REWARD-GUIDED KNOWLEDGE DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

The goal of knowledge distillation is to improve a student model's performance using output samples of a teacher model. The performance can be measured, yet this explicit signal has not been used to control the impact of individual teacher samples during training. In this work, we establish Reward-Guided Knowledge Distillation (RG-KD), a unified and principled framework that weights the distillation loss according to the expected downstream utility of the teacher's predictive distribution. RG-KD further extends to sequential prediction through trajectory-level reward weighting and critic-based step-level value estimation. Extensive evaluations across five diverse tasks demonstrate that RG-KD consistently outperforms unweighted and entropy-weighted baselines as well as recent adaptive and selective distillation methods by effectively weighting teacher signals and aligning the student with high-reward outcomes. RG-KD also improves existing approaches through plug-and-play integration.

1 INTRODUCTION

As foundation models continue to scale, knowledge distillation (KD) (Hinton et al., 2015) is attracting growing interest as a means of deploying more practical and compact models. Standard KD trains a smaller, weaker *student* model to match the predictive distribution of a larger, stronger *teacher* model, typically by minimizing a divergence between their output distributions. This distribution-matching principle has been applied across diverse domains, including vision (Hinton et al., 2015; Romero et al., 2015; Park et al., 2019), language generation and reasoning (Kim & Rush, 2016; Gu et al., 2024; Agarwal et al., 2024; Ko et al., 2024), and sequential decision-making (Rusu et al., 2015; Schmitt et al., 2018; Czarnecki et al., 2019). Various extensions have further improved KD by modifying the distillation target, objective, temperature schedule, or training data distribution (Zhao et al., 2022; Li et al., 2023; He et al., 2025).

Given the current state of the field, we make two key observations. First, the ultimate goal of KD is to improve the student's task performance, which can be measured using a verifier, reward model, or critic. Second, teacher supervision contributes unequally to this improvement across samples (Li et al., 2023; Su et al., 2025). Existing adaptive KD methods (Li et al., 2023; Su et al., 2025; He et al., 2025) instead modulate teacher supervision using heuristic signals such as confidence, entropy, or sample difficulty, which can be viewed as implicit estimates of its utility. This issue becomes particularly important in sequence-level distillation, where the usefulness of teacher supervision can vary across tokens, actions, or states. Some steps may be crucial for achieving high reward, while others may provide limited or misaligned guidance. These observations motivate a principled criterion for controlling the strength of teacher imitation based on the expected downstream utility of the teacher's predictive distribution.

In this work, we establish Reward-Guided Knowledge Distillation (RG-KD), a unified and principled framework that connects the distribution-matching principle of KD with the reward-maximization principle of reinforcement learning. RG-KD weights the distillation loss according to the expected downstream utility of the teacher's predictive distribution, providing a unified framework under which standard KD corresponds to uniform weighting and adaptive KD to heuristic signal-based weighting. For sequential prediction, the same principle naturally extends to trajectory-level weighting using rewards from complete trajectories and step-level weighting using critic-based value estimation at individual prediction steps.

Evaluations across five diverse tasks demonstrate that RG-KD consistently outperforms established baselines, with its trajectory- and step-level extensions driving clear improvements in sequential

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.

generation. Furthermore, RG-KD outperforms recent adaptive and selective distillation methods and improves existing approaches such as GKD (Agarwal et al., 2024) and DistiLLM (Ko et al., 2024) through plug-and-play integration, highlighting the flexibility and compatibility of the framework.

Our contributions are summarized as follows:

- We establish Reward-Guided Knowledge Distillation (RG-KD), a unified and principled framework that controls distillation strength based on the expected downstream utility of the teacher's predictive distribution.
- We extend RG-KD from sample-level to sequence-level distillation, introducing reward-based trajectory-level weighting and critic-based step-level weighting that estimates utility at individual prediction steps.
- We validate RG-KD across five diverse tasks and demonstrate its effectiveness against recent distillation methods and its compatibility with existing approaches.

2 Related Work

Standard and Autoregressive Knowledge Distillation. Knowledge Distillation (KD) (Hinton et al., 2015) trains a compact student model to match the predictive distribution of a larger teacher. Beyond output distributions, prior work has explored intermediate representations (Romero et al., 2015) and inter-sample relations (Park et al., 2019). KD has also been extended to autoregressive models: SeqKD (Kim & Rush, 2016) distills teacher-generated sequences, while recent LLM methods explore alternative divergence objectives and training distributions, including MiniLLM (Gu et al., 2024), GKD (Agarwal et al., 2024), and DistiLLM (Ko et al., 2024). In contrast, RG-KD focuses on how strongly each teacher distribution should be distilled based on its expected downstream utility.

Adaptive and Selective Knowledge Distillation. Several studies have questioned the uniform use of teacher supervision in KD. CA-MKD (Zhang et al., 2022) assigns reliability scores to teacher predictions, while Selective KD (Wang et al., 2021) selectively applies teacher supervision where it is beneficial. CTKD (Li et al., 2023) adapts distillation difficulty through a learned temperature, and EA-KD (Su et al., 2025) reweights the distillation loss using teacher and student entropy. Recent autoregressive and LLM distillation methods further adapt supervision at the token or sample level: ATKD (Zhong et al., 2024) adjusts token-level teaching modes, DA-KD (He et al., 2025) modifies the distillation data based on sample difficulty, and TSD-KD (Kim & Baek, 2026) selectively distills informative reasoning tokens. While these methods relax uniform teacher supervision, their adaptation signals are primarily based on reliability, entropy, difficulty, or selection heuristics. In contrast, RG-KD grounds the weighting signal in the expected downstream utility of the teacher's predictive distribution, estimated through reward or value signals.

Reward Signals in Language Generation. Reward signals have recently been incorporated into knowledge distillation for language models. GKD (Agarwal et al., 2024) trains students on self-generated on-policy sequences with teacher feedback and shows that distillation can be integrated with RL fine-tuning. ADPA (Gao et al., 2025) uses teacher-derived advantages to transfer preference-alignment knowledge, while RLKD (Xu et al., 2026) uses reward-based structural alignment between teacher and student reasoning paths for RL-based reasoning distillation. ASKD (Zhang et al., 2026) further adjusts the skewness of the KD objective according to sample quality signals. These methods use reward or value signals for training distribution design, alignment, RL-style updates, or objective adaptation. In contrast, RG-KD uses reward- or value-based utility estimates to evaluate the utility of the teacher-estimated distribution itself and directly reweights the distillation loss accordingly.

3 Methodology

In this section, we establish a unified and principled framework that connects reinforcement learning (RL) and knowledge distillation (KD) through task utility. By incorporating task reward into KD, the framework provides a theoretical foundation for adaptively controlling the strength of teacher imitation. We first show how KD and RL can be integrated within a unified distillation objective, and then extend this framework to sequential prediction and decision-making, from trajectory-level weighting to critic-based step-level weighting using value estimation from RL.

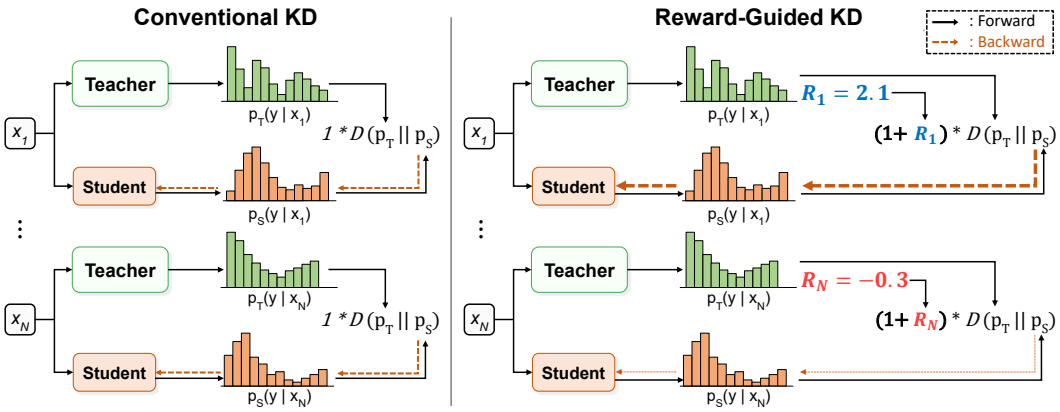


Figure 1: **Comparison between conventional KD (Left) and Reward-Guided KD (Right).** While conventional KD uniformly matches the student to the teacher distribution, RG-KD reweights the KD loss according to the estimated utility of each teacher distribution, emphasizing beneficial teacher supervision and down-weighting less useful guidance.

3.1 Knowledge Distillation

Knowledge distillation (KD) aims to train a compact student model by transferring knowledge from a larger or more capable teacher model. For an input x, standard KD encourages the student distribution $p_S^\theta(\cdot \mid x)$ to match the teacher distribution $p_T(\cdot \mid x)$. This objective can be generally written as

$$\mathcal{L}_{\mathrm{KD}}(\theta) = \mathbb{E}_x \left[D\left(p_T(\cdot \mid x) \,\|\, p_S^\theta(\cdot \mid x)\right) \right], \tag{1}$$

where $D(\cdot\|\cdot)$ denotes a divergence measure between two distributions, such as Kullback–Leibler divergence (KLD) and Jensen–Shannon divergence (JSD).

The above formulation optimizes distribution matching for each sample, but does not explicitly assess whether the teacher distribution provides a useful distillation target for that sample.

However, the teacher distribution may not be equally beneficial as a distillation target for all samples. Even if the teacher is stronger than the student on average, its predictive distribution for a given sample may still assign probability mass to outputs that are suboptimal, uninformative, or difficult for the student to benefit from. This creates a gap between the training principle of KD—matching the teacher distribution—and the ultimate goal of learning—improving task performance. Bridging this gap motivates incorporating task reward into the distillation objective to adapt the strength of teacher imitation according to the task utility of the teacher distribution for each sample, rather than applying it uniformly.

3.2 Reward-Guided Knowledge Distillation

Motivated by this gap between teacher imitation and task-oriented learning, we establish Reward-Guided Knowledge Distillation (RG-KD), which provides a principled learning framework that connects knowledge distillation with the reward-maximization principle of reinforcement learning. When reward signals are available, they provide a principled basis for estimating the expected utility of the teacher distribution for each sample. RG-KD leverages this information to evaluate how beneficial the teacher distribution is as a distillation target and adjusts the distillation strength accordingly. Instead of assigning a uniform weight to all teacher distributions, RG-KD places larger weights on teacher distributions that are expected to promote useful student learning, while down-weighting those that are less reliable or less beneficial.

We first define the expected return of a teacher distribution with respect to the return $r(x, y)$. Given an input sample x and the teacher-estimated distribution $p_T(\cdot \mid x)$, the expected return of the teacher distribution is defined as

$$R(x, p_T) = \mathbb{E}_{y \sim p_T(\cdot|x)} \left[r(x, y) \right]. \tag{2}$$

This expected return characterizes the task utility of the teacher distribution for a given sample. A higher expected return indicates that matching the student to the teacher distribution for that sample provides greater downstream task utility than matching to teacher distributions with lower expected returns. This motivates stronger teacher imitation for samples with higher expected returns.

Using this return, we define the RG-KD objective as

$$\mathcal{L}_{\text{RG-KD}}(\theta) = \mathbb{E}_x \left[(1 + R(x, p_T))\, D\left(p_T(\cdot \mid x) \,\|\, p_S^\theta(\cdot \mid x)\right) \right] \tag{3}$$

$$\approx \mathbb{E}_x \left[\left(1 + \frac{1}{s} \sum_{i=1}^{s} r(x, y_i) \right) D\left(p_T(\cdot \mid x) \,\|\, p_S^\theta(\cdot \mid x)\right) \right], \quad y_i \sim p_T(\cdot \mid x). \tag{4}$$

The constant term 1 within the weight $1 + R$ preserves the original KD signal, while the reward term adjusts the influence of each teacher distribution according to its estimated utility as shown in Eq. 3. Thus, teacher distributions with higher estimated return exert stronger influence on student learning, whereas those with lower return are relatively down-weighted. In practice, the expected return in Eq. 3 is estimated by Monte Carlo sampling with s samples from $p_T(\cdot \mid x)$, as described in Eq. 4.

The formulation is agnostic to the reward source: $r(x, y)$ can be instantiated by any task-relevant utility signal, such as a task metric, verifier, reward model, or environment return; we examine different reward sources in Appendix E. Importantly, RG-KD does not replace teacher imitation with reward optimization, but uses reward as a task-oriented signal to modulate the distillation process.

3.3 Reward-Guided Knowledge Distillation for Sequential Prediction

The formulation in Sec. 3.2 establishes task utility as the principle for controlling the strength of teacher imitation. In sequential prediction, however, applying this principle requires accounting for interdependent decisions, delayed reward, and variations in the utility of teacher supervision across trajectories and intermediate histories. To make this framework operational in such settings, we consider two practical realizations of the same principle: trajectory-level weighting using the return of a complete trajectory, and step-level weighting using critic-based value estimation from RL to estimate the future utility of each intermediate history.

3.3.1 Trajectory-level RG-KD

For sequential prediction, we write the output as an autoregressive trajectory $z_{1:T} = (z_1, \ldots, z_T)$ and denote the history at step t by $h_t = (x, z_{<t})$. The teacher and student therefore induce trajectory distributions through their step-wise conditional distributions $p_T(z_t \mid h_t)$ and $p_S^\theta(z_t \mid h_t)$, respectively. We also replace the sample-level reward with a trajectory-level reward $r(x, z_{1:T})$.

Starting from the sample-level RG-KD objective in Eq. 3, we derive a trajectory-level formulation by factorizing the autoregressive trajectory distribution over prediction steps. After reorganizing the step-wise terms, we obtain the following trajectory-level RG-KD objective:

$$\mathcal{L}_{\text{RG-KD}}(\theta) = \mathbb{E}_x \left[\left(1 + \mathbb{E}_{z_{1:T} \sim p_T(\cdot|x)} \left[r(x, z_{1:T}) \right] \right) D\left(p_T(z_{1:T} \mid x) \,\|\, p_S^\theta(z_{1:T} \mid x) \right) \right] \tag{5}$$

$$\approx \mathbb{E}_x \left[\left(1 + \mathbb{E}_{z_{1:T} \sim p_T(\cdot|x)} \left[r(x, z_{1:T}) \right] \right) \mathbb{E}_{z_{1:T} \sim p_T(\cdot|x)} \left[\sum_{t=1}^{T} D\left(p_T(z_t \mid h_t) \,\|\, p_S^\theta(z_t \mid h_t) \right) \right] \right]. \tag{6}$$

For practical sequence modeling, we approximate the sequence-level divergence by summing over step-wise conditional terms. While such decomposition is generally approximate, it is exact for forward KLD, whereas the corresponding step-wise formulations for JSD and Skew KL remain practical approximations. Further details are provided in Appendix A.

The final approximation comprises two distinct expectations, both taken over the output trajectories of the teacher distribution p_T for a given input x. Theoretically, the first expectation yields a single, global weighting factor that represents the expected utility of the teacher's distribution. This scalar weight is then applied to the second factor—the total distillation loss—which itself is computed as an expectation across the entire space of possible trajectories. Essentially, the mean quality of the teacher's output determines the overall importance of distilling its knowledge for that specific input.

In practice, however, evaluating these exhaustive expectations is computationally intractable. We typically resort to Monte Carlo (MC) sampling: the first expectation is used to estimate the reward-guided weight, while the second facilitates distillation over a subset of sampled trajectories. Therefore, naively applying independent sampling to these two expectations introduces a critical misalignment: a weight estimated from one specific trajectory would be used to scale the distillation loss of an entirely different trajectory. To mitigate this, we employ a coupled sampling heuristic, where a single trajectory is sampled and applied to both factors simultaneously. This approach effectively moves the expectation outside the product, resulting in a unified objective where each trajectory's distillation loss is weighted by its own specific reward where the "product of expectations" is replaced by a "single expectation of the product," yielding:

$$\mathcal{L}_{\text{RG-KD}}(\theta) \approx \mathbb{E}_x \mathbb{E}_{z_{1:T} \sim p_T(\cdot|x)} \left[(1 + r(x, z_{1:T})) \sum_{t=1}^{T} D_{\text{KL}} \left(p_T(\cdot \mid h_t) \,\|\, p_S^\theta(\cdot \mid h_t) \right) \right]. \tag{7}$$

We refer to $1 + r(x, z_{1:T})$ as the trajectory-level reward-guided weight, which provides a trajectory-level realization of the RG-KD principle by weighting the distillation loss of each sampled trajectory according to its observed reward.

3.3.2 Step-level RG-KD with a Critic

The trajectory-level RG-KD objective directly extends reward-guided distillation to sequential prediction. However, it assigns a single, global trajectory-level reward to all conditional distillation terms. Since only specific decisions within a sequence may actually be responsible for the final reward, uniformly weighting every prediction step can be too coarse and introduces high variance. This motivates a finer-grained, step-level weighting scheme.

To this end, we cast the autoregressive generation process as a Markov Decision Process where the state is defined by the full generation history $h_t = (x, z_{<t})$ (Sutton & Barto, 2018). We can then apply the standard reward-guided distillation at each individual timestep by substituting the history h_t for the input x:

$$\mathcal{L}_{\text{step-RG-KD}}(\theta) = \mathbb{E}_{h_t} \left[(1 + R(h_t, p_T)) \, D \left(p_T(z_t \mid h_t) \,\|\, p_S^\theta(z_t \mid h_t) \right) \right] \tag{8}$$

where the teacher and student distributions now predict a single next output given state h_t.

The primary challenge in this formulation arises in measuring the expected return $R(h_t, p_T)$ for the teacher distribution. In sequence generation, reward signals are typically sparse and are often only available after the final output of the sequence has been generated. This is precisely the delayed-reward estimation problem extensively studied in reinforcement learning. We therefore leverage a critic $V_\phi(h_t)$, which estimates the expected future return from a given history h_t (Sutton & Barto, 2018; Konda & Tsitsiklis, 2000).

Value Estimation via the Student Policy. Leveraging this critic, the expected utility $R(h_t, p_T)$ for a teacher's distribution is obtained by simulating a single token prediction from the teacher and evaluating the resulting subsequent state:

$$R(h_t, p_T) = \mathbb{E}_{z_t \sim p_T(\cdot|h_t)} \left[V_\phi([h_t; z_t]) \right] \approx \frac{1}{s} \sum_{i=1}^{s} V_\phi([h_t; z_t^{(i)}]) \tag{9}$$

where $[h_t; z_t]$ denotes the next state h_{t+1} formed by inserting the predicted token z_t into the history h_t, and the expectation is approximated via Monte Carlo sampling. Crucially, the critic V_ϕ is designed to estimate the expected return under the *student's* policy, rather than the teacher's. The rationale for this is highly practical: once the distillation is complete, the student model is the one that the system will deploy autonomously. Therefore, the value of a single-step action taken by the teacher should be measured by how well the student can complete the remaining sequence from that new state. Accordingly, the critic serves as a step-level estimator of the same expected utility introduced in Sec. 3.2, rather than defining a separate distillation criterion.

Online Critic Training. To provide accurate value estimates during distillation, the critic must be trained online alongside the student model. At every time step, we construct a hybrid rollout: we sample a single-step output from the teacher, and then predict the remaining sequential outputs using the current student policy p_S^θ. Upon reaching the end of the sequence, we obtain the ground-truth

reward for this hybrid trajectory. The critic is then updated using a Temporal Difference loss to minimize the error between its state-value prediction and the estimated return from the rollout (Sutton, 1988; Sutton & Barto, 2018).

3.3.3 Trade-offs Between Trajectory-level and Step-level RG-KD

The choice between trajectory-level and step-level RG-KD presents a fundamental trade-off across four key dimensions. **Computational efficiency:** The trajectory-level approach introduces zero architectural overhead during distillation, whereas the step-level approach incurs a significant computational burden due to active critic training, separate hybrid rollouts, and additional forward passes for value estimation. **Credit assignment:** Trajectory distillation suffers from signal sparsity by assigning a single global reward to an entire sequence, potentially penalizing optimal local choices. In contrast, step-level distillation resolves this by providing dense, fine-grained reward signals that accurately weight each conditional term based on its specific contribution to downstream utility. **Estimation accuracy:** By utilizing the actual, observed return of a complete sequence, the trajectory-level method avoids the bias inherent in learned value estimation, while the effectiveness of the step-level approach remains inherently bounded by the approximation errors of the critic's predictions. **Rollout flexibility:** Finally, the trajectory formulation is strictly tied to complete teacher rollouts, whereas the step-level approach—armed with a critic capable of evaluating arbitrary intermediate states—seamlessly extends to diverse trajectory samples, including student-generated or hybrid rollouts adopted in several studies (Agarwal et al., 2024; Lin et al., 2020; Gu et al., 2024).

3.4 Understanding Existing Distillation Methods under RG-KD

Existing adaptive KD methods can be interpreted as heuristic special cases of the RG-KD framework. Methods based on confidence, entropy, sample difficulty, or related signals implicitly estimate the utility of teacher supervision through different heuristic proxies and use these estimates to modulate distillation strength. RG-KD generalizes this perspective by formulating distillation weighting in terms of the expected downstream utility of the teacher's predictive distribution, providing a unified basis for interpreting and designing adaptive distillation methods. Under this view, standard KD is recovered as a special case with uniform weighting, where the estimated utility is treated as constant across samples. Reward-based filtering and resampling methods (Padarha, 2025) can likewise be viewed as restricted, offline trajectory-level instantiations of this framework: they estimate the utility of teacher supervision from rewards associated with completed responses and use these estimates to modify the training distribution or assign post-hoc weights to individual responses. RG-KD places these approaches within a broader utility-based formulation that can operate at different levels of granularity, naturally extending from trajectory-level weighting to step-level weighting through the utility of conditional teacher distributions. We further provide an empirical comparison with representative reward-based trajectory selection and weighting methods in Appendix F. This unified perspective also provides a natural bridge between knowledge distillation and reinforcement learning. Our critic-based step-level RG-KD provides one concrete example by incorporating value estimation developed in RL into distillation. More broadly, RG-KD provides a foundation for bringing techniques developed in RL into knowledge distillation within a unified utility-based framework.

4 Experiments

4.1 Experimental Settings

Tasks and Environments. To rigorously assess the versatility and robustness of RG-KD, we conduct experiments across four distinct domains spanning both single-step and sequential decision-making. First, we evaluate **semantic segmentation** to validate the efficacy of RG-KD in an isolated, single-step prediction context. Second, we utilize a **Maze navigation** environment as a controlled, toy setup to explicitly analyze the dynamics of sequence-level prediction. Then, we benchmark on **Atari games** to investigate the performance of different RG-KD variants across complex, high-dimensional environments. This suite allows us to demonstrate how varying game dynamics dictate the relative effectiveness of trajectory-level versus step-level distillation. Finally, we scale our method to **LLM tasks**, specifically targeting code generation and mathematical reasoning, to highlight the practical applicability of RG-KD in state-of-the-art autoregressive generation.

Baseline Methods and Divergence Metrics. We compare our proposed **RG-KD** against two primary distillation paradigms: **Standard KD** (Hinton et al., 2015), which represents the conventional,

Table 1: **Results on semantic segmentation and maze reasoning.** For semantic segmentation, we evaluate on ADE20K (Zhou et al., 2017) using mIoU, with a fine-tuned SegFormer-B5 teacher and a MiT-B0 student (Xie et al., 2021). For maze reasoning, we report success rate (SR), with teacher and student scores of 76.04 and 60.64, respectively. TL and SL denote trajectory- and step-level weighting, respectively, and the best result for each divergence objective is highlighted in bold.

(a) Semantic segmentation results on ADE20K.

Method	**KLD**	**JSD**	**SKL**
Standard KD	31.27	30.90	30.73
EA-KD	31.36	30.81	30.47
RG-KD (Ours)	**31.65**	**31.11**	**31.26**

(b) Maze reasoning results.

Method	**TL/SL**	**KLD**	**JSD**	**SKL**
Standard KD	-	62.12	62.04	62.92
EA-KD	TL	61.76	61.96	62.16
RG-KD (Ours)	TL	63.28	63.00	63.68
RG-KD (Ours)	SL	**64.80**	**64.48**	**64.16**

Table 2: **Results on Atari.** We report mean episodic return over 50 test seeds. The upper and lower blocks use teacher-generated and student-generated trajectories, respectively. TL and SL denote trajectory-level and step-level weighting, and the best result in each game is highlighted in bold.

		Teacher: 4325.0 *Student: 1025.0*			*Teacher: 98.0* *Student: 29.0*			*Teacher: 18902.2* *Student: 3492.2*		
		Qbert			**Amidar**			**Phoenix**		
Method	**TL/SL**	**KLD**	**JSD**	**SKL**	**KLD**	**JSD**	**SKL**	**KLD**	**JSD**	**SKL**
Teacher Generated Trajectories										
Standard KD	-	7950.0	4450.0	8575.0	337.0	126.0	386.0	473.0	894.6	641.8
EA-KD	TL	7950.0	8075.0	7950.0	403.0	97.0	398.0	456.0	795.2	766.0
RG-KD (Ours)	TL	11475.0	12000.0	11550.0	511.0	237.0	502.0	501.6	1637.0	1210.6
RG-KD (Ours)	SL	8050.0	4425.0	8050.0	**538.0**	453.0	502.0	584.8	2843.2	2833.4
Student Generated Trajectories										
Standard KD	-	7825.0	9025.0	4950.0	274.0	31.0	387.0	6058.0	5889.2	8154.0
RG-KD (Ours)	SL	8550.0	9050.0	**12200.0**	241.0	349.0	338.0	15005.2	9670.4	**15646.6**

uniformly weighted distillation approach, and **Entropy-based Adaptive KD (EA-KD)** (Su et al., 2025), which dynamically scales distillation weights based on the teacher's predictive uncertainty. Furthermore, to thoroughly understand the interaction between reward weighting and distribution matching, we evaluate these distillation objectives across three distinct divergence metrics: the forward Kullback-Leibler Divergence (KLD) (Hinton et al., 2015), the generalized Jensen-Shannon Divergence (JSD) (Agarwal et al., 2024), and the Skew Kullback-Leibler Divergence (SKL) (Ko et al., 2024).

Implementation Details. To reduce the variance of the reward multiplier and stabilize the distillation process, we employ an advantage-based formulation. Specifically, we set the estimation of the student model's expected reward as a baseline, ensuring the distillation weight reflects the teacher's relative advantage over the student's current capability. For RG-KD variants requiring step-level value estimation, the critic network is initially pretrained using offline rollouts generated by the student policy. Following this initialization, the critic is continually fine-tuned online throughout the distillation process to accurately track and evaluate the student's evolving policy. Detailed experimental setups for each experiment are provided in Appendix B.

4.2 Experiments on Semantic Segmentation

Table 1a reports the semantic segmentation results on ADE20K. EA-KD yields mixed performance relative to standard KD, providing a slight improvement with KLD but underperforming with JSD and SKL. This suggests that predictive entropy alone may be insufficient to reliably capture the true quality of the teacher's predictions. In contrast, RG-KD leverages a concrete reward signal (mIoU) to accurately measure the quality of the estimated distribution, consistently outperforming both standard KD and EA-KD across all evaluated divergence metrics. These results demonstrate the effectiveness of utility-guided weighting in the single-step prediction setting. Further results on image classification, another single-step prediction setting, are provided in Appendix G, with detailed semantic segmentation settings in Appendix B.1.

Table 3: **Results on code generation.** We report pass@1 accuracy on HumanEval and MBPP. Avg. aggregates HumanEval, MBPP. TL and SL denote trajectory-level and step-level weighting, respectively. The best result in each column is highlighted in bold.

		Teacher: 91.46 *Student: 70.73*			*Teacher: 82.28* *Student: 68.78*			*Teacher: 86.87* *Student: 69.76*		
		HumanEval			**MBPP**			**Avg.**		
Method	**TL/SL**	**KLD**	**JSD**	**SKL**	**KLD**	**JSD**	**SKL**	**KLD**	**JSD**	**SKL**
Teacher Generated Trajectories										
Standard KD	-	74.39	75.61	75.61	71.96	72.75	71.69	73.18	74.18	73.65
EA-KD	TL	75.61	75.61	74.39	71.69	72.49	71.69	73.65	74.05	73.04
RG-KD (Ours)	TL	**76.83**	**76.22**	**76.22**	72.75	73.28	**72.75**	**74.79**	74.75	**74.49**
RG-KD (Ours)	SL	73.17	75.61	75.61	**73.28**	**74.87**	**72.75**	73.23	**75.24**	74.18

Table 4: **Results on mathematical reasoning.** We report accuracy on GSM8K and MATH. Avg. denotes the average over the two benchmarks. TL denotes trajectory-level weighting, and the best result in each column is highlighted in bold.

		Teacher: 89.39 *Student: 76.73*			*Teacher: 73.96* *Student: 60.62*			*Teacher: 81.68* *Student: 68.67*		
		GSM8K			**MATH**			**Avg.**		
Method	**TL/SL**	**KLD**	**JSD**	**SKL**	**KLD**	**JSD**	**SKL**	**KLD**	**JSD**	**SKL**
Teacher Generated Trajectories										
Standard KD	-	77.41	77.56	77.86	66.08	67.00	66.42	71.74	72.28	72.14
EA-KD	TL	77.64	77.71	77.48	66.54	66.98	66.60	72.09	72.35	72.04
RG-KD (Ours)	TL	77.64	78.09	78.17	66.56	**67.24**	66.98	72.10	72.66	72.57
RG-KD (Ours)	SL	**80.52**	**79.83**	**80.67**	**67.22**	67.20	**67.38**	**73.87**	**73.52**	**74.02**

4.3 Experiments on Maze Reasoning

Table 1b reports the results on maze reasoning. While all evaluated KD methods improve upon the student's initial performance, EA-KD consistently underperforms standard KD across all divergence metrics. This further suggests that entropy-based quality estimation may be an insufficient proxy for the utility of teacher supervision in sequential prediction settings. In contrast, both trajectory-level and step-level RG-KD consistently outperform standard KD, with step-level RG-KD achieving the best performance across all divergence metrics. This indicates the benefit of finer-grained utility estimation in this maze environment. As shown in the subsequent Atari experiments, however, the relative advantage of trajectory-level and step-level weighting varies depending on the underlying task dynamics. Detailed experimental settings are provided in Appendix B.2.

4.4 Experiments on Atari

Table 2 reports the results across the three Atari games. The effectiveness of distillation varies substantially with both the rollout source and the underlying game dynamics. In particular, standard KD on teacher-generated trajectories performs strongly in some games but severely degrades the student's initial performance in others, highlighting the importance of trajectory selection. This observation is consistent with prior work studying the choice of distillation trajectories (Agarwal et al., 2024; Czarnecki et al., 2019). EA-KD also shows mixed results, remaining largely comparable to standard KD rather than providing consistent improvements.

In contrast, RG-KD provides substantial gains when paired with an appropriate weighting granularity and rollout source, although the preferred configuration varies across games. On *Qbert*, trajectory-level RG-KD on teacher-generated trajectories achieves the best performance, whereas on *Amidar*, step-level RG-KD performs best. The clearest case appears in *Phoenix*, where standard KD on teacher-generated trajectories substantially degrades performance. Step-level RG-KD on student-generated trajectories overcomes this failure mode and achieves the strongest overall performance, improving over the corresponding standard KD baselines by 101.3% on average. This recovery is enabled by the critic-based step-level formulation, which allows RG-KD to evaluate and

Table 5: **Comparison and integration with existing distillation methods.** We report accuracy on GSM8K and MATH for (a) comparisons with recent adaptive and selective methods and (b) plug-and-play integration with existing approaches. The best result in each column is highlighted in bold.

(a) Comparison with adaptive and selective distillation methods.

Method	GSM8K	MATH
TSD-KD (Kim & Baek, 2026)	78.54	66.22
ASKD (Zhang et al., 2026)	79.53	67.44
RG-KD (Ours)	**80.52**	**68.14**

(b) Plug-and-play integration of RG-KD with existing distillation methods.

Method	GSM8K	MATH
GKD (Agarwal et al., 2024)	78.70	67.42
+ RG-KD	**80.36**	**68.42**
DistiLLM (Ko et al., 2024)	77.56	66.26
+ RG-KD	**79.08**	**66.46**

distill from student-generated trajectories. Overall, these results demonstrate the flexibility of RG-KD across different task dynamics. Detailed experimental settings are provided in Appendix B.3.

4.5 Experiments on LLM Tasks: Code Generation and Mathematical Reasoning

Table 3 presents the code generation results on HumanEval and MBPP. While all evaluated distillation methods improve upon the student baseline, RG-KD achieves the best overall performance on both benchmarks, with the preferred weighting granularity varying across tasks. On HumanEval, trajectory-level RG-KD achieves the strongest performance, whereas on MBPP, step-level RG-KD performs better in several configurations. Extended results on HumanEval+ and MBPP+, together with results on student-generated trajectories, are provided in Appendix H.

For mathematical reasoning, Table 4 reports the results on GSM8K and MATH. Step-level RG-KD achieves the best overall performance on both benchmarks and provides particularly strong gains on GSM8K, demonstrating the benefit of finer-grained utility estimation for mathematical reasoning. Results using student-generated trajectories are further provided in Appendix I. Detailed experimental settings for both tasks are provided in Appendix B.4.

4.6 Comparison and Integration with Existing Distillation Methods

We further evaluate RG-KD against recent adaptive and selective distillation methods and examine its compatibility with existing distillation approaches. First, we compare step-level RG-KD with TSD-KD (Kim & Baek, 2026) and ASKD (Zhang et al., 2026) on GSM8K and MATH. TSD-KD selectively distills informative reasoning tokens, while ASKD adapts the skewness of the KD objective according to sample-quality signals. As shown in Table 5a, RG-KD achieves the best performance on both benchmarks, outperforming both TSD-KD and ASKD. These results demonstrate the effectiveness of reward-guided distillation relative to recent adaptive and selective approaches. Detailed experimental settings are provided in Appendix B.5.

Beyond direct comparison, we evaluate RG-KD as a plug-and-play component of existing distillation methods. Specifically, we integrate RG-KD with GKD (Agarwal et al., 2024) and DistiLLM (Ko et al., 2024). As shown in Table 5b, incorporating RG-KD consistently improves both methods across GSM8K and MATH. This suggests that utility-guided weighting is complementary to different distillation strategies, including methods based on student-generated trajectories and adaptive divergence objectives. Overall, these results demonstrate that RG-KD can be readily integrated with existing distillation approaches to further improve their effectiveness. Detailed experimental settings for the plug-and-play integration are provided in Appendix B.6.

5 Conclusion

In this work, we established Reward-Guided Knowledge Distillation (RG-KD), a unified and principled framework that controls distillation strength based on the expected downstream utility of the teacher's predictive distribution. We extended this principle from sample-level to sequential distillation through trajectory-level reward weighting and critic-based step-level value estimation. Across diverse tasks, RG-KD consistently improves over established baselines across sample-level, trajectory-level, and step-level distillation settings. RG-KD also outperforms recent adaptive and selective distillation methods and improves existing approaches through plug-and-play integration, demonstrating the flexibility of the proposed framework.

AI USE STATEMENT

In this work, generative AI tools were used only to assist with drafting and editing parts of the manuscript, including improving clarity, readability, and presentation. We did not use generative AI tools to develop the methodology, design experiments, process or analyze experimental data, or interpret results. All AI-assisted text was reviewed and revised by the authors. We take responsibility for the final content of this work, including text produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide detailed experimental settings, implementation details, model configurations, and training procedures in the appendix. Additional results and analyses are also included to support the reproducibility of our findings. We will release the code necessary to reproduce the experiments upon publication.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The twelfth international conference on learning representations*, 2024.

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.

Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

Sahil Chaudhary. Code alpaca: An instruction-following llama model for code generation. `https://github.com/sahil280114/codealpaca`, 2023.

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.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Wojciech M Czarnecki, Razvan Pascanu, Simon Osindero, Siddhant Jayakumar, Grzegorz Swirszcz, and Max Jaderberg. Distilling policy distillation. In *The 22nd international conference on artificial intelligence and statistics*, pp. 1331–1340. PMLR, 2019.

Shiping Gao, Fanqi Wan, Jiajian Guo, Xiaojun Quan, and Qifan Wang. Advantage-guided distillation for preference alignment in small language models. *arXiv preprint arXiv:2502.17927*, 2025.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. In *The twelfth international conference on learning representations*, 2024.

Changyi He, Yifu Ding, Jinyang Guo, Ruihao Gong, Haotong Qin, and Xianglong Liu. Da-kd: difficulty-aware knowledge distillation for efficient large language models. In *Forty-second International Conference on Machine Learning*, 2025.

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.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

Michael Igorevich Ivanitskiy, Rusheb Shah, Alex F. Spies, Tilman Räuker, Dan Valentine, Can Rager, Lucia Quirke, Chris Mathwin, Guillaume Corlouer, Cecilia Diniz Behn, and Samy Wu Fung. A configurable library for generating and manipulating maze datasets, 2023. URL http://arxiv.org/abs/2309.10498.

Minsang Kim and Seung Baek. Explain in your own words: Improving reasoning via token-selective dual knowledge distillation. In *International Conference on Learning Representations*, volume 2026, pp. 103174–103196, 2026.

Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 1317–1327, 2016.

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

Vijay R. Konda and John N. Tsitsiklis. Actor-critic algorithms. In *Advances in Neural Information Processing Systems*, volume 12, pp. 1008–1014, 2000.

Zheng Li, Xiang Li, Lingfeng Yang, Borui Zhao, Renjie Song, Lei Luo, Jun Li, and Jian Yang. Curriculum temperature for knowledge distillation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 1504–1512, 2023.

Alexander Lin, Jeremy Wohlwend, Howard Chen, and Tao Lei. Autoregressive knowledge distillation through imitation learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6121–6133, 2020.

Chris Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiacai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, et al. Skywork-reward-v2: Scaling preference data curation via human-ai synergy. *arXiv preprint arXiv:2507.01352*, 2025.

Shreyansh Padarha. Enhancing reasoning capabilities in slms with reward guided dataset distillation, 2025. URL https://arxiv.org/abs/2507.00054.

Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3967–3976, 2019.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets, 2015. URL https://arxiv.org/abs/1412.6550.

Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015.

Simon Schmitt, Jonathan J Hudson, Augustin Zidek, Simon Osindero, Carl Doersch, Wojciech M Czarnecki, Joel Z Leibo, Heinrich Kuttler, Andrew Zisserman, Karen Simonyan, et al. Kickstarting deep reinforcement learning. *arXiv preprint arXiv:1803.03835*, 2018.

Max Schwarzer, Johan Samir Obando Ceron, Aaron Courville, Marc G. Bellemare, Rishabh Agarwal, and Pablo Samuel Castro. Bigger, better, faster: Human-level atari with human-level efficiency. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 30365–30380, 2023.

Chi-Ping Su, Ching-Hsun Tseng, Bin Pu, Lei Zhao, Jiewen Yang, Zhuangzhuang Chen, and Shin-Jye Lee. Ea-kd: Entropy-based adaptive knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 731–740, 2025.

Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018.

Fusheng Wang, Jianhao Yan, Fandong Meng, and Jie Zhou. Selective knowledge distillation for neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 6456–6466, 2021.

Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *CoRR*, abs/2105.15203, 2021. URL https://arxiv.org/abs/2105.15203.

Shicheng Xu, Liang Pang, Yunchang Zhu, Jia Gu, Zihao Wei, Jingcheng Deng, Feiyang Pan, Huawei Shen, and Xueqi Cheng. Rlkd: Distilling llms' reasoning via reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 34151–34159, 2026.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

Hailin Zhang, Defang Chen, and Can Wang. Confidence-aware multi-teacher knowledge distillation. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4498–4502. IEEE, 2022.

Mingjie Zhang, Xiaoling Zhou, Yuxiao Luo, Yiyu Liu, Shikun Zhang, and Wei Ye. Askd: Reinforcement learning-style knowledge distillation with quality-adaptive skewness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 34781–34789, 2026.

Borui Zhao, Quan Cui, Renjie Song, Yiyu Qiu, and Jiajun Liang. Decoupled knowledge distillation. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pp. 11953–11962, 2022.

Qihuang Zhong, Liang Ding, Li Shen, Juhua Liu, Bo Du, and Dacheng Tao. Revisiting knowledge distillation for autoregressive language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10900–10913, 2024.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 633–641, 2017.

A Derivation of the Trajectory-Level RG-KD Objective

Below is the derivation of the trajectory-level RG-KD Objective:

$$\mathcal{L}_{\text{RG-KD}}(\theta) = \mathbb{E}_x \left[(1 + R(x, p_T))\, D\left(p_T(\cdot \mid x) \,\|\, p_S^\theta(\cdot \mid x) \right) \right] \tag{10}$$

$$= \mathbb{E}_x \left[\left(1 + \mathbb{E}_{z_{1:T} \sim p_T(\cdot|x)} \left[r(x, z_{1:T}) \right] \right) D\left(p_T(z_{1:T} \mid x) \,\|\, p_S^\theta(z_{1:T} \mid x) \right) \right] \tag{11}$$

$$= \mathbb{E}_x \left[\left(1 + \mathbb{E}_{z_{1:T} \sim p_T(\cdot|x)} \left[r(x, z_{1:T}) \right] \right) \mathbb{E}_{z_{1:T} \sim p_T(\cdot|x)} \left[\log \frac{p_T(z_{1:T} \mid x)}{p_S^\theta(z_{1:T} \mid x)} \right] \right] \tag{12}$$

$$= \mathbb{E}_x \left[\left(1 + \mathbb{E}_{z_{1:T} \sim p_T(\cdot|x)} \left[r(x, z_{1:T}) \right] \right) \mathbb{E}_{z_{1:T} \sim p_T(\cdot|x)} \left[\log \frac{\prod_{t=1}^T p_T(z_t \mid h_t)}{\prod_{t=1}^T p_S^\theta(z_t \mid h_t)} \right] \right] \tag{13}$$

$$= \mathbb{E}_x \left[\left(1 + \mathbb{E}_{z_{1:T} \sim p_T(\cdot|x)} \left[r(x, z_{1:T}) \right] \right) \mathbb{E}_{z_{1:T} \sim p_T(\cdot|x)} \left[\sum_{t=1}^T \log \frac{p_T(z_t \mid h_t)}{p_S^\theta(z_t \mid h_t)} \right] \right] \tag{14}$$

$$= \mathbb{E}_x \left[\left(1 + \mathbb{E}_{z_{1:T} \sim p_T(\cdot|x)} \left[r(x, z_{1:T}) \right] \right) \sum_{t=1}^T \mathbb{E}_{z_{<t} \sim p_T(\cdot|x)} \left[D\left(p_T(\cdot \mid h_t) \,\|\, p_S^\theta(\cdot \mid h_t) \right) \right] \right] \tag{15}$$

$$= \mathbb{E}_x \left[\left(1 + \mathbb{E}_{z_{1:T} \sim p_T(\cdot|x)} \left[r(x, z_{1:T}) \right] \right) \mathbb{E}_{z_{1:T} \sim p_T(\cdot|x)} \left[\sum_{t=1}^T D\left(p_T(\cdot \mid h_t) \,\|\, p_S^\theta(\cdot \mid h_t) \right) \right] \right]. \tag{16}$$

The above derivation is mathematically valid for the forward KL divergence because the KL divergence between autoregressive trajectory distributions decomposes into the expected sum of conditional KL divergences. However, the same exact decomposition does not generally hold for Jensen–Shannon divergence or Skew KL divergence. Jensen–Shannon divergence depends on a mixture distribution over full trajectories, which does not decompose into a simple sum of per-step conditional divergences between the teacher and student distributions. Similarly, Skew KL divergence evaluates the KL divergence against a skewed mixture of the teacher and student trajectory distributions. Since this skewed mixture is defined at the trajectory level, it does not generally factor into a simple per-step skewed mixture of conditional distributions. Consequently, applying the same step-wise trajectory formulation to JSD or Skew KL should be understood as a heuristic approximation rather than an exact mathematical reduction.

B Experimental Settings

Across all experiments, RG-KD uses a single-sample estimate of teacher utility. For trajectory-level RG-KD, we generate one complete rollout for each input and use the reward of that rollout to weight the distillation loss computed along the same trajectory. For step-level RG-KD, at each prediction step, we sample one next output from the teacher distribution, append it to the current state, and evaluate the resulting state with the critic to obtain the distillation weight for that step.

B.1 Semantic Segmentation

Setup. We evaluate RG-KD on semantic segmentation as a single-step prediction task using ADE20K (Zhou et al., 2017). A SegFormer-B5 (Xie et al., 2021) fine-tuned on ADE20K at 640×640 resolution serves as the teacher, while the student uses a pretrained MiT-B0 encoder (Xie et al., 2021) with the remaining components randomly initialized. We compute the mean Intersection over Union (mIoU) between a sampled teacher prediction and the ground-truth mask as the reward signal, and use this reward to determine the distillation weight for the corresponding input.

B.2 Maze Reasoning

Setup. We evaluate sequential distillation on a Maze reasoning task (Ivanitskiy et al., 2023), where the model autoregressively generates spatial coordinates connecting the start and end points of a 5×5 maze. Performance is measured by success rate (SR) using a binary reward for successful navigation. We distill a 6.6M-parameter teacher Transformer (76.04% SR) into a 0.4M-parameter

student (60.64% SR), and evaluate EA-KD and both trajectory- and step-level RG-KD using teacher-generated trajectories. For trajectory-level RG-KD, we generate one complete teacher trajectory for each maze and use its terminal navigation reward to weight the distillation loss over that trajectory. For step-level RG-KD, at each prediction step, we sample one next spatial action from the teacher distribution and evaluate the resulting state with the critic to obtain the step-specific distillation weight. The critic is initialized from the student backbone and implemented as a CNN-conditioned Transformer with two action-value heads trained using an Expected-SARSA temporal-difference objective. It is pretrained on 1M transitions sampled from the initial student policy and continually updated during distillation.

B.3 Atari

Setup. We evaluate RG-KD on Atari control (Bellemare et al., 2013) using *Qbert*, *Amidar*, and *Phoenix*. We distill a 37M-parameter teacher into a 0.1M-parameter student, both initialized from checkpoints pretrained following (Schwarzer et al., 2023), and use the true episodic return from the simulator as the reward signal. We evaluate trajectory- and step-level RG-KD on teacher-generated trajectories and additionally evaluate step-level RG-KD on student-generated trajectories (Lin et al., 2020; Agarwal et al., 2024). For trajectory-level RG-KD, one complete teacher rollout is generated and its episodic return is used to weight the distillation loss over the same rollout. For step-level RG-KD, at each visited state, we sample one teacher action and evaluate the resulting next state with the critic to determine the distillation weight for that step. We use a CNN-based action-value critic initialized from the student backbone and trained with a distributional temporal-difference objective under discrete-action SAC. The critic is pretrained on 500K transitions from the initial student policy and continually updated during distillation. Final performance is reported as the mean episodic return over 50 test seeds, with checkpoints selected using 50 disjoint validation seeds.

B.4 LLM Tasks: Code Generation and Mathematical Reasoning

Setup. For code generation, we train on CodeAlpaca-20k (Chaudhary, 2023) and evaluate on HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021), distilling Qwen2.5-Coder-7B-Instruct into Qwen2.5-Coder-1.5B-Instruct (Hui et al., 2024). For mathematical reasoning, we train and evaluate on GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021), distilling Qwen2.5-Math-7B-Instruct into Qwen2.5-Math-1.5B-Instruct (Yang et al., 2024). In both domains, Skywork-Reward-V2-Llama-3.1-8B-40M (Liu et al., 2025) scores generated responses and provides the reward signal. We train the student for two epochs using BF16 LoRA fine-tuning with rank 8, $\alpha = 16$, and dropout 0.05 on the query, key, value, and output projection modules, with Adam, a learning rate of 5×10^{-6}, and a batch size of 4. Responses are generated with temperature 0.8, top-p 0.95, and a maximum of 1,024 new tokens. For trajectory-level RG-KD, we generate one complete response per prompt and use its reward-model score to weight the distillation loss over that response. For step-level RG-KD, at each token position, we sample one candidate next token from the teacher distribution, append it to the current history, and evaluate the resulting history with the critic to obtain the token-level distillation weight. The critic is initialized from the corresponding student backbone with a scalar value head that estimates the expected terminal reward-model score, and is trained using GAE returns and PPO's clipped value-function objective. It is pretrained for 30 epochs using rollouts from the initial student policy and continually updated during distillation.

B.5 Comparison with Adaptive and Selective Distillation Methods

Setup. We compare step-level RG-KD with TSD-KD (Kim & Baek, 2026) and ASKD (Zhang et al., 2026) on GSM8K and MATH using Qwen2.5-Math-7B-Instruct as the teacher and Qwen2.5-Math-1.5B-Instruct as the student. All methods are trained for one epoch with a batch size of 4 using BF16 LoRA fine-tuning with rank 8, $\alpha = 16$, and dropout 0.05 on the query, key, value, and output projection modules, with Adam at a learning rate of 5×10^{-6}. TSD-KD and RG-KD use one student-generated response per prompt, whereas ASKD uses one teacher-generated and one student-generated response per prompt, following their respective training procedures. Responses are generated with temperature 0.8, top-p 0.95, and a maximum of 1,024 new tokens. Step-level RG-KD follows the critic setup in Appendix B.4: at each visited token position, we sample one candidate next token from the teacher distribution and use the critic value of the resulting history as the distillation weight for that step.

B.6 Plug-and-Play Integration with Existing Distillation Methods

Setup. We integrate step-level RG-KD with GKD (Agarwal et al., 2024) and DistiLLM (Ko et al., 2024) on GSM8K and MATH using the same Qwen2.5-Math-7B-Instruct teacher and Qwen2.5-Math-1.5B-Instruct student as in Appendix B.4. For GKD, we use generalized JSD with $\beta = 0.9$ and set the student data fraction to $\lambda = 1$, resulting in fully on-policy training with student-generated sequences. For DistiLLM, we use skew KLD with $\alpha = 0.1$ and its adaptive SGO scheduler initialized at $\phi = 0.4$, which dynamically balances student-generated outputs and fixed-dataset samples based on validation loss. All configurations are trained for one epoch with a batch size of 4 using the same BF16 LoRA configuration and optimizer settings as Appendix B.4. Student-generated responses use temperature 0.8, top-p 0.95, and a maximum of 1,024 new tokens. RG-KD uses the same critic architecture, pretraining, and online updating procedure as Appendix B.4. For each RG-KD-weighted update, at every visited token position we sample one next token from the teacher distribution and use the critic value of the resulting history as the step-level distillation weight. For DistiLLM, this weighting is applied only to SGO-based updates, while fixed-dataset updates retain the original DistiLLM objective because fixed responses do not provide sample-based estimates of the teacher distribution's utility.

C Compute Resources

We conducted several experiments using different GPU resources depending on the task scale. For semantic segmentation, training was performed on an NVIDIA Pro6000 GPU and took approximately 3 hours. Maze reasoning experiments were run on an NVIDIA RTX 3090 GPU and required about 30 minutes. Atari experiments were also conducted on an NVIDIA RTX 3090 GPU, with training taking approximately 24 hours. For the LLM-based tasks, including code generation and mathematical reasoning, experiments were performed on a single NVIDIA H100 GPU and required approximately 48 hours of training.

D Limitations.

RG-KD relies on the availability and quality of reward or value signals. If the reward model is noisy, biased, sparse, or misaligned with the true task objective, the resulting weights may emphasize unhelpful or even harmful teacher distributions. This limitation is especially important in open-ended language generation, where reward models may fail to capture factuality, safety, or long-term utility. Step-level RG-KD further depends on a learned critic, whose approximation error can affect the quality of the distillation weights and introduce additional training instability. Our method can also incur extra computational cost due to reward estimation, Monte Carlo sampling, critic training, or additional rollouts, particularly in sequential prediction and reinforcement learning settings. Finally, although we evaluate RG-KD across diverse domains, our experiments cover a limited set of model scales, datasets, and reward formulations. Further work is needed to understand its robustness under different teachers, larger-scale models, alternative reward models, and more complex deployment scenarios.

E Ablation on Reward Sources for Mathematical Reasoning

We conducted an ablation study on the reward source for mathematical reasoning tasks. Specifically, we evaluated trajectory-level RG-KD using two types of reward signals: reward-model scores and verifiable rewards. As shown in Table 6, the results show mixed trends. While RG-KD with verifiable rewards outperformed the one with reward-model scores using KLD and JSD on GSM8K, it performed worse with SKL on GSM8K and across all divergence losses on MATH. One possible explanation is that binary verifiable rewards provide a more reliable correctness signal but cannot capture fine-grained, sample-specific differences in quality. By contrast, a reward model assigns continuous rewards to individual samples, which may allow RG-KD to weight them more effectively.

Table 6: **Ablation on reward sources for trajectory-level RG-KD on mathematical reasoning.** We compare reward-model scores and binary verifiable rewards on GSM8K and MATH using teacher-generated trajectories. Results are reported as accuracy across KLD, JSD, and SKL objectives, with Standard KD included as a reference.

		GSM8K			MATH		
Method	**Reward Source**	**KLD**	**JSD**	**SKL**	**KLD**	**JSD**	**SKL**
Teacher Generated Trajectories							
Standard KD	–	77.41	77.56	77.86	66.08	67.00	66.42
RG-KD	Reward Model	77.64	78.09	**78.17**	**66.56**	**67.24**	**66.98**
RG-KD	Verifier	**77.71**	**78.24**	77.79	66.14	66.70	66.66

Table 7: **Comparison with reward-based trajectory selection and weighting methods.** We compare AdvDistill, reward-based resampling, and trajectory-level RG-KD on GSM8K and MATH using four rollouts per prompt. Results are reported as accuracy, and the best result on each benchmark is highlighted in bold.

Method	**GSM8K**	**MATH**
AdvDistill	76.27	59.42
Reward-based Resampling	77.33	65.88
RG-KD (Ours)	**80.06**	**66.86**

F Comparison with Reward-Based Trajectory Selection and Weighting Methods

Existing reward-based distillation approaches can use reward signals to filter, resample, or reweight completed responses. For example, response filtering and weighting methods such as AdvDistill (Padarha, 2025) use response-level rewards to weight the SFT objective, while reward-based resampling preferentially selects high-reward responses for distillation. In contrast, RG-KD defines distillation strength from the expected utility of the teacher's predictive distribution, with trajectory rewards serving as sample-based estimates of this utility. From the unified view in Sec. 3.4, these methods can be viewed as restricted, offline trajectory-level instantiations of the RG-KD framework.

To empirically compare these approaches, we evaluate AdvDistill, reward-based resampling, and trajectory-level RG-KD on GSM8K and MATH using four rollouts per prompt. All methods are trained for two epochs. For AdvDistill, we compute group-relative advantages, filter the resulting training samples, and weight the SFT objective using these advantages. For reward-based resampling, we sample one response from the four rollouts according to their reward-model scores and distill the selected response using KLD. For trajectory-level RG-KD, the reward of each sampled trajectory directly modulates its distillation loss according to the RG-KD objective. As shown in Table 7, trajectory-level RG-KD achieves the best performance on both GSM8K and MATH, outperforming both AdvDistill and reward-based resampling. These results provide complementary empirical evidence that directly incorporating trajectory utility into the distillation objective can transfer teacher knowledge more effectively than the compared response-filtering and resampling strategies.

G Experiments on Image Classification

To further evaluate RG-KD on a standard image classification benchmark, we conduct experiments on CIFAR-100 using a pretrained ResNet32×4 as the teacher and a randomly initialized ResNet8×4 as the student. The student is distilled for 240 epochs with a batch size of 64. For reward estimation, we sample a class from the teacher distribution and assign a binary reward according to whether the sampled class matches the ground-truth label. As shown in Table 8, RG-KD consistently outperforms the corresponding baselines across all evaluated divergence objectives, further demonstrating its applicability to conventional image classification.

Table 8: **Results on CIFAR-100 image classification.** We report accuracy for distillation from a pretrained ResNet32×4 teacher to a randomly initialized ResNet8×4 student. The best result for each divergence objective is highlighted in bold.

Method	KLD	JSD	SKL
Standard KD	73.34	72.59	73.55
EA-KD	74.72	72.59	74.41
RG-KD (Ours)	**75.12**	**72.66**	**74.74**

H CODE GENERATION.

Table 9: **Results on code generation.**

Method	TL/SL	HumanEval	HumanEval+	MBPP	MBPP+	Avg.
Teacher	-	91.46	84.15	82.28	71.43	82.33
Student	-	70.73	65.24	68.78	59.26	66.00
Teacher Generated Trajectories						
KLD	-	74.39	69.51	71.96	61.90	69.44
JSD	-	75.61	69.51	72.75	62.70	70.14
SKL	-	75.61	70.12	71.69	61.90	69.83
EA-KLD	TL	75.61	70.12	71.69	61.64	69.77
EA-JSD	TL	75.61	69.51	72.49	62.17	69.94
EA-SKL	TL	74.39	68.90	71.69	62.17	69.29
RG-KLD (Ours)	TL	**76.83**	**71.95**	72.75	62.70	**71.06**
RG-JSD (Ours)	TL	76.22	70.73	73.28	63.76	71.00
RG-SKL (Ours)	TL	76.22	70.73	72.75	62.70	70.60
RG-KLD (Ours)	SL	73.17	68.29	73.28	62.96	69.43
RG-JSD (Ours)	SL	75.61	70.12	**74.87**	**64.29**	**71.22**
RG-SKL (Ours)	SL	75.61	69.51	72.75	62.17	70.01
Student Generated Trajectories						
KLD	-	74.39	70.12	71.96	63.23	69.92
JSD	-	**75.61**	**70.73**	72.22	61.11	69.92
SKL	-	**75.61**	69.51	72.49	62.17	69.94
RG-KLD (Ours)	SL	74.39	70.12	73.28	**63.49**	70.32
RG-JSD (Ours)	SL	**75.61**	70.12	73.02	62.70	70.36
RG-SKL (Ours)	SL	**75.61**	70.12	**74.07**	**63.49**	**70.82**

I MATHEMATICAL REASONING.

Table 10: **Results on mathematical reasoning.**

Method	TL/SL	GSM8K	MATH	Avg.
Teacher	-	89.39	73.96	81.68
Student	-	76.73	60.62	68.67
Teacher Generated Trajectories				
KLD	-	77.41	66.08	71.74
JSD	-	77.56	67.00	72.28
SKL	-	77.86	66.42	72.14
EA-KLD	TL	77.64	66.54	72.09
EA-JSD	TL	77.71	66.98	72.35
EA-SKL	TL	77.48	66.60	72.04
RG-KLD (Ours)	TL	77.64	66.56	72.10
RG-JSD (Ours)	TL	78.09	67.24	72.66
RG-SKL (Ours)	TL	78.17	66.98	72.57
RG-KLD (Ours)	SL	80.52	67.22	73.87
RG-JSD (Ours)	SL	79.83	67.20	73.52
RG-SKL (Ours)	SL	**80.67**	**67.38**	**74.02**
Student Generated Trajectories				
KLD	-	77.94	67.34	72.64
JSD	-	79.00	67.26	73.13
SKL	-	79.23	67.36	73.30
RG-KLD (Ours)	SL	78.62	67.46	73.04
RG-JSD (Ours)	SL	**80.82**	67.38	**74.10**
RG-SKL (Ours)	SL	80.59	**67.50**	74.05

J VISUALIZATION OF WEIGHT DYNAMICS



Q_T	1.166	1.244	1.270	1.247
Q_S	0.006	0.498	0.187	0.432
ΔQ	+1.160	+0.746	+1.083	+0.815
w	**47.40**	**30.83**	**44.33**	**33.60**

Figure 2: Visualization of weight dynamics in the maze reasoning task.

To provide a qualitative view of how step-level RG-KD modulates the distillation signal, we visualize the weights actually applied during maze reasoning. At each step, we sample an action from the teacher policy and evaluate its value using the critic. We then use the value of an action sampled from the student policy as a baseline and compute the teacher action's relative advantage. Specifically, given the critic values of the teacher- and student-sampled actions, Q_T and Q_S, respectively, we compute the relative advantage as $\Delta Q = Q_T - Q_S$. The resulting advantage is then scaled by a constant factor to obtain the step-level distillation weight w. Figure 2 illustrates this process, showing how the assigned weights vary across states depending on the relative value of the teacher and student actions.