ToDi: Token-wise Distillation via Fine-Grained Divergence Control

Seongryong Jung[1,2], **Suwan Yoon**[1], **DongGeon Kim**[1], **Hwanhee Lee**[1]*
[1]Department of Artificial Intelligence, Chung-Ang University, [2]Dmtlabs
{jungsr1116, swyoon0312, golddonggun, hwanheelee}@cau.ac.kr

Abstract

Large language models (LLMs) offer impressive performance but are impractical for resource-constrained deployment due to high latency and energy consumption. Knowledge distillation (KD) addresses this by transferring knowledge from a large teacher to a smaller student model. However, conventional KD, notably approaches like Forward KL (FKL) and Reverse KL (RKL), apply uniform divergence loss across the entire vocabulary, neglecting token-level prediction discrepancies. By investigating these representative divergences via gradient analysis, we reveal that FKL boosts underestimated tokens, while RKL suppresses overestimated ones, showing their complementary roles. Based on this observation, we propose **Token-wise Distillation (ToDi)**, a novel method that adaptively combines FKL and RKL per token using a sigmoid-based weighting function derived from the teacher-student probability log-ratio. ToDi dynamically emphasizes the appropriate divergence for each token, enabling precise distribution alignment. We demonstrate that ToDi consistently outperforms recent distillation baselines using uniform or less granular strategies across instruction-following benchmarks. Extensive ablation studies and efficiency analysis further validate ToDi's effectiveness and practicality.[1]

1 Introduction

Recent advances in large language models (LLMs), driven by scaling up model size, have substantially enhanced their ability to follow user instructions and generate contextually appropriate responses (Brown et al., 2020; Sanh et al., 2022; Wei et al., 2022; Chung et al., 2024). However, the continued enlargement of model size introduces several challenges, including increased inference latency, high energy consumption, and inefficiency in resource-constrained environments. To address these issues, knowledge distillation (KD; Hinton et al., 2015) has been widely adopted; this approach aims to minimize the performance gap between teacher and student models by transferring knowledge from a high-performing large teacher model to a smaller student model. Recently, various knowledge distillation techniques for enhancing the efficiency of LLMs have been proposed, and research surrounding these methods is actively underway (Zhang et al., 2024b; Feng et al., 2024; Shing et al., 2025).

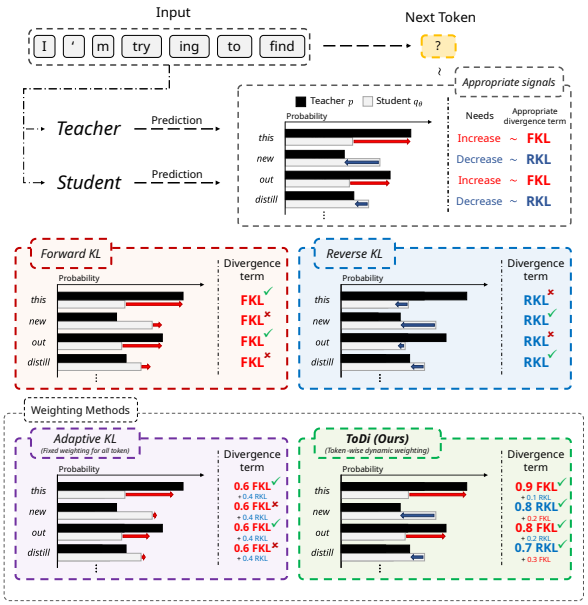


Figure 1: Token-wise learning signals for KL-based distillation objectives. Conventional methods apply a fixed divergence across the entire vocabulary, while ToDi dynamically blends Forward and Reverse KL per-token based on the teacher–student probability ratio, balancing gradients across all tokens.

Conventional knowledge distillation methods often employ divergences such as Forward KL (FKL) and Reverse KL (RKL) to minimize the discrepancy between teacher and student distributions (Hinton et al., 2015; Gu et al., 2024). How-

*Corresponding author.
[1]The code is available at https://github.com/jungseongryong/ToDi

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 8078–8091
November 4-9, 2025

ever, as depicted in the example for FKL and RKL in Figure 1, these approaches apply a single divergence uniformly across the *entire vocabulary*, regardless of how severely the student misestimates each token. This uniform-loss assumption persists in symmetric and hybrid variants of FKL/RKL (Wen et al., 2023; Ko et al., 2024; Agarwal et al., 2024), and even dynamic combinations at the vocabulary-set or time-step level like Adaptive KL (Wu et al., 2025). We hypothesize that such uniform treatment is sub-optimal because different tokens may require different correction signals.

In this paper, we analyze the limitation of uniform application by investigating token-specific optimal signals through a gradient-based analysis of divergences in existing KD methods (Section 3). This analysis reveals that FKL effectively increases the probability of tokens that the student model underestimates relative to the teacher model, whereas RKL excels at suppressing the probability of tokens that it overestimates, showing their distinct and complementary roles. However, existing methods apply a uniform divergence loss across the entire vocabulary, failing to leverage these complementary signals effectively at the token level. As shown in Figure 1, this uniformity prevents appropriate training signals for individual tokens, particularly when the student significantly over- or underestimates the teacher's distribution.

Motivated by this insight, we propose a novel distillation method, **Token-wise Distillation (ToDi)** (Section 4). As illustrated in Figure 1, ToDi dynamically balances the contributions of FKL and RKL based on token-level prediction discrepancies by adaptively combining them per-token using a token-specific weighting function. This approach directly provides tailored training signals that capture fine-grained differences between the teacher and student distributions, going beyond uniform loss application.

We demonstrate ToDi's effectiveness through extensive experiments and show that ToDi consistently outperforms recent distillation baselines on various instruction-following benchmarks, achieving superior ROUGE-L scores and higher win rates in GPT-4-based pairwise evaluations. Furthermore, we validate the critical importance of ToDi's token-wise divergence control. We also show that ToDi maintains stable training and linear time complexity with respect to vocabulary size, highlighting its efficiency and practicality.

The principal contributions of this paper are as follows:

- We analyze and show the complementary roles of FKL and RKL for KD through gradient analysis.
- Based on this analysis, we propose ToDi, a new KD method that adaptively combines FKL and RKL per token according to prediction discrepancies and enables fine-grained distribution alignment.
- We provide theoretical grounding for ToDi and demonstrate its superior performance over existing methods through extensive experiments on instruction following tasks.

2 Related Work

2.1 Objective Functions of KD

In knowledge distillation (Hinton et al., 2015), the student model is trained to mimic the teacher's output distribution by minimizing the divergence loss. The FKL induces mode averaging, smoothing a multimodal teacher distribution, while the RKL causes mode collapse, driving the student to focus on a single mode (Koller and Friedman, 2009; Chan et al., 2022; Wang et al., 2024). To counter these extremes, Wen et al. (2023) adopted the symmetric Jensen–Shannon Divergence (JSD), and Agarwal et al. (2024) generalized it to interpolate between FKL and RKL. Skewed KL variants (SKL, SRKL) further mix the student distribution into the teacher's distribution for stability (Ko et al., 2024), while TAID (Shing et al., 2025) inserts a time-varying intermediate distribution between teacher and student.

Despite these advances, all prior work on applying KD for language models still processes the *entire* vocabulary distribution at every sequence position and applies a uniform loss across tokens. This coarse treatment misses token-level mismatches between teacher and student, limiting the student's ability to replicate the teacher's fine-grained predictive structure. Our proposed method aims to overcome this limitation by applying a token-wise dynamic divergence control, precisely addressing these fine-grained mismatches.

2.2 Dynamic Combination of FKL and RKL

Several studies have explored combining FKL and RKL to take advantage of both methods. Lee et al. (2023) proposed a straightforward additive combination, whereas Amara et al. (2022) introduced

BD-KD, which adjusts the weights of FKL and RKL on a per-sample basis via the entropy gap between teacher and student distributions. Wu et al. (2025) presented AKL—tailored for LLM distillation—that adaptively combines the two divergences based on the observation that, in early training, FKL primarily learns head predictions while RKL focuses on tail predictions. More recently, Ko et al. (2025) proposed DistiLLM-2, a contrastive framework that applies distinct divergence functions depending on whether the responses are generated by the teacher or the student.

Nevertheless, such approaches still dynamically apply FKL and RKL to the entire vocabulary distribution at every sequence position without assigning *dynamic weights to individual tokens*. This limitation prevents a fine-grained reflection of token-level prediction differences between teacher and student, thereby hindering the learning of detailed predictive structures. In contrast, our proposed ToDi method dynamically balances FKL and RKL on a per-token basis, capturing fine-grained probability discrepancies and enabling more precise predictive structure learning.

3 Gradient Behavior of FKL and RKL

In this section, we formalize knowledge distillation for autoregressive LLMs and analyze the FKL and RKL objectives from a gradient perspective. By understanding the gradients, we precisely examine how the learning signal for each vocabulary token depends on the relative magnitudes of the teacher probability $p(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$ and the student probability $q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$, providing insight into token-specific optimal signals.

3.1 Preliminaries

An autoregressive LLMs generates an output sequence $\mathbf{y} = [y_1, \ldots, y_{|\mathbf{y}|}]$ conditioned on an input sequence $\mathbf{x}$. At each time step t, it selects one token from a finite vocabulary $\mathcal{V} = \{v_1, \ldots, v_{|\mathcal{V}|}\}$.

KD minimizes the discrepancy between the teacher's distribution $p(y_t \mid \mathbf{y}_{<t}, \mathbf{x})$ and the student's distribution $q_\theta(y_t \mid \mathbf{y}_{<t}, \mathbf{x})$, where θ denotes the student parameters and $\mathbf{y}_{<t} = [y_1, \ldots, y_{t-1}]$ are the tokens generated before step t.

During KD, the loss is typically instantiated as either the *FKL* or the *RKL*. At time step t, the contribution of each divergence for a token $v_i \in \mathcal{V}$ is defined as:

$$D_{\text{FKL}}^{(t,i)}(p, q_\theta) = p(v_i \,|\, \mathbf{y}_{<t}, \mathbf{x}) \log \frac{p(v_i \,|\, \mathbf{y}_{<t}, \mathbf{x})}{q_\theta(v_i \,|\, \mathbf{y}_{<t}, \mathbf{x})}, \tag{1}$$

$$D_{\text{RKL}}^{(t,i)}(p, q_\theta) = q_\theta(v_i \,|\, \mathbf{y}_{<t}, \mathbf{x}) \log \frac{q_\theta(v_i \,|\, \mathbf{y}_{<t}, \mathbf{x})}{p(v_i \,|\, \mathbf{y}_{<t}, \mathbf{x})}. \tag{2}$$

Training Objective We accumulate the token-level divergences (from Equations 1 and 2) over all time steps and vocabulary entries to obtain the total forward and reverse KL divergence losses:

$$\mathcal{L}_{\text{FKL}} = \sum_{t=1}^{|\mathbf{y}|} \sum_{i=1}^{|\mathcal{V}|} D_{\text{FKL}}^{(t,i)}\big(p,\, q_\theta\big), \tag{3}$$

$$\mathcal{L}_{\text{RKL}} = \sum_{t=1}^{|\mathbf{y}|} \sum_{i=1}^{|\mathcal{V}|} D_{\text{RKL}}^{(t,i)}\big(p,\, q_\theta\big). \tag{4}$$

3.2 Theoretical Analysis

We theoretically analyze the FKL and RKL training signals. In particular, we examine how the two divergences exert opposite corrective effects depending on the relative magnitudes of the teacher distribution $p(y_t \mid \mathbf{y}_{<t}, \mathbf{x})$ and the student distribution $q_\theta(y_t \mid \mathbf{y}_{<t}, \mathbf{x})$. The analysis is grounded in the token-level definitions given in Equations 1 and 2.

Gradient Form. The partial derivatives of each divergence with respect to q_θ are:

$$\frac{\partial}{\partial q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})} D_{\text{FKL}}^{(t,i)}(p, q_\theta) = -\frac{p(v_i \mid \mathbf{y}_{<t}, \mathbf{x})}{q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})}, \tag{5}$$

$$\frac{\partial}{\partial q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})} D_{\text{RKL}}^{(t,i)}(p, q_\theta) = \log \frac{q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})}{p(v_i \mid \mathbf{y}_{<t}, \mathbf{x})} + 1. \tag{6}$$

We describe the detailed derivations in Appendix A.

Difference in Training Signals by Relative Probability. The two gradients can be compared through a single ratio $r = \frac{p\left(v_i|\mathbf{y}_{<t},\mathbf{x}\right)}{q_\theta\left(v_i|\mathbf{y}_{<t},\mathbf{x}\right)}$:

- $r > 1$ **(the student model underestimates).** Here, the FKL gradient $-r$ is a negative value whose magnitude exceeds 1, pushing q_θ to *increase* sharply. The RKL gradient, $\log \frac{1}{r} + 1$, turns negative only when $r > e$ and its magnitude is smaller, producing a relatively weak corrective signal. Thus, for tokens underestimated by the student, FKL provides the dominant "push-up" signal.

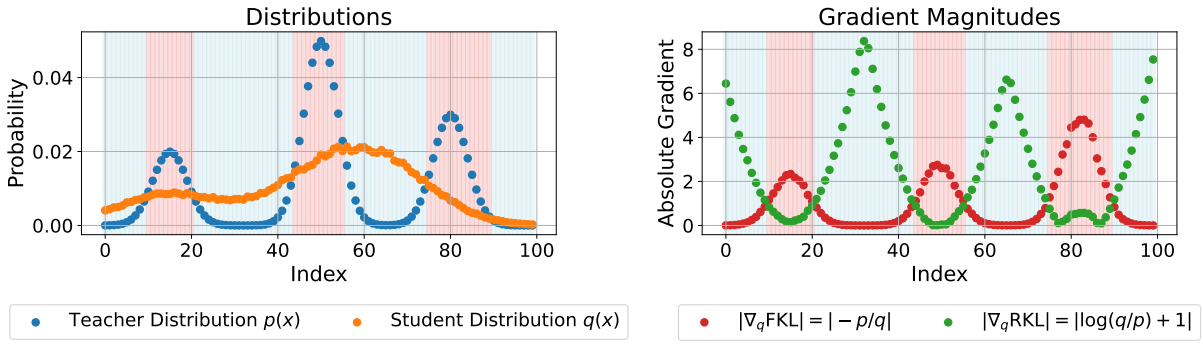


Figure 2: Toy example demonstrating the behavior of FKL and RKL gradients. In regions where $p > q$, FKL provides stronger gradients, while in regions where $q > p$, RKL provides stronger learning signals.

- $r < 1$ **(the student model overestimates).** In this case, the FKL gradient remains a small negative value, whereas the RKL gradient is a positive value greater than 1, providing a strong signal to *decrease* q_θ. Consequently, when the student overestimates, RKL provides the dominant "pull-down" signal.

Case	Forward KL	Reverse KL
$\mathbf{p} > \mathbf{q}_\theta$	↑ **Strong push-up**	≈ *Weak push-up*
$\mathbf{p} < \mathbf{q}_\theta$	≈ *Weak pull-down*	↓ **Strong pull-down**

Table 1: Complementary training signals of FKL vs. RKL.

In summary, as organized in Table 1, our theoretical analysis reveals that FKL and RKL provide complementary training signals around the boundary $r = 1$: FKL strongly encourages increasing student probability (i.e. push-up) for underestimated tokens ($p > q_\theta$), while RKL strongly encourages decreasing student probability (i.e. pull-down) for overestimated tokens ($q_\theta > p$).

3.3 Empirical Analysis of a Toy Example

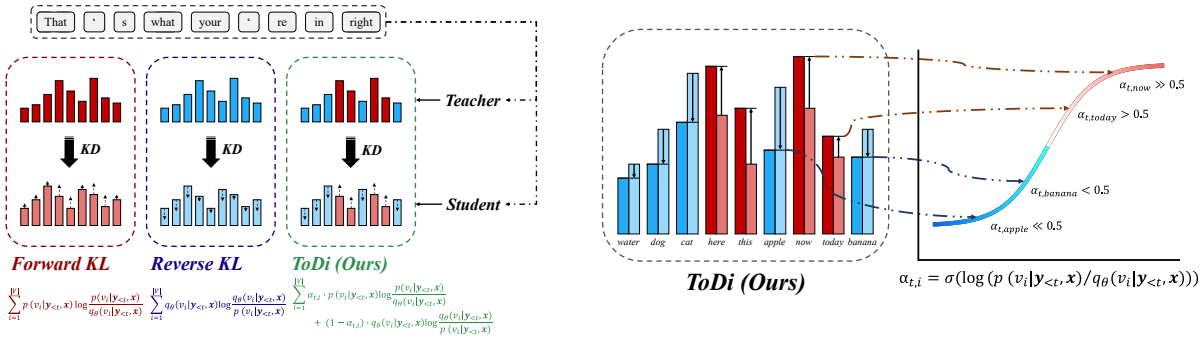
To empirically examine how FKL and RKL gradient magnitudes depend on the relative teacher–student probabilities at each token, we construct a toy example by defining teacher distribution $p(x)$ and student distribution $q(x)$. Figure 2 illustrates the comparison of gradient magnitudes according to the relative relationship between the teacher distribution $p(x)$ and the student distribution $q(x)$ in a toy example. The left panel shows where the two distributions intersect, with the regions $p(x) > q(x)$ and $q(x) > p(x)$ shaded separately. The right panel visualizes, for each index, the gradient magnitudes induced by FKL and RKL.

Consistent with the theoretical analysis, we observe in the toy example that in the region where $p(x) > q(x)$, FKL produces substantially larger gradients than RKL, delivering a strong corrective signal for tokens that the student under-estimates relative to the teacher. Conversely, in the region where $q(x) > p(x)$, the magnitude of the RKL gradient is greater, indicating a strong signal to suppress over-estimation. Consequently, FKL and RKL provide specialized training signals in different scenarios.

4 ToDi

In this section, we introduce **Token-wise Distillation (ToDi)**, which dynamically adjusts the contributions of FKL and RKL based on the token-level probability ratios in the teacher and student distributions.

Objective Functions for ToDi. As shown in the gradient analysis of Section 3, for each vocabulary token v_i, when $p(v_i \mid \mathbf{y}_{<t}, \mathbf{x}) > q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$, the FKL provides a learning signal that effectively increases q_θ, and conversely, when $q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x}) > p(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$, the RKL offers a signal that reduces q_θ. Building on this insight into their complementary roles, we propose a novel distillation method, **Token-wise Distillation (ToDi)**, which dynamically combines FKL and RKL according to the relative magnitudes of the teacher probability $p(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$ and the student probability $q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$. Unlike conventional approaches that apply a single loss uniformly across the entire vocabulary, ToDi computes a specific loss for each token v_i at time step t, denoted $D_{\text{ToDi}}^{(t,i)}$. This token-level loss is a weighted sum of the token's FKL and RKL divergences. Specifically, the token-level loss



Token-wise Distillation (ToDi) **Weighting Function for ToDi**

Figure 3: Illustration of the Token-wise Distillation. **(Left)** For each vocabulary token, the contributions of FKL and RKL are dynamically combined using a token-specific weight $\alpha_{t,i}$. **(Right)** The weight $\alpha_{t,i}$, determined by the teacher–student probability ratio, smoothly increases FKL emphasis when $p > q_\theta$ and RKL emphasis when $q_\theta > p$.

$D_{\text{ToDi}}^{(t,i)}$ is defined as follows:

$$D_{\text{ToDi}}^{(t,i)}(p, q_\theta) = \alpha_{t,i} \cdot D_{\text{FKL}}^{(t,i)}(p, q_\theta) + \left(1 - \alpha_{t,i}\right) \cdot D_{\text{RKL}}^{(t,i)}(p, q_\theta), \tag{7}$$

where $\alpha_{t,i}$ is a **token-specific weight** dynamically computed for each token v_i based on the relative teacher and student probabilities.

As illustrated in Figure 3 (Left), we utilize the weighting function to amplify the contribution of FKL when needed (when $p > q_\theta$) and amplify the contribution of RKL when needed (when $q_\theta > p$).

The overall distillation loss is then the sum of these token-level losses over all time steps and vocabulary entries:

$$\mathcal{L}_{\text{ToDi}} = \sum_{t=1}^{|\mathbf{y}|} \sum_{i=1}^{|\mathcal{V}|} D_{\text{ToDi}}^{(t,i)}(p, q_\theta). \tag{8}$$

Weighting Function for ToDi. The core of ToDi's token-wise control lies in the weighting function that determines $\alpha_{t,i}$. This weight must dynamically adjust according to the relative magnitudes of $p(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$ and $q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$ to effectively leverage the complementary nature of FKL and RKL.

Specifically, the token-specific weight $\alpha_{t,i}$ is defined by a function W of these probabilities:

$$\alpha_{t,i} = W\big(p(v_i \mid \mathbf{y}_{<t}, \mathbf{x}),\, q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})\big) \tag{9}$$

The function W should assign a larger value (thus increasing the contribution of FKL) when $p(v_i \mid \mathbf{y}_{<t}, \mathbf{x}) > q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$, so as to boost the student's probability. Conversely, when $q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x}) > p(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$, a smaller function value (favoring RKL) is appropriate. To satisfy these requirements and enable fine-grained control, the function W must meet the following four conditions:

- If $p(v_i \mid \mathbf{y}_{<t}, \mathbf{x}) > q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$, then $\alpha_{t,i}$ should be greater than 0.5 to emphasize FKL.
- If $q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x}) > p(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$, then $\alpha_{t,i}$ should be less than 0.5 to emphasize RKL.
- To allocate more extreme weights when the teacher–student probability gap is larger, $\alpha_{t,i}$ must be a monotonically increasing function of the ratio $p(v_i \mid \mathbf{y}_{<t}, \mathbf{x})/q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$.
- $\alpha_{t,i}$ must lie within the valid weight range $[0, 1]$.

To satisfy all four conditions, we adopt the sigmoid function for W, defining $\alpha_{t,i}$ as:

$$\alpha_{t,i} = \text{sg}\left[\sigma\left(\log \frac{p\big(v_i \mid \mathbf{y}_{<t}, \mathbf{x}\big)}{q_\theta\big(v_i \mid \mathbf{y}_{<t}, \mathbf{x}\big)}\right)\right] \tag{10}$$

Here, $\sigma(\cdot)$ denotes the sigmoid function, and $\text{sg}[\cdot]$ the stop-gradient operator. By applying $\text{sg}[\cdot]$, we block gradient flow through its arguments, effectively treating the weight $\alpha_{t,i}$ as a fixed value during the backpropagation of the loss.

As illustrated in Figure 3 (Right), $\alpha_{t,i}$ smoothly varies between 0 and 1 according to the magnitude of $p(v_i \mid \mathbf{y}_{<t}, \mathbf{x})/q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$, naturally reflecting the teacher–student probability discrepancy. A detailed proof that the sigmoid satisfies all four conditions is provided in Appendix B. Furthermore, we implement the stop-gradient operator $\text{sg}[\cdot]$ as a `detach` operation during training; its effects are discussed in detail in Appendix C.

Function	$\alpha_{t,i}(r)$ $(r = p/q_\theta)$	β
Sigmoid	$\frac{1}{1+e^{-\log r}} = \frac{r}{1+r}$	1
Scaled tanh	$\frac{1}{2}\big(1 + \tanh(\log r)\big)$	2
Jeffreys (fixed)	$\frac{1}{2}$	0
Step function	$\mathbf{1}[r > 1]$	$\beta \to \infty$

Table 2: Various weighting functions can be unified under the Generalized ToDi, where each can be expressed in the form $\alpha_{t,i}(r) = \sigma(\beta \log r)$ with an appropriate scaling factor β.

Generalized ToDi. Any function satisfying the four weight conditions introduced above can take many forms. To explore this design space and unify various weighting strategies, we introduce a *scaling hyperparameter* $\beta \in \mathbb{R}$. By incorporating β into the sigmoid input, we can express a variety of weighting functions in a *single unified form*. In this case, the ToDi weight function $\alpha_{t,i}$ is defined as:

$$\alpha_{t,i} = \mathrm{sg}\left[\sigma\left(\beta \cdot \log \frac{p\big(v_i \mid \mathbf{y}_{<t}, \mathbf{x}\big)}{q_\theta\big(v_i \mid \mathbf{y}_{<t}, \mathbf{x}\big)}\right)\right] \quad (11)$$

As summarized in Table 2, by simply varying the value of β, this unified framework can represent a range of weighting functions, such as the standard sigmoid ($\beta = 1$), scaled tanh ($\beta = 2$), Jeffreys divergence (Jeffreys, 1946) ($\beta = 0$), and approximating a step function ($\beta \to \infty$).

5 Experiments

5.1 Experimental Setup

Training Configuration. We follow the experimental setup of Zhang et al. (2024b) to evaluate ToDi. For training, we use the `databricks/dolly-15k` dataset, which comprises 11K training samples, 1K validation samples, and 500 test samples. As student models for the main experiment, we employ GPT2-120M (Radford et al., 2019) and TinyLLaMA-1.1B (Zhang et al., 2024a). We train GPT2-120M via full fine-tuning using GPT2-1.5B as the teacher model, whereas we train TinyLLaMA-1.1B with LoRA (Hu et al., 2022) using LLaMA2-7B (Touvron et al., 2023) as the teacher.

Evaluation Protocol. We conduct performance evaluation following the protocol of Gu et al. (2024), using the ROUGE-L metric (Lin, 2004). We assess instruction-following ability across five datasets: **DollyEval**, **S-NI** (Wang et al., 2022), **UnNI** (Honovich et al., 2023), **SelfInst** (Wang et al., 2023), and **VicunaEval** (Zheng et al., 2023). We repeat each evaluation with five different random seeds, and we report the average scores. Further details of the experimental setup are provided in Appendix D.

Baseline Methods. We use the following methods as baselines to compare the performance of ToDi:

- **SFT**: Fine-tuning the student model directly on the dataset without knowledge distillation.
- **FKL/RKL** (Hinton et al., 2015; Gu et al., 2024): Knowledge distillation using Forward or Reverse KL divergence.
- **JS/TVD** (Wen et al., 2023): Symmetric divergences—Jensen–Shannon and Total Variation—minimizing the distance between the teacher and student distributions.
- **SKL/SRKL** (Ko et al., 2024): Skewed KL and Skewed Reverse KL, which mix teacher and student distributions at ratio λ; SKL uses $\lambda p + (1-\lambda)q_\theta$ while SRKL uses $(1-\lambda)p + \lambda q_\theta$.
- **AKL** (Wu et al., 2025): Adaptive KL that combines FKL and RKL by considering head–tail differences in the distributions.

To evaluate ToDi's performance, we select various divergence-based knowledge distillation methods as baselines and compare their performance based on the choice of divergence.

5.2 Results

Overall Performance We first evaluate the overall instruction-following performance of ToDi against baselines using ROUGE-L. Table 3 presents the performance of the teacher and student models under different teacher–student configurations, compared across various knowledge distillation methods. Our proposed ToDi achieves the highest average score on all five instruction-following tasks for both teacher–student pairs, outperforming all baseline methods, showing that ToDi effectively transfers the knowledge of the teacher to the student. We demonstrate that ToDi consistently outperforms all single-divergence baselines and even surpasses an approach that uses a single, global weight across the entire vocabulary. These results indicate that dynamic, token-level adjustment of

Methods	DollyEval	S-NI	UnNI	SelfInst	VicunaEval	Average
GPT2 1.5B → GPT2 120M						
Teacher	26.66±0.30	27.17±0.33	31.60±0.13	14.42±0.49	16.32±0.41	23.23
SFT	23.09±0.53	16.44±0.39	18.96±0.08	9.72±0.43	14.81±0.34	16.61
FKL	24.06±0.43	18.43±0.22	21.42±0.04	11.13±0.34	15.53±0.45	18.12
RKL	24.22±0.18	18.60±0.10	21.99±0.07	**11.42±0.33**	**15.65±0.51**	18.38
JS	23.77±0.29	17.31±0.17	19.74±0.07	10.08±0.37	15.08±0.32	17.20
TVD	23.90±0.61	17.89±0.24	20.87±0.12	10.73±0.71	15.20±0.30	17.72
SKL	24.05±0.31	17.18±0.31	20.43±0.08	10.54±0.55	14.93±0.29	17.42
SRKL	24.20±0.40	18.02±0.18	21.67±0.09	11.05±0.48	15.07±0.22	18.00
AKL	24.67±0.29	18.29±0.23	21.46±0.12	10.62±0.68	15.28±0.16	18.07
ToDi (Ours)	**24.81±0.62**	**19.42±0.18**	**22.16±0.21**	11.30±0.41	15.61±0.34	**18.66**
LLaMA2 7B → TinyLLaMA 1.1B						
Teacher	28.88±0.23	30.72±0.36	32.02±0.08	19.89±0.58	18.76±0.59	26.05
SFT	23.36±0.26	26.19±0.18	26.69±0.08	15.76±1.04	15.88±0.63	21.58
FKL	25.40±0.50	30.13±0.43	29.47±0.06	**18.22±1.12**	16.77±0.31	24.00
RKL	24.11±0.31	**32.09±0.37**	30.29±0.11	17.97±0.84	16.02±0.73	24.09
JS	24.41±0.34	28.55±0.33	28.69±0.10	17.31±0.32	16.21±0.52	23.03
TVD	24.71±0.74	29.23±0.25	29.12±0.05	16.64±0.83	16.19±0.63	23.18
SKL	25.32±0.54	31.10±0.38	29.89±0.11	17.45±0.69	16.32±0.33	24.01
SRKL	24.93±0.18	30.52±0.31	30.62±0.15	17.17±0.68	16.41±0.36	23.93
AKL	25.50±0.53	30.41±0.28	30.55±0.08	17.52±0.57	16.79±0.34	24.15
ToDi (Ours)	**26.26±0.31**	31.53±0.22	**31.29±0.17**	18.14±0.23	**16.96±0.23**	**24.83**

Table 3: Across all distillation settings, our proposed ToDi consistently outperforms every baseline in ROUGE-L score. The best result is shown in **bold**, and the second best is underlined.

divergence weights—tailored to each token's predicted probability discrepancy—yields significant performance gains.

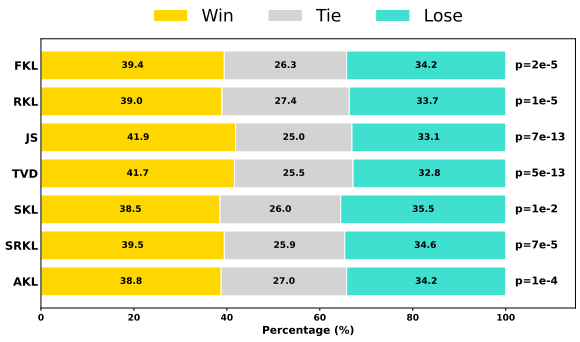


Figure 4: GPT-4 pairwise evaluation of TinyLLaMA models trained with various KD methods on 5,000 UnNI examples. Bars show Win/Tie/Lose proportions; p-values on right.

Preference Evaluation via GPT-4 We further evaluate ToDi through a pairwise comparison experiment using GPT-4. We also evaluate the subjective quality of responses generated by models trained with ToDi using a GPT-4. We randomly select 5,000 samples from the UnNI dataset and compare the responses generated by a TinyLLaMA model trained with ToDi to those produced by models trained with alternative divergence objectives. GPT-4 judged which response was superior. As shown in Figure 4, ToDi consistently achieved higher win rates across all comparisons. In most cases, these improvements were statistically significant ($p < 0.001$), confirming ToDi's superiority over the baselines. For additional details, refer to Appendix E.

Evaluation on Various Model Configurations In addition to GPT2 and LLaMa2 in Table 3, we further evaluate ToDi's performance across diverse teacher–student configurations. We further experiment with OLMo2 (OLMo et al., 2025) (7B -> 1B), Qwen2.5 (Qwen et al., 2025) (1.5B -> 0.5B), and Gemma3 (Team et al., 2025) (4B -> 1B) for the DollyEval benchmark. As shown in Table 4, ToDi consistently outperforms existing baselines under all five configurations. This demonstrates that ToDi can transfer knowledge robustly and effectively across different teacher–student setups.

5.3 Analysis

Training Stability and Convergence We analyze the training dynamics of ToDi to assess its stability and convergence behavior. As shown in Figure 5, ToDi maintains a large performance margin over other methods at every epoch, achieving the highest scores throughout training. In particular, ToDi outperforms all baselines by a wide margin in the first epoch and exhibits a steady upward tra-

Methods	GPT2	LLaMa2	OLMo2	Qwen2.5	Gemma3
Teacher	26.66±0.30	28.88±0.23	30.24±0.48	27.42±0.63	30.60±0.42
SFT	23.09±0.53	23.36±0.26	24.53±0.41	24.89±0.25	24.12±0.37
FKL	24.06±0.43	25.40±0.50	26.88±0.57	26.71±0.56	26.88±0.35
RKL	24.22±0.18	24.11±0.31	25.98±0.46	27.14±0.32	28.69±0.14
JS	23.77±0.29	24.41±0.34	25.39±0.59	26.82±0.12	25.10±0.40
TVD	23.90±0.61	24.71±0.74	25.60±0.34	26.78±0.52	26.06±0.21
SKL	24.05±0.31	25.32±0.54	25.86±0.31	27.04±0.17	26.16±0.35
SRKL	24.20±0.40	24.93±0.18	26.03±0.12	26.74±0.54	25.90±0.59
AKL	24.67±0.29	25.50±0.53	25.97±0.13	26.66±0.22	28.53±0.37
ToDi (Ours)	**24.81±0.62**	**26.26±0.31**	**26.94±0.41**	**27.20±0.34**	**29.03±0.43**

Table 4: ROUGE-L scores on the DollyEval benchmark across diverse distillation settings with varying teacher-student model pairs, including GPT2-1.5B → GPT2-120M, LLaMA2-7B → TinyLLaMA-1.1B, OLMo2-7B → OLMo2-1B, Qwen2.5-1.5B → Qwen2.5-0.5B and Gemma3-4B → Gemma3-1B. The best result is shown in **bold**.

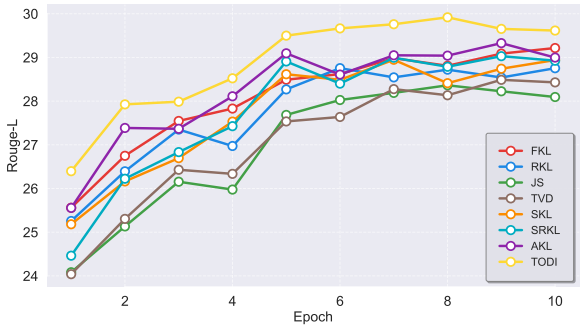


Figure 5: Validation ROUGE-L scores per epoch for TinyLLaMA using various KD methods.

jectory during the middle epochs (2–6 epochs). In the later stages (6–10 epochs), its learning curve remains smooth and converges stably without oscillation. These results indicate that ToDi not only provides a strong training signal as a KD loss function but also ensures reliable convergence.

Computational Efficiency We compare the computational complexity of ToDi with existing methods to assess its efficiency. The efficiency of ToDi is evident not only in its performance but also in its computational complexity. For instance, AKL—which dynamically adjusts the weights of FKL and RKL globally across the entire vocabulary—incurs a time complexity of $O(V \log V)$ due to the required sorting operations. In contrast, ToDi performs computations adaptively on a per-token basis without any sorting during loss computation. As a result, it preserves linear time complexity $O(V)$ with respect to vocabulary size, identical to both FKL and RKL.

Effect of the Generalization Parameter β To analyze the impact of the scaling parameter β, we compare the three settings $\beta \in \{1, 0, -1\}$ in generalized ToDi. $\beta = 1$ corresponds to the default ToDi configuration; $\beta = 0$ fixes $\alpha = 0.5$, resulting in an equal combination of FKL and RKL (i.e., Jeffreys divergence); and $\beta = -1$ reverses the weighting direction, amplifying FKL when $q_\theta > p$ and RKL when $p > q_\theta$. Experimental results with GPT2-120M are shown in Figure 6 (Left). The dynamic weighting scheme ($\beta = 1$) outperforms both the static setting ($\beta = 0$) and the reversed setting ($\beta = -1$), with the reversed setting exhibiting even lower performance than the static scheme, indicating that ToDi's adaptive weight adjustment contributes to performance improvements.

β	Dolly	S-NI	UnNI	Self	Vicuna	Average
0.6	24.44	18.17	22.44	10.88	16.09	18.40
0.8	24.50	19.15	22.04	10.76	15.74	18.44
1	24.81	19.42	22.16	11.30	15.61	18.66
1.2	24.29	18.85	21.86	11.15	15.69	18.37
∞	24.30	18.96	21.89	10.93	15.11	18.24

Table 5: Comparison of ROUGE-L scores of GPT-2 student models under different values of the scaling parameter β.

Sensitivity Analysis on β Table 5 reports ROUGE-L scores as a function of the scaling parameter $\beta \in \{0.6, 0.8, 1.0, 1.2, \infty\}$. The experiments show that $\beta = 1.0$ achieves the highest average score of 18.66. Two key trends are observed:

- **Low-sensitivity regime** ($\beta < 1$): As β decreases, the sigmoid's slope becomes shallower, causing the weight $\alpha_{t,i}$ to converge toward 0.5. This nearly fixed combination of FKL and RKL reduces responsiveness to token-level prediction discrepancies, degrading training effectiveness. Indeed, at $\beta = 0.6$, the average performance drops to 18.40.
- **High-sensitivity regime** ($\beta \to \infty$): As β

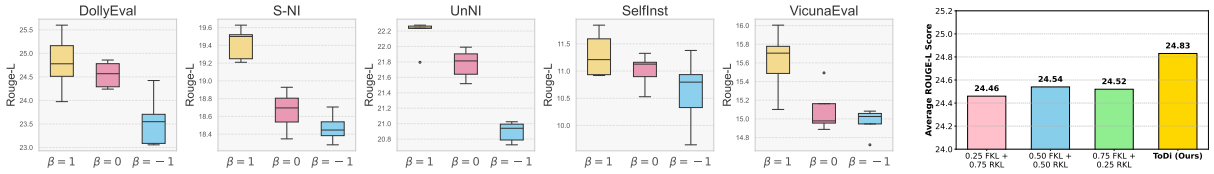


Figure 6: **(Left)** Performance comparison of Generalized ToDi with different scaling parameters $\beta \in \{1, 0, -1\}$ across five evaluation datasets. The dynamic weighting scheme ($\beta = 1$) outperforms the static setting ($\beta = 0$), while the reversed weighting ($\beta = -1$) shows clear performance degradation on all datasets. **(Right)** Average ROUGE-L scores on five instruction-following benchmarks for fixed-ratio FKL–RKL mixtures uniformly applied across the entire vocabulary distribution versus ToDi's token-wise weighting strategy.

grows large, the sigmoid approaches a step function and the weight $\alpha_{t,i}$ becomes discrete:

$$\alpha_{t,i} \xrightarrow{\beta \to \infty} \mathbf{1}\left[p\left(v_i \mid \mathbf{y}_{<t}, \mathbf{x}\right) > q_\theta\left(v_i \mid \mathbf{y}_{<t}, \mathbf{x}\right)\right].$$

This fully separates the application of FKL and RKL, introducing discontinuities in the learning signal near the boundary $p \approx q_\theta$. Such abrupt transitions undermine training stability, and the average performance declines to 18.24.

Token-wise vs. Uniform Divergence Control Rather than applying a fixed FKL–RKL ratio uniformly across all tokens, ToDi dynamically adjusts this balance on a per-token basis. To validate this effect, we conduct comparative experiments on a TinyLLaMA model using the fixed FKL–RKL mixtures schemes. As shown in Figure 6 (Right), ToDi consistently achieves higher ROUGE-L scores than all fixed-ratio schemes. This demonstrates that flexible, token-level ratio adjustment, rather than a uniform application across the vocabulary, is the key to performance improvements.

Methods	GPT2	TinyLLaMA
AKL	0.477	0.599
ToDi	0.482	0.610

Table 6: Pearson similarities for AKL and ToDi using trained GPT-2 and TinyLLaMA models in Section 5, with distributions computed from the `databricks/dolly-15k` training set.

Coarse vs. Fine-Grained Weighting To demonstrate that a student model trained with ToDi more accurately learns the teacher distribution than one trained with AKL, we compare the distributions generated by each student model to the teacher distribution following Huang et al. (2022). Table 6 summarizes our analysis by reporting the Pearson similarity between the teacher and student model distributions. ToDi achieves higher Pearson similarity than AKL, which—despite adaptively combining forward and reverse KL at each time step—applies a uniform mixing ratio across the entire vocabulary. This indicates that ToDi's dynamic, per-token mixing more accurately captures the teacher distribution.

6 Conclusion

We present ToDi, a novel token-wise distillation method that dynamically balances FKL and RKL based on per-token prediction discrepancies. Our gradient analysis shows that FKL corrects underestimation while RKL suppresses overestimation, and ToDi leverages this by using a sigmoid-based weight per token. Experiments on multiple instruction-following benchmarks demonstrate that ToDi consistently outperforms existing baselines, and GPT-4 pairwise preference evaluations confirm its superiority. Finally, we introduce a unified weighting framework and validate its effectiveness via extensive ablations.

Limitations

ToDi precisely captures token-level prediction discrepancies between the teacher and student models, thereby enabling effective distribution alignment. However, ToDi assumes that the teacher and student share an identical vocabulary, which limits its direct applicability when the two models employ different vocabularies. Moreover, ToDi requires access to the full token probability distribution of the teacher model, restricting its use to open-source LLMs that expose per-token logits.

Experiments on extremely large-scale models were not conducted due to computational resource constraints. Nevertheless, ToDi consistently outperforms existing methods across a diverse range of models, including GPT2-120M and TinyLLaMA-1.1B, demonstrating its practicality and efficiency.

Acknowledgments

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT)(RS-2025-24683575). This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) [RS-2021-II211341, Artificial Intelligence Graduate School Program (Chung-Ang University)]. This work was supported by the ICT Credit-Linked Internship Program. We would also like to thank Prof. Changhee Lee (Korea University) for his valuable feedback and discussions, and Yonghyun Jun and Junhyuk Choi (Undergraduate students at Chung-Ang University) for their assistance and contributions to this work.

References

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 *The Twelfth International Conference on Learning Representations*.

Ibtihel Amara, Nazanin Sepahvand, Brett H Meyer, Warren J Gross, and James J Clark. 2022. Bd-kd: balancing the divergences for online knowledge distillation. *arXiv preprint arXiv:2212.12965*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Alan Chan, Hugo Silva, Sungsu Lim, Tadashi Kozuno, A Rupam Mahmood, and Martha White. 2022. Greedification operators for policy optimization: Investigating forward and reverse kl divergences. *Journal of Machine Learning Research*, 23(253):1–79.

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.

Tao Feng, Yicheng Li, Li Chenglin, Hao Chen, Fei Yu, and Yin Zhang. 2024. Teaching small language models reasoning through counterfactual distillation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5831–5842, Miami, Florida, USA. Association for Computational Linguistics.

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

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

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2023. Unnatural instructions: Tuning language models with (almost) no human labor. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14409–14428, Toronto, Canada. Association for Computational Linguistics.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Tao Huang, Shan You, Fei Wang, Chen Qian, and Chang Xu. 2022. Knowledge distillation from a stronger teacher. In *Advances in Neural Information Processing Systems*, volume 35, pages 33716–33727. Curran Associates, Inc.

Harold Jeffreys. 1946. An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 186(1007):453–461.

Jongwoo Ko, Tianyi Chen, Sungnyun Kim, Tianyu Ding, Luming Liang, Ilya Zharkov, and Se-Young Yun. 2025. DistiLLM-2: A contrastive approach boosts the distillation of LLMs. In *Forty-second International Conference on Machine Learning*.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Seyoung Yun. 2024. Distillm: Towards streamlined distillation for large language models. In *The Forty-first International Conference on Machine Learning*. ICML.

Daphne Koller and Nir Friedman. 2009. *Probabilistic graphical models: principles and techniques*. MIT press.

Hyoje Lee, Yeachan Park, Hyun Seo, and Myungjoo Kang. 2023. Self-knowledge distillation via dropout. *Computer Vision and Image Understanding*, 233:103720.

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

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark,

Pradeep Dasigi, Nouha Dziri, and 21 others. 2025. 2 olmo 2 furious. *Preprint*, arXiv:2501.00656.

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, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, and 21 others. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Makoto Shing, Kou Misaki, Han Bao, Sho Yokoi, and Takuya Akiba. 2025. TAID: Temporally adaptive interpolated distillation for efficient knowledge transfer in language models. In *The Thirteenth International Conference on Learning Representations*.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

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*.

Chaoqi Wang, Yibo Jiang, Chenghao Yang, Han Liu, and Yuxin Chen. 2024. Beyond reverse KL: Generalizing direct preference optimization with diverse divergence constraints. In *The Twelfth International Conference on Learning Representations*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, and 16 others. 2022. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. 2023. f-divergence minimization for sequence-level knowledge distillation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10817–10834, Toronto, Canada. Association for Computational Linguistics.

Taiqiang Wu, Chaofan Tao, Jiahao Wang, Runming Yang, Zhe Zhao, and Ngai Wong. 2025. Rethinking Kullback-Leibler divergence in knowledge distillation for large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 5737–5755, Abu Dhabi, UAE. Association for Computational Linguistics.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024a. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*.

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

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

A Gradient Derivations

A.1 Derivation of FKL Gradient

We consider the forward KL divergence term at time step t and vocabulary token v_i, defined as:

$$D_{\text{FKL}}^{(t,i)}(p, q_\theta) = p_i \log \frac{p_i}{q_i} \tag{12}$$

where:

$$p_i := p(v_i \mid \mathbf{y}_{<t}, \mathbf{x}), \quad q_i := q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x}) \tag{13}$$

To compute the gradient with respect to q_i, we apply the product rule:

$$\frac{\partial}{\partial q_i} D_{\text{FKL}}^{(t,i)}(p, q_\theta) = \frac{\partial}{\partial q_i} \left[p_i \log \frac{p_i}{q_i} \right] \tag{14}$$

Since p_i is independent of q_i, we treat it as a constant:

$$= p_i \cdot \frac{\partial}{\partial q_i} (\log p_i - \log q_i) = -p_i \cdot \frac{1}{q_i} \tag{15}$$

Thus, the gradient becomes:

$$\frac{\partial}{\partial q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})} D_{\text{FKL}}^{(t,i)}(p, q_\theta) = -\frac{p(v_i \mid \mathbf{y}_{<t}, \mathbf{x})}{q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})} \tag{16}$$

A.2 Derivation of RKL Gradient

We now derive the gradient for the reverse KL divergence, defined as:

$$D_{\text{RKL}}^{(t,i)}(p, q_\theta) = q_i \log \frac{q_i}{p_i} \tag{17}$$

where the same definitions apply:

$$p_i := p(v_i \mid \mathbf{y}_{<t}, \mathbf{x}), \quad q_i := q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x}) \tag{18}$$

Applying the product rule:

$$\begin{aligned} \frac{\partial}{\partial q_i} D_{\text{RKL}}^{(t,i)}(p, q_\theta) &= \frac{\partial}{\partial q_i} \left[q_i \log \frac{q_i}{p_i} \right] \\ &= \frac{\partial}{\partial q_i} (q_i \log q_i - q_i \log p_i) \end{aligned} \tag{19}$$

Since $\log p_i$ is constant w.r.t. q_i, the derivative simplifies to:

$$\begin{aligned} \frac{\partial}{\partial q_i} D_{\text{RKL}}^{(t,i)}(p, q_\theta) &= (\log q_i + 1) - \log p_i \\ &= \log \frac{q_i}{p_i} + 1 \end{aligned} \tag{20}$$

Hence, the final gradient expression is:

$$\begin{aligned} &\frac{\partial}{\partial q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})} D_{\text{RKL}}^{(t,i)}(p, q_\theta) \\ &= \log \frac{q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})}{p(v_i \mid \mathbf{y}_{<t}, \mathbf{x})} + 1 \end{aligned} \tag{21}$$

B Proof of Sigmoid Weight-Function Properties

For the ToDi weight function

$$\alpha_{t,i} = \sigma\left(\log \frac{p\big(v_i \mid \mathbf{y}_{<t}, \mathbf{x}\big)}{q_\theta\big(v_i \mid \mathbf{y}_{<t}, \mathbf{x}\big)} \right) \tag{22}$$

we prove the following:

- If $p(v_i \mid \mathbf{y}_{<t}, \mathbf{x}) > q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$, then $\log \frac{p(v_i|\mathbf{y}_{<t},\mathbf{x})}{q_\theta(v_i|\mathbf{y}_{<t},\mathbf{x})} > 0 \Rightarrow \alpha_{t,i} > 0.5$, which increases the contribution of FKL.

- If $q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x}) > p(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$, then $\log \frac{p(v_i|\mathbf{y}_{<t},\mathbf{x})}{q_\theta(v_i|\mathbf{y}_{<t},\mathbf{x})} < 0 \Rightarrow \alpha_{t,i} < 0.5$, which increases the contribution of RKL.

- Let $r = p(v_i \mid \mathbf{y}_{<t}, \mathbf{x}) / q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$, so that $\alpha_{t,i} = \sigma(\log r)$. Then

  $$\frac{d\alpha_{t,i}}{dr} = \frac{\sigma(\log r)\big(1 - \sigma(\log r)\big)}{r} > 0$$

  implying that $\alpha_{t,i}$ is monotonically increasing in r.

- Since $\forall z,\ \sigma(z) \in (0, 1)$, it follows that $\alpha_{t,i} \in (0, 1)$.

C Jeffreys-Inspired Weighting with Stop-Gradient

The token-wise weight $\alpha_{t,i}$ in ToDi is inspired by Jeffreys divergence. In this section, we outline this connection and, in particular, show analytically how applying a stop-gradient (`detach`) to $\alpha_{t,i}$ yields gradients that differ from those of standard Jeffreys divergence.

At time step t for token $v_i \in \mathcal{V}$, the Jeffreys divergence can be written using Equation 1 and Equation 2 as:

$$D_{\text{Jeffreys}}^{(t,i)}(p, q_\theta) \;=\; D_{\text{FKL}}^{(t,i)}(p, q_\theta) + D_{\text{RKL}}^{(t,i)}(p, q_\theta) \tag{23}$$

The ToDi weighting function $\alpha_{t,i}$ can then be

derived from Jeffreys divergence as:

$$
\begin{aligned}
& p_i \log \frac{p_i}{q_i} + q_i \log \frac{q_i}{p_i} \\
&= p_i \log \frac{p_i}{q_i} - q_i \log \frac{p_i}{q_i} \\
&= (p_i - q_i) \log \frac{p_i}{q_i} \\
&= \frac{p_i^2 - q_i^2}{p_i + q_i} \log \frac{p_i}{q_i} \\
&= \frac{p_i^2}{p_i + q_i} \log \frac{p_i}{q_i} - \frac{q_i^2}{p_i + q_i} \log \frac{p_i}{q_i} \\
&= \frac{p_i^2}{p_i + q_i} \log \frac{p_i}{q_i} + \frac{q_i^2}{p_i + q_i} \log \frac{q_i}{p_i} \\
&= \frac{p_i}{p_i + q_i} \left(p_i \log \tfrac{p_i}{q_i}\right) + \frac{q_i}{p_i + q_i} \left(q_i \log \tfrac{q_i}{p_i}\right) \\
&= \sigma\left(\log \frac{p_i}{q_i}\right) \left(p_i \log \frac{p_i}{q_i}\right) \\
&\quad + \left(1 - \sigma\left(\log \frac{p_i}{q_i}\right)\right) \left(q_i \log \frac{q_i}{p_i}\right)
\end{aligned}
\tag{24}
$$

where, for brevity, we denote $p_i := p(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$ and $q_i := q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$. In ToDi, $\sigma(\log \frac{p_i}{q_i})$ is `detached` so that no gradient flows through it. As a result, $\alpha_{t,i}$ acts purely as a constant weight, leading to an optimization behavior that diverges from Jeffreys divergence.

To clarify this difference, we compare derivatives with respect to $q_\theta(v_i \mid \mathbf{y}_{<t}, \mathbf{x})$:

- Jeffreys divergence derivative:

$$
\frac{\partial}{\partial q_\theta} \left[p \log \frac{p}{q_\theta} + q_\theta \log \frac{q_\theta}{p} \right] = -\frac{p}{q_\theta} + \log \frac{q_\theta}{p} + 1 \tag{25}
$$

- ToDi derivative ($\alpha_{t,i}$ is `detached`, so treated as constant):

$$
\begin{aligned}
&\frac{\partial}{\partial q_\theta} \left[\alpha_{t,i} \cdot p \log \frac{p}{q_\theta} + (1 - \alpha_{t,i}) \cdot q_\theta \log \frac{q_\theta}{p} \right] \\
&= \alpha_{t,i} \left(-\frac{p}{q_\theta} \right) + (1 - \alpha_{t,i}) \left(\log \frac{q_\theta}{p} + 1 \right)
\end{aligned}
\tag{26}
$$

Using the `detached` weight $\alpha_{t,i}$, ToDi increases the weight on $D_{\text{FKL}}^{(t,i)}(p, q_\theta)$ when $p > q_\theta$, elevating the student probability, and increases the weight on $D_{\text{RKL}}^{(t,i)}(p, q_\theta)$ when $q_\theta > p$, suppressing the student probability. Unlike Jeffreys divergence, which applies divergence uniformly across the vocabulary, ToDi adaptively refines divergence intensity at the token level.

Settings	GPT2	TinyLLaMA	LLaMA2
Epoch	20	10	10
Learning Rate	5e-4	1e-3	1e-3
Batch Size	32	32	32
Fine-Tuning Method	Full	LoRA	LoRA
LoRA Rank	-	256	256
LoRA Alpha	-	8	8
LoRA Dropout	-	0.1	0.1

Table 7: Hyperparameter settings for KD.

D Experimental Details

D.1 Training details

Training was conducted based on the setup of Zhang et al. (2024b). For GPT2-1.5B, we employed the publicly released model from Gu et al. (2024), while GPT2-120M was trained for 20 epochs with a learning rate of 5×10^{-4}. The TinyLLaMA and LLaMA2 models were trained for 10 epochs with a learning rate of 1×10^{-3}. All experiments were carried out on a single RTX A6000 GPU. The training loss was composed by combining the KD loss and the cross-entropy loss in equal proportions (0.5:0.5). Detailed hyperparameter settings for each model are summarized in Table 7.

D.2 Evaluation details

All test sets were processed following Gu et al. (2024). The number of samples in each test set is as follows: DollyEval contains 500 examples; S-NI includes 1,694 examples with response lengths exceeding 11 tokens; UnNI comprises 10,000 examples with response lengths exceeding 11 tokens; SelfInst has 242 examples; and VicunaEval consists of 80 examples. For response generation, we used random seeds {10, 20, 30, 40, 50} and report the average ROUGE-L score across these seeds.

E Details of GPT-4 Evaluation

Pairwise comparison of model responses was performed using the `gpt-4o-2024-11-20` API, with response order randomized in the prompt to mitigate position bias. We followed the LLM-as-a-Judge evaluation protocol of Zheng et al. (2023), employing the pairwise comparison prompt shown in Figure 7.

[System]
Please act as an impartial judge and evaluate the quality of the responses provided by two AI assistants to the user question displayed below. You should choose the assistant that follows the user's instructions and answers the user's question better. Your evaluation should consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and level of detail of their responses. Begin your evaluation by comparing the two responses and provide a short explanation. Avoid any positional biases and ensure that the order in which the responses were presented does not influence your decision. Do not allow the length of the responses to influence your evaluation. Do not favor certain names of the assistants. Be as objective as possible. After providing your explanation, output your final verdict by strictly following this format: "[[A]]" if assistant A is better, "[[B]]" if assistant B is better, and "[[C]]" for a tie.

[Question]
{}

[The Start of Assistant A's Answer]
{}
[The End of Assistant A's Answer]

[The Start of Assistant B's Answer]
{}
[The End of Assistant B's Answer]

Figure 7: Prompt for GPT-4o based Evaluation.