KETCHUP: K-Step Return Estimation for Sequential Knowledge Distillation

Anonymous ACL submission

Abstract

We propose a novel K-step return estimation method (called KETCHUP) for Reinforcement Learning(RL)-based knowledge distillation (KD) in text generation tasks. Our idea is to induce a K-step return by using the Bellman Optimality Equation for multiple steps. Theoretical analysis shows that this K-step formulation reduces the variance of the gradient estimates, thus leading to improved RL optimization especially when the student model size is large. Empirical evaluation on three text generation tasks demonstrates that our approach yields superior performance in both standard task metrics and large language model (LLM)-based evaluation. These results suggest that our K-step return induction offers a promising direction for enhancing RL-based KD in LLM research. [1]

1 Introduction

Knowledge distillation (KD; Hinton et al., 2015) refers to training a (typically) small student model from a teacher's output. KD has been increasingly important in the LLM era, as larger models achieve higher performance (Kaplan et al., 2020) but are more difficult to deploy in low-resource scenarios.

KD approaches can be generally categorized into two types: intermediate-layer matching and prediction matching. Intermediate-layer matching aims to match the student's and teacher's hidden states, encouraging the student to mimic the teacher's behavior layer by layer (Sun et al., 2019; Jiao et al., 2020; Wang et al., 2021). Prediction matching informs the student of the task to solve, typically by minimizing the divergence of output distributions (Kim and Rush, 2016; Wen et al., 2023).

Classic KD for text generation suffers from the exposure bias problem (Bengio et al., 2015), as the student learns word by word following the teacher's or ground truth's prefix, without accounting for its own previous predictions. RL alleviates this issue by enabling the student to learn through exploration. Hao et al. (2022) induce a step-wise reward function from a language model trained in a supervised way. Building on this, Li et al. (2024) apply RL to text generation KD, where a student model is trained by the REINFORCE algorithm (Williams, 1992) maximizing the cumulative reward suggested by the teacher.

However, REINFORCE is known to suffer from high variance because it estimates gradient by sampled trajectories (i.e., sequences), which can vary significantly (Sutton and Barto, 2018). This issue is further exacerbated in text generation scenarios due to the large action space (i.e., vocabulary size), resulting in unstable learning.

In this paper, we propose KETCHUP, a novel ***K***-step return **E**stimation **TeCH**nique to **U**pdate **P**olicy for RL-based knowledge distillation. Our work is inspired by Li et al. (2024), who derive a Q-value function from the teacher's policy (next-token probabilities) and induce a reward function based on the Bellman-Optimality Equation (Bellman, 1952). We propose to estimate the total reward based on K-step Bellman optimality. Theoretical analysis shows that our KETCHUP reduces the variance of the total reward, thus effectively mitigating the high variance issue of RL-based text generation KD.

We evaluated our approach on three text generation datasets categorized into different domains: XSum (Narayan et al., 2018) for summarization, the Europarl corpora (Koehn, 2005) for machine translation, and GSM8K (Cobbe et al., 2021) for arithmetic reasoning. Experiments show that our proposed KETCHUP consistently achieves an add-on performance improvement when combined with the recent KD through the RL method (Li et al., 2024). We also conduct an empirical analysis to show that the KETCHUP demonstrates lower vari-

[1] Our code is released at https://anonymous.4open.science/r/KETCHUP-4956

ance and converges better than Li et al. (2024), i.e., achieving a higher return and being more stable.

2 Methodology

2.1 RL Formulation of Text Generation

Text generation can be formulated as an undiscounted Markov Decision Process (MDP) with tuple $(\mathcal{S}, \mathcal{A}, T, r)$. The *state* space $\mathcal{S}$ includes all possible (sub)sequences and each of them is represented by $\mathbf{y}_{<t}$ for some time step t; notice that text generation may also depend on an input sequence, which is omitted here. The *action* $a_t \in \mathcal{A}$ at step t corresponds to the next token y_t from the vocabulary $\mathcal{V}$. The *state transition* T is a deterministic process in text generation, as s_{t+1} is essentially the concatenation of s_t and the newly generated word a_t. The *reward* function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ provides feedback based on (s_t, a_t). The goal of RL is to find a *policy* (distribution over actions) to maximize the expected *return* (cumulative rewards).

A key challenge in applying RL to text generation is the lack of well-defined step-wise reward functions. To address this, Hao et al. (2022) and Li et al. (2024) assume that a language model generates the next word from a Boltzmann distribution based on the *Q-value function*,[2] given by

$$\pi_{\mathrm{LM}}(a \mid s) = \frac{\exp\big(q(s,a)\big)}{\sum_{a'} \exp\big(q(s,a')\big)}, \qquad (1)$$

Due to the shared formula, a language model's pre-softmax logit can be viewed as the Q-value function, and with the Bellman optimality equation (Bellman, 1952), a step-wise reward function can be induced by

$$r(s_t, a_t) = q(s_t, a_t) - \max_{a' \in \mathcal{A}} q(s_{t+1}, a'). \qquad (2)$$

Then, the goal of RL for text generation KD is to optimize the student's policy, denoted by π_θ, to maximize the expected cumulative reward:

$$J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=1}^{T} r(s_t, a_t)\right], \qquad (3)$$

The REINFORCE algorithm (Williams, 1992) is a policy gradient method, which is widely used for RL in NLP (Hao et al., 2022; Li et al., 2024).

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=1}^{T} G_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\right] \qquad (4)$$

where $G_t = \sum_{i=t}^{T} r(s_i, a_i)$ is a cumulative reward (i.e., return) from step t, and the expectation is approximated by Monte Carlo samples from the distribution π_θ.

2.2 Our KETCHUP Method

In this work, we address RL-based KD and propose to refine the learning signal G_t in Eqn. (4) by extending the one-step reward induction to K steps, which alleviates the high variance issue of RL. The key idea is to apply the Bellman optimality equation for multiple steps, therefore directly connecting the Q-values at the current state with those of a future state.

We begin by considering the sum of rewards in Eqn. (2) over K consecutive steps starting from step t, denoted by $G_{t:t+K}$:

$$\begin{aligned} G_{t:t+K} &:= \sum_{i=0}^{K-1} r(s_{t+i}, a_{t+i}) \\ &= \sum_{i=0}^{K-1} \Big[q(s_{t+i}, a_{t+i}) - \max_{a' \in \mathcal{A}} q(s_{t+i+1}, a')\Big] \\ &= q(s_t, a_t) - \max_{a' \in \mathcal{A}} q(s_{t+K}, a') \qquad (5) \end{aligned}$$

where Eqn. (5) assumes that an optimal action $a_{t+i+1} = \arg\max_{a' \in \mathcal{A}} q\big(s_{t+i+1}, a'\big)$ is taken. However, a student's policy may not be optimal; therefore, Eqn. (5) becomes an approximation, denoted by $\hat{G}_{t:t+K}$,:

$$\hat{G}_{t:t+K} = q(s_t, a_t) - \max_{a' \in \mathcal{A}} q(\hat{s}_{t+K}, a') \qquad (6)$$

where $\hat{s}_{t+K}$ is the state at the $(t+K)$th step after following the student's policy. This is a reasonable approximation because, in KD, a student is usually pretrained in a meaningful way (Turc et al., 2019; Lee et al., 2023; Kim et al., 2024) and the approximation will be more accurate as the optimization proceeds.

Building upon the K-step reward formulation, we can obtain an approximate return $\hat{G}_t$ by considering intervals of K steps, i.e.,

[2] The Q-value function estimates the expected return (cumulative reward) of taking action a in state s and then following a given policy thereafter, defined by $q_\pi(s,a) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \;\middle|\; s_0 = s, a_0 = a\right]$.

Algorithm 1 KETCHUP

Input: Non-parallel dataset D; teacher Q-value function $q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$; student policy π_θ with initial parameters θ; segment length K; learning rate η; maximum rollout length T; number of iterations U

Output: Trained student policy π_θ

for $j \leftarrow 1$ **to** U **do**
  Sample a source sentence $\mathbf{x} \in D$
  Set the initial state $s_0 \leftarrow \mathbf{x}$
  Generate a trajectory $\tau = \{(s_0, a_0), (s_1, a_1), \ldots, (s_T, a_T)\}$ by sampling from π_θ
  Initialize gradient accumulator: $g \leftarrow 0$
  for $t \leftarrow T$ **to** 0 **do**
    if $t = T$ **then**
      $\hat{G}_T \leftarrow q(s_T, a_T)$
    else if $T - t < k$ **then**
      $\hat{G}_t \leftarrow \Big[q(s_t, a_t) - \max_{a' \in \mathcal{A}} q(s_{t+1}, a')\Big] + \hat{G}_{t+1}$
    else
      $\hat{G}_t \leftarrow \Big[q(s_t, a_t) - \max_{a' \in \mathcal{A}} q(s_{t+K}, a')\Big] + \hat{G}_{t+K}$
    end
    $g \leftarrow g + \hat{G}_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$
  end
  $\theta \leftarrow \theta + \eta g$
end
return π_θ

$\hat{G}_{t:t+K}, \hat{G}_{t+K:t+2K}, \cdots$. Formally, we have

$$\begin{aligned}\hat{G}_t &= \sum_{i=0}^{\lfloor \frac{T-t+1}{K} \rfloor} \hat{G}_{t+iK:t+(i+1)K} \\ &= \sum_{i=0}^{\lfloor \frac{T-t+1}{K} \rfloor} \Big[q(s_{t+iK}, a_{t+iK}) - \max_{a' \in \mathcal{A}} q(\hat{s}_{t+(i+1)K}, a')\Big].\end{aligned} \tag{7}$$

which will be used in our RL-based generation KD.

In particular, the student's policy is used to sample a sequence of actions (i.e., output words). Then, the sequence is fed to the teacher model, which evaluates the sequence by Eqn. (7). Finally, we follow the policy gradient formula, but use the approximate return for the update:

$$\nabla_\theta J(\theta) \approx \mathbb{E}_{\pi_\theta}\left[\sum_{t=1}^{T} \hat{G}_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\right] \tag{8}$$

where $\hat{G}_t$ is our approximate return defined in Eqn. (7). The process is shown in Algorithm 1.

2.3 Bias and Variance Analysis

Although the REINFORCE algorithm (Williams, 1992) estimates gradients in an unbiased way, it is known to be noisy and prone to high variance in the gradient estimation, which may lead to instability in learning (Greensmith et al., 2004; Mnih et al., 2016; Bjorck et al., 2022).

A standard method to mitigate this issue is to subtract a *baseline* term b_t from the actual return:

$$\hat{G}_t = G_t - b_t. \tag{9}$$

For example, the average return over a batch (Rosenberg, 2021) is commonly used as the baseline term to stabilize the REINFORCE algorithm.

Our KETCHUP approach is a variant of REINFORCE with baseline. This can be seen by examining the difference between the actual return G_t and our approximate return $\hat{G}_t$. In our KD application, the actual return G_t is given by accumulating the reward defined in Eqn. (2). In other words, we have

$$G_t = \sum_{i=0}^{T}\Big(q(s_{t+i}, a_{t+i}) - \max_{a' \in \mathcal{A}} q(s_{t+i+1}, a')\Big). \tag{10}$$

Combining Eqns. (7), (9), and (10), we can interpret our approximate return $\hat{G}_t$ as introducing a baseline term with the following form

$$b_t = \sum_{\substack{i=0 \\ i \not\equiv 0 \pmod k}}^{T-1} \Big[q(s_{t+Ki+1}, a_{t+Ki+1}) - \max_{a' \in \mathcal{A}} q(s_{t+Ki+1}, a')\Big]. \tag{11}$$

Unlike conventional, policy-independent baselines (Sutton and Barto, 2018; Rosenberg, 2021), our baseline depends on the selected actions and thus introduces bias into the expected return estimation. However, our approach can alleviate the high variance issue of REINFORCE with mild assumptions, as shown by the following theorem.

Theorem 1 (Variance Reduction via K-Step Return)**.** *Let G_t be the actual return and $\hat{G}_t$ be the K-step approximate return for some sequences sampled from the student policy π. Assuming that the state–action–reward tuples (s_t, a_t, r_t) are iid drawn at different steps, we have:*

$$\mathrm{Var}[\hat{G}_t] \leq \mathrm{Var}[G_t]. \tag{12}$$

Proof. See Appendix A. □

The iid assumption is reasonable and widely adopted in theoretical RL research (Kearns and Singh, 2000; Bhandari et al., 2018; Xu et al., 2020), because in many environments the dependencies decay rapidly and correlation is further weakened when a large batch of samples are considered.

Overall, Theorem 1, along with the derivations in Appendix A, indicates that our KETCHUP alleviates variance at a power rate as K increases. Although this method introduces a bias term in the gradient estimation, the bias is effectively mitigated: it diminishes for smaller values of K and converges to zero as the student policy becomes more optimal. Detailed bias analysis is given in Appendix B. Such a trade-off is wildly applied in existing RL literature, as seen in Temporal Difference (TD) learning (Sutton, 1988), Actor–Critic algorithms (Konda and Tsitsiklis, 1999; Mnih et al., 2016), and Deep Q-Network (DQN; Mnih et al., 2015).

3 Experiments

In this section, we present the empirical evaluation and analysis of our proposed KETCHUP. We begin by describing the datasets, baseline methods, and implementation details, followed by the main results and detailed analyses.

3.1 Settings

Tasks, Datasets, and Metrics. We evaluate our approach on various text generation tasks that are frequently considered in previous literature (Maruf et al., 2018; Magister et al., 2023; Wen et al., 2023; Touvron et al., 2023; Biderman et al., 2024; Wang et al., 2024).

- **XSum Summarization.** The Extreme Summarization (XSum) is a challenging dataset for text summarization introduced by Narayan et al. (2018), where the summaries are highly abstractive as they emphasize key ideas with novel wordings. The dataset consists of approximately 226,000 BBC articles paired with single-sentence summaries. We employ ROUGE (Lin, 2004) as the primary metric, which is common practice in summarization (Ravaut et al., 2024; Van Veen et al., 2024; Agarwal et al., 2025).
- **Europarl EN–NL Translation.** Europarl (Koehn, 2005) is a high-quality, multilingual parallel corpus extracted from European Parliament proceedings. Its texts are professionally produced and carefully aligned, ensuring reliable, well-edited data. We choose English-to-Dutch, a relatively low-resource translation direction, to facilitate our distillation experiments. We report the BLEU score (Papineni et al., 2002), character-level F score (chrF, Popović, 2015), and translation edit rate (TER, Snover et al., 2006), following the standard evaluation in machine translation (Barrault et al., 2019; Hrabal et al., 2024).
- **GSM8K Reasoning.** Grade School Math 8K (GSM8K, Cobbe et al., 2021) is a popular dataset consisting of around 8,000 grade school-level math problems with detailed step-by-step solutions. It is designed to evaluate a model's abilities in mathematical reasoning and multi-step problem-solving. The standard evaluation metric for GSM8K is solution accuracy (Wang et al., 2024; Setlur et al., 2025), which is adopted in our experiments.

We employ the standard training, validation, and test splits for XSUM (Narayan et al., 2018) and Europarl (Koehn, 2005). For GSM8K, the standard split comprises only training and test sets (Cobbe et al., 2021). We adopt the open-source split provided by Wang et al. (2024), where the validation set is constructed by randomly selecting examples from the original training data.

Implementation Details. In our KD, the teacher is the 3B-parameter FLAN-T5-XL model (Chung et al., 2024), which shares the same architecture as prior work (Li et al., 2024). For the summarization task, we directly prompt FLAN-T5-XL as it has already been instruction-finetuned for summarization. On the other tasks, FLAN-T5-XL yields subpar performance if prompted directly; we fine-tune the model as the teacher, which is commonly practiced in KD research (De Gibert et al., 2024; Setiawan, 2024; Ye et al., 2025).

The student uses the 250M-parameter T5-base model Raffel et al. (2020), which is consistent with the configuration in Wen et al. (2023) and Li et al. (2024).

Following previous KD studies (Wen et al., 2023; Li et al., 2024), we perform pre-distillation, where the student is pretrained by the cross-entropy loss based on the teacher's outputs. This ensures a meaningful initialization of the student model and enables effective exploration for reinforcement learning. Notice that text generation has a much larger state–action space than a typical RL environment such as Atari games (Mnih et al., 2015).

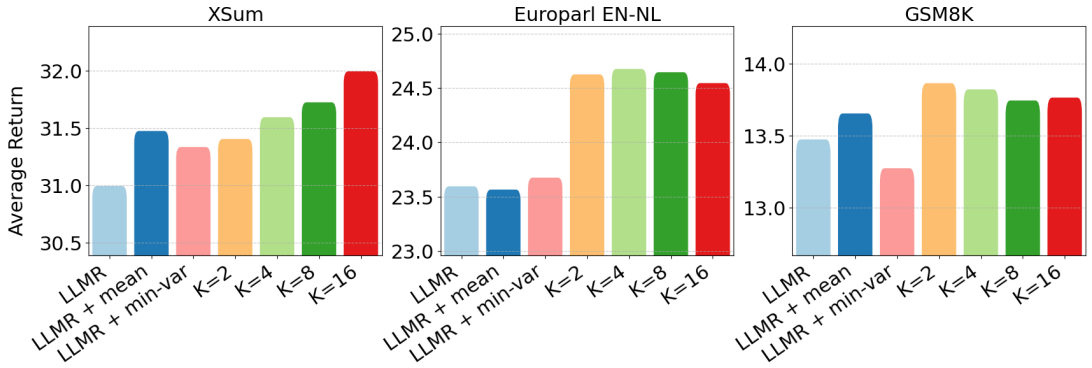


Figure 1: Average predicted return vs Approaches.

The student performs greedy action selection when generating a sequence. Our return induction builds upon K-step Bellman optimality equations, and the hyperparameter K is critical in our framework. We report performance for $K \in \{2, 4, 8, 16\}$ in our experiments.

Competing Methods. We compare our KETCHUP against both divergence-based and RL-based text generation KD:

- **SeqKD** (Kim and Rush, 2016). This is a classic method where the student maximizes likelihood of teacher-generated sequences.
- **KL Distillation** (Hinton et al., 2015). It minimizes the Kullback–Leibler (KL) divergence between student and teacher distributions. Notice that SeqKD is a hard version of KL distillation.
- **JS Distillation** (Wen et al., 2023). Jensen–Shannon (JS) divergence is a symmetric divergence that overcomes the over-smoothing problem of KL divergence (Wei et al., 2019; Wen et al., 2023).
- **TVD Distillation** (Wen et al., 2023). The Total Variation Distance (TVD) is another symmetric divergence and is shown to outperform other methods (Wen et al., 2023). Such a method is also explored in Agarwal et al. (2024) with a tunable ratio between the two terms of TVD.
- **LLMR** (Li et al., 2024). In this method, a reward function is induced from a teacher language model by one-step Bellman optimality (Hao et al., 2022). Then, the student model is trained by RL towards the induced reward.

Since our approach reduces the variance of RL, we consider alternative variance reduction techniques under the LLMR framework:

- **LLMR + Mean Baseline.** Using the average reward in a batch as a baseline is commonly used for stabilizing RL training (Sutton and Barto, 2018).
- **LLMR + Min-Variance Baseline.** This is an advanced variant that is shown to be theoretically optimal when the baseline is derived from batch data (Rosenberg, 2021).

For a fair comparison, we apply the same settings in Section 3.1 (when applicable) to the competing methods as we do to our approach. Specifically, all methods adopt pre-distillation to ensure a meaningful student initialization, and all RL methods use the same action selection procedure.

3.2 Main Results

As mentioned in Section 2.2, the primary advantage of our KETCHUP is its enhanced RL optimization compared with classic REINFORCE. In this part, we will first show that our approach indeed achieves a higher return (cumulative reward) in RL. Then, we will show that our approach leads to improved performance in NLP tasks.

Return in RL. The goal of RL is to learn a policy maximizing the cumulative reward, also known as the return. Therefore, we may use it to evaluate the outcome of RL training.

Figure 1 shows the return score that is defined in Eqn. (10), where the return is averaged over different test samples, using various RL methods in the three NLP tasks. As seen, our KETCHUP consistently achieves a higher average return than competing approaches across all the tasks. This indicates that our KETCHUP learns a superior policy in terms of the return, which is precisely the RL objective.

In addition, we observe that an increased K may not necessarily improve the return. This is because our KETCHUP introduces bias despite its reduced variance (Section 2.3). Therefore, a trade-off should be sought when choosing the K value.

NLP Task Performance. Table 1 presents the results of our approach in terms of text generation

Model		XSum			Europarl EN-NL			GSM8K
		ROUGE-1↑	ROUGE-2↑	ROUGE-L↑	BLEU4↑	chrF↑	TER↓	Accuracy(%)↑
Teacher		41.32	18.86	33.79	25.36	51.11	63.17	40.71
Student		19.60	3.19	13.72	0.95	24.80	100.21	0.00
Distilled Student	SeqKD (Kim and Rush, 2016)	33.54	11.90	26.67	22.09	48.33	66.18	20.02
	KL (Hinton et al., 2015)	34.36	12.86	27.38	22.35	48.58	65.93	23.96
	JS (Wen et al., 2023)	34.87	13.18	27.84	22.55	48.71	65.74	24.72
	TVD (Wen et al., 2023)	35.17	13.30	28.10	22.63	48.66	65.79	24.94
	LLMR (Li et al., 2024)	35.54	13.70	28.56	22.72	49.04	65.38	25.21
	LLMR + Mean baseline	35.60	13.76	28.64	22.67	49.03	65.39	25.39
	LLMR + Min-Var baseline	35.59	13.78	28.66	22.70	48.97	65.55	25.10
	KETCHUP ($K = 2$)	**36.03**	13.95	**28.89**	22.93	**49.25**	**65.15**	25.32
	KETCHUP ($K = 4$)	35.96	**13.96**	28.87	22.93	49.21	65.21	25.40
	KETCHUP ($K = 8$)	35.68	13.88	28.76	**22.95**	49.23	65.20	**25.71**
	KETCHUP ($K = 16$)	35.31	13.68	28.51	22.94	49.24	65.18	25.47

Table 1: Main results on XSum, Europarl EN–NL, and GSM8K datasets. The best student result is in **bold**. ↑/↓The higher/lower, the better. We prompt the teacher and off-the-shelf student in a zero-shot manner to gain the first two rows. We select the best checkpoint based on the performance of the held-out validation set and report the performance of these checkpoints on the test set for all distilled students.

performance.

We first examine the performance of directly prompting the teacher and the non-distilled student model in a zero-shot manner, offering empirical lower and upper bounds for the KD process. Note that the bounds are not theoretically guaranteed; instead, KD is empirically expected to improve the student's performance but may still underperform the teacher, especially when the student is small. In our setup, the student is a T5-base model, which does not yield reasonable performance when prompted directly.

We then consider divergence-based distillation methods, including SeqKD and KL/JS/TVD distillations. As seen from the table, symmetric methods (JS, TVD)—which involve both exploitation of teacher predictions and exploration based on student predictions—tend to surpass asymmetric methods (SeqKD, KL), where the student follows teacher predictions without any exploration. The results are consistent with previous findings (Wen et al., 2023; Agarwal et al., 2024).

Next, we evaluate LLMR (Li et al., 2024), a text generation KD approach using REINFORCE. Results show that LLMR provides certain performance gain over non-RL KD methods, which is likely stemmed from the student's self-exploration, aligning with the observations in Li et al. (2024) and other recent RL-based text generation research (Ouyang et al., 2022; Liu et al., 2024; DeepSeek-AI et al., 2025).

To mitigate the high variance of REINFORCE in LLMR, we incorporate classic RL baseline terms (mean baseline and min-variance baseline) that are estimated from batch data. However, these methods are not effective in our scenario, as text generation has a very large state–action space, which makes the generated outputs in a batch less representative and the baseline term less useful.

By contrast, our KETCHUP employs a novel baseline formulation that largely reduces the variance of RL (Theorem 1) and improves RL optimization (Figure 1). Consequently, it delivers a noteworthy add-on performance gain on top of LLMR across three text generation tasks.

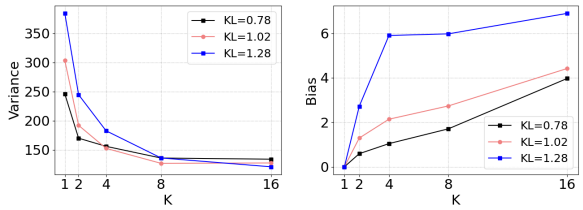
In the experiment, we also observe that a moderate K between 2 to 8 leads to the highest NLP performance, which is consistent with the return analysis in Figure 1. It is also noticed that RL return and NLP performance are not perfectly correlated, as the induced reward may not fully reflect the task metric such as BLEU and ROUGE scores, which is also known as reward hacking (Amodei et al., 2016; Hao et al., 2022; Ouyang et al., 2022).

Summary. Our main results show that the proposed KETCHUP (with a moderate K) improves RL optimization, which is generally translated to higher performance in various NLP tasks.

3.3 In-Depth Analyses

Variance and bias analysis. As shown by the theoretical analysis in Section 2.3, our approach provides a bias–variance trade-off by largely reducing the variance, although introducing a bias term. We empirically verify them in this analysis.

Figure 2a shows the variance of the K-step return, where we sample 32 outputs for a given input and use Eqn. (20) to estimate the variance of return;



(a) Variance (b) Bias

Figure 2: Variance and bias with different K values.

the variance is further averaged over 10K input samples. For the bias, we use Eqn. (25) for empirical estimation, and the results are shown in Figure 2b. We choose the value of K from $\{1, 2, 4, 8, 16\}$ to see the trends. Note that $K = 1$ corresponds to the competing approach LLMR (Li et al., 2024). In addition, we examine the impact of the initial student policy by considering students with various KL divergence levels from the teacher policy: a smaller KL divergence indicates that the student and teacher are more resemblant.

We observe that the variance decreases drastically as K increases, while the bias term increases steadily. The observations align with our theoretical analysis in Section 2.3 and Appendix B, suggesting the need for seeking a moderate K value to balance bias and variance.[3]

We also observe that when the student policy is initialized closer to the teacher policy (i.e., a smaller KL divergence), our KETCHUP generally demonstrates lower bias and variance. The bias reduction is predicted by our theoretical analysis in Appendix B, whereas the variance reduction is an empirical observation. Overall, the results demonstrate that pre-distillation is important to RL training for text generation, which is consistent with previous work (Ouyang et al., 2022; Li et al., 2024; DeepSeek-AI et al., 2025).

Model Size. We analyze RL-based KD approaches with different student sizes. Figure 3 presents the learning curves for student models initialized from T5-small (77M parameters), T5-base (250M parameters), and T5-large (800M parameters) using our K-step approach and the competing LLMR approach.

[3]Our bias–variance trade-off is different from that in a regression analysis (Hastie et al., 2009; Vapnik, 2013), where the total squared error is the sum of variance and squared bias, plus an irreducible noise. By contrast, the variance of return affects the smoothness of RL training, while bias affects the optimum quality (if converging); their total effect is not given by a simple addition.

Dataset	Method	Overall	Informativeness	Coherence
XSum	TVD	67.50%	68.15%	65.90%
	LLMR	69.95%	70.55%	66.30%
	KETCHUP	**73.50%**	**73.90%**	**70.40%**
EN-NL	TVD	53.80%	54.15%	54.85%
	LLMR	56.45%	55.85%	56.30%
	KETCHUP	**58.85%**	**57.95%**	**58.45%**

Table 2: LLM-based evaluation on the summarization and translation tasks. EN-NL refers to Europarl EN-NL dataset. We show the winning rates of each method over the KL distillation baseline in terms of overall quality, informativeness, and coherence.

As seen from the learning curves in Figure 3, the LLMR approach exhibits notable instability during RL training as the model size increases, especially when scaling to T5-large. Such a phenomenon is also reported in the RL literature: a large network is prone to overfit the limited sampled outputs, consequently leading to unstable performance on test data (Henderson et al., 2018; Cobbe et al., 2019).

On the contrary, our KETCHUP largely alleviates this issue by reducing the variance, which stabilizes the learning curves. Overall, our method achieves smoother training and higher performance with all model sizes, compared with the LLMR approach.

LLM Evaluation. We conduct an LLM evaluation as a surrogate of human evaluation, as classic NLP metrics (such as ROUGE and BLEU) may not fully reflect the quality of generated text. Specifically, we prompt the `Qwen2.5-72B-Instruct` (Qwen et al., 2025) LLM to conduct a pairwise evaluation of system outputs, against the commonly used KL distillation. We select TVD, LLMR, and our KETCHUP from Table 1 as the competitors, as pairwise evaluation is expensive. Our LLM evaluation considers multiple criteria, including overall quality, informativeness, and coherence. For each comparison, we query the LLM four times by swapping the two candidates and their IDs (namely, A and B), as LLM is prone to ID bias (Zheng et al., 2023) and positional bias (Shen et al., 2023). The detailed prompts are presented in Appendix D.

Table 2 shows the results of the LLM evaluation. We observe that our KETCHUP achieves the best winning rate in terms of all criteria (overall quality, informativeness, and coherence) on both datasets. These compelling results are consistent with the traditional task metrics in Table 1 and further demonstrate the effectiveness of our KETCHUP.

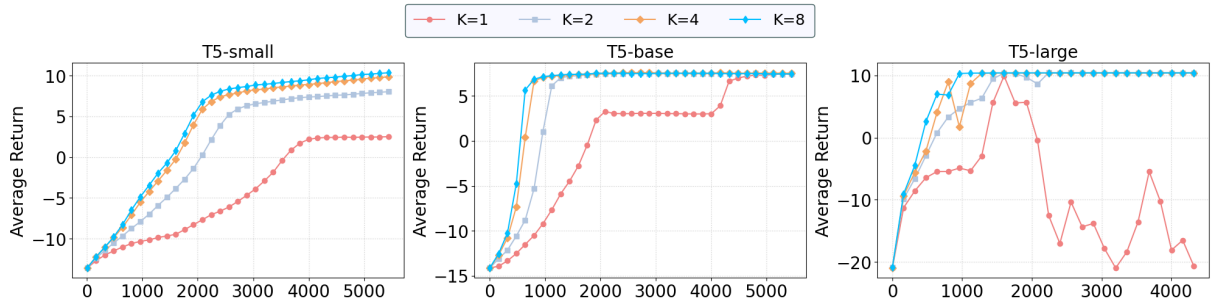


Figure 3: Learning curves with different K values and model sizes, where the x-axis is the number of training steps.

4 Related Work

Knowledge Distillation. The foundation of KD is laid by Buciluǎ et al. (2006), who performs KD by aligning the logits of the student with those of a teacher through squared error minimization. This framework is extended by Hinton et al. (2015), who propose to use KL divergence to match the output probability distributions of the teacher and student. Kim and Rush (2016) extend KD to the sequence level for auto-regressive models, and Wen et al. (2023) further propose a general framework of f-divergence minimization to mitigate the mode averaging and collapsing issues. These divergence-based KD approaches heavily rely on imitation of the teacher's predictions, neglecting the student's active exploration during learning.

Reinforcement Learning for Text Generation. Reinforcement learning (RL) offers a framework that enables a language model to explore during training. A key challenge in RL-based text generation lies in designing reward signals. Early efforts by Wu et al. (2018) employ task-specific metrics (e.g., BLEU for machine translation) as rewards, while Ouyang et al. (2022) leverage human preference data to train discriminative reward models. However, such methods require human engineering or human annotation.

Bridging RL and text generation KD. Recent work has sought to combine RL and KD by deriving rewards from teacher models. Hao et al. (2022) interpret a supervised-trained language model's pre-softmax logits as Q-values, deriving a step-wise reward function via Bellman Optimality equation, which alleviates the sparse reward issue commonly existing in other RL text generation scenarios (Wu et al., 2018; Ouyang et al., 2022). Building on this, Li et al. (2024) extend this approach to KD settings, where they induce a reward function from a large language model (serves as a teacher) and train a student model to maximize the teacher-induced cumulative reward. However, RL is known to suffer from high variance, and our paper proposes KETCHUP that largely reduces the variance of RL training.

Variance Reduction in RL. REINFORCE with baseline (Sutton and Barto, 2018; Rosenberg, 2021) mitigates the high variance issue by subtracting a baseline term derived from batch data. Actor–Critic methods (Konda and Tsitsiklis, 1999; Mnih et al., 2016) address this by learning a value function (critic) as the baseline term, but the inaccurate value estimates from the critic can lead to harmful updates in the actor's policy, while a poor decision by the actor can adversely affect the critic's learning. This often results in divergence of RL training (Bhatnagar et al., 2007; Fujimoto et al., 2018; Parisi et al., 2019). Recent RL work for large language models avoids learning a critic as the baseline term (DeepSeek-AI et al., 2025). Our KETCHUP exploits the mathematical structure of LM-induced rewards to derive a principled baseline for variance reduction, without learning an auxiliary neural network like a critic.

Another line of studies develops conservative policy optimization techniques like TRPO (Schulman et al., 2015) and PPO (Schulman et al., 2017), which constrain policy updates to prevent instability. Our work of estimating K-step return is compatible with this line of research. This goes beyond the scope of our paper, but can be explored in future work.

5 Conclusion

In this paper, we introduce KETCHUP, a K-step return induction framework for reinforcement learning for knowledge distillation in the text generation domain. Compared with conventional RL methods, our approach effectively reduces gradient variance, shown by both theoretical and empirical analyses. Extensive experiments across diverse text generation tasks verify that our approach improves RL training and boosts NLP task performance.

Limitations

While our work demonstrates both theoretical depth and empirical effectiveness, it is not without limitations. First, our RL-based knowledge distillation optimizes an induced reward function, which may not fully align with the NLP task (Ouyang et al., 2022; Pan et al., 2022; Gao et al., 2023). Nevertheless, our experiments support the claim that a better RL optimization generally leads to improved NLP metrics, as shown in Table 1. Also, traditional NLP metrics (such as ROUGE and BLEU scores) may not fully reflect human judgment. Therefore, we have also conducted LLM evaluation as a surrogate of human studies (Chiang and Lee, 2023; Liu et al., 2023; Lin and Chen, 2023), during which we have carefully eliminated the bias of LLMs (Zheng et al., 2023; Shen et al., 2023).

References

Rishabh Agarwal, Avi Singh, Lei Zhang, Bernd Bohnet, Luis Rosias, Stephanie Chan, Biao Zhang, Ankesh Anand, Zaheer Abbas, Azade Nova, and 1 others. 2025. Many-shot in-context learning. In *Advances in Neural Information Processing Systems*, pages 76930–76966.

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

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. 2016. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the Conference on Machine Translation (WMT19). In *Proceedings of the Conference on Machine Translation*, pages 1–61.

Richard Bellman. 1952. On the theory of dynamic programming. In *Proceedings of the National Academy of Sciences*, pages 716–719.

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*, pages 1171–1179.

Jalaj Bhandari, Daniel Russo, and Raghav Singal. 2018. A finite time analysis of temporal difference learning with linear function approximation. In *Proceedings of the Conference on Learning Theory*, pages 1691–1692.

Shalabh Bhatnagar, Mohammad Ghavamzadeh, Mark Lee, and Richard S Sutton. 2007. Incremental natural actor-critic algorithms. In *Advances in Neural Information Processing Systems*, pages 105–112.

Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, Cody Blakeney, and John Patrick Cunningham. 2024. LoRA learns less and forgets less. *Transactions on Machine Learning Research*, pages 2835–8856.

Johan Bjorck, Carla P Gomes, and Kilian Q Weinberger. 2022. Is high variance unavoidable in RL? A case study in continuous control. In *International Conference on Learning Representations*.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 535–541.

Cheng-Han Chiang and Hung-Yi Lee. 2023. Can large language models be an alternative to human evaluations? In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 15607–15631.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, and 1 others. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. 2019. Quantifying generalization in Reinforcement Learning. In *Proceddings of International Conference on Machine Learning*, pages 1282–1289.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Ona De Gibert, Mikko Aulamo, Yves Scherrer, and Jörg Tiedemann. 2024. Hybrid distillation from rbmt and nmt: Helsinki-NLP's submission to the shared task on translation into low-resource languages of Spain. In *Proceedings of the Conference on Machine Translation*, pages 908–917.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong

Shao, and et al. 2025. DeepSeek-R1: Incentivizing reasoning capability in LLMs via Reinforcement Learning.

Scott Fujimoto, Herke Hoof, and David Meger. 2018. Addressing function approximation error in actor-critic methods. In *Proceddings of the International Conference on Machine Learning*, pages 1587–1596.

Leo Gao, John Schulman, and Jacob Hilton. 2023. Scaling laws for reward model overoptimization. In *Proceddings of the International Conference on Machine Learning*, pages 10835–10866.

Evan Greensmith, Peter L Bartlett, and Jonathan Baxter. 2004. Variance reduction techniques for gradient estimates in Reinforcement Learning. *Journal of Machine Learning Research*, pages 1471–1530.

Yongchang Hao, Yuxin Liu, and Lili Mou. 2022. Teacher forcing recovers reward functions for text generation. In *Advances in Neural Information Processing Systems*, pages 12594–12607.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. 2018. Deep Reinforcement Learning that matters. In *Proceedings of the AAAI conference on artificial intelligence*, pages 3207–3214.

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

Miroslav Hrabal, Josef Jon, Martin Popel, Nam Luu, Danil Semin, and Ondřej Bojar. 2024. CUNI at WMT24 general translation task: LLMs,(Q)LoRA, CPO and model merging. In *Proceedings of the Conference on Machine Translation*, pages 232–246.

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

Jaehun Jung, Peter West, Liwei Jiang, Faeze Brahman, Ximing Lu, Jillian Fisher, Taylor Sorensen, and Yejin Choi. 2024. Impossible distillation for paraphrasing and summarization: How to make high-quality lemonade out of small, low-quality model. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4439–4454.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Michael J Kearns and Satinder Singh. 2000. Bias-variance error bounds for temporal difference updates. In *Proceedings of the Conference on Computational Learning Theory*, page 142–147.

Gyeongman Kim, Doohyuk Jang, and Eunho Yang. 2024. PromptKD: Distilling student-friendly knowledge for generative language models via prompt tuning. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 6266–6282.

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

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation*, pages 79–86.

Vijay Konda and John Tsitsiklis. 1999. Actor-critic algorithms. In *Advances in Neural Information Processing Systems*, pages 1008–1014.

Hayeon Lee, Rui Hou, Jongpil Kim, Davis Liang, Sung Ju Hwang, and Alexander Min. 2023. A study on knowledge distillation from weak teacher for scaling up pre-trained language models. In *Findings of the Association for Computational Linguistics: ACL*, pages 11239–11246.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Dongheng Li, Yongchang Hao, and Lili Mou. 2024. LLMR: Knowledge distillation with a large language model-induced reward. In *Proceedings of the Joint International Conference on Computational Linguistics, Language Resources and Evaluation*, pages 10657–10664.

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

Yen-Ting Lin and Yun-Nung Chen. 2023. LLM-Eval: Unified multi-dimensional automatic evaluation for open-domain conversations with large language models. In *Proceedings of the Workshop on NLP for Conversational AI (NLP4ConvAI 2023)*, pages 47–58.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-Eval: NLG Evaluation using GPT-4 with better human alignment. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. 2023. Teaching small language models to reason. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 1773–1781.

Sameen Maruf, André FT Martins, and Gholamreza Haffari. 2018. Contextual neural model for translating bilingual multi-speaker conversations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 101–112.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep Reinforcement Learning. In *Proceedings of the International Conference on Machine Learning*, pages 1928–1937.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, and 1 others. 2015. Human-level control through deep Reinforcement Learning. *Nature*, pages 529–533.

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. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, page 1797–1807.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, pages 27730–27744.

Alexander Pan, Kush Bhatia, and Jacob Steinhardt. 2022. The effects of reward misspecification: Mapping and mitigating misaligned models. In *International Conference on Learning Representations*.

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

Simone Parisi, Voot Tangkaratt, Jan Peters, and Mohammad Emtiyaz Khan. 2019. TD-Regularized Actor-Critic methods. *Machine Learning*, 108:1467–1501.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Workshop on Statistical Machine Translation*, page 392–395.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, and 24 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Qwen-Team. 2024. Introducing qwen1.5.

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

Mathieu Ravaut, Aixin Sun, Nancy Chen, and Shafiq Joty. 2024. On context utilization in summarization with large language models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 2764–2781.

David S. Rosenberg. 2021. Variance reduction in policy gradient. Lecture slides, DS-GA 3001: Tools and Techniques for ML.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *Proceedings of the International Conference on Machine Learning*, pages 1889–1897.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Hendra Setiawan. 2024. Accurate knowledge distillation via n-best reranking. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1330–1345.

Amrith Setlur, Saurabh Garg, Xinyang Geng, Naman Garg, Virginia Smith, and Aviral Kumar. 2025. RL on incorrect synthetic data scales the efficiency of llm math reasoning by eight-fold. In *Advances in Neural Information Processing Systems*, pages 43000–43031.

Chenhui Shen, Liying Cheng, Xuan-Phi Nguyen, Yang You, and Lidong Bing. 2023. Large language models are not yet human-level evaluators for abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 4215–4233.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the Conference of the Association*

for Machine Translation in the Americas: Technical Papers, pages 223–231.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for BERT model compression. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 4323–4332.

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

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: The impact of student initialization on knowledge distillation. *arXiv preprint arXiv:1908.08962*.

Dave Van Veen, Cara Van Uden, Louis Blankemeier, Jean-Benoit Delbrouck, Asad Aali, Christian Bluethgen, Anuj Pareek, Malgorzata Polacin, Eduardo Pontes Reis, Anna Seehofnerová, and 1 others. 2024. Adapted large language models can outperform medical experts in clinical text summarization. *Nature Medicine*, pages 1134–1142.

Vladimir Vapnik. 2013. *The Nature of Statistical Learning Theory*. Springer Science & Business Media.

Linyong Wang, Lianwei Wu, Shaoqi Song, Yaxiong Wang, Cuiyun Gao, and Kang Wang. 2025. Distilling structured rationale from large language models to small language models for abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25389–25397.

Tianduo Wang, Shichen Li, and Wei Lu. 2024. Self-training with direct preference optimization improves chain-of-thought reasoning. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 11917–11928.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. 2021. MiniLMv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP*, page 2140–2151.

Bolin Wei, Shuai Lu, Lili Mou, Hao Zhou, Pascal Poupart, Ge Li, and Zhi Jin. 2019. Why do neural dialog systems generate short and meaningless replies? a comparison between dialog and translation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7290–7294.

Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. 2023. f-divergence minimization for sequence-level knowledge distillation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 10817–10834.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist Reinforcement Learning. *Machine learning*, 8:229–256.

Lijun Wu, Fei Tian, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2018. A study of reinforcement learning for neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 3612–3621.

Tengyu Xu, Zhe Wang, Yi Zhou, and Yingbin Liang. 2020. Reanalysis of variance reduced temporal difference learning. In *International Conference on Learning Representations*.

Dezhi Ye, Junwei Hu, Jiabin Fan, Bowen Tian, Jie Liu, Haijin Liang, and Jin Ma. 2025. Best practices for distilling large language models into BERT for web search ranking. In *Proceedings of the International Conference on Computational Linguistics: Industry Track*, pages 128–135.

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. 2023. Large language models are not robust multiple choice selectors. In *International Conference on Learning Representations*.

A Proof of Theorem 1

Using K-step returns as a learning signal to learn a student policy π guarantees reduced variance in return estimation compared to the full trajectory return, i.e., $\mathrm{Var}[\hat{G}_t] \leq \mathrm{Var}[G_t]$. (Detailed in Theorem 1).

Proof. We denote the variance of $q(s,a)$ and $\max_{a' \in \mathcal{A}} q(s,a')$ as:

$$\sigma^2_{\mathcal{S},\mathcal{A}} = \mathrm{Var}_{s,a}\big[q(s,a)\big], \tag{13}$$

$$\sigma^2_{\mathcal{S}} = \mathrm{Var}_{s}\Big[\max_{a' \in \mathcal{A}} q(s,a')\Big]. \tag{14}$$

We first decompose the variance of the actual return G_t:

$$\begin{aligned}
\mathrm{Var}[G_t] &= \mathrm{Var}\Big[\sum_{i=0}^{T-t} r_{t+i}\Big] && \text{[definition of } G_t\text{]} && (15)\\
&= \sum_{i=0}^{T-t} \mathrm{Var}\Big[q(s_{t+i}, a_{t+i}) - \max_{a' \in \mathcal{A}} q(s_{t+i+1}, a')\Big] && \text{[iid assumption]} && (16)\\
&= \sum_{i=0}^{T-t} \Big(\mathrm{Var}\big[q(s_{t+i}, a_{t+i})\big] + \mathrm{Var}\big[\max_{a' \in \mathcal{A}} q(s_{t+i+1}, a')\big]\Big) && \text{[iid assumption]} && (17)\\
&= \sum_{i=0}^{T-t} \big(\sigma^2_{\mathcal{S},\mathcal{A}} + \sigma^2_{\mathcal{S}}\big) && && (18)\\
&= (T-t+1)\big(\sigma^2_{\mathcal{S},\mathcal{A}} + \sigma^2_{\mathcal{S}}\big). && && (19)
\end{aligned}$$

Next, we decompose the variance of our K-step approximate return $\hat{G}_t$:

$$\begin{aligned}
\mathrm{Var}[\hat{G}_t] &= \mathrm{Var}\Bigg[\sum_{i=0}^{\lfloor \frac{T-t}{k} \rfloor} \Big(q(s_{t+ik}, a_{t+ik}) - \max_{a' \in \mathcal{A}} q(s_{t+(i+1)k}, a')\Big)\Bigg] && \text{[by Eqn. (7)]} && (20)\\
&= \sum_{i=0}^{\lfloor \frac{T-t}{k} \rfloor} \mathrm{Var}\Big[q(s_{t+ik}, a_{t+ik}) - \max_{a' \in \mathcal{A}} q(s_{t+(i+1)k}, a')\Big] && \text{[iid assumption]} && (21)\\
&= \sum_{i=0}^{\lfloor \frac{T-t}{k} \rfloor} \Big(\mathrm{Var}\big[q(s_{t+ik}, a_{t+ik})\big] + \mathrm{Var}\big[\max_{a' \in \mathcal{A}} q(s_{t+(i+1)k}, a')\big]\Big) && \text{[iid assumption]} && (22)\\
&= \sum_{i=0}^{\lfloor \frac{T-t}{k} \rfloor} \big(\sigma^2_{\mathcal{S},\mathcal{A}} + \sigma^2_{\mathcal{S}}\big) && && (23)\\
&= \Big(\Big\lfloor \frac{T-t}{k} \Big\rfloor + 1\Big)\big(\sigma^2_{\mathcal{S},\mathcal{A}} + \sigma^2_{\mathcal{S}}\big). && && (24)
\end{aligned}$$

Comparing Eqns. (19) and (24), we immediately have $\mathrm{Var}[\hat{G}_t] \leq \mathrm{Var}[G_t]$, completing the proof. □

B Bias Analysis

In this section, we analyze the bias introduced by using the K-step return $\hat{G}_t$ in place of the actual return G_t. Recall that they differ by a baseline term shown in Eqns. (9) and (11), and this discrepancy introduces bias in the return estimation:

$$\text{bias of return} = \mathbb{E}_{\pi_\theta}\Big[(\hat{G}_t - G_t)\Big] = \mathbb{E}_{\pi_\theta}\Bigg[\sum_{\substack{i=0 \\ i \not\equiv 0 \,(\mathrm{mod}\, k)}}^{T-1} \Big[q(s_{t+Ki+1}, a_{t+Ki+1}) - \max_{a' \in \mathcal{A}} q(s_{t+Ki+1}, a')\Big]\Bigg] \tag{25}$$

gradient estimation:

$$\text{bias of gradient} = \mathbb{E}_{\pi_\theta}\Big[(\hat{G}_t - G_t)\nabla_\theta \log \pi_\theta(a_t \mid s_t)\Big] = \mathbb{E}_{\pi_\theta}\Big[-b_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\Big] \tag{26}$$

We show below that a smaller value of K reduces bias, providing a bias-variance tradeoff for REINFORCE. Further, we will show that the bias converges to zero as the student policy becomes more optimal, assuming all Q-values are distinct.

Bias Reduction with Smaller K. The baseline term defined in Eqn. (11) is given by

$$b_t = \sum_{\substack{i=0 \\ i \not\equiv 0 \,(\mathrm{mod}\, k)}}^{T-1} \Big[q(s_{t+Ki+1}, a_{t+Ki+1}) - \max_{a' \in \mathcal{A}} q(s_{t+Ki+1}, a')\Big]. \tag{27}$$

Since

$$q(s_{t+Ki+1}, a_{t+Ki+1}) - \max_{a' \in \mathcal{A}} q(s_{t+Ki+1}, a') \leq 0, \tag{28}$$

a smaller K reduced the number of terms in the summation. This decreases $|b_t|$, which in turn decreases the magnitude of the gradient bias in Eqn. (26).

Bias Convergence to Zero. Suppose the student policy is optimal, i.e., greedy with respect to the teacher's Q-value function $q(s, a)$, given by

$$a_{t+i} = \arg\max_{a' \in \mathcal{A}} q(s_{t+i}, a'). \tag{29}$$

It is easy to see from Eqn. (27) that $b_t = 0$, implying that

$$\mathbb{E}_{\pi_\theta}\Big[b_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\Big] = 0. \tag{30}$$

Suppose the Q-values for different actions are distinct (in which case argmax is continuous), the result further suggests that the bias term would converge to zero, if the student policy is closer to optimal during training.

Dataset	Task	# of Samples		
		Train	Dev	Test
XSum (Narayan et al., 2018)	Summarization	202,926	11,332	11,333
Europarl EN-NL (Koehn, 2005)	Machine Translation	1,167,808	10,014	10,016
GSM8K (Cobbe et al., 2021)	Arithmetic reasoning	6,705	768	1,319

Table 3: Statistics of our datasets.

C Experimental setting Details

Training settings: we used the AdamW optimizer (Loshchilov and Hutter, 2019) with default hyperparameters $\beta = (0.9, 0.999)$ on these three datasets. We chose a small batch size of 8 to fit the student. The learning rate is set as $3e^{-5}$. All student models were trained for 5 epochs for pre-distillation and another 2 epochs for each advanced distilling method (i.e. JSD and TVD).

For further RL-based training, we kept using the same AdamW optimizer with default hyperparameters, as well as the batch size of 8, and we set the learning rate to $1e^{-5}$. Hyperparameter details for RL-based training is shown in Table 4 & 5.

Inference settings: We follow previous work and use greedy decoding consistently for all distillations in three datasets.

It should also be noted that in the arithmetic reasoning task, we follow Wang et al. (2024) and integrate an external calculator into the decoding process of both teacher and student models, which largely improves the models' performance. More implementation details can be found in Section 3.2 in Wang et al. (2024).

Hyperparameter	**Value**
Training Epochs	10
Train Batch size	8
Eval Batch size	32
Optimizer	AdamW
Grad Accumulation Steps	32
Eval Split	Test
Reward Clip Range	[-100, 100]
Dropout	0.0
Learning Rate (LR)	0.00001
Max Input Length	1024 (Xsum) / 80 (Europarl EN-NL)
Max Output Length	64 (Xsum) / 80 (Europarl EN-NL)
Evaluation	Greedy

Table 4: Hyperparameter Details for experiments on Xsum and Europarl EN-NL.

Hyperparameter	**Value**
Training Epochs	5
Train Batch size	8
Eval Batch size	32
Optimizer	AdamW
Grad Accumulation Steps	4
Eval Split	Test
Reward Clip Range	[-100, 100]
Dropout	0.0
Learning Rate (LR)	0.00001
Max Input Length	200
Max Output Length	300
Evaluation	Greedy

Table 5: Hyperparameter Details for experiments on GSM8K.

D Prompts templates for LLM Evaluation

Table 6 and Table 7 present our prompts template for LLM evaluation on the summarization task and machine translation task, respectively.

Please evaluate the overall quality of the following summaries given the document.

Evaluation Criteria:
Overall Quality: A good summary should be both precise and concise, summarizing the most important points in the given document, without including unimportant or irrelevant details

Document: **[Source]**
Summary **[ID1]**: **[Summary-A]**
Summary **[ID2]**: **[Summary-B]**

FIRST, provide a one-sentence comparison of the two summaries for overall quality, explaining which you prefer and why.
SECOND, on a new line, state only the ID to indicate your choice. Your response should use the format:
Overall Quality: <one-sentence comparison and explanation>
Preferred: <summary ID>

Please evaluate the informativeness of the following summaries given the document.

Evaluation Criteria:
Informativeness: Does it include the most important details while excluding irrelevant content?

Document: **[Source]**
Summary **[ID1]**: **[Summary-A]**
Summary **[ID2]**: **[Summary-B]**

FIRST, provide a one-sentence comparison of the two summaries for informativenss, explaining which you prefer and why.
SECOND, on a new line, state only the ID to indicate your choice. Your response should use the format:
Informativeness: <one-sentence comparison and explanation>
Preferred: <summary ID>

Please evaluate the coherence of the following summaries given the document.

Evaluation Criteria:
Coherence: Is the summary logically structured and easy to follow?

Document: **[Source]**
Summary **[ID1]**: **[Summary-A]**
Summary **[ID2]**: **[Summary-B]**

FIRST, provide a one-sentence comparison of the two summaries for coherence, explaining which you prefer and why.
SECOND, on a new line, state only the ID to indicate your choice. Your response should use the format:
Informativeness: <one-sentence comparison and explanation>
Preferred: <summary ID>

Table 6: Prompt templates for LLM evaluation on the summarization task in terms of overall quality, informativeness, and coherence. Here, "**Source**" is the document to be summarized. The choices of IDs are "A" and "B"; "**Summary-A**" and "**Summary-B**" are replaced with model-generated texts. Since LLMs are not robust to ID and order (Zheng et al., 2023; Shen et al., 2023), we enumerate different combinations for a given pair, resulting in four LLM queries.

E Results on more models

KD studies on seq2seq tasks have largely centred on encoder-decoder structures such as T5 (Raffel et al., 2020; Chung et al., 2024) and BART (Lewis et al., 2020) models (Wen et al., 2023; Li et al., 2024; Agarwal et al., 2024; Jung et al., 2024; Wang et al., 2025). To answer reviewers' likely question about KETCHUP's behaviour on recent popular decoder-only architectures, we also applied it to the Qwen1.5 model series (Qwen-Team, 2024) and report the results in Table 8.

Please evaluate the overall quality of the following translations from English to Dutch.

Evaluation Criteria:
Overall Quality: A good translation should: 1) faithfully reflect the meaning of the source text; 2) avoid adding unnecessary or irrelevant details. 3) use natural and fluent Dutch.

Source: **[Source]**
Translation **[ID1]**: **[Translation-A]**
Translation **[ID2]**: **[Translation-B]**

FIRST, provide a one-sentence comparison of the two translations for overall quality, explaining which you prefer and why.
SECOND, on a new line, state only the ID to indicate your choice. Your response should use the format:
Overall Quality: <one-sentence comparison and explanation>
Preferred: <translation ID>

Please evaluate the informativeness of the following translations from English to Dutch.

Evaluation Criteria:
Informativeness: Does the translation preserve all key information without adding irrelevant details?

Source: **[Source]**
Translation **[ID1]**: **[Translation-A]**
Translation **[ID2]**: **[Translation-B]**

FIRST, provide a one-sentence comparison of the two translations for informativeness, explaining which you prefer and why.
SECOND, on a new line, state only the ID to indicate your choice. Your response should use the format:
Informativeness: <one-sentence comparison and explanation>
Preferred: <translation ID>

Please evaluate the coherence of the following translations from English to Dutch.

Evaluation Criteria:
Coherence: Is the translation fluent, logically structured, and easy to understand in Dutch?

Source: **[Source]**
Translation **[ID1]**: **[Translation-A]**
Translation **[ID2]**: **[Translation-B]**

FIRST, provide a one-sentence comparison of the two translations for coherence, explaining which you prefer and why.
SECOND, on a new line, state only the ID to indicate your choice. Your response should use the format:
Informativeness: <one-sentence comparison and explanation>
Preferred: <translation ID>

Table 7: Prompt templates for LLM evaluation on the machine translation task in terms of overall quality, informativeness, and coherence. Here, "**Source**" is the source sentence to be translated. The choices of IDs are "A" and "B"; "**Translation-A**" and "**Translation-B**" are replaced with model-generated texts. We still enumerate different combinations for a given pair, resulting in four LLM queries.

Model	XSum (ROUGE-1↑)	Europarl (BLEU4↑)	GSM8K (Acc. (%)↑)
Teacher (Qwen1.5-4B)	38.15	21.32	42.08
Student (Qwen1.5-0.5B)	8.80	0.02	0.00
KL (Hinton et al., 2015)	31.29	15.76	26.31
TVD (Wen et al., 2023)	31.18	16.22	26.99
LLMR (Li et al., 2024)	31.61	15.90	27.29
KETCHUP	**32.28**	**16.46**	**28.13**

Table 8: Distillation results on XSum, Europarl EN–NL, and GSM8K using Qwen1.5 models. Higher ↑ is better. The best K values are 2, 2, and 16 for the three datasets, respectively.