PLaD: Preference-based Large Language Model Distillation with Pseudo-Preference Pairs

Rongzhi Zhang[1*]   **Jiaming Shen**[2†]   **Tianqi Liu**[2†]
Haorui Wang[1]   **Zhen Qin**[2†]   **Feng Han**[2†]   **Jialu Liu**[2]
Simon Baumgartner[2†]   **Michael Bendersky**[2†]   **Chao Zhang**[1]

[1]Georgia Institute of Technology, GA, USA
[2]Google, NY, USA
rongzhi.zhang@gatech.edu, jmshen@google.com

Abstract

Large Language Models (LLMs) have exhibited impressive capabilities in various tasks, yet their vast parameter sizes restrict their applicability in resource-constrained settings. Knowledge distillation (KD) offers a viable solution by transferring expertise from large teacher models to compact student models. However, traditional KD techniques face specific challenges when applied to LLMs, including restricted access to LLM outputs, significant teacher-student capacity gaps, and the inherited mis-calibration issue. In this work, we present PLaD, a novel preference-based LLM distillation framework. PLaD exploits the teacher-student capacity discrepancy to generate *pseudo-preference pairs* where teacher outputs are preferred over student outputs. Then, PLaD leverages a ranking loss to re-calibrate student's estimation of sequence likelihood, which steers the student's focus towards understanding the relative quality of outputs instead of simply imitating the teacher. PLaD bypasses the need for access to teacher LLM's internal states, tackles the student's expressivity limitations, and mitigates the student mis-calibration issue. Through extensive experiments on two sequence generation tasks and with various LLMs, we demonstrate the effectiveness of our PLaD framework.

1 Introduction

Large language models (LLMs) have shown remarkable abilities across a wide range of tasks (OpenAI, 2022; Anil et al., 2023). However, their huge parameter sizes and computational requirements pose significant challenges

for practical deployment, especially in environments with limited resources. Knowledge distillation (KD) has emerged as a technique for addressing these challenges by transferring insights from a large, sophisticated teacher model to a compact student model with reduced memory footprints and inference costs. The seminal work (Hinton et al., 2015) proposes to train a student model to match the output class distribution of the teacher model. Kim and Rush (2016) further extends this idea to the sequence level and teaches the student to directly produce teachers' decoded sequences. Another line of work (Jiao et al., 2019; Wang et al., 2020) seeks to align the student model's intermediate-layer representations with the teacher's. All these approaches employ a teacher-forcing strategy, training the student to fully match the outputs or representations of the teacher model.

Applying conventional KD methods to LLMs presents several significant challenges. First, those LLM teachers are typically only available through API calls. The absence of direct access to the full output logits or internal states of LLM teachers hinders the implementation of traditional distillation techniques. Second, the capacity gap between the student model and LLM teachers becomes significantly larger compared to the previous instances when a relatively smaller teacher model was employed. This disparity exacerbates the student model's limited ability to fully match the teacher LLM's output distribution. Third, as LLMs increase in size, they often encounter a mis-calibration issue (Zhao et al., 2023) where sequences that are highly likely according to the model don't necessarily exhibit high quality for target tasks. Consequently, when a student model is trained to mimic these outputs from the teacher LLM,

15623

it inherits this mis-calibration, leading to sub-optimal performance. Although some recent studies have enhanced the standard teacher-forcing KD paradigm with improved loss functions (Zhang et al., 2023b; Wen et al., 2023; Gu et al., 2023) or learning strategies (Agarwal et al., 2023), these advancements have not yet fully addressed the above challenges, leaving efficient and effective LLM distillation as an open research question.

In this work, we present **P**reference-based Large **La**nguage Model **D**istillation (PLaD), a novel framework for distilling LLM knowledge with preference data. PLaD is developed based on the following observation: sequences decoded by the teacher model typically surpass the output sequences of the student in quality. By sampling outputs from both the teacher and student, PLaD generates *pseudo-preference pairs* and calculates a ranking loss that re-calibrates sequence likelihood on the student side. This innovation acknowledges the complex teacher-student interaction dynamics within the realm of LLMs and thus shifts the student's learning focus towards understanding the relative quality of different outputs. Without strictly adhering to teacher forcing, we address the student's inherent limitations in expressivity.

Moreover, the introduction of calibration loss directly ties the quality of generation to its likelihood, allowing for a targeted optimization of output quality through calibration. This strategy also skillfully bypasses the requirement for teacher model internal access and presents an annotation-free method to construct preference pairs based on the inherent capacity gap between teacher and student models. PLaD is also flexible enough to be applied in scenarios where additional reward models or ranking metrics are available. This versatility makes it a powerful framework for LLM distillation.

We evaluate PLaD on Anthropic helpful dialogue generation (Bai et al., 2022) and Reddit TL;DR summarization (Stiennon et al., 2020). with two different model families LLaMA-2 (Touvron et al., 2023) and GPT-Neo (Black et al., 2021). The student model learned by our PLaD framework can outperform the one learned with other state-of-the-art KD methods

with respect to the win rate of model generations compared to the target sequences. We also show that PLaD are universally applicable across model families: from PaLM2-L (Anil et al., 2023) to T5-Large (Raffel et al., 2020).

Contributions. The major contributions of this work are summarized as follows: (1) We propose PLaD, a novel framework that distills LLMs with preference data; (2) We present a metric-free approach to construct pseudo-preference pairs without human annotations; (3) We facilitate the LLM distillation with an explicit calibration objective and improve the student model's generation capability; (4) We conduct comprehensive experiments on multiple tasks with different-sized teacher models to demonstrate the effectiveness of PLaD.

2 Related Work

Sequence Knowledge Distillation. Knowledge distillation (KD) is first proposed in (Buciluǎ et al., 2006) to compress the large models to smaller, faster models without a significant performance drop. Hinton et al. (2015) generalizes this technique by introducing a temperature parameter to smooth the teacher model prediction. SeqKD (Kim and Rush, 2016), initially targeting the neural machine translation task, extends the scope of KD from multi-class classification to sequence generation and learns a distill student model to generate a sequence holistically. Further developments have seen the incorporation of contrastive learning (Tian et al., 2019) and patient distillation techniques (Sun et al., 2019), where the student learns from multiple layers of the teacher model. Transformer-specific distillation methods have also been proposed (Sanh et al., 2019; Jiao et al., 2019), focusing on attention and hidden state alignment for efficient knowledge transfer. These advancements underscore the ongoing efforts to refine SeqKD for natural language processing tasks, balancing model size with linguistic performance.

Learning from Preference Data. The pivotal Reinforcement Learning from Human Feedback (RLHF) framework (Christiano et al., 2017) first uses preference data to fit a reward model and then fine-tunes the LM to maximize the given

reward using reinforcement learning algorithms. In practice, however, using RL to directly fine-tune LMs are challenging, incurring significant computational costs and requiring extensive hyperparameter tuning. To mitigate this issue, DPO (Rafailov et al., 2023) proposes to directly train the policy LM using the pairwise logistic loss without explicitly fitting a reward model. SLiC (Zhao et al., 2023) and RRHF (Yuan et al., 2023) adopt the pairwise hinge loss to train the policy LM. RSO (Liu et al., 2023) further introduces the statistical rejection sampling method to improve DPO by addressing the distribution drift issue. More recently, LiPO (Liu et al., 2024) leverages a ranked list of responses (either rated by humans for ranked by models (Qin et al., 2023)) for LLM alignment. All these studies require either human annotations or external reward models to obtain the preference data and focus on aligning a single model with human preference. Additionally, some works have attempted to explain the distillation mechanisms from multiple perspectives(Lopez-Paz et al., 2015; Lopes et al., 2017; Zhang et al., 2020; Menon et al., 2021; Zhang et al., 2022), but none have yet bridged the preference learning and knowledge distillation, while it is a natural approach given the capacity gap between teacher and student models. In this work, we focus on distilling a teacher model into a student model with self-supervised preference pairs.

LLM Distillation. Recent efforts in the distillation of Large Language Models (LLMs) have introduced innovative approaches to refine the knowledge transfer process. (Liang et al., 2023) employ a task-aware, layer-wise distillation method which effectively tackles the challenges of tasking student models mimicking the hidden representation of a much larger teacher, but it requires access to the teacher model's intermediate layers, whereas our research focuses on the scenarios where only the teacher model's final sequence-level output is available, which is common in the context of LLMs. (Zhang et al., 2023a) seek to bridge the capacity gap between student models and teacher models by deploying the mixture of experts (MoE) architecture on the student side, thereby increasing its capacity. We alternatively address the capacity gap by

shifting from traditional teacher-forcing distillation to a preference-based distillation, leveraging synthetic preference data and avoid memory footprint overhead increased by complex architectures of the student model.

Another line of works leverages additional knowledge to improve LLM distillation. For instance, "Distillation step-by-step" by (Hsieh et al., 2023) enriches the student model training by integrating LLM output rationales, aiming for a deeper understanding and replication of the teacher model's reasoning pathways. Similarly, Fu et al. (2023) advance the methodology by focusing on LLM output Chains of Thought (CoT), engaging in per token distribution matching to capture nuanced decision-making processes. Shridhar et al. (2023) take a specialized approach towards reasoning tasks, proposing a dual-student model framework where one student model is dedicated to problem decomposition and the other to solving the identified subproblems, facilitating a more segmented yet comprehensive distillation strategy. Additionally, Zhang et al. (2023b) also introduce a non-teacher forcing distillation approach, where the teacher output is perturbed to get a proxy teacher with a distribution closer to the ground truth. Despite these advancements, none of these works incorporate preference data into the distillation process, highlighting a key contribution of this study.

3 Preliminaries

3.1 Auto-Regressive Text Generation

We denote the input and output sequence as x and y, respectively. Let V denote the vocabulary comprising of M tokens, $y_{<n+1} = (y_1, y_2, \ldots, y_n)$ denote the generated output sequence up to the n-th token, and L_y denote the length of sequence y. An auto-regressive policy $p(.|y_{<n}, x)$ outputs a next-token probability distribution over all tokens in V, conditioned on the input x and output sequence $y_{<n}$. The probability $p(y_n|x)$ of predicting the n-th token in sequence y is determined by a softmax function with temperature γ as follows:

$$p(y_n|x) = \frac{\exp(z_n/\gamma)}{\sum_{i=1}^{M} \exp(z_i/\gamma)}, \tag{1}$$

where z_i is the logit score for the i-th token in V. Higher values of γ introduce more randomness, while a lower value makes the output more deterministic by favoring the most probable words. In conditional language generation tasks, the model produces a response y conditioning on a prompt x sampled from the distribution p_x. The output sequence y is sampled from the generative model in an auto-regressive manner as described above.

3.2 Sequence Knowledge Distillation

We approach knowledge distillation (KD) as an optimization problem to minimize the difference between a fixed teacher model output distribution $p(y|x)$ and a student model output distribution $q_\theta(y|x)$, parameterized by θ. The standard KD method for generative models aims to minimize the forward KL divergence:

$$KL[p||q] = \mathbb{E}_{x \sim p_x, y \sim p'} \log \frac{p(y|x)}{q_\theta(y|x)}, \qquad (2)$$

where p' represents either the real data distribution for word-level KD or the teacher distribution p for sequence-level KD. However, $KL[p||q]$ tends to overestimate the low-value regions of p in language generation tasks when q_θ lacks the expressiveness to cover all the modes of p'. This issue is particularly pertinent for LLMs that perform a variety of tasks in a generative manner, since the low-capacity student models are unable to imitate the complex language generation distribution of their teacher models or that of humans perfectly.

4 PLaD: Preference-based Large Language Model Distillation

4.1 Framework Overview

Our proposed PLaD framework starts from the supervised fine-tuning (SFT) phase for both the teacher and the student models. First, the teacher model undergoes SFT to optimize its parameters for the target tasks. The student model is similarly fine-tuned to prepare for the subsequent distillation phase. Then, we construct pseudo-preference pairs to calibrate the likelihood of the student generation. Specifically, we conduct inference on a distillation set, which

consists of task-specific inputs without corresponding target outputs. The sampled generations from both the teacher and student models are used to form the pseudo-preference data, where we assume the teacher output is preferred over student output due to their capacity difference. Finally, with the pseudo-preference data, we implement a calibration loss to optimize the generation quality in the distillation phase explicitly. The final distilled student model is evaluated by win rate and ROUGE scores to compare with its SFT predecessor and other baselines. We outline the proposed PLaD framework in Figure 1.

4.2 Pseudo Preference Pairs Generation

Our PLaD framework pivots around the concept of pseudo-preference data, offering a practical and efficient alternative to human annotated preference pairs. Traditional approaches to preference learning, such as Deep Preference Optimization (DPO) (Rafailov et al., 2023) and Reinforcement Learning from Human Feedback (RLHF) (Christiano et al., 2017), rely on preference data obtained through costly human annotations or inferences from state-of-the-art models. These methods, while effective, are prohibitively expensive and time-consuming for large-scale applications.

To mitigate these issues, we capitalize on the reliable assumption that the teacher model's generative quality supersedes that of the student due to its greater capacity. Consequently, we can generate pseudo-preference pairs by sampling outputs from both models on the distillation set and assuming the teacher output is preferred over the student output. Formally, the generation process for a given input x from the distillation set can be expressed as:

$$(\hat{y}_+, \hat{y}_-) := (\hat{y}^T, \hat{y}^S) = (p(y|x), q_\theta(y|x)) \qquad (3)$$

where $\hat{y}^T$ and $\hat{y}^S$ represent the generations from the teacher and student models, respectively. We then construct the preference pairs $(\hat{y}_+, \hat{y}_-)$, with $\hat{y}_+ := \hat{y}^T$ signifying the teacher's higher-quality output and $\hat{y}_- := \hat{y}^S$ indicating the student's output. These pseudo-preference pairs provide a cost-effective and scalable alternative to human-annotated or AI-annotated data, en-
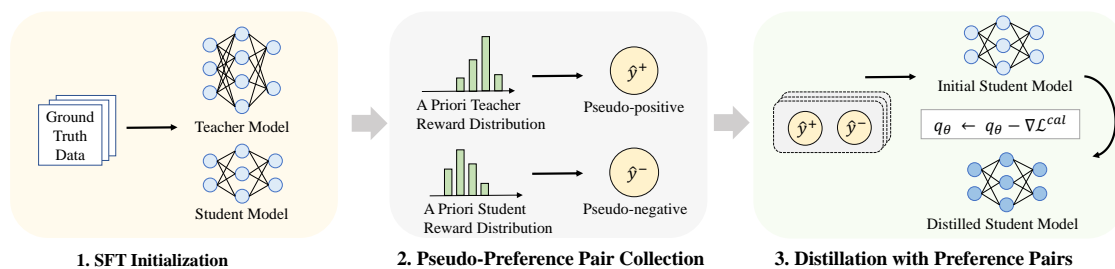
Figure 1: Our PLaD framework starts with separate SFT processes for both the teacher and student models. The best checkpoint of the teacher model is selected based on the win rate over targets, while the student model went through the whole SFT for initialization. In the next stage, we generate pseudo-preference data by sampling generation pairs from the teacher and the student. The student model then undergoes preference distillation using this pseudo-preference data to produce a distilled student model.

abling the student model to redirect the teacher to the relative importance of learning.

4.3 Distillation with Preference Pairs

The distillation with preference pairs is formalized through a calibration loss, designed to bridge the student's generative quality to its likelihood. We employ two types of losses: the ranking calibration loss, $L_{\text{rank}}^{\text{cal}}$, and the margin calibration loss, $L_{\text{margin}}^{\text{cal}}$.

The ranking calibration loss is defined as:

$$L_{\text{rank}}^{\text{cal}} = \max(0, \beta - \log P_\theta(\hat{y}_+|x) + \log P_\theta(\hat{y}_-|x)), \tag{4}$$

where β is a margin hyper-parameter, and $\hat{y}_+$ ($\hat{y}_-$) represents the teacher (student) outputs from the pseudo preference pairs, respectively. This loss encourages the student model to increase the probability of the preferred output while decreasing the likelihood of the less preferred one.

The margin calibration loss is introduced to refine the student's output by considering a scoring function s, which provides an additional quality measure for the generated sequences:

$$\begin{aligned} L_{\text{margin}}^{\text{cal}} = \max(0, &\beta(s(y; \hat{y}_+; x) - s(y; \hat{y}_-; x)) \\ & - \log P_\theta(\hat{y}_+|x) + \log P_\theta(\hat{y}_-|x)) \end{aligned} \tag{5}$$

In this equation, $s(y; \hat{y}_+; x)$ and $s(y; \hat{y}_-; y; x)$ represent the scores of the preferred and less-preferred outputs, respectively. This loss function penalizes the student model when the score of the less-preferred output is too close to that of the preferred one, promoting a distinction between high and low-quality generations. In prac-

---

Algorithm 1 Teacher-Student Knowledge Distillation with Calibration Loss

1: **Require:** Teacher model p, student model q_θ, SFT dataset $\mathcal{D}_0$ with labeled target sequences, and distillation set $\mathcal{D}$.
2: // Step 1: *Initialization*
3: Learn SFT teacher p and initial student q_θ on $\mathcal{D}_0$.
4: // Step 2: *Pseudo-Preference Pair Construction*
5: **for** each input sequence $x \in D$ **do**
6:     Sample teacher output $y_T = p(y|x)$.
7:     Sample student output $y_S = q_\theta(y|x)$.
8:     Make pseudo-preference pairs $(\hat{y}_+, \hat{y}_-) :=$ (y_T, y_S).
9: **end for**
10: // Step 3: *Student Distillation with Preference Pairs*
11: **for** each batch in $\mathcal{D}$ **do**
12:     Compute loss $\mathcal{L}^{cal}$ via Eq. 4 or 5.
13:     Update student model $q_\theta \leftarrow q_\theta - \nabla \mathcal{L}^{cal}$.
14: **end for**
15: **Output:** Distilled student model q_θ.

---

tice, we choose the learning objectives based on the performance on the validation set.

By leveraging the calibration loss in conjunction with pseudo-preference pairs, our method enables an effective distillation process, fostering a student model that not only performs well on text generation tasks but also exhibits calibrated confidence in its outputs. We summarize the framework in Algorithm.1.

5 Experiments

5.1 Experiment Setup

Datasets. We conduct experiments on the following datasets: (1) **TL;DR** (Stiennon et al., 2020) which comprises around 140k Reddit posts along with their TL;DR summarizations, providing a rich source for training and eval-

uating text summarization models, and (2) **Anthropic-HH** (Bai et al., 2022) which initially designed for training preference models in a dialog system with Reinforcement Learning from Human Feedback (RLHF). We use its helpful slice for experiments. More detailed dataset statistics are listed in Appendix A.1.

Models. We evaluate two model families in our main experiments: (1) **LLaMA-2 Models** (Touvron et al., 2023) include LLaMA-2-13B as the teacher model and LLaMA-2-7B as the student model, and (2) **GPT-Neo Models** (Black et al., 2021) include GPT-Neo-2.7B as the teacher model and GPT-Neo-1.3B as the student model. Besides, we also extend PLaD to PaLM-2 (Anil et al., 2023) and T5 models (Raffel et al., 2020) to show its broad applicability.

Baseline Methods. We compare PLaD to both classic KD techniques and LLM KD techniques, including (1) **Standard KD** (Hinton et al., 2015): The foundational knowledge distillation technique that trains a student model to replicate the teacher model's output distributions; (2) **SeqKD** (Kim and Rush, 2016): An extension of standard KD to sequence generation that distills the student model directly on the teacher's generations; (3) **f-distill** (Wen et al., 2023): A framework that addresses KL divergence's the mode averaging and collapsing problems by minimizing a symmetric f-divergence; and (4) **MiniLLM** (Gu et al., 2023): A framework that distills LLMs into their smaller counterparts using reverse KL divergence.

Evaluation Schemes. We calculate the win rate as the major metric for model evaluation. Win rate is defined as the ratio of preferred generation compared to the target text. Specifically, we deploy a task-specific reward model and use the human-written reference sequence as the target. Both the reward model and the reference sequence are pre-provided in the open-source community and the datasets. Besides, we also provide ROUGE scores for reference. More details are provided in Appendix A.2.

Implementation Details. We split each task's original training set into two equal shards for training and distillation. Given the training set $\mathcal{D}_0$, we do 1-epoch fully supervised training to get the teacher model and initialize the student model. We run inference and construct teacher-student pseudo-preference pairs for the distillation set $\mathcal{D}$. The original test set is kept for evaluation. For model training, the learning rate is $1e^{-4}$ with a linear scheduler, and the per-device batch size is 8. We also use LoRA (Hu et al., 2021) for all experiments for training efficiency. Specifically, we set LoRA rank as 8, LoRA dropout as 0.1, and LoRA alpha as 32. We list more details in Appendix A.6.

5.2 Main Results

Table 1 presents our main experiment results. PLaD exhibits a notable capacity for closing the teacher-student performance gap. Compared to the initial student and the SFT baseline, the student model learned by PLaD not only significantly improves the win rate but also enhances the ROUGE scores. Impressively, for the Anthropic-HH task, the student model learned by PLaD even surpasses the teacher model in terms of the win rate (student's 27.74% win rate against the teacher's 26.96% win rate).

In comparison to KD baselines, our approach consistently delivers superior performance across different settings. This is evident from our method exhibiting the highest win rate among all the student models distilled by baseline methods. We also count the generation length by the number of words to verify if the quality improvement comes from the output verbosity. We notice that the word count across different methods is relatively stable. We thus further conduct an experiment in Section 5.4 to justify the performance on different generation length ranges.

Moreover, the Teacher-Student Win Rate (T-S WR) emerges as a critical determinant in the final performance. For the stronger teachers like the LLaMA teacher and the GPT-Neo teacher on the TL;DR task, the distilled student improves by relatively larger margins. Conversely, when learning from a mediocre teacher model, as in the case of the Anthropic-HH task with a lower T-S WR, the distilled student models show only marginal advancements. This highlights the importance of the teacher model's quality in the distillation process and its impact on the student model's ultimate efficacy.

15628

Model	Method	TL-DR	T-S WR = 64.86				Anthropic-HH	T-S WR = 53.97			
		Word Count	R-1	R-2	R-L	RM WR (%)	Word Count	R-1	R-2	R-L	RM WR (%)
LLaMA-2	Teacher	25.51	33.14	11.40	25.31	45.28	67.78	25.15	6.50	16.32	26.95
	Initial Student	27.24	26.68	8.98	20.60	33.42	67.80	24.96	6.48	16.26	25.63
	SFT	25.77	30.06	9.95	22.97	35.13	67.64	25.33	6.66	16.51	25.52
	Standard KD	26.32	31.45	10.04	23.78	37.70	67.60	25.06	6.57	16.31	25.83
	SeqKD	25.67	31.67	10.42	24.20	37.92	67.73	25.02	6.51	16.28	26.24
	f-distill	25.98	32.09	10.65	24.42	38.35	68.24	25.23	6.72	16.40	25.98
	MiniLLM	25.80	31.32	10.48	24.25	38.07	68.05	25.42	6.76	16.53	26.61
	Ours	26.02	31.46	10.79	24.91	40.46	67.93	25.54	6.82	16.70	27.74

Model	Method	TL-DR	T-S WR = 60.80				Anthropic-HH	T-S WR = 55.23			
		Word Count	R1	R2	RL	RM WR (%)	Word Count	R1	R2	RL	RM WR (%)
GPT-Neo	Teacher	25.81	29.64	8.82	22.22	15.35	72.55	21.47	4.12	13.66	15.52
	Initial Student	26.35	29.04	8.34	21.66	11.51	73.53	19.77	3.36	12.67	8.73
	SFT Student	25.88	29.08	8.44	21.76	11.40	73.49	18.51	2.76	11.93	8.79
	Standard KD	26.50	29.15	8.51	21.84	12.47	73.09	20.35	3.56	12.78	8.94
	SeqKD	26.14	29.23	8.57	21.82	13.18	73.11	20.74	3.75	13.05	9.25
	f-distill	25.73	29.34	8.62	21.91	13.55	72.68	20.67	3.84	13.74	9.66
	MiniLLM	25.86	29.37	8.62	22.03	13.73	72.38	20.90	3.80	13.56	9.81
	Ours	25.77	29.50	8.69	22.08	14.76	72.76	21.12	3.82	12.83	10.38

Table 1: Main results with the LLaMA-2 and GPT-Neo models. For the LLaMA-2 group, the teacher model is LLaMA-2-13B, and the student model is LLaMA-2-7B. For the GPT-Neo group, the teacher model is GPT-Neo-2.7B, and the student model is GPT-Neo-1.3B. WR stands for win rate, and RM refers to a task-specific reward model. The teacher-student win rate (T-S WR) on the distillation set evaluated by a reward model is provided for reference.

5.3 Impact of Real Preference Pairs

In this set of experiments, we investigate the effect of using real preference pairs compared to pseudo-preference pairs. We use both the LLaMA-2 model and the GPT-Neo model on TL;DR and Anthropic-HH datasets, each setting comes with varying ratios of real to pseudo preference pairs. Here we use a reward model to evaluate all the pairs and adjust the ranking within each pair based on their reward values.

In Figure 2, we observe a slight improvement in the win rate against using pseudo pairs as the ratio of real preference pairs increases. This trend is evident across both models and settings, albeit to varying degrees. The win rate improvement is more pronounced in the LLaMA-2 models on the TL;DR dataset, indicating a model-specific benefit from real preference data.

Considering the original improvements from using pseudo-preference pairs, the gain of replacing the pseudo-preference pairs with real ones is marginal. While the use of real preference pairs does yield improvements in model performance, it also incurs additional human annotation costs. This presents a trade-off scenario where the gains from real preference data are weighed against the resource expenditure associated with their use.

In conclusion, the experiment outcomes support the reliability of employing pseudo-preference pairs in LLM distillation, supplemented by the proposed calibration objective. Despite the slight edge provided by real preference pairs, the cost-effective and time-efficient nature of pseudo pairs makes them a viable alternative. This study suggests that pseudo-preference pairs, when properly constructed, can serve as a practical proxy to real preference data without significantly compromising the learning efficacy of language models.

5.4 Performance in Length Ranges

In this experiment, we examine the correlation between generation length and win rate. This experiment is conducted on the TL-DR task using LLAMA-2 models. Figure. 3 indicates all methods maintain a relatively stable win rate improvement across varying generation lengths. It also illustrates that our method consistently leads to better performance across all generation lengths of student models, with a notable peak within the (20, 30] interval, achieving the highest win rate improvement to the initial student and the highest margin over the baselines. Considering the average generation length is around 26 on this task,
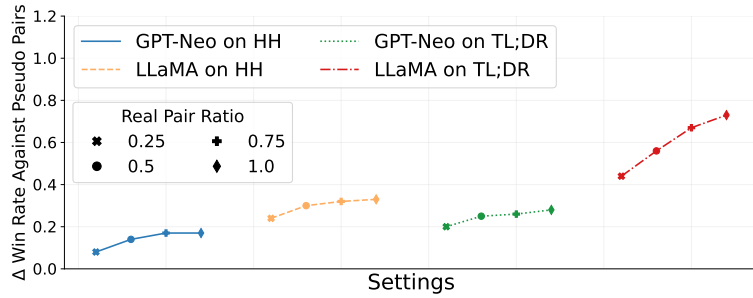
Figure 2: Comparison between using real preference pairs compared to using pseudo preference pairs.
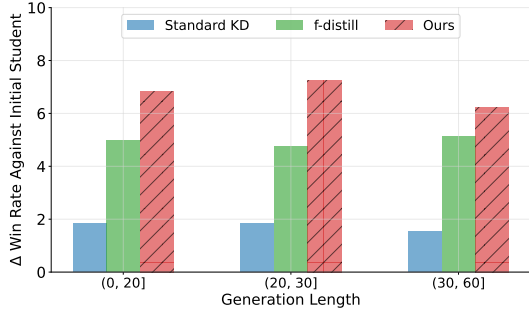

Figure 3: The △ win rate against the initial student v.s. the length of student model generation. Experiments are conducted on TL-DR with LLaMA-2.

this suggests that our approach is particularly effective in the most commonly encountered scenario within the TL-DR task, and the enhancement at this central interval is especially valuable given its prevalence in the dataset.

5.5 Scaling with Distillation Data

We present a scaling analysis in Figure 4 to demonstrate how our distillation method's effectiveness varies with changes in the amount of distillation data. We keep the same setting as the LLaMA-TL;DR group in the main experiments, and perform distillation with different distillation data usage.

The figure illustrates that as the percentage of distillation data increases from 5% to 100%, all methods show an upward trend in win rate over the target, indicating improved model performance with access to more distillation data. Notably, PLaD demonstrates a steep improvement curve, outperforming the other methods, particularly at higher data usage levels. This suggests that PLaD is highly efficient in refining the student model.
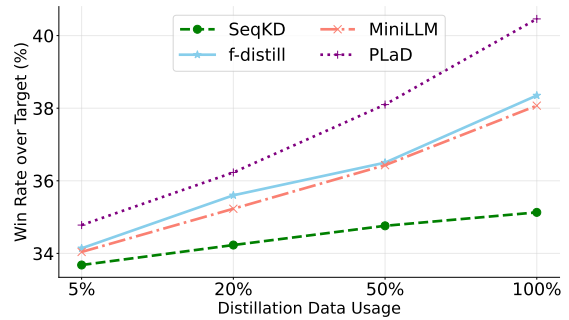

Figure 4: Scaling properties of the distillation data.

Task	Teacher	Initial	Distill
TL-DR	T5-xxl	36.67	44.46
Anthropic-HH	PaLM-2-S	15.32	17.88

Table 2: Win rates of using PALM-2 and T5 models. The teacher model is T5-XXL and PaLM-2-S, respectively, and the student model is T5-large.

5.6 Results on More LLMs

To demonstrate the generalizability of our framework, we extend our investigations to more LLMs including PaLM 2 (Anil et al., 2023) and T5 (Raffel et al., 2019). Specifically, we employ the T5-XXL as the teacher model for the TL;DR task and the PaLM-2-S as the teacher model for the Anthropic-HH task. They are distilled into a T5-Large student model in both tasks. The results are presented in Table 2, and the same conclusion holds that the student distilled with PLaD gets a significantly higher win rate compared to the initial student. Our results, spanning LLaMA-2, GPT, T5, and PaLM models, underscore the framework's adaptability across a range of LLM families.

5.7 Efficacy of General-Purpose Models in Win Rate Calculation

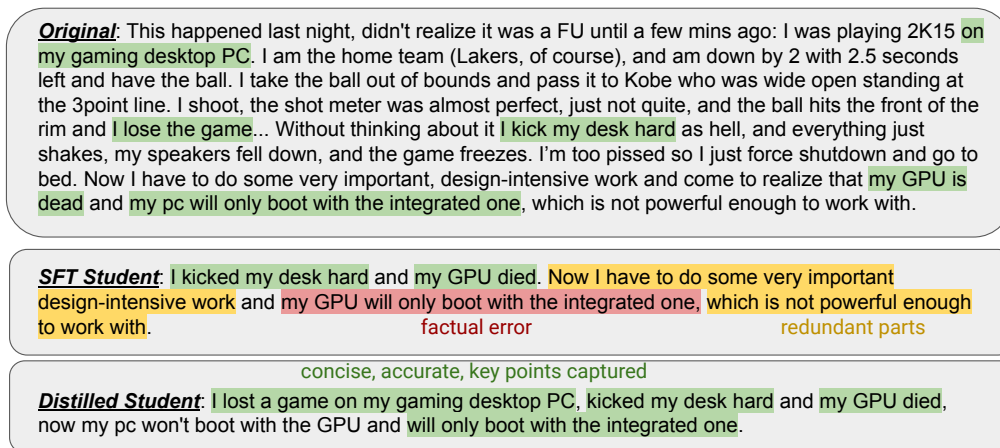We explore the efficacy of using general-purpose models, specifically GPT-3.5-turbo,

Original: This happened last night, didn't realize it was a FU until a few mins ago: I was playing 2K15 on my gaming desktop PC. I am the home team (Lakers, of course), and am down by 2 with 2.5 seconds left and have the ball. I take the ball out of bounds and pass it to Kobe who was wide open standing at the 3point line. I shoot, the shot meter was almost perfect, just not quite, and the ball hits the front of the rim and I lose the game... Without thinking about it I kick my desk hard as hell, and everything just shakes, my speakers fell down, and the game freezes. I'm too pissed so I just force shutdown and go to bed. Now I have to do some very important, design-intensive work and come to realize that my GPU is dead and my pc will only boot with the integrated one, which is not powerful enough to work with.

SFT Student: I kicked my desk hard and my GPU died. Now I have to do some very important design-intensive work and my GPU will only boot with the integrated one, which is not powerful enough to work with.   factual error   redundant parts

concise, accurate, key points captured

Distilled Student: I lost a game on my gaming desktop PC, kicked my desk hard and my GPU died, now my pc won't boot with the GPU and will only boot with the integrated one.

Figure 5: The case study on the TL;DR dataset.

TL;DR	T-S WR	Initial	Baseline†	PLaD
DeBERTa RM	60.80	33.42	38.35	40.46
GPT-3.5-turbo	63.81	31.69	37.42	38.17
Anthropic-HH	**T-S WR**	**Initial**	**Baseline***	**PLaD**
DeBERTa RM	55.23	25.63	26.24	27.74
GPT-3.5-turbo	58.95	24.43	25.06	26.22

Table 3: Comparison of win rates calculated by the task-specific reward model DeBERTa and the general-purpose model GPT-3.5-turbo. The teacher model is LLaMA-2-13B and the student model is LLaMA-2-7B. We present the best baseline methods where † refers to f-distill and * refers to MiniLLM.

for calculating win rates, to provide a parallel evaluation with our deployed task-specific reward models. Table 3 shows that the performance advantage in terms of win rate still holds when evaluated by the general-purpose model. Notably, the T-S WR increases when evaluated by GPT-3.5-turbo, albeit accompanied by a marginal decline in the win rate of model-generated text over the target text. It indicates that compared to the preference made by the reward model, GPT-3.5-turbo favors teacher generation in the teacher-student comparison, while simultaneously showing a slight preference for the target text over model-generated content. Nevertheless, the results affirm that PLaD secures an enhancement in win rate compared to the initial student and the most competitive baseline methods.

5.8 Case Study

In Figure 5, we present a comparative case study from TL;DR to illustrate the efficacy of PLaD. We examine the quality of summaries generated by both an SFT (Self-Training) student model and our distilled student model against the original text. Key points from the original narrative are highlighted, providing a benchmark for evaluating the student models' performance. We detail the discussion in Appendix A.4.

6 Conclusions and Future Work

In this work, we introduce PLaD, a novel preference-based LLM distillation framework that leverages pseudo preference pairs to efficiently transfer knowledge from a large LLM teacher to a compact student model. By focusing on the relative ranking of outputs, we allow the student model to learn in a way that is both resource-efficient and aligned with the qualitative nuances of language generation tasks. Our experiments demonstrate that PLaD learns a student model that retains a high level of performance, measured by traditional metrics like ROUGE, and shows significant improvements in terms of generation quality measured by win rate. The ablation studies further underscore the importance of our design choices, particularly the use of pseudo-preference pairs over real reward pairs and the implementation of length normalization techniques.

As LLMs continue to expand their role in various applications, methods like PLaD that optimize for efficiency and quality without requiring extensive computational resources will become increasingly vital. We hope our work will pave the way for more sustainable and accessible AI language systems in the future.

7 Limitations

Potential limitations are: (1) the work depends on the assumption that the teacher model is better than the student model. While the assumption might hold at the beginning stage of student model training, it might not hold when the student model is very carefully trained. This might explicitly create a ceiling for the student model performance. The iterative methods could be considered for future work. (2) Our approach requires computation resources for conducting bulk inference for both teacher and student models. (3) To accurately obtain ranking pairs, we can resort to a reward model in the loop. However, it brings computational overhead to the current pipeline.

References

Rishabh Agarwal, Nino Vieillard, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. 2023. Gkd: Generalized knowledge distillation for auto-regressive sequence models. *arXiv preprint arXiv:2306.13649*.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. 2023. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. Gpt-neo: Large scale autoregressive language modeling with mesh-tensorflow.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Yao Fu, Hao-Chun Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing smaller language models towards multi-step reasoning. In *ICML*.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2023. Knowledge distillation of large language models. *arXiv preprint arXiv:2306.08543*.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander J. Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *ArXiv*, abs/2305.02301.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*.

Chen Liang, Simiao Zuo, Qingru Zhang, Pengcheng He, Weizhu Chen, and Tuo Zhao. 2023. Less is more: Task-aware layer-wise distillation for language model compression. In *International Conference on Machine Learning*, pages 20852–20867. PMLR.

Tianqi Liu, Zhen Qin, Junru Wu, Jiaming Shen, Misha Khalman, Rishabh Joshi, Yao Zhao, Mohammad Saleh, Simon Baumgartner, Jialu Liu, Peter J. Liu, and Xuanhui Wang. 2024. Lipo: Listwise preference optimization through learning-to-rank. *ArXiv*, abs/2402.01878.

Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J Liu, and Jialu Liu. 2023. Statistical rejection sampling improves preference optimization. *arXiv preprint arXiv:2309.06657*.

Raphael Gontijo Lopes, Stefano Fenu, and Thad Starner. 2017. Data-free knowledge distillation for deep neural networks. *arXiv preprint arXiv:1710.07535*.

David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. 2015. Unifying distillation and privileged information. *arXiv preprint arXiv:1511.03643*.

Aditya K Menon, Ankit Singh Rawat, Sashank Reddi, Seungyeon Kim, and Sanjiv Kumar. 2021. A statistical perspective on distillation. In *International Conference on Machine Learning*, pages 7632–7642. PMLR.

OpenAI. 2022. Chatgpt.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and Michael Bendersky. 2023. Large language models are effective text rankers with pairwise ranking prompting. *ArXiv*, abs/2306.17563.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

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

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 21:140:1–140:67.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling multi-step reasoning capabilities of large language models into smaller models via semantic decompositions. *ACL*, abs/2212.00193.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. 2019. Contrastive representation distillation. *arXiv preprint arXiv:1910.10699*.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.

Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. 2023. f-divergence minimization for sequence-level knowledge distillation. *arXiv preprint arXiv:2307.15190*.

Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. 2023. RRHF: Rank responses to align language models with human feedback. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Chen Zhang, Yang Yang, Jiahao Liu, Jingang Wang, Yunsen Xian, Benyou Wang, and Dawei Song. 2023a. Lifting the curse of capacity gap in distilling language models. *arXiv preprint arXiv:2305.12129*.

Rongzhi Zhang, Jiaming Shen, Tianqi Liu, Jialu Liu, Michael Bendersky, Marc Najork, and Chao Zhang. 2023b. Do not blindly imitate the teacher: Using perturbed loss for knowledge distillation. *arXiv preprint arXiv:2305.05010*.

Rongzhi Zhang, Yue Yu, Pranav Shetty, Le Song, and Chao Zhang. 2022. Prboost: Prompt-based rule discovery and boosting for interactive weakly-supervised learning. *arXiv preprint arXiv:2203.09735*.

Rongzhi Zhang, Yue Yu, and Chao Zhang. 2020. Seqmix: Augmenting active sequence labeling via sequence mixup. *arXiv preprint arXiv:2010.02322*.

Yao Zhao, Rishabh Joshi, Tianqi Liu, Misha Khalman, Mohammad Saleh, and Peter J Liu. 2023. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*.

A Appendix

A.1 Dataset Statistics

Table 4 shows the statistics of studied datasets.

Dataset	Train	Distillation	Dev	Test
TL;DR	56k	56k	0.5k	12k
Anthropic-HH	58k	58k	6.4k	6.6k

Table 4: Dataset Statistics

A.2 Model Details

We leverage the open-sourced pre-trained models in our main experiments. The model cards can be found via LLaMA-2 and GPT-Neo on Hugging Face.

A.3 Reward Model

For the reward model, we use the reward-model-deberta-v3-large-v2 released on Hugging Face. It is trained on

- webgpt_comparisons;
- summarize_from_feedback;
- synthetic-instruct-gptj-pairwise;
- anthropic_hh-rlhf,

which covers the related datasets of our studied tasks. Its performance is listed in Table 6.

Model	Summary	Anthropic RLHF
deberta-v3-large-v2	71.47	69.25

Table 6: Reward Model Performance.

A.4 Case Study

Figure 5 presents the case study on the TL;DR dataset. In the original text, critical details are highlighted in green. Similarly, in the student's summary, critical details are also highlighted in green, with redundant parts in yellow and factual errors in red. Compared to the summaries produced by the SFT student model, those generated by the student model distilled with our method are notably more concise and accurate, capturing a broader range of key points.

The original text describes an incident with details. The SFT student's summary, while capturing the essence of the event, includes factual errors and redundant elements. It incorrectly

states the GPU's issue, which is a misinterpretation of the original event. Furthermore, it redundantly mentions details not central to the original account's focus. In contrast, the summary produced by our distilled student model is concise and free of factual inaccuracies. It accurately captures the original text and summarizes it into a brief one, maintaining the critical details: the loss of the game, the subsequent damage to the GPU, and the resulting limitation on the computer's functionality. This case study shows an evident improvement after distillation with PLaD , in the model's ability to preserve key information while eliminating superfluous details.

A.5 Computing Resources

We test our code on the System Ubuntu 18.04.4 LTS with CPU: Intel(R) Xeon(R) Silver 4214 CPU @ 2.20GHz and GPU: NVIDIA A100 (80G). We implement our method using Python 3.9, PyTorch 2.0.1, and transformers 4.32.

A.6 Hyper-parameters

We list the search range of hyperparamters in Table 5. The search for batch size and learning rate is applied to all the methods. And for each baseline, we search for the best baseline-specific hyper-parameters. For those method-specific hyperparameters, the LoRA rank r does not impact the final performance much, while the margin β and the temperature τ slightly impacts the final performance, and we choose them carefully. Specifically, we have $\tau = 0.7$ and $\beta = 1.0$ in the main experiments.

A.7 Score Function

The score function in Eq. 5 is defined as

$$s(\hat{y}, y; x) = \sum_n F_n \left(e(\hat{y}, y), e(\hat{y}, x) \right), \quad (6)$$

where $F_n = 2P_n R_n / (P_n + R_n)$. The definitions of P_n, R_n, F_n can be found in (Zhang et al., 2019). When $n > 1$, we have

$$R_n = \frac{1}{|e|} \sum_{i+n \in e} \max_{j+n \in e} e_i^T e_{j+n}, \quad (7)$$

and

$$P_n = \frac{1}{|e|} \sum_{j:n \in e} e_i^T e_{j:n+n}. \quad (8)$$

Hyper-parameter	Search Range
Learning Rate	$\{1, 2, 3, 5\} \times 10^{-4}$
Batch Size for LLaMA	$\{1, 2, 4, 8, 16\}$
Batch Size for GPT-Neo	$\{4, 8, 16, 32\}$
Temperature τ	$\{0.1, 0.3, 0.5, 0.7, 0.9\}$
Margin β	$\{0.1, 0.2, 0.4, 0.6, 0.8, 1.0, 1.5, 2.0, 2.5\}$
LoRA rank	$\{8, 16, 32\}$

Table 5: The search range of hyper-parameters.

	TL;DR $\mathcal{L}_{cal}^{rank}$	TL;DR $\mathcal{L}_{cal}^{margin}$	HH $\mathcal{L}_{cal}^{rank}$	HH $\mathcal{L}_{cal}^{margin}$
LLaMA-2	40.46	40.21	27.65	27.74
GPT-Neo	14.70	14.76	10.38	9.94

Table 7: Ranking calibration loss v.s. margin calibration loss.

This score function measures the similarity between the positive (negative) text and the reference text with negligible computational overhead, because it uses the student model to obtain the representation instead of an external model. By integrating it into Eq. 5, we aim to scale the margin β with the score difference $s(\hat{y}, y'; x) - s(\hat{y}, y''; x)$. This modulation increases the margin when the positive sequence closely resembles the reference text. A larger margin means that the positive sequences should be preferred more via a higher likelihood in the generation process. This dynamic adjustment of the margin essentially encourages a clearer distinction between positive and negative sequence pairs during training.

For the empirical validation, we choose between the ranking calibration loss and the margin calibration loss based on their performance on the validation set. We report the numbers of each loss in the main experiments setting here.