PromptKD: Distilling Student-Friendly Knowledge for Generative Language Models via Prompt Tuning

Anonymous ACL submission

Abstract

Recent advancements in large language models (LLMs) have raised concerns about inference costs, increasing the need for research into model compression. While knowledge distillation (KD) is a prominent method for this, research on KD for generative language models like LLMs is relatively sparse, and the approach of distilling student-friendly knowledge, which has shown promising performance in KD for classification models, remains unexplored in generative language models. To explore this approach, we propose PromptKD, a simple yet effective method that utilizes prompt tuning - for the first time in KD - to enable generative language models to transfer student-friendly knowledge. Unlike previous works in classification that require fine-tuning the entire teacher model for extracting student-friendly knowledge, PromptKD achieves similar effects by adding a small number of prompt tokens and tuning only the prompt with student guidance. Extensive experiments on instruction-following datasets using the GPT-2 model family show that PromptKD achieves state-of-the-art performance while adding only 0.0007% of the teacher's parameters as prompts. Further analysis suggests that distilling student-friendly knowledge alleviates exposure bias effectively throughout the entire training process, leading to performance enhancements.

1 Introduction

With the massive improvement of generative language models, such as the emerging abilities (Wei et al., 2022) observed in large language models (LLMs), there is a growing need for model compression research to efficiently deploy models in various tasks (Touvron et al., 2023; Taori et al., 2023). However, among notable compression methods such as knowledge distillation (KD; Hinton et al., 2015; Kim and Rush, 2016; Gu et al., 2024), pruning (Ma et al., 2023), and quantization (Tao et al., 2022), KD has not been successfully applied to generative language models.

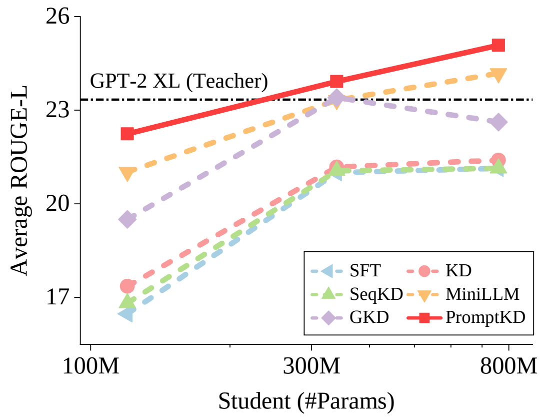


Figure 1: Comparison of instruction-following performance of KD methods using the GPT-2 model family. Owing to the student-friendly knowledge, our PromptKD outperforms others with only an additional 11K parameters. Dashed reference line represents the performance of the teacher model.

Since most KD methods are devised with models like BERT (Devlin et al., 2019) for classification tasks, the challenge arises when attempting to directly apply these KD methods to generative language models, which have different architectures and are designed for tasks other than classification. While there have been some methods proposed for generative language models, such as Supervised KD (Sanh et al., 2019) or SeqKD (Kim and Rush, 2016), they tend to be naive approaches. Even recently proposed works (Agarwal et al., 2024; Gu et al., 2024), like previous research, have focused on distribution discrepancy metrics or pseudo-targets. Therefore, despite the rapid advancement of LLMs in recent times, the drawback is that they are not designed with the extension to LLMs in mind.

Moreover, attempts to distill student-friendly knowledge in a generative language model have yet to be explored. Recent KD studies (Yang et al., 2022; Park et al., 2021a; Zhou et al., 2022) for classification tasks aim to distill such knowledge. This idea emerges because previous works extract knowledge from fixed teacher without knowing the student's capacity, and the observation (Cho and Hariharan, 2019) that larger teacher models do not necessarily improve student performance. However, there hasn't been any exploration of applying these ideas to generative language models. Since the capacity gap between teacher and student persists in KD for generative language models, it is reasonable to expect that distilling student-friendly knowledge would be beneficial.

To address this issues, we propose PromptKD, which utilizes prompts in generative language models to distill student-friendly knowledge. Extracting student-friendly knowledge from the teacher requires modifying the teacher, as in previous studies (Ren et al., 2023; Zhou et al., 2022). However, modifying a large teacher model can incur significant computational costs. PromptKD addresses this concern by exploiting prompt tuning. By appending prompt tokens to the beginning of the input, we can efficiently fine-tune the teacher model with notably fewer parameters. While there are other parameter-efficient fine-tuning methods such as prefix-tuning (Li and Liang, 2021) and LoRA (Hu et al., 2022), they suffer from the disadvantage that the number of parameters to be trained increases proportionally with the number of layers. Moreover, there is an observation (Lester et al., 2021) that prompt tuning shows similar performance to full-parameter fine-tuning as the model size increases, making prompt tuning a more reasonable choice. PromptKD learns prompts that stimulate the teacher to distill student-friendly knowledge with guidance from the student. Additionally, it employs regularization loss during the early stages of training to prevent significant divergence from the original teacher when appending prompts, ensuring stable training.

For evaluation, we measure the instruction-following performance (Ouyang et al., 2022), aiming to cover a variety of tasks that generative language models can perform. Compared to the existing baseline, PromptKD achieves state-of-the-art performance by adding prompt parameters equivalent to only 0.0007% of the teacher parameters, as depicted in Figure 1. Additionally, the analysis of exposure bias suggests that remarkable alleviation of exposure bias through student-friendly knowledge is likely the cause of performance improvement. Lastly, through ablation studies, we confirm the necessity of regularization loss and the importance of prompt initialization.

To summarize, our contribution is four-fold:

- We investigate the effect of student-friendly knowledge, which has not been previously explored in knowledge distillation (KD) for generation tasks.
- We propose PromptKD, the first usage of prompt tuning in KD, enabling memory-efficient extraction of student-friendly knowledge from teacher.
- Through extensive experiments on 5 instruction-following datasets, PromptKD achieves state-of-the-art performance.
- We suggest that the superiority of PromptKD lies in its ability to fully mitigate exposure bias in the training phase.

2 Related Work

KD for text classification Knowledge distillation (KD) (Hinton et al., 2015) is a model compression technique where the knowledge of a teacher model is transferred to improve the performance of a student model. Most KD research has been focused on text classification tasks. It has evolved from simple approaches (Song et al., 2020) that match the class distributions between teacher and student to more complex methods (Jiao et al., 2020; Sun et al., 2019; Wang et al., 2020; Park et al., 2021b) that involve matching hidden states or attention matrices between models. Recently, concerns have been raised about the observation (Cho and Hariharan, 2019) that larger teacher models do not necessarily produce better students and the issue of teachers distilling knowledge while being unaware of the student's capacity. To address this, Park et al. (2021a); Zhou et al. (2022); Ren et al. (2023) transfer student-friendly knowledge, which requires the teacher to transform during the distillation process, influenced by specific objectives aimed at benefiting the student. Additionally, focusing on the capacity gap between the teacher and student during training, Yang et al. (2022) proposes gradually pruning the teacher, while Liang et al. (2023a) suggests initializing the student as a model of the same size as the teacher and then pruning it during training.

KD for text generation For text generation, Sanh et al. (2019) minimizes the KL divergence between the next token prediction distributions of the teacher and student at each time step. In addition, some research (Calderon et al., 2023; Agarwal et al., 2024) focus on the sentences inputted to the teacher and student during the distillation process. For example, Kim and Rush (2016) uses sentences generated by the teacher as pseudo-targets instead of ground truth. Moreover, black-box KD methods (Hsieh et al., 2023; Ho et al., 2023) that use inference-only black-box LLMs as teachers and augment existing data before training are proposed. Recently, Agarwal et al. (2024); Gu et al. (2024) explored discrepancy metrics between model distributions and used sentences generated by the student as pseudo-targets to minimize exposure bias. However, there have been no attempts yet to distill student-friendly knowledge while the teacher is aware of the student's capacity. Although Liang et al. (2023b) incorporates task-aware filters into both teacher and student to transfer knowledge, its scalability is limited due to the addition of filters at each layer for layer distillation. Crucially, it encourages knowledge to be task-specific, making it diverges from what we aim to explore in this paper.

Prompt tuning After Brown et al. (2020) demonstrates that pre-trained language models can perform specific tasks by prepending text prompts to input, many studies have tried to either manually craft (Schick and Schütze, 2021) or automatically discover (Shin et al., 2020; Jiang et al., 2020; Gao et al., 2021) such hard prompts, which are discrete tokens. Subsequently, research (Hambardzumyan et al., 2021; Zhong et al., 2021) emerged to advance prompts into the form of soft prompts composed of embeddings, making prompt updates via back-propagation easier and resulting in better performance compared to hard prompts. Presently, prompt tuning (Lester et al., 2021) has become a prominent parameter-efficient fine-tuning technique. Although Ma et al. (2022) uses hard prompts to generate input data for knowledge extraction, we are pioneering the use of prompts for parameter-efficient fine-tuning in KD research.

3 PromptKD

PromptKD is devised in the instruction-following (Ouyang et al., 2022) setting for application to generative language models. We formulate instruction-following as a conditional text generation task, where the request $\boldsymbol{x} = \{x_1, x_2, \ldots, x_n\}$ sampled from the data distribution p_x comprises instruction and input to describe the task. Then, given the request $\boldsymbol{x}$ as a condition, the model generates a response $\boldsymbol{y} = \{y_1, y_2, \ldots, y_T\}$. For prompt tuning, soft prompts $P = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_m\}$, where $\boldsymbol{p}_i$ is an embedding vector of the same dimension as the token embedding, are initialized with text and prepended to the input request $\boldsymbol{x}$. Formally, given the request $\boldsymbol{x}$, the teacher model distribution conditioned on the prompt P is denoted as $p(\boldsymbol{y}|P, \boldsymbol{x})$ (here we suppress the teacher's model parameter since it is fixed), and the student's model distribution parameterized by θ is denoted as $q_\theta(\boldsymbol{y}|\boldsymbol{x})$, where only the student model parameters θ and the prompt P are trainable. The training process consists of 3 steps per iteration, as shown in Figure 2. First, generating input data used for knowledge distillation (*pseudo-target generation*). Then, updating the prompt based on guidance from the student and teacher models to facilitate adaptive teaching (*prompt tuning for adaptive teaching*). Finally, distilling student-friendly knowledge to the student using the updated prompt (*student-friendly knowledge distillation*).

3.1 Pseudo-Target Generation

PromptKD uses the response $\boldsymbol{y}$ generated by the student for the prompt tuning and knowledge distillation processes, treating it as the pseudo-target. This approach addresses exposure bias, which arises due to the discrepancy between the sentences used during training and those generated during inference, leading to degraded performance in free-run generation (Zhang et al., 2019). Based on the insight (Agarwal et al., 2024) that incorporating sentences that the model can generate during free-run generation into the training process can mitigate exposure bias, we devise the approach accordingly. It is worth noting that for the sake of method simplicity, back-propagation during this sampling process is not conducted.

3.2 Prompt Tuning for Adaptive Teaching

Initially, we concatenate the request $\boldsymbol{x}$ and response $\boldsymbol{y}$, including the prompt P for the teacher, and input them into both models. Prompt P is updated to minimize the KD loss $\mathcal{L}_{\text{kd}}$, which computes the distribution discrepancy of the response part. This encourages the prompt to enable the teacher to generate sentences at a similar level to the student

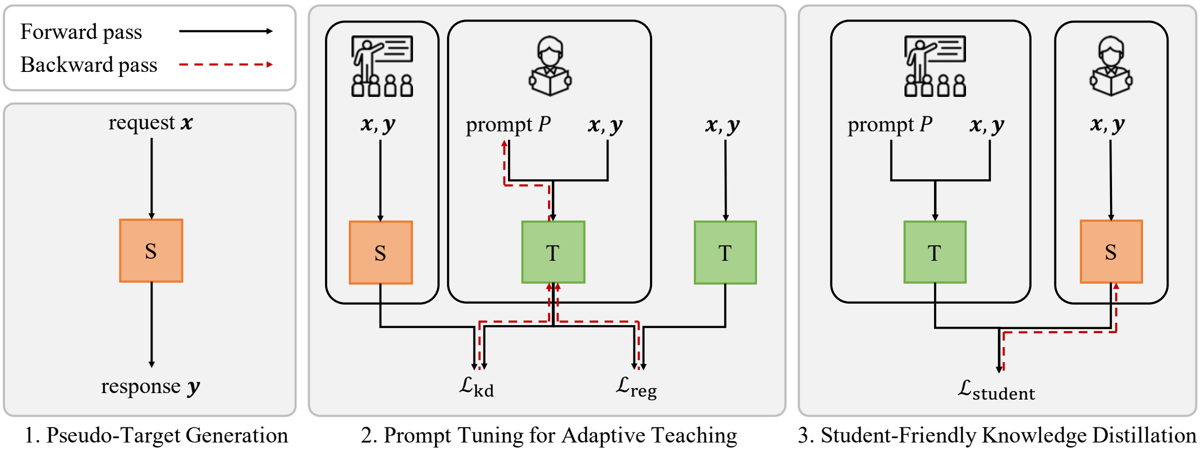


Figure 2: Training procedure of PromptKD. To mitigate exposure bias, responses are generated by the student to be used as pseudo-targets. Then, for adaptive teaching, the prompt input to the teacher is trained based on guidance from the student. During this process, regularization loss is also employed to address instability stemming from the prompt. Lastly, teacher distills student-friendly knowledge to the student using the trained prompt.

when it is prepended to the teacher's input. Drawing inspiration from the concept of adaptive teaching in education, we design this objective with the aim of enabling students to receive knowledge from the teacher at a level they can comprehend.

However, during the early stages of training, the influence of the prompt may cause significant deviations or inaccuracies in the teacher model distribution, leading to unstable learning (Hou et al., 2022). To address this issue, we initialize the prompt with text embedding and devise an additional regularization loss $\mathcal{L}_{\text{reg}}$ to ensure that the teacher model distribution remains similar whether the prompt is used or not. The regularization loss $\mathcal{L}_{\text{reg}}$ is computed in a similar manner to the KD loss $\mathcal{L}_{\text{kd}}$, but with the difference that it is measured based on the teacher model distribution when the prompt is excluded from the input given to the teacher. This approach allows for the continued use of the fixed teacher model, making it memory-efficient. However, since the fixed teacher is unaware of the student's capacity, $\mathcal{L}_{\text{reg}}$ deviates from our ultimate goal. Therefore, we introduce a coefficient that starts at 1 for $\mathcal{L}_{\text{reg}}$ and linearly decreases to 0 during training, focusing solely on stabilizing the early stages of learning.

Regarding the two objectives, we opt for minimizing the reverse KL divergence instead of the forward KL divergence to measure the discrepancy, as it exhibits mode-seeking behavior (Nowozin et al., 2016) and benefits generation tasks. Hence, summarizing the two objectives, the final loss $\mathcal{L}_{\text{prompt}}$, which updates only the prompt, is determined by their summation, as follows:

Algorithm 1 PromptKD

Input: teacher T, student's output distribution q_θ, data distribution p_x, prompt P, training step K, learning rate η
for each step $k = 1, ..., K$ **do**
  Sample a request $\boldsymbol{x}$ from p_x
  Sample a response $\boldsymbol{y}$ from $q_\theta(\cdot|\boldsymbol{x})$
  Update $P \leftarrow P - \eta \nabla \mathcal{L}_{\text{prompt}}$ ▷ Eq. (3)
  Update $\theta \leftarrow \theta - \eta \nabla \mathcal{L}_{\text{student}}$ ▷ Eq. (4)
end for
return q_θ

$$\mathcal{L}_{\text{kd}} = D_{KL}\big(p(\boldsymbol{y}|P,\boldsymbol{x}) \parallel q_\theta(\boldsymbol{y}|\boldsymbol{x})\big), \quad (1)$$

$$\mathcal{L}_{\text{reg}} = D_{KL}\big(p(\boldsymbol{y}|P,\boldsymbol{x}) \parallel p(\boldsymbol{y}|\boldsymbol{x})\big), \quad (2)$$

$$\mathcal{L}_{\text{prompt}} = \mathcal{L}_{\text{kd}} + \left(\frac{K-k}{K}\right)\mathcal{L}_{\text{reg}}, \quad (3)$$

where K represents the total training steps, and k denotes the current step.

3.3 Student-Friendly Knowledge Distillation

The updated prompt is utilized as a trigger to extract student-friendly knowledge from the teacher and distill it to the student. The student loss $\mathcal{L}_{\text{student}}$ minimizes the distribution discrepancy between teacher and student through reverse KL divergence, as follows:

$$\mathcal{L}_{\text{student}} = D_{KL}\big(q_\theta(\boldsymbol{y}|\boldsymbol{x}) \parallel p(\boldsymbol{y}|P,\boldsymbol{x})\big). \quad (4)$$

For a clear understanding, we summarize the PromptKD algorithm in Algorithm 1.

#Params	Method	Instruction-following datasets				
		Dolly	SelfInst	Vicuna	S-NI	UnNI
1.5B	Teacher	27.3	14.5	16.2	27.1	31.6
120M	SFT	22.9	10.2	14.5	16.3	18.5
	KD	22.6	11.0	15.1	18.0	20.1
	SeqKD	23.3	10.3	14.7	16.6	19.2
	MiniLLM	24.2	12.7	16.9†	25.1	26.2
	GKD	24.8	11.1	**17.7**†	20.7	23.2
	PromptKD	**25.6**	**13.1**	16.8†	**26.8**	**28.9**
340M	SFT	25.1	12.9	15.9	23.7	27.4
	KD	25.1	13.0	15.6	24.5	27.7
	SeqKD	25.3	12.7	16.0	23.8	27.5
	MiniLLM	26.3	14.8†	**17.9**†	26.4	31.2
	GKD	26.9	14.8†	17.8†	26.6	30.9
	PromptKD	**27.3**†	**15.0**†	17.6†	**27.1**†	**32.6**†
760M	SFT	24.9	13.4	15.8	24.0	27.6
	KD	25.7	13.7	15.9	24.0	27.7
	SeqKD	25.2	13.3	15.8	24.0	27.4
	MiniLLM	26.2	15.8†	16.9†	28.5†	33.5†
	GKD	**26.9**	14.1	17.1†	25.4	29.6
	PromptKD	**26.9**	**16.4**†	**17.8**†	**29.5**†	**34.8**†

Table 1: Evaluation results on 5 instruction-following datasets. Each ROUGE-L score is averaged over 5 random seeds. The best score for each model size is highlighted in **boldface**. †Results surpass those of the teacher.

4 Experiments

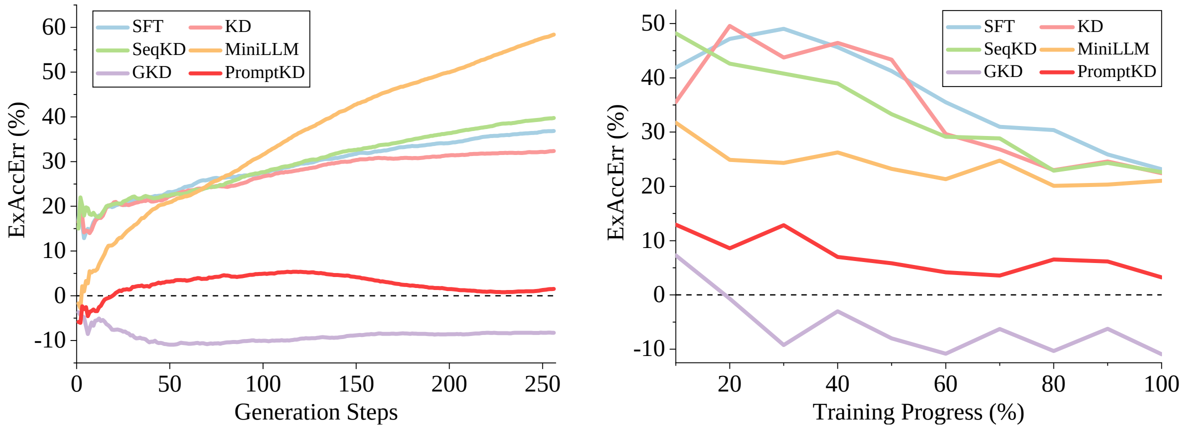
4.1 Experimental Setup

Following Gu et al. (2024), we evaluate PromptKD using 5 instruction-following datasets.

Settings We split the Dolly (Conover et al., 2023), consisting of 15,000 human-written instruction-response pairs, into 14,000 for training and 500 for validation and testing. For evaluation, we employ not only the Dolly but also 4 additional datasets: SelfInst (Wang et al., 2023), consisting of user-oriented instruction-following sets; Vicuna (Chiang et al., 2023), comprising 80 questions used in the Vicuna evaluation; S-NI, the test set of Super-NaturalInstructions (Wang et al., 2022); and UnNI, the core dataset of UnnaturalInstructions (Honovich et al., 2023). Similar to Gu et al. (2024), data samples with ground truth response lengths of 11 or more are utilized for S-NI and UnNI. We generate 5 responses for each request in each dataset using different random seeds and evaluate them to report the average scores for reliability. We choose the ROUGE-L score (Lin, 2004) as the metric for evaluation, as it aligns well with human preferences (Wang et al., 2022) in instruction-following evaluations. The best checkpoint based on the ROUGE-L score on the validation set is used for evaluation.

Models To evaluate the instruction-following performance of PromptKD, we utilize pre-trained models from the GPT-2 family. GPT-2 XL (1.5B params) is employed for the teacher model, and GPT-2 Base (120M params), GPT-2 Medium (340M params), GPT-2 Large (760M params) are used for the student model. Before knowledge distillation, the teacher model undergoes supervised fine-tuning on the Dolly training set. Similarly, the student model is also fine-tuned on the same training data for only three epochs, following the previous works (Agarwal et al., 2024; Gu et al., 2024).

Baselines PromptKD is compared with various approaches ranging from supervised fine-tuning (SFT), which does not involve knowledge distillation, to commonly used methods in generation tasks such as Supervised KD (KD; Sanh et al., 2019), SeqKD (Kim and Rush, 2016), and more recent proposals like MiniLLM (Gu et al., 2024) and GKD (Agarwal et al., 2024). KD and SeqKD both aim to minimize the discrepancy between the model distributions of teacher and student at each token step. The difference lies in whether the input



(a) exposure bias against generation steps

(b) exposure bias against training progress

Figure 3: The measurement of exposure bias. Excess accumulated error (ExAccErr) is measured with respect to generation steps and training progress, where values closer to 0 indicate alleviation of exposure bias.

sentence is ground truth or pseudo-target generated by the teacher. MiniLLM replaces forward KL divergence with reverse KL divergence and updates the student model using policy gradient. On the other hand, GKD focuses on distribution discrepancy metrics and pseudo-targets to propose a general method. In this paper, GKD computes reverse KL divergence and utilizes sentences generated by the student as pseudo-targets, and this choice is based on the reported performance in their paper. Additionally, it is worth noting that the students for MiniLLM, GKD, and PromptKD all commence from the same supervised fine-tuned checkpoint, while other methods start from pre-trained models. For training details, please see the Appendix A.

4.2 Experimental Results

We report the instruction-following performance of PromptKD and baselines on 5 datasets in Table 1.

Firstly, PromptKD achieves state-of-the-art performance overall in the instruction-following setting, outperforming other KD baselines. Additionally, it also outperforms on 4 datasets not used in training, demonstrating PromptKD's superb generalization ability. It's worth noting that despite MiniLLM incorporating language modeling loss through the corpus used for pre-training, PromptKD exhibits better performance.

Furthermore, only PromptKD shows superior performance to the teacher across all datasets. This demonstrates that modifying the teacher to extract student-friendly knowledge for distillation works not only for classification tasks but also for generation tasks. Moreover, the better performance of PromptKD, MiniLLM, and GKD, which utilize responses generated by the student as pseudo-targets, compared to other baselines, can be interpreted as exposure bias mitigation contributing to the performance improvement.

Lastly, as the model size increases, PromptKD outperforms in more datasets. This can be attributed to the fact that prompt tuning exhibits a similar effect to full-parameter fine-tuning as the model size grows (Lester et al., 2021). Thanks to the scalability and efficiency of prompt tuning, PromptKD can be expected to yield outstanding results even when applied to larger models.

PromptKD and the baselines' qualitative results are summarized in the Appendix B, where it is shown that PromptKD generates responses most similar to the ground truth.

4.3 Analysis

Exposure bias In this section, we investigate exposure bias to understand why PromptKD performs well. Exposure bias refers to the mismatch in distribution between the sentences seen during training and those generated during inference. If exposure bias is significant, the tokens generated during inference may diverge from those seen during training, leading to accumulated errors in the generation process. Following Arora et al. (2022), exposure bias up to l generation steps can be quantified as follows:

$$\text{ExAccErr}(l) = \frac{R(l) - E(l)}{E(l)} \times 100\%, \quad (5)$$

$$R(l) = \sum_{t=1}^{l} \mathop{\mathbb{E}}_{\substack{\boldsymbol{y}_{<t} \sim q_\theta(\cdot|\boldsymbol{x}) \\ y_t \sim p(\cdot|\boldsymbol{y}_{<t}, \boldsymbol{x})}} \log \frac{p(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})}{q_\theta(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})}, \quad (6)$$

$$E(l) = \sum_{t=1}^{l} \mathop{\mathbb{E}}_{\substack{\boldsymbol{y}_{<t} \sim p(\cdot|\boldsymbol{x}) \\ y_t \sim p(\cdot|\boldsymbol{y}_{<t}, \boldsymbol{x})}} \log \frac{p(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})}{q_\theta(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})}. \quad (7)$$

$R(l)$ represents the average forward KL divergence up to l time steps when the student-generated response is used as the pseudo-target, while $E(l)$ is similar to $R(l)$ but differs in that it uses the teacher-generated response as the pseudo-target. $R(l)$ can be interpreted as the distribution gap between the teacher and the student due to low-quality pseudo-targets generated by the student, while $E(l)$ serves as a lower-bound of distribution gap between the teacher and the student. Therefore, ExAccErr calculates the relative error caused solely by exposure bias. If exposure bias is alleviated, the student should exhibit a nearly identical distribution gap regardless of which model generated the response. Therefore, the ExAccErr value should approach 0.

We depict the ExAccErr at each generation step and the variation of ExAccErr up to 50 generation steps during the model training in Figure 3. In this experiment, a fixed pre-trained teacher is used as the teacher, while the student employs models distilled using each KD method.

When examining the ExAccErr over generation steps in Figure 3(a), it can be observed that for most methods, the error due to exposure bias accumulates as the generation length increases, increasing ExAccErr values. In the case of GKD, the objective used in training leads the student to minimize $R(l)$. Consequently, the value becomes negative, indicating that the distribution gap between the student and the teacher approaches 0 when using a student-generated response as a pseudo-target. However, there still exists a distribution gap for the teacher's oracle response, and this means exposure bias also still exists. Nevertheless, PromptKD maintains ExAccErr values close to 0 at all generation steps, indicating that error accumulation does not occur. This demonstrates that PromptKD is the most effective in alleviating exposure bias compared to other baselines.

Furthermore, ExAccErr is measured up to 50 generation steps in Figure 3(b) to focus on the early generations where errors tend to accumulate. To observe how it changes during the training process, the total training step of best checkpoint is divided by 10, and the model is saved at each time step for ExAccErr measurement. It is apparent that PromptKD, MiniLLM, and GKD, which utilize student's responses, exhibit consistently lower ExAccErr values compared to other baselines from the early stages of training. Among them, PromptKD demonstrates the most stable maintenance of ExAccErr close to 0, signifying that distilling student-friendly knowledge aids in mitigating exposure bias during training.

Method	MA (GB)	CA (GB)	Time (hour)
SFT	2.36	2.75	0.43
KD	5.91	6.45	0.85
SeqKD	5.91	6.45	0.86
MiniLLM	12.63	**21.45**	7.59
GKD	6.23	6.93	8.48
PromptKD	**13.47**	14.40	**9.78**

Table 2: Comparison of computational costs. Where MA denotes the maximum allocated memory on the GPU and CA denotes the maximum cached memory on the GPU. Time indicates the total training time for each method. All computational costs are calculated on 8 NVIDIA GeForce RTX 3090 (24 GB) GPUs.

Computational cost To demonstrate the efficiency of PromptKD, we compare its computational cost with baselines in Table 2. GPT-2 XL and GPT-2 Base are used as the teacher and the student, with measurements conducted on 8 NVIDIA GeForce RTX 3090 GPUs. From a time perspective, methods that sample the student at each iteration to create pseudo-targets take significantly more time than those that do not. Additionally, while PromptKD introduces only a small amount of additional parameters, namely the product of the prompt length and input embedding dimension, it requires a considerable amount of memory due to back-propagation during training. In contrast, MiniLLM, which does not add parameters, requires more cached memory than PropmtKD. This is because it receives guidance not only from the teacher but also from the corpus used in pre-training and calculates rewards in advance at every iteration. When compared to MiniLLM and GKD, which show superior performance to other baselines, PromptKD shows competitive advantages because it performs reliably better despite requiring similar or slightly more time and memory resources.

#Params	w/o $\mathcal{L}_{\text{reg}}$	w/ $\mathcal{L}_{\text{reg}}$
120M	21.97	**22.25**
340M	**24.13**	23.92
760M	24.47	**25.08**

Table 3: Ablation on regularization loss. We assess the average instruction-following performance of student models without and with regularization loss to verify the effectiveness of regularization.

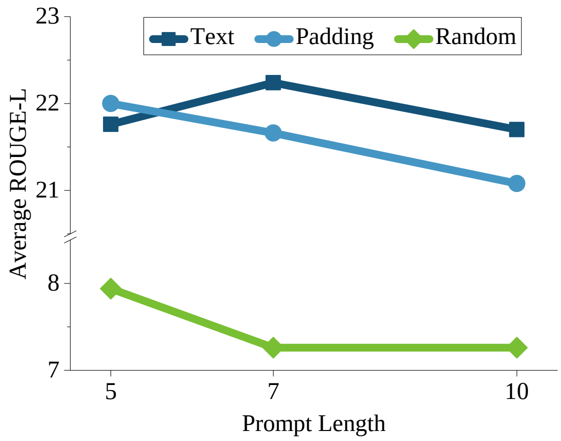


Figure 4: Ablation on prompt settings. To validate the impact of prompt initialization method and length, we evaluate the average ROUGE-L score over varying these settings.

4.4 Ablation Study

Regularization loss To confirm the effectiveness of the introduced regularization loss in alleviating instability when the prompt is prepended, we conduct experiments by excluding this objective. The average performance across the 5 datasets is reported in Table 3. Although there is a slight performance drop when using regularization loss with GPT-2 Medium, we observe a more significant performance increase with the other two models. This suggests the necessity of regularization loss for improving performance.

Prompt settings Although the regularization loss effectively mitigates the initial instability, the prompt's length and initialization also significantly influence the prompt tuning process (Hou et al., 2022). Therefore, the average instruction-following performance is measured by varying the prompt length m from 5, 7, 10 and the initialization method from random, padding, text. Results are summarized in Figure 4. In the padding method, all prompt tokens are initialized with the embedding of the "[PAD]" token, while in the text method, the sentence "Suppose you are a student." is tokenized, and these embeddings are used for initializing prompt tokens from the beginning. In this case, if the number of prompt tokens is smaller, the sentence is truncated, while if it is larger, all embeddings of the sentence are assigned, and then the embeddings are assigned again from the beginning for the next prompt token.

Firstly, considering the emphasis on the importance of prompt initialization in previous works, it is found that training does not proceed properly with random initialization. Moreover, generally, the text initialization method shows better performance than the padding method. Regarding length, when initialized with text, better performance is observed with a length of 7, while with padding initialization, shorter lengths exhibit better performance. This is presumably because, in text initialization, the sentence is fully encoded since it is tokenized into 7 tokens, while in padding initialization, longer lengths exert a greater influence on the instability of teacher model distribution when prepended. Therefore, all experiments in this paper are performed with a prompt length of 7, initialized using text initialization.

KL divergences To assess the impact of distribution discrepancy metrics, we conduct ablation study on this. We observe the best performance when measuring discrepancy using reverse KL divergence in both KD loss $\mathcal{L}_{\text{kd}}$ and regularization loss $\mathcal{L}_{\text{reg}}$, similar to previous observations. Detailed experimental results are provided in the Appendix C.

5 Conclusions

In this work, we have pioneered the exploration of extracting and distilling student-friendly knowledge for generative language models. To achieve this, we have proposed a novel method called PromptKD, which leverages prompt tuning in knowledge distillation for the first time. Thanks to the memory-efficient nature of prompts and the advantage of replacing full-parameter fine-tuning, particularly beneficial for larger models like LLMs, PromptKD has proven to be an efficient approach. Through extensive experiments, PromptKD has achieved state-of-the-art performance, confirming the effectiveness of student-friendly knowledge in generation tasks. Additionally, through exposure bias analysis, we have demonstrated that PromptKD successfully alleviates exposure bias throughout the training process.

Limitations

While PromptKD has achieved state-of-the-art performance by distilling student-friendly knowledge, it still has limitations in terms of its naive extraction approach. Considering that knowledge distillation (KD) research for classification tasks employs various methods to distill student-friendly knowledge, it is expected that there are alternative approaches to effectively transfer student-friendly knowledge in a generative language model. Furthermore, although PromptKD is designed for instruction-following settings based on task-specific KD, there is a need for expansion towards task-agnostic KD to make it applicable during the pre-training process.

Ethics Statement

PromptKD utilizes pre-trained models, exposing it to risks similar to those highlighted by Weidinger et al. (2021); Bommasani et al. (2021), regarding the vulnerability of pre-trained language models to ethical and social risks. Additionally, Hooker et al. (2020) mentions that the process of model compression can introduce biases. However, since most model compression studies leverage pre-trained models, these issues are general risks and not specific to PromptKD. Nevertheless, these risks should be addressed in the future through advanced pre-training objectives and dataset collection methods (Lee et al., 2023).

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. Generalized knowledge distillation for auto-regressive language models. In *The Twelfth International Conference on Learning Representations*.

Kushal Arora, Layla El Asri, Hareesh Bahuleyan, and Jackie Cheung. 2022. Why exposure bias matters: An imitation learning perspective of error accumulation in language generation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 700–710, Dublin, Ireland. Association for Computational Linguistics.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Nitay Calderon, Subhabrata Mukherjee, Roi Reichart, and Amir Kantor. 2023. A systematic study of knowledge distillation for natural language generation with pseudo-target training. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14632–14659, Toronto, Canada. Association for Computational Linguistics.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*.

Jang Hyun Cho and Bharath Hariharan. 2019. On the efficacy of knowledge distillation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4794–4802.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. 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*.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. WARP: Word-level Adversarial ReProgramming. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4921–4933, Online. Association for Computational Linguistics.

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

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2023. Large language models are reasoning teachers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14852–14882, Toronto, Canada. Association for Computational Linguistics.

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.

Sara Hooker, Nyalleng Moorosi, Gregory Clark, Samy Bengio, and Emily Denton. 2020. Characterising bias in compressed models. *arXiv preprint arXiv:2010.03058*.

Yutai Hou, Hongyuan Dong, Xinghao Wang, Bohan Li, and Wanxiang Che. 2022. MetaPrompting: Learning to learn better prompts. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3251–3262, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex 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. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8003–8017, Toronto, Canada. Association for Computational Linguistics.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

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

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

Hwaran Lee, Seokhee Hong, Joonsuk Park, Takyoung Kim, Gunhee Kim, and Jung-woo Ha. 2023. KoSBI: A dataset for mitigating social bias risks towards safer large language model applications. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 208–224, Toronto, Canada. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Chen Liang, Haoming Jiang, Zheng Li, Xianfeng Tang, Bing Yin, and Tuo Zhao. 2023a. Homodistil: Homotopic task-agnostic distillation of pre-trained transformers. In *The Eleventh International Conference on Learning Representations*.

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

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.

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

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. *CoRR*, abs/2305.11627.

Xinyin Ma, Xinchao Wang, Gongfan Fang, Yongliang Shen, and Weiming Lu. 2022. Prompting to distill: Boosting data-free knowledge distillation via reinforced prompt. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 4296–4302. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. 2016. f-gan: Training generative neural samplers using variational divergence minimization. *Advances in neural information processing systems*, 29.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Dae Young Park, Moon-Hyun Cha, Daesin Kim, Bohyung Han, et al. 2021a. Learning student-friendly teacher networks for knowledge distillation. *Advances in neural information processing systems*, 34:13292–13303.

Geondo Park, Gyeongman Kim, and Eunho Yang. 2021b. Distilling linguistic context for language model compression. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 364–378, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yuxin Ren, Zihan Zhong, Xingjian Shi, Yi Zhu, Chun Yuan, and Mu Li. 2023. Tailoring instructions to student's learning levels boosts knowledge distillation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1990–2006, Toronto, Canada. Association for Computational Linguistics.

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*.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Kaitao Song, Hao Sun, Xu Tan, Tao Qin, Jianfeng Lu, Hongzhi Liu, and Tie-Yan Liu. 2020. Light{paff}: A two-stage distillation framework for pre-training and fine-tuning.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for BERT model compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4323–4332, Hong Kong, China. Association for Computational Linguistics.

Chaofan Tao, Lu Hou, Wei Zhang, Lifeng Shang, Xin Jiang, Qun Liu, Ping Luo, and Ngai Wong. 2022. Compression of generative pre-trained language models via quantization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4821–4836, Dublin, Ireland. Association for Computational Linguistics.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

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

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, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. 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, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, et al. 2021. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*.

Yi Yang, Chen Zhang, and Dawei Song. 2022. Sparse teachers can be dense with knowledge. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3904–3915, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. 2019. Bridging the gap between training and inference for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4334–4343, Florence, Italy. Association for Computational Linguistics.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [MASK]: Learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5017–5033, Online. Association for Computational Linguistics.

Wangchunshu Zhou, Canwen Xu, and Julian McAuley. 2022. BERT learns to teach: Knowledge distillation with meta learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7037–7049, Dublin, Ireland. Association for Computational Linguistics.

A Training Details

In our study, we employ the AdamW (Loshchilov and Hutter, 2019) optimizer for training, with batch sizes of 32 for GPT-2 Base and 8 for GPT-2 Medium and Large. The learning rates of prompt and student are set at 5e-5 for Base, 1e-5 for Medium, and 5e-6 for Large. For the generation, we sample with top-k and top-p parameters at 0 and 1.0, respectively, and use a fixed temperature of 1.0. Training and generation phases both have a maximum sequence length of 512 and a maximum prompt length of 256. Please note that we preprocess each instruction-following dataset by converting the instruction-response pairs into a standardized sentence structure, as shown in Table 4. For reproducibility of our PromptKD, we will make both the code and the checkpoints public.

B Qualitative Results

For the qualitative results, we present samples generated by student models trained using various methods. The prompts are drawn from the S-NI dataset and utilize GPT-2 XL as the teacher model, with GPT-2 Large employed as the student model. Results are shown in Table 6.

Prompt Format
Below is an instruction that describes a task. Write a response that appropriately completes the request. ### Instruction: {instruction} ### Input: {input} ### Response:

Table 4: Prompt format used for training and evaluation.

$\mathcal{L}_{\text{kd}}$ & $\mathcal{L}_{\text{reg}}$	ROUGE-L
Reverse KL & Reverse KL	**22.25**
Reverse KL & Forward KL	21.91
Forward KL & Reverse KL	22.20
Forward KL & Forward KL	22.13

Table 5: Ablation on distribution discrepancy metric. Since each loss can compute distribution discrepancy with either forward or reverse, we report the average instruction-following performance for each pair.

C Ablation on KL Divergences

During prompt tuning, PromptKD minimizes the reverse KL divergence between the teacher distribution and the student distribution ($\mathcal{L}_{\text{kd}}$) or between the teacher distribution and the teacher distribution excluding the prompt ($\mathcal{L}_{\text{reg}}$). In this context, forward KL divergence can also be considered instead of reverse KL divergence. As shown in Table 5, experimental results indicate that using reverse KL divergence yields the best performance. However, there is barely any significant difference. We conjecture that since the model distribution being trained is derived from the teacher, resulting in similar or even more modes in distribution, which prevent undesirable behaviors such as mode-covering even during forward KL divergence minimization.

Case 1	
Instruction	Convert a disfluent question to a proper question. A disfluent question is a question that has some interruptions in it while framing. A proper question is the correct form of the question without any disfluency.
Input	What do no I meant Who do some tax agents want to make an impression on during an arrest?
Ground Truth	Who do some tax agents want to make an impression on during an arrest?
SFT	Some tax agents want to make an impression on during an arrest.
KD	The tax agents want to make an impression on during an arrest.
SeqKD	No I meant to ask who are some tax agents that want to make an impression on an arrest.
MiniLLM	A disfluent question is one that has some interruptions while framing. A proper questions is the correct form of a question without any disfluidity.
GKD	No I mean that there are no tax agents who want to make an impression on the arrest.
PromtKD	Who do some tax agents want make an impression on during a arrest?
Case 2	
Instruction	Given an entity, a before event, an after event, and an attribute related to the entity, generate a sentence as output. Your sentence should show the changes in the attribute of the entity.
Input	entity: rope, before: separate from ball, after: through slot in ball, attr: location
Ground Truth	The location of the rope was separate from the ball before and through a slot in the ball afterwards.
SFT	Rope was previously connected to a ball through a slot in the ball. When the rope is pulled through the ball, it changes location and is no longer connected to the ball.
KD	Rope is a string. before: ball, after: rope stroke
SeqKD	Rope is made of rope.
MiniLLM	through slot in ball
GKD	before: separate from Ball, after: through slot In Ball, attr: location
PromptKD	Rope was separate from Ball until through slot in Ball.

Table 6: Sample response from the S-NI dataset. PromptKD follows the instructions most accurately.