcal{D}_{\mathrm{H}}\}$ denote the set of original inputs used for student generation. The human-labeled set is preserved throughout all cycles. IOD augments this fixed set with filtered student-generated trajectories. At cycle $c \in \{1, \ldots, C\}$, the previous student $\pi_S^{(c-1)}$ generates K candidate outputs for each original input x_i:

$$\hat{y}_{i,k}^{(c)} \sim \pi_S^{(c-1)}(\cdot \mid x_i), \quad k = 1, \ldots, K. \tag{2}$$

The newly generated student pool is

$$\mathcal{P}^{(c)} = \{(x_i, \hat{y}_{i,k}^{(c)})\}_{i=1,k=1}^{N,K}. \tag{3}$$

This pool reflects the current student's on-policy generation behavior. The main question is how to select and reuse these student-generated samples across cycles while preventing noisy or poorly aligned generations from dominating training.

For each student-generated pair (x, y), the fixed teacher assigns a length-normalized conditional log-likelihood score:

$$s_T(x, y) = \frac{1}{|y|} \sum_{t=1}^{|y|} \log \pi_T(y_t \mid x, y_{<t}). \tag{4}$$

The length normalization reduces the effect of sequence length when comparing candidates. The teacher is not used to generate training outputs. It is used as a fixed scorer for filtering and as a source of token-level distribution supervision during student update.

3.2 Theoretical Motivation

IOD is motivated by the hypothesis that filtered accumulation can improve on-policy distillation. Prior work suggests that training on model-generated data can be fragile when low-quality synthetic examples are reused without control, and that data selection is an important design axis in language model training (Shumailov et al., 2023; Albalak et al., 2024). If student-generated samples are filtered by a fixed teacher, the retained samples form a set of teacher-approved trajectories. Reusing these trajectories in later cycles increases the amount of on-policy data available for teacher supervision. At the same time, accumulation must be controlled, since indiscriminately keeping all generated samples can introduce noisy or poorly aligned outputs into training.

We provide a justification for teacher-based top-q filtering. The statements below are intended only to motivate the filtering rule. They do not assume that teacher likelihood is a perfect measure of task quality.

Let $\widetilde{\mathcal{A}} = \{(x_i, y_i)\}_{i=1}^{n}$ be a student-generated candidate pool before filtering, and let $z_i = s_T(x_i, y_i)$ be the teacher score. Let $\mathcal{A}_q$ denote the subset containing the top $m = \lceil qn \rceil$ examples under z_i.

Lemma 1 (Top-q filtering increases average teacher score)**.** *For any pre-filter candidate pool $\widetilde{\mathcal{A}}$ and top-q subset $\mathcal{A}_q$,*

$$\frac{1}{|\mathcal{A}_q|} \sum_{(x,y)\in\mathcal{A}_q} s_T(x, y) \geq \frac{1}{|\widetilde{\mathcal{A}}|} \sum_{(x,y)\in\widetilde{\mathcal{A}}} s_T(x, y). \tag{5}$$

To relate score-based selection to training, we use the following assumption.

Assumption 1 (Teacher-score monotonicity)**.** *For a fixed student parameter θ, define*

$$L_\theta(x, y) = \mathbb{E}[\ell_\theta(x, y)].$$

Higher teacher scores are associated with lower expected per-example loss: for two student-generated examples (x_i, y_i) and (x_j, y_j),

$$s_T(x_i, y_i) \geq s_T(x_j, y_j) \Rightarrow L_\theta(x_i, y_i) \leq L_\theta(x_j, y_j). \tag{6}$$

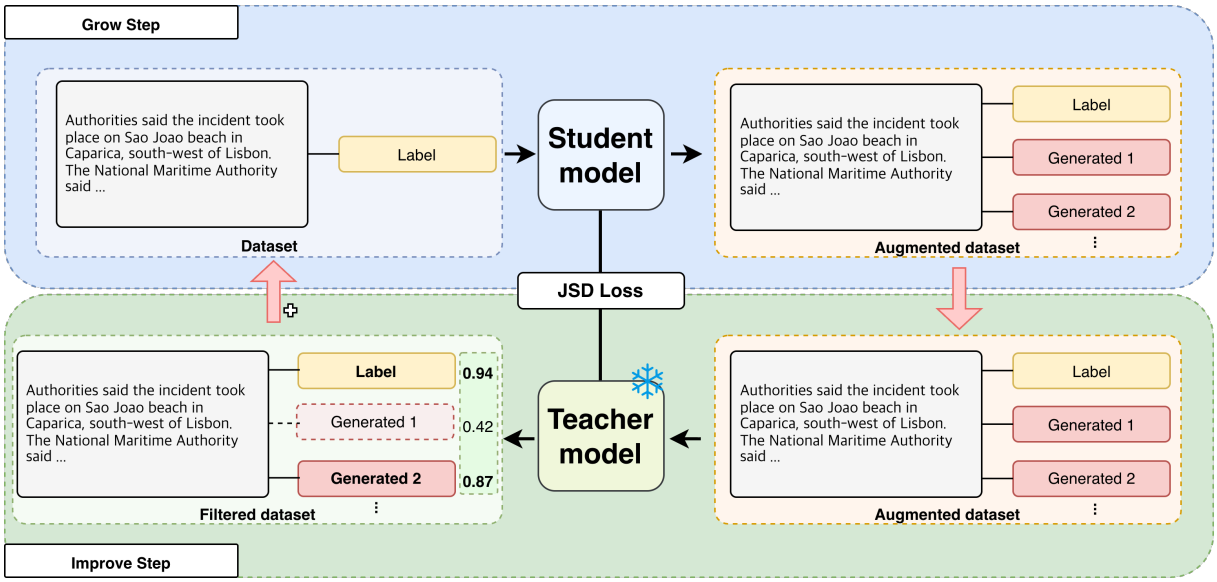

Figure 1: Overview of Iterative On-policy Distillation. In each cycle, the student generates candidates from the original inputs. The fixed teacher scores only the student-generated candidates and retains a high-scoring subset. Previously retained student generations are combined with newly generated candidates and re-filtered. The student is then updated on the original human data together with the filtered student-generated curriculum.

Theorem 1 (Filtering reduces expected loss under monotonicity)**.** *Under Assumption 1, the expected loss over the top-q filtered set is no larger than that over the full candidate pool:*

$$\mathbb{E}_{(x,y)\sim\mathcal{A}_q}[\ell_\theta(x,y)] \leq \mathbb{E}_{(x,y)\sim\widetilde{\mathcal{A}}}[\ell_\theta(x,y)]. \quad (7)$$

Lemma 1 and Theorem 1 clarify the role of teacher-based filtering: if higher teacher scores are positively associated with more useful training examples, then selecting high-scoring samples improves the expected quality of the student-generated training pool. We empirically examine this assumption through teacher-score dynamics and ablations in Section 4. Proofs are provided in Appendix A.1, and Appendix A.2 provides additional analysis on how filtering can reduce the retained fraction of noisy generated samples under a simple retention-separation condition.

3.3 Iterative On-policy Distillation

IOD proceeds through repeated Grow-Evaluate/Filter-Improve cycles. Figure 1 illustrates the cycle-wise data flow, and Algorithm 1 summarizes the procedure.

Candidate generation (Grow). At cycle c, the current student $\pi_S^{(c-1)}$ generates the candidate pool $\mathcal{P}^{(c)}$ as defined in Eq. (3). This step provides the on-policy samples used in the current cycle.

Teacher-based scoring (Evaluate). For each student-generated pair (x, y), we compute the teacher score $s_T(x, y)$ defined in Eq. (4). This score is used only for candidate selection. Since π_T is fixed across all cycles, the scoring criterion remains stable even as the student changes.

Filtering. Filtering is applied only to student-generated examples. The original human-labeled examples are never removed by the teacher-score filter and are preserved throughout all cycles. Given a pre-filter student-generated pool $\widetilde{\mathcal{A}}$ and a retention ratio $q \in (0, 1]$, we define

$$\mathrm{TopQ}(\widetilde{\mathcal{A}}, s_T, q) = \{(x, y) \in \widetilde{\mathcal{A}} : s_T(x, y) \geq \tau_q\}, \quad (8)$$

where τ_q is the cutoff score that retains the top $\lceil q|\widetilde{\mathcal{A}}| \rceil$ candidates. Ties at the cutoff are resolved deterministically. In our implementation, the cutoff is computed globally over the student-generated pool rather than separately for each input.

Filtered accumulation across cycles. Let $\mathcal{A}^{(c)}$ denote the filtered student-generated set retained after cycle c, with $\mathcal{A}^{(0)} = \emptyset$. At cycle c, IOD combines the previously retained student-generated set with the newly generated candidate pool:

$$\widetilde{\mathcal{A}}^{(c)} = \mathcal{A}^{(c-1)} \cup \mathcal{P}^{(c)}. \quad (9)$$

The combined student-generated pool is then re-filtered using the same teacher-based criterion:

$$\mathcal{A}^{(c)} = \mathrm{TopQ}(\widetilde{\mathcal{A}}^{(c)}, s_T, q). \tag{10}$$

This filtered accumulation expands the set of teacher-approved on-policy samples while reducing the risk of accumulating noisy or poorly aligned generations. The full training set used in cycle c is the union of the original human-labeled data and the filtered student-generated curriculum:

$$\mathcal{D}_{\text{train}}^{(c)} = \mathcal{D}_{\text{H}} \cup \mathcal{A}^{(c)}. \tag{11}$$

Thus, the teacher-score filter controls only the student-generated portion of the training set, while the original human data remain available for distillation in every cycle.

Student update (Improve). It is important to distinguish the two uses of teacher feedback. Filtering uses the sequence-level teacher score $s_T(x, y)$ to select student-generated trajectories, whereas student update uses token-level JSD to match teacher and student next-token distributions under the target prefixes. For each $(x, y) \in \mathcal{D}_{\text{train}}^{(c)}$, define

$$p_t = \pi_T(\cdot \mid x, y_{<t}), \quad q_t = \pi_\theta(\cdot \mid x, y_{<t}), \tag{12}$$

where π_θ is the student being optimized. We use the Jensen–Shannon divergence:

$$D_{\text{JSD}}(p_t \| q_t) = \frac{1}{2} D_{\text{KL}}(p_t \| m_t) + \frac{1}{2} D_{\text{KL}}(q_t \| m_t), \quad m_t = \frac{1}{2}(p_t + q_t). \tag{13}$$

The cycle-c objective is

$$\mathcal{L}_{\text{IOD}}^{(c)} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{train}}^{(c)}} \left[\frac{1}{|y|} \sum_{t=1}^{|y|} D_{\text{JSD}}(p_t \| q_t) \right]. \tag{14}$$

After optimization, the updated student becomes $\pi_S^{(c)}$ and is used to generate candidates in the next cycle.

Algorithm 1 Iterative On-policy Distillation

Require: Human data $\mathcal{D}_{\text{H}} = \{(x_i, y_i^{\text{H}})\}_{i=1}^{N}$, fixed teacher π_T, initial student $\pi_S^{(0)}$, Cycles C, candidates per input K, filter ratio q, accumulation flag b_{acc}

$\mathcal{A}^{(0)} \leftarrow \emptyset$
for $c = 1$ to C **do**
  Grow: Generate $\mathcal{P}^{(c)}$ by sampling $\hat{y}_{i,k}^{(c)} \sim \pi_S^{(c-1)}(\cdot \mid x_i)$ for each $(x_i, y_i^{\text{H}}) \in \mathcal{D}_{\text{H}}$ and $k = 1, \ldots, K$
  Pool update:
  if b_{acc} **then**
    $\widetilde{\mathcal{A}}^{(c)} \leftarrow \mathcal{A}^{(c-1)} \cup \mathcal{P}^{(c)}$
  else
    $\widetilde{\mathcal{A}}^{(c)} \leftarrow \mathcal{P}^{(c)}$
  Evaluate/Filter: Compute $s_T(x, y)$ for all $(x, y) \in \widetilde{\mathcal{A}}^{(c)}$, $\mathcal{A}^{(c)} \leftarrow \mathrm{TopQ}(\widetilde{\mathcal{A}}^{(c)}, s_T, q)$
  Improve: Update $\pi_S^{(c-1)}$ on the resulting training set $\mathcal{D}_{\text{train}}^{(c)} \leftarrow \mathcal{D}_{\text{H}} \cup \mathcal{A}^{(c)}$ and optimize $\mathcal{L}_{\text{IOD}}^{(c)}$ on $\mathcal{D}_{\text{train}}^{(c)}$ to obtain $\pi_S^{(c)}$
return $\pi_S^{(C)}$

4 Experiments

We evaluate IOD on two representative sequence prediction tasks: Abstractive summarization and Machine translation. Our experiments are designed to test both overall effectiveness and the contribution of individual components. In addition to the main comparisons, we conduct ablation studies to quantify the effects of teacher-based filtering, cross-cycle curriculum accumulation, and the choice of training objective. We use $C = 3$ cycles in the main experiments, based on preliminary observations that both task performance and teacher-score distributions largely stabilized by the third cycle.

4.1 Experiments setup

4.1.1 Student/Teacher Models

We follow the initialization protocol of On-policy Distillation (Agarwal et al., 2024). As base checkpoints, we start from LM-adapted T5v1.1 models, which are initialized from T5v1.1 and further trained for 100K steps with the language-modeling objective described in the T5 work (Raffel et al., 2020). We then initialize both teacher and student for distillation by additional supervised fine-tuning (SFT) on the original task training set, matching the OPD setup. The teacher remains fixed throughout training and is used only to score candidates for filtering; the student is updated by the objectives described in Equation 14.

4.1.2 Training Objectives

To disentangle the effect of distribution matching from direct quality optimization, we evaluate four training objectives. In all settings, the student π_θ is

updated using the cycle-k curriculum $\mathcal{D}_k$.

Distillation via generalized JSD. Our default objective matches the teacher's and student's next-token distributions on student-generated prefixes using a Jensen–Shannon-type divergence, which is used for stable on-policy distillation (Agarwal et al., 2024). For $(x, y) \sim \mathcal{D}_k$, let $p_t = \pi_T(\cdot \mid x, y_{<t})$ and $q_t = \pi_\theta(\cdot \mid x, y_{<t})$, and define $m_t = \frac{1}{2}(p_t + q_t)$. We minimize

$$\mathcal{L}_{\text{JSD}}^{(c)} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{train}}^{(c)}} \left[\frac{1}{|y|} \sum_{t=1}^{|y|} D_{\text{JSD}}(p_t \| q_t) \right]. \tag{15}$$

Hybrid objective with teacher-derived feedback. We also study a hybrid objective that augments token-level distillation with a REINFORCE term driven by a teacher-derived sequence score $R(x, y)$. Let b be a batch-average baseline to reduce variance.

$$\mathcal{L}_{\text{RL}}(\theta) = -\mathbb{E}_{(x,y)\sim\mathcal{D}_k} \left[(R(x, y) - b) \log \pi_\theta(y \mid x) \right], \tag{16}$$

and optimize $\mathcal{L}_{\text{Total}} = \lambda \mathcal{L}_{\text{JSD}} + (1 - \lambda)\mathcal{L}_{\text{RL}}$. To establish a neutral baseline for this hybrid approach without introducing heavy hyperparameter search, we fix $\lambda = 0.5$, assigning equal importance to both dense token-level matching and sparse sequence-level rewards.

Ablations. To isolate the source of improvements, we consider two ablations. (i) **Filtered SFT:** supervised fine-tuning on the filtered student generations using the standard NLL loss. (ii) **RL-only with behavior cloning:** removing $\mathcal{L}_{\text{JSD}}$ and optimizing $\mathcal{L}_{\text{RL}}$ together with a behavior cloning term, i.e., NLL on the replay curriculum.

4.1.3 Baselines

To validate the effectiveness of IOD, we benchmark against established distillation methods. Instead of comparing absolute scores across disparate studies, which are prone to framework-specific artifacts, we reference the unified baseline evaluations reported by Ko et al. (2024), who benchmarked these methods under the same T5-XL to T5-Small configuration on XSum. Specifically, we consider the following baselines: (i) **Supervised Fine-Tuning (SFT):** The student model trained solely on ground-truth references via NLL, serving as the fundamental lower bound. (ii) **Standard Knowledge Distillation (SeqKD):** An off-policy approach where the student mimics the teacher's distribution on fixed gold data (Kim and Rush, 2016). (iii) **On-Policy Distillation (GKD):** A single-cycle on-policy baseline (Agarwal et al., 2024), which helps quantify the gains achieved specifically through our iterative curriculum construction.

4.2 Case Study: Abstractive Summarization

Task Setup. We evaluate our framework on the XSUM dataset (Narayan et al., 2018), a widely used benchmark for abstractive summarization requiring high-level abstraction and understanding. Performance is measured using standard ROUGE metrics (Lin, 2004) to quantify the quality of generated summaries and demonstrate the performance gains over existing baselines. Following the standard protocol of prior On-Policy Distillation (OPD) work (Agarwal et al., 2024), we truncate inputs to 1024 tokens and targets to 64 tokens. To ensure a fair comparison, all hyperparameters including model architecture (T5-Small student, T5-XL teacher), optimizer settings are kept identical to the OPD baseline, unless otherwise specified for our proposed iterative components.

Main Results. Table 1 summarizes distillation performance on XSum. To ensure fair comparison and mitigate evaluation toolkit variances across codebases, we report the relative capability gap rather than raw scores. For transparency, our T5-XL teacher and IOD student achieved absolute ROUGE-L scores of 39.28 and 35.50, respectively (a 9.63% gap). By strictly following the GKD protocol, we fairly compare this gap against the comprehensive benchmarks in DistiLLM (Ko et al., 2024). As shown, IOD substantially reduces the teacher-student gap, outperforming single-cycle distillation baselines.

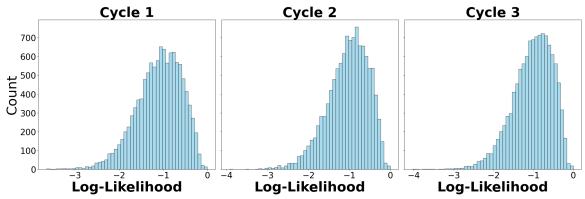

Figure 2: The evolution of Teacher Log-Likelihood distributions across cycles on the XSum test set. The distribution progressively shifts towards zero (higher likelihood), indicating that the student model generates summaries with higher confidence under the teacher model as training proceeds

T5-XL (3B) → T5-Small (0.06B)	
KD (Hinton et al., 2015)	30.42 %
SeqKD (Kim and Rush, 2016)	30.65 %
ImitKD (Lin et al., 2020)	28.83 %
GKD (Agarwal et al., 2024)	27.86 %
IOD (ours)	**9.63** %

Table 1: Comparative results on the XSUM summarization task. Our primary methodological baseline is the on-policy distillation approach, GKD (Agarwal et al., 2024). However, to mitigate evaluation discrepancies arising from cross-framework variance, the absolute scores used to calculate the baseline relative gaps are derived from the unified benchmark reported in DistiLLM. (Ko et al., 2024), which strictly reproduced GKD and other baselines under the identical T5-XL to T5-Small setting.

Method	**ROUGE-L**
JSD, w/ Accum	**35.50**
JSD, w/ Accum, No Filtering	32.79
JSD + SFT, w/ Accum	31.25
JSD, w/o Accum	31.01
JSD + REINFORCE, w/o Accum	30.97
JSD + REINFORCE, w/ Accum	31.06
REINFORCE + BC, w/ Accum	27.31
REINFORCE, w/ Accum	30.96
BC, w/ Accum	30.99

Table 2: Ablation study of training objectives and data strategies on XSum. We report ROUGE-L scores. **w/ Accum** denotes filtered accumulation across cycles, while **w/o Accum** uses only the data generated in the current cycle. **No Filtering** removes teacher-score filtering. JSD denotes the Jensen-Shannon Divergence and BC refers Behavior Cloning

Evolution of Teacher Log-Likelihood. Beyond standard lexical overlap metrics, we analyze the distributional shift of the student's generation quality using the teacher model as a proxy oracle. We compute the **length-normalized log-likelihood** of each student-generated summary y given the source document x under the teacher model P_T: $\frac{1}{|y|}\sum_{t=1}^{|y|}\log P_T(y_t|x, y_{<t})$. This metric quantifies how plausible the student's output is from the teacher's perspective. Figure 2 visualizes the histograms of these scores across training cycles. As the cycles progress, we observe a distinct shift in the distribution towards zero, indicating a consistent increase in the likelihood assigned by the teacher model. This trend demonstrates that the student is progressively learning to generate summaries that align more closely with the teacher's high-confidence regions, validating the effectiveness of our iterative filtering strategy in distilling high-quality trajectories. We provide a detailed theoretical justification for using Teacher LL as a quality proxy and its correlation with generation performance in Appendix A.3.

Ablation Study. To dissect the contributions of each component in our framework, we conducted an extensive ablation study, as summarized in Table 2. We primarily report ROUGE-L scores, as this metric captures the longest common subsequence, thereby reflecting the sentence-level fluency and structural coherence crucial for summarization.

Effect of Filtering & Accumulation. Comparing the full IOD method (*JSD, w/ Accum*) with its variants reveals the critical role of our curriculum strategy. Removing the quality filtering mechanism (*No Filter*) leads to a performance drop from 35.50 to 32.79, indicating that indiscriminately accumulating student generations introduces noise that hampers learning. Furthermore, omitting cross-cycle accumulation (*w/o Accum*) significantly degrades performance to 31.01, confirming that a growing replay buffer of high-quality samples is essential for robust self-improvement.

Effect of Objectives. We observe that hard-label or reward-based variants do not match the performance of token-level distribution matching. The REINFORCE + BC configuration, corresponding to the RL-only ablation, yields the lowest ROUGE-L score. In addition, adding REINFORCE to JSD does not improve over the JSD-based IOD variant. This suggests that under a balanced integration ($\lambda = 0.5$), the sparse sequence-level reward may actually dilute the dense supervisory signals of the teacher's logits. Overall, the combination of JSD-based distillation and quality-filtered accumulation provides the strongest configuration as it does not require exhaustive hyperparameter tuning to achieve high performance.

Sensitivity to Filtering Ratio. We analyze the effect of the filtering ratio q on XSum. Here, q denotes the retained fraction after teacher-score filtering; for example, $q = 70\%$ keeps the top 70% of student-generated candidates. As shown in Table 3, the best performance is obtained at $q = 70\%$. Retaining all generated samples without filtering reduces ROUGE-L, while overly strict filtering also hurts performance. These results suggest that IOD benefits from a balance between teacher-based quality control and sufficient on-policy data coverage.

Initial Decoder-only Study. To examine

Setting	ROUGE-L
$q = 100\%$ (No Filtering)	32.79
$q = 70\%$	**35.50**
$q = 50\%$	35.13
$q = 10\%$	31.26
$q = 0\%$	30.94

Table 3: Sensitivity to the filtering ratio q on XSum. Here, q denotes the fraction of student-generated candidates retained after teacher-score filtering; for example, $q = 70\%$ keeps the top 70% candidates under the teacher score. The $q = 100\%$ setting corresponds to no filtering, where all student-generated candidates are retained.

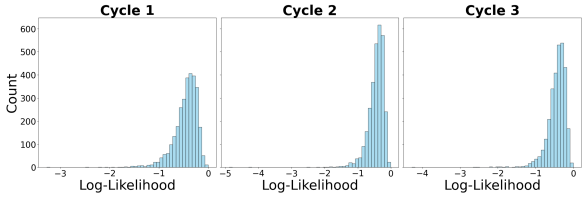

Figure 3: Evolution of Teacher Log-Likelihood distributions on WMT14 En-De. Similar to the summarization task, the student's generation quality consistently improves across cycles, aligning more closely with the teacher's distribution.

whether IOD is specific to encoder-decoder models(T5), we also conduct an initial decoder-only experiments using QWEN models (Team et al., 2024). The setup and results are provided in Appendix B.

4.3 Case Study: Machine Translation

To verify the versatility of our framework beyond summarization, we extend our evaluation to the WMT14 English-to-German (En-De) translation task (Bojar et al., 2014). While summarization requires high-level abstraction, machine translation demands precise syntactic alignment and fidelity. Validating IOD on this distinct modality serves to demonstrate that our iterative curriculum construction is a robust, task-agnostic strategy for self-improvement.

Task Setup. We adhere to the standard WMT14 En-De benchmark protocols, employing the widely used BLEU metric (Papineni et al., 2002) for evaluation. We utilize a T5-Small student and a T5-XL teacher, with both inputs and targets truncated to 80 tokens to ensure a direct comparison with existing baselines.

Method	BLEU
Baseline (SFT)	25.10
GKD (On-Policy) (Agarwal et al., 2024)	25.58
IOD (Ours)	**26.25**

Table 4: Performance on WMT14 En-De translation task. IOD outperforms the single-cycle on-policy baseline, demonstrating its robustness across different domains.

Results. Table 4 presents the translation performance. IOD achieves a BLEU score of **26.25**, surpassing the strong on-policy distillation baseline (GKD) reported in previous literature (25.58) (Agarwal et al., 2024). Furthermore, as illustrated in Figure 3, the distribution of teacher log-likelihoods progressively shifts towards higher values across cycles. This mirrors the trend observed in summarization, confirming that our iterative filtering mechanism consistently successfully guides the student towards the teacher's high-confidence manifold, regardless of the task domain. These results reinforce the position of IOD as a generalized framework capable of enhancing student capabilities across diverse sequence generation challenges.

5 Conclusion

In this work, we proposed Iterative On-policy Distillation (IOD), a multi-cycle framework that connects on-policy distillation with data-filtered self-training by constructing a teacher-filtered curriculum from student-generated trajectories. In each cycle, the student generates on-policy candidates, a fixed teacher scores them with length-normalized likelihood, and selected trajectories are accumulated and re-filtered across cycles. The student is then updated with token-level JSD distribution matching.

Experiments on summarization and machine translation show that IOD improves over supervised fine-tuning and standard on-policy distillation baselines. Ablations demonstrate that both teacher-based filtering and cross-cycle accumulation are important. Overall, our results highlight quality-controlled accumulation as a simple and effective mechanism for organizing student-generated data in sequence generation distillation.

Limitations

Computational overhead of accumulated curriculum. IOD accumulates teacher-filtered trajectories across cycles, which increases the size of the training pool and can raise offline training

cost compared with single-cycle on-policy distillation. This overhead comes from student generation, teacher scoring, and additional student updates over the accumulated curriculum. At the same time, the teacher is used for likelihood-based scoring rather than autoregressive generation, giving IOD a different cost profile from self-training pipelines that require large-model generation or separate reward-model training. The added cost is confined to offline training: at inference time, only the distilled student is used, so the deployment latency and memory footprint remain unchanged.

Scope of decoder-only validation. We include an initial decoder-only study using Qwen2.5 models on XSum, but we do not yet evaluate decoder-only IOD on WMT14 En-De. This leaves open whether teacher-filtered accumulation yields similar benefits for decoder-only machine translation, which we plan to examine in future work.

Auxiliary RL objective. Although we explored REINFORCE-style auxiliary objectives, our experiments did not show clear gains over the JSD-based IOD variant. This suggests that, in our setting, the main benefits come from teacher-filtered data construction and dense token-level distribution matching rather than from sequence-level policy-gradient updates. We do not view this as evidence that reinforcement learning is ineffective for iterative distillation in general. More advanced RL formulations, such as lower-variance advantage estimation or better off-policy correction, may provide additional benefits and are left for future work.

Comparison with reward-based self-training. IOD is related to iterative generate-filter-improve frameworks such as ReST, but it differs in its supervision and computational assumptions. IOD uses the existing distillation teacher as a fixed scorer and does not train an additional reward model or rely on an external evaluator. In contrast, reward-based self-training methods often depend on a task reward, learned reward model, or other external scoring mechanism. For this reason, our main experiments focus on standard on-policy distillation baselines under the same teacher–student setting. A fully controlled comparison with reward-based self-training methods, with matched reward sources and compute budgets, is left for future work.

References

Josh Achiam and 1 others. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

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*.

Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, Colin Raffel, Shiyu Chang, Tatsunori Hashimoto, and William Yang Wang. 2024. A survey on data selection for language models. *arXiv preprint arXiv:2402.16827*.

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*, volume 28.

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

Cristian Bucilua, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 535–541. Association for Computing Machinery.

Aakanksha Chowdhery and 1 others. 2023. PaLM: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

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*.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. 2023. Reinforced self-training (ReST) for language modeling. *arXiv preprint arXiv:2308.08998*.

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

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

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

Alex M. Lamb, Anirudh Goyal, Ying Zhang, Saizheng Zhang, Aaron Courville, and Yoshua Bengio. 2016. Professor forcing: A new algorithm for training recurrent networks. In *Advances in Neural Information Processing Systems*, volume 29.

Alexander Lin, Jeremy Wohlwend, Howard Chen, and Tao Lei. 2020. Autoregressive knowledge distillation through imitation learning. *arXiv preprint arXiv:2009.07253*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv preprint arXiv:1808.08745*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *International Conference on Learning Representations*.

H. Scudder. 1965. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11(3):363–371.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, and Ross Anderson. 2023. The curse of recursion: Training on generated data makes models forget. *arXiv preprint arXiv:2305.17493*.

Qwen Team, An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayang Liu, Jieming Yin, Jingren Zhou, and 33 others. 2024. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. LIMA: Less is more for alignment. In *Advances in Neural Information Processing Systems*, volume 36.

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. 2023. A survey on model compression for large language models. *arXiv preprint arXiv:2308.07633*.

A Appendix

A.1 Proofs for Theoretical Motivation

This section provides proofs for Lemma 1 and Theorem 1. The results are intended as lightweight justification for teacher-based filtering, not as a claim that teacher likelihood is a perfect task-quality metric.

A.1.1 Proof of Lemma 1

Proof. Let

$$\widetilde{\mathcal{A}} = \{(x_i, y_i)\}_{i=1}^{n}$$

be the pre-filter student-generated pool, and let

$$z_i = s_T(x_i, y_i)$$

be the teacher score. Without loss of generality, sort the examples so that

$$z_1 \geq z_2 \geq \cdots \geq z_n.$$

Let $m = \lceil qn \rceil$, where $q \in (0, 1]$, and let

$$\mathcal{A}_q = \{(x_i, y_i)\}_{i=1}^{m}$$

be the top-q subset under the teacher score.

We need to show that the average teacher score of the selected subset is at least the average teacher score of the full pool:

$$\frac{1}{m}\sum_{i=1}^{m} z_i \geq \frac{1}{n}\sum_{i=1}^{n} z_i.$$

If $m = n$, the two sides are equal. Consider $m < n$. Since $z_1, \ldots, z_m$ are the largest m values, the average of the top m values is no smaller than the average of the remaining $n - m$ values:

$$\frac{1}{m}\sum_{i=1}^{m} z_i \geq \frac{1}{n-m}\sum_{i=m+1}^{n} z_i.$$

Let

$$\bar{z}_{\text{top}} = \frac{1}{m}\sum_{i=1}^{m} z_i, \qquad \bar{z}_{\text{rest}} = \frac{1}{n-m}\sum_{i=m+1}^{n} z_i.$$

Then the full-pool average can be written as the weighted average

$$\frac{1}{n}\sum_{i=1}^{n} z_i = \frac{m}{n}\bar{z}_{\text{top}} + \frac{n-m}{n}\bar{z}_{\text{rest}}.$$

Since $\bar{z}_{\text{top}} \geq \bar{z}_{\text{rest}}$, this weighted average is no larger than $\bar{z}_{\text{top}}$. Therefore,

$$\frac{1}{|\mathcal{A}_q|}\sum_{(x,y)\in\mathcal{A}_q} s_T(x, y) \geq \frac{1}{|\widetilde{\mathcal{A}}|}\sum_{(x,y)\in\widetilde{\mathcal{A}}} s_T(x, y).$$

This proves the lemma. □

A.1.2 Proof of Theorem 1

Proof. Let

$$L_i = L_\theta(x_i, y_i) = \mathbb{E}[\ell_\theta(x_i, y_i)]$$

denote the expected per-example loss for the i-th student-generated example. As in the proof above, sort examples by teacher score:

$$z_1 \geq z_2 \geq \cdots \geq z_n,$$

and let $\mathcal{A}_q = \{(x_i, y_i)\}_{i=1}^{m}$ be the selected top-q subset, where $m = \lceil qn \rceil$.

By Assumption 1, higher teacher score implies no larger expected per-example loss. Hence, for any selected example $i \leq m$ and any unselected example $j > m$, we have

$$z_i \geq z_j \quad \Rightarrow \quad L_i \leq L_j.$$

Therefore, the average expected loss over selected examples is no larger than the average expected loss over unselected examples:

$$\bar{L}_{\text{top}} = \frac{1}{m}\sum_{i=1}^{m} L_i \leq \frac{1}{n-m}\sum_{i=m+1}^{n} L_i = \bar{L}_{\text{rest}},$$

when $m < n$. If $m = n$, the result is immediate.

The full-pool average expected loss is

$$\frac{1}{n}\sum_{i=1}^{n} L_i = \frac{m}{n}\bar{L}_{\text{top}} + \frac{n-m}{n}\bar{L}_{\text{rest}}.$$

Since $\bar{L}_{\text{top}} \leq \bar{L}_{\text{rest}}$, the weighted average above is at least $\bar{L}_{\text{top}}$. Thus,

$$\frac{1}{m}\sum_{i=1}^{m} L_i \leq \frac{1}{n}\sum_{i=1}^{n} L_i.$$

Equivalently, under the uniform distributions over the selected set and the full pre-filter pool,

$$\mathbb{E}_{(x,y)\sim\mathcal{A}_q}[\ell_\theta(x, y)] \leq \mathbb{E}_{(x,y)\sim\widetilde{\mathcal{A}}}[\ell_\theta(x, y)].$$

This proves the theorem. □

A.2 Additional Filtering Analysis

This section provides an additional stylized analysis of how filtering can reduce the retained fraction of harmful generated samples. We use the term "noisy" broadly to refer to student-generated samples that are poorly aligned with the teacher scoring criterion or are otherwise undesirable for training. This analysis is conditional on a retention-separation property of the filtering rule.

Lemma 2 (Retained noisy-sample rate under filtering)**.** *Consider a pre-filter student-generated pool $\widetilde{\mathcal{A}}$ whose samples are a mixture of clean and noisy outputs. Let H denote the event that a sample is noisy, and let $\alpha = \Pr(H)$ be the noisy-sample rate in the pre-filter pool. Let R denote the event that a sample is retained by the top-q filtering rule. Assume that the filtering rule satisfies*

$$\Pr(R \mid H) \le \varepsilon_H, \qquad \Pr(R \mid H^c) \ge \varepsilon_C,$$

where $\varepsilon_H, \varepsilon_C \in (0, 1]$. Then the noisy-sample rate after filtering,

$$\alpha_q = \Pr(H \mid R),$$

satisfies

$$\alpha_q \le \frac{\alpha\varepsilon_H}{(1-\alpha)\varepsilon_C + \alpha\varepsilon_H} \le \frac{\alpha\varepsilon_H}{(1-\alpha)\varepsilon_C}. \quad (17)$$

Proof. By Bayes' rule,

$$\Pr(H \mid R) = \frac{\Pr(R \mid H)\Pr(H)}{\Pr(R)}.$$

Using the law of total probability,

$$\Pr(R) = \Pr(R \mid H)\Pr(H) + \Pr(R \mid H^c)\Pr(H^c).$$

Substituting $\Pr(H) = \alpha$ and $\Pr(H^c) = 1 - \alpha$, we obtain

$$\alpha_q = \frac{\Pr(R \mid H)\alpha}{\Pr(R \mid H)\alpha + \Pr(R \mid H^c)(1-\alpha)}.$$

The right-hand side is increasing in $\Pr(R \mid H)$ and decreasing in $\Pr(R \mid H^c)$. Therefore, using $\Pr(R \mid H) \le \varepsilon_H$ and $\Pr(R \mid H^c) \ge \varepsilon_C$,

$$\alpha_q \le \frac{\alpha\varepsilon_H}{\alpha\varepsilon_H + (1-\alpha)\varepsilon_C}.$$

Since

$$\alpha\varepsilon_H + (1-\alpha)\varepsilon_C \ge (1-\alpha)\varepsilon_C,$$

the looser second inequality follows:

$$\alpha_q \le \frac{\alpha\varepsilon_H}{(1-\alpha)\varepsilon_C}.$$

□

[Bounded contribution of noisy samples] Assume that the per-example loss is bounded:

$$0 \le \ell_\theta(x, y) \le L_{\max}.$$

Then the contribution of noisy samples to the expected loss over the filtered set is bounded by

$$\mathbb{E}_{(x,y)\sim\mathcal{A}_q}\left[\ell_\theta(x,y)\mathbf{1}\{(x,y) \in H\}\right] \le L_{\max}\alpha_q, \quad (18)$$

where α_q is bounded as in Lemma 2.

Proof. By the definition of conditional expectation,

$$\begin{aligned}&\mathbb{E}_{(x,y)\sim\mathcal{A}_q}\left[\ell_\theta(x,y)\mathbf{1}\{(x,y) \in H\}\right] \\ &\quad = \Pr(H \mid R)\mathbb{E}[\ell_\theta(x,y) \mid H, R].\end{aligned}$$

Since $\ell_\theta(x, y) \le L_{\max}$,

$$\mathbb{E}[\ell_\theta(x,y) \mid H, R] \le L_{\max}.$$

Thus,

$$\begin{aligned}&\mathbb{E}_{(x,y)\sim\mathcal{A}_q}\left[\ell_\theta(x,y)\mathbf{1}\{(x,y) \in H\}\right] \\ &\quad \le L_{\max}\Pr(H \mid R) = L_{\max}\alpha_q.\end{aligned} \quad (19)$$

□

The lemma and corollary do not assume that the teacher score perfectly detects noisy samples. They show that, whenever the teacher-based filtering rule retains noisy samples at a lower rate than clean samples, the retained pool has a reduced noisy-sample fraction and a bounded noisy-sample loss contribution. This supports the use of filtering together with accumulation: accumulation expands the amount of on-policy data, while re-filtering helps control the quality of the retained pool.

A.3 Teacher Likelihood as a Filtering Score

IOD uses the length-normalized teacher log-likelihood as the fixed scoring criterion for filtering student-generated candidates:

$$s_T(x, y) = \frac{1}{|y|}\sum_{t=1}^{|y|} \log \pi_T(y_t \mid x, y_{<t}). \quad (20)$$

This score is used only for selecting student-generated trajectories. The student update is not performed by matching teacher and student sequence likelihoods; instead, the student is trained with token-level distribution matching as described in Eq. (14). Therefore, the teacher-score dynamics reported in the experiments should be interpreted as a diagnostic of the filtering criterion used by IOD.

Relation to teacher-distribution alignment. For a fixed input x, let $P_S(\cdot \mid x)$ denote the sequence distribution induced by the student generator and $P_T(\cdot \mid x)$ denote the teacher sequence distribution. Ignoring length normalization for the moment, the expected teacher log-likelihood of student samples is

$$\mathbb{E}_{y\sim P_S(\cdot|x)}\left[\log P_T(y \mid x)\right]. \quad (21)$$

This quantity is the negative cross-entropy from the student sequence distribution to the teacher sequence distribution:

$$H(P_S, P_T) = -\mathbb{E}_{y \sim P_S(\cdot|x)} \left[\log P_T(y \mid x) \right]. \tag{22}$$

Thus, when the expected teacher log-likelihood of student generations increases, the generated sequences become more likely under the fixed teacher distribution.

The same quantity is also related to the sequence-level KL divergence:

$$\begin{aligned} D_{\mathrm{KL}}(P_S \| P_T) &= -H(P_S) + H(P_S, P_T) \\ &= -H(P_S) - \mathbb{E}_{y \sim P_S}[\log P_T(y \mid x)]. \end{aligned} \tag{23}$$

where $H(P_S)$ is the entropy of the student sequence distribution. Rearranging gives

$$\mathbb{E}_{y \sim P_S} \left[\log P_T(y \mid x) \right] = -D_{\mathrm{KL}}(P_S \| P_T) - H(P_S). \tag{24}$$

Therefore, when the entropy of the student generator is comparable across cycles, a higher expected teacher log-likelihood is consistent with the student distribution placing more probability mass in regions preferred by the teacher. This motivates using teacher log-likelihood as a diagnostic of teacher alignment. However, because the entropy term can also change, teacher likelihood alone should not be interpreted as a complete measure of task quality.

Length normalization. Raw sequence log-likelihood scales with sequence length:

$$\log P_T(y \mid x) = \sum_{t=1}^{|y|} \log \pi_T(y_t \mid x, y_{<t}). \tag{25}$$

Without length normalization, shorter outputs may receive higher raw log-likelihood simply because they contain fewer probability terms. We therefore use the per-token average in Eq. (20). This makes teacher scores more comparable across student-generated sequences of different lengths.

Interpretation of teacher-score dynamics. In our experiments, we compare the distribution of $s_T(x, y)$ across IOD cycles using the same fixed teacher and the same evaluation protocol. A shift toward higher values, i.e., values closer to zero, indicates that student generations become more likely under the fixed teacher. This supports the role of teacher-based filtering in IOD: the curriculum formed by retained student-generated trajectories becomes increasingly aligned with the teacher scoring criterion. Importantly, this analysis is complementary to task metrics such as ROUGE and BLEU; it does not by itself prove that task quality improves. Task performance is evaluated separately with the corresponding task metrics.

B Appendix:Initial Decoder-only Experiment

To examine whether IOD is specific to encoder–decoder architectures, we conduct an additional decoder-only experiment on XSum. We use Qwen2.5-1.5B as the student model and Qwen2.5-3B as the fixed teacher model. The experiment follows the same IOD recipe used in the main experiments: the student generates candidates on-policy, the teacher scores student-generated candidates, high-scoring samples are retained through teacher-based filtering, and the retained student-generated data are accumulated across cycles. The student is then updated using token-level JSD distribution matching on the resulting training set.

Results. Table 5 reports ROUGE-L scores across IOD cycles. The initial student obtains a validation ROUGE-L score of 31.05. After one IOD cycle, ROUGE-L improves to 32.89, showing that the teacher-filtered accumulation pipeline can also improve a decoder-only student. The model remains above the initial validation performance in later cycles, although the best validation ROUGE-L is reached at Cycle 1. The final model obtains a test ROUGE-L score of 31.72.

Cycle	ROUGE-L
Cycle 0 (SFT)	31.05
Cycle 1	**32.89**
Cycle 2	32.34
Cycle 3	31.94

Table 5: Initial decoder-only study on XSum using Qwen2.5 models. The student is Qwen2.5-1.5B and the fixed teacher is Qwen2.5-3B. ROUGE-L improves from the initial validation score after applying the IOD pipeline with teacher-based filtering and cross-cycle accumulation.

Accumulated student-generated data. The number of retained and accumulated student-generated samples increases across cycles: 571K after Cycle 1, 972K after Cycle 2, and 1.25M after Cycle 3. This confirms that the decoder-only experiment follows the same accumulated curriculum construction used in IOD. These results suggest

that quality-controlled reuse of student-generated trajectories can be applied beyond the T5 encoder–decoder setting. However, since this experiment is conducted only on XSum and does not include separate decoder-only ablations for filtering and accumulation, we treat it as an initial architecture validation rather than a complete study of decoder-only generalization.