Controlling Distillation Trust in PromptKD : A Multi-Strategy Approach to Uncertainty

Korea University COSE461 Final Project

Hyeonwoo Baek
Department of Industrial Management Engineering
Team 5
2020170851

Chanhong Min
Department of Industrial Management Engineering
Team 5
2020170831

Hyolim Son
Department of Industrial Management Engineering
Team 5
2022170831

Seunghyun Lee
Department of Industrial Management Engineering
Team 5
2023170829

Abstract

In reality, LLM outputs are not always reliable. Without considering the uncertainty in the teacher's predictions, the student model may inherit misleading or incorrect knowledge. PromptKD, while effective, also overlooks this aspect of reliability.

To address this limitation, we propose an enhanced version of PromptKD that incorporates uncertainty-aware techniques: (1) *entropy-based loss weighting*, which emphasizes confident teacher outputs; (2) *soft blending of student distributions*, which stabilizes learning by mixing student and teacher probabilities; and (3) *teacher assistant filtering*, which retains only the most informative teacher logits.

We evaluate our method on the `databricks-dolly-15k` dataset in an instruction-following setting. Results show that blending-based strategies, such as soft and dynamic blending, achieve slight improvements over the PromptKD baseline in ROUGE-L and BERTScore. However, other techniques like loss weighting and teacher assistant filtering underperform, highlighting the importance of careful calibration when incorporating uncertainty. These findings suggest that softly balancing teacher and student distributions may be more effective than aggressively modifying the teacher signal.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities across a wide range of natural language tasks, including instruction-following, dialogue generation, and question answering. However, the immense computational cost associated with their deployment has led to increased interest in model compression techniques such as pruning, quantization, and knowledge distillation (KD). Among these, KD is especially attractive as it enables a compact student model to learn from a larger teacher model by mimicking its behavior.

While KD has been extensively studied for classification tasks, its application to generative language models remains challenging. Recently, PromptKD [1] introduced a novel approach that incorporates prompt tuning into the distillation process. Instead of updating the entire teacher model, PromptKD learns soft prompts to extract *student-friendly knowledge*, which effectively mitigates exposure bias and leads to improved instruction-following performance.

Although PromptKD provides a strong foundation, it does not explicitly consider the uncertainty of the teacher's predictions or the confidence of the student. We argue that uncertainty-awareness is crucial when distilling knowledge from one probabilistic model to another, especially in generative settings where multiple valid responses may exist.

To address this limitation, we propose three enhancements to PromptKD that incorporate uncertainty-aware mechanisms into the distillation process:

- **Entropy-based Loss Weighting**: We assign greater importance to low-entropy teacher outputs, which are more confident and likely to provide reliable supervision. This encourages the student to focus on clearer learning signals.
- **Soft Blending of Student Distributions**: During training, we mix the student's own output distribution with the teacher's distribution. This prevents over-reliance on potentially overconfident teacher signals and promotes smoother, more stable learning.
- **Teacher Assistant Filtering**: We filter the teacher's output by retaining only the top-k logits or by applying top-p (nucleus) filtering before computing the KL divergence loss. This reduces noise from low-probability tokens and focuses distillation on the most informative outputs.

We evaluate our method on the `dolly` dataset in the instruction-following setting. Experimental results reveal that while blending-based strategies—such as soft blending and dynamic blending—achieve marginal improvements over the PromptKD baseline, other techniques like entropy-based loss weighting and teacher assistant filtering do not consistently outperform the baseline. These findings suggest that carefully balancing student-teacher distributions may be more beneficial than aggressively filtering or reweighting teacher outputs in uncertainty-aware distillation.

In summary, our work builds on PromptKD by introducing uncertainty-aware techniques that further enhance knowledge distillation for generative language models. Our findings suggest that incorporating confidence information—either explicitly via entropy or implicitly via distribution shaping—can significantly improve the effectiveness of prompt-based distillation.

2 Related Work

2.1 Knowledge Distillation

Knowledge Distillation (KD) [2] is a widely used technique for transferring knowledge from a large teacher model to a smaller student model. It has shown effectiveness in compressing models for both classification and generative tasks. Traditional KD methods assume that the teacher's outputs are always reliable, which may not hold true in open-ended generative settings.

2.2 Prompt Tuning and PromptKD

Prompt tuning [3] allows efficient adaptation of language models through soft prompts while freezing the backbone. PromptKD [1] combines this idea with KD, using soft prompts as a mechanism to transfer knowledge from the teacher. While PromptKD improves computational efficiency and reduces exposure bias, it does not account for the uncertainty in teacher outputs.

2.3 Uncertainty-Aware Distillation

Recent works have introduced uncertainty-aware distillation methods to address unreliable or noisy teacher signals. Entropy-based loss weighting [4] and top-k filtering [5] help prioritize confident outputs, while soft blending [6] mitigates optimization instability by mixing teacher and student distributions. Our work adopts and extends these ideas into the prompt-based distillation setting, evaluating their effectiveness in instruction-following generation.

3 Approach

3.1 PromptKD Framework

PromptKD trains a student generative model to mimic the behavior of a teacher through a three-stage process that combines prompt tuning and knowledge distillation.

1. **Pseudo-Target Generation**: The student model receives an input request x and generates a response y through autoregressive decoding. This synthetic output serves as a pseudo-target to mitigate exposure bias, ensuring that the teacher is trained on the types of responses the student is likely to produce.
2. **Prompt Tuning for Adaptive Teaching**: The teacher model, augmented with a learnable soft prompt P, receives the student-generated (x, y) pair as input. The prompt is updated to minimize the KL divergence between the teacher and student distributions on the response, encouraging the teacher to produce more student-aligned outputs. An additional regularization loss $\mathcal{L}_{\text{reg}}$ is applied to prevent divergence from the teacher's original distribution.
3. **Student-Friendly Knowledge Distillation**: Using the updated prompt P, the teacher generates a response distribution over y, which is then used to supervise the student. The student is trained to match this distribution by minimizing the Kullback-Leibler (KL) divergence:

$$\mathcal{L}_{\text{KD}} = \sum_{t \in T} \text{KL}\left(p_t^{\text{teacher}} \parallel q_t\right) \tag{1}$$

   where q_t denotes the student's output distribution at time step t, and p_t^{teacher} represents the teacher's output distribution conditioned on the updated prompt. This objective encourages the student to closely mimic the teacher's behavior across the sequence.

This multi-stage design enables PromptKD to not only transfer knowledge effectively but also adapt teacher behavior based on student predictions—resulting in better alignment, reduced exposure bias, and more robust instruction-following performance.

3.2 Uncertainty-Aware Strategies for PromptKD

To improve the robustness and reliability of PromptKD in the face of noisy or uncertain teacher outputs, we propose three complementary uncertainty-aware strategies.

3.2.1 Entropy-Based Loss Weighting

Teacher outputs with high entropy may reflect ambiguity or low confidence. To down-weight such uncertain predictions, we compute a normalized entropy score w_t for each time step t:

$$H(p_t^{\text{teacher}}) = -\sum_i p_{t,i}^{\text{teacher}} \log p_{t,i}^{\text{teacher}}$$

$$w_t = 1 - \frac{H(p_t^{\text{teacher}})}{\log |V|}$$

where $|V|$ is the vocabulary size. The final weighted KL loss becomes:

$$\mathcal{L}_{\text{entropy}} = \sum_{t \in \mathcal{T}} w_t \cdot \text{KL}(p_t^{\text{teacher}} \| q_t)$$

This approach prioritizes confident supervision from the teacher while reducing the influence of high-entropy outputs.

3.2.2 Soft Blending (Fixed Alpha)

In soft blending, the training target is a weighted average of teacher and student output distributions:

$$p_t^{\text{blend}} = \alpha \cdot p_t^{\text{teacher}} + (1 - \alpha) \cdot p_t^{\text{student}}$$

$$\mathcal{L}_{\text{soft}} = \sum_{t \in \mathcal{T}} \text{KL}(p_t^{\text{blend}} \| q_t)$$

where $\alpha = 0.7$ is fixed and $\mathcal{T}$ is the set of response tokens.

3.2.3 Dynamic Blending (Adaptive Alpha)

Dynamic blending adapts α during training:

$$\alpha_{\text{step}} = \max(0.5, 0.9 - 0.4 \cdot \text{step/total\_steps})$$

$$p_t^{\text{blend}} = \alpha_{\text{step}} \cdot p_t^{\text{teacher}} + (1 - \alpha_{\text{step}}) \cdot p_t^{\text{student}}$$

This allows the student to rely more on the teacher early on and gradually transition to its own predictions.

Strategy	Target Distribution	Alpha Schedule	Robustness
Entropy Loss Weighting	$w_t \cdot \text{KL}(p_T \| q)$	N/A	High
Soft Blending	$\alpha p_T + (1-\alpha) p_S$	Fixed ($\alpha = 0.7$)	Moderate
Dynamic Blending	$\alpha_{\text{step}} p_T + (1-\alpha_{\text{step}}) p_S$	Decaying	High

Table 1: Summary of uncertainty-aware strategies for PromptKD.

3.2.4 Teacher Assistant Filtering

While blending strategies focus on the integration of teacher and student distributions, TA (Teacher Assistant) filtering serves as a complementary mechanism to refine the transferred knowledge. Inspired by prior works on token-level distillation and output filtering, we apply a top-k and top-p sampling-based mask to the teacher's logits:

- Retain only top-k highest probability tokens per timestep (e.g., $k = 50$).
- Optionally apply top-p nucleus filtering ($p = 0.95$).
- If filtered logits become degenerate (e.g., empty or near-zero sum), fallback to temperature-scaled softmax.

This filtering ensures that only confident, informative tokens are used for distillation, reducing the impact of noisy teacher outputs.

4 Experiments

4.1 Data

We use the **databricks-dolly-15k** dataset[1], a publicly available instruction-following dataset containing 15,000 examples curated for training and evaluating large language models. Each data point consists of an `instruction`, an optional `input`, and a reference `response` written by a human.

To fit our Prompt-based Knowledge Distillation (PromptKD) framework, we preprocess each example into the following standardized input format:

```
User:  <instruction and input>
Context:  <additional context if available>
Assistant:
```

For example:

```
User:  Given this paragraph about video game consoles, what
was the first console?
Context:  The history of video game consoles, both home and
handheld, had their origins in the 1970s.  The concept of
home consoles used to play games on a television set was
founded by the 1972 Magnavox Odyssey, first conceived by
Ralph H. Baer in 1966.  Handheld consoles bore out from
electro-mechanical games that had used mechanical controls and
```

[1]`https://huggingface.co/datasets/databricks/databricks-dolly-15k`

```
light-emitting diodes (LED) as visual indicators.  Handheld
electronic games had replaced the mechanical controls with
electronic and digital components, and with the introduction
of Liquid-crystal display (LCD) to create video-like screens
with programmable pixels, systems like the Microvision and the
Game & Watch became the first handheld video game consoles,
and fully realized by the Game Boy system.
Assistant:
```

This format ensures that the model clearly distinguishes between the user query, contextual information, and the expected assistant response.

We divide the dataset into 14,000 training examples, 500 validation examples, and 500 test examples. All inputs are tokenized using the GPT-2 tokenizer with a maximum sequence length of 512 tokens. Sequences exceeding the limit are truncated, and shorter ones are padded accordingly.

This dataset is particularly suited for evaluating instruction-following capabilities of models and enables effective testing of different distillation strategies in a realistic setting.

4.2 Evaluation method

In our evaluation, we filter out examples where the ground-truth response simply echoes the instruction to avoid trivial matching. Each sample is pre-processed into a standardized format that combines instruction, optional context, and assistant prompt. We use greedy decoding (i.e., `do_sample=False`) with a maximum of 32 newly generated tokens to ensure deterministic outputs.

The evaluation set consists of 500 held-out examples (indices 14500–14999) from the databricks-dolly-15k dataset. We compute the following automatic metrics:

- **Exact Match (EM)**: Measures strict string match between model output and reference.
- **ROUGE-L**: via a custom implementation measuring the longest common subsequence between generated and reference outputs. ROUGE-L captures sequence-level similarity and is particularly useful when evaluating coherence and word order, which is essential in instruction-following tasks.
- **BERTScore**: computed using the `bert-score` package (F1 score) with English pre-trained models to assess semantic similarity. Unlike lexical metrics, BERTScore compares contextual embeddings to evaluate whether the generated text conveys similar meaning, even when using different words. This is particularly valuable in open-ended generation tasks where diversity is acceptable.

4.3 Experimental details

The student model and the teacher model are initialized from GPT-2-Medium and GPT-2-Large, respectively, and both are fine-tuned separately on the `databricks-dolly-15k` dataset using instruction-style supervised learning. This fine-tuning step is designed to adapt each model to the task of generating prompt-conditioned responses in downstream applications.

Fine-tuning for both models is conducted with a batch size of 2 and gradient accumulation over 4 steps, resulting in an effective batch size of 8. This configuration allows for stable training under limited GPU memory. After fine-tuning, the teacher model remains frozen and is used only during the distillation phase to provide pseudo-labels and compute KL divergence, while the student model continues to be updated.

We conduct all experiments using the `databricks-dolly-15k` dataset, filtered to remove echoing responses. The dataset is split into 14,000 training, 500 validation, and 500 test samples. A soft prompt of 7 tokens is prepended to each input sequence during training and inference.

During the distillation stage, pseudo-labels are generated by the student model using sampling-based decoding with top-k, top-p, and temperature control. The generated tokens (excluding the prompt) are fed into the frozen teacher model, with the same soft prompt prepended, to compute KL divergence. Only the student model and the teacher's soft prompt are updated during this stage. For validation, we use greedy decoding to ensure deterministic comparison.

All experiments were conducted in a Google Colab environment equipped with an A100 GPU, using the PyTorch and Hugging Face Transformers frameworks.

Hyperparameter	**Value (applied to both student and teacher)**
Dataset	`databricks-dolly-15k`
Instruction style	Yes
Batch size	2
Gradient accumulation	4
Effective batch size	8
Learning rate	1×10^{-5}
Epochs	3
Max input length	512 tokens
Prompt masking	Yes (label = -100)
Optimizer	AdamW
Scheduler	Linear with warmup

Table 2: Fine-tuning hyperparameters shared by both models. Student: GPT-2-Medium; Teacher: GPT-2-Large.

Hyperparameter	**Value**
Batch size	4
Learning rate	1×10^{-5}
Epochs	3
Soft prompt length	7 tokens
Max input length	512 tokens
Max new tokens	32
KL loss scaling	0.5
Gradient clipping norm	0.5
Top-k(sampling)	0
Top-p(sampling)	1.0
Temperature(sampling)	1.0
Optimizer	AdamW

Table 3: Training hyperparameters used in our experiments.

4.4 Results

Method	**EM**	**ROUGE-L**	**BERTScore**
Baseline	1.60	23.06	86.19
Soft Blending	**2.40**	**24.88**	**85.04**
Dynamic Blending	**2.40**	**24.86**	**85.04**
Loss Weighting	0.80	14.72	75.96
TA Filtering (top_k=200, top_p=0.9)	1.00	21.04	83.23

Table 4: Evaluation results on Exact Match (EM), ROUGE-L, and BERTScore.

Uncertainty-aware strategies, such as loss weighting and TA filtering, performed worse than the baseline on several metrics. For example, loss weighting showed a significant drop in both ROUGE-L and BERTScore, and TA filtering similarly underperformed in EM and ROUGE-L. These results suggest that simply incorporating uncertainty—without careful calibration—can hurt performance, possibly by suppressing useful teacher signals or disrupting gradient flow in generative tasks.

5 Analysis

5.1 Blending

Performance over PromptKD Baseline. Both blending strategies outperform the PromptKD baseline across key metrics. Exact Match (EM) improves from 1.60 to 2.40, and ROUGE-L increases from 23.06 to 24.88 (Soft) and 24.86 (Dynamic), indicating better surface-level alignment. Although BERTScore slightly decreases compared to the baseline (from 86.19 to 85.04), it remains within a reasonable range—suggesting that semantic fidelity is preserved.

Soft vs. Dynamic Blending. Soft Blending shows a slight advantage in ROUGE-L, whereas Exact Match and BERTScore remain virtually identical between the two methods. This implies that both strategies maintain semantic similarity, with Soft Blending exhibiting slightly better token-level overlap. Dynamic Blending, however, offers architectural flexibility by allowing the blending ratio to adapt to contextual uncertainty, making it a more robust candidate for generalization in diverse scenarios.

Interpretation. Blending-based approaches offer a balanced alternative to strict teacher imitation. While both strategies provide marginal yet consistent improvements over the baseline, Dynamic Blending stands out for its adaptability and potential for future extension in uncertainty-aware distillation frameworks.

5.2 Entropy-based Loss weighting

Despite its intuitive design, the entropy-based loss weighting strategy underperforms compared to the PromptKD baseline. As shown in Table 4, the model trained with loss weighting achieves a lower ROUGE-L (14.72 vs. 23.06), Exact Match (0.80 vs. 1.60), and BERTScore (75.96 vs. 86.19), indicating degraded performance in both lexical overlap and semantic alignment.

This suggests that naively scaling the distillation loss by entropy-derived weights may suppress meaningful learning signals. While the goal was to emphasize confident teacher predictions and downweight uncertain ones, this approach may inadvertently:

(1) Undermine gradient contributions from less confident—but still informative—tokens, (2) Overfit to high-confidence predictions, which may not generalize well, (3) Disrupt the balance between exploration and exploitation during generation.

Furthermore, the entropy weights are computed independently for each token, which can lead to noisy or unstable gradients—particularly in long-form generation, where local uncertainty may not correlate with global response quality.

In summary, although the technique aims to introduce uncertainty awareness, the lack of calibration or adaptive control may explain the performance degradation. Future work might explore smoothed entropy weighting, token-level filtering, or entropy-informed curriculum learning as alternative means of leveraging uncertainty more effectively in generative KD.

5.3 TA Filtering

Quantitative Interpretation . Across the three evaluation axes-Exact-Match (EM), ROUGE-L, and BERTScore-F1-the proposed *TA-Filtering* variant with the conservative setting *(top-k = 200, top-p = 0.90)* displays a profile that is almost indistinguishable, in semantic terms, from the PromptKD baseline while reducing verbatim copy-rate by $\tilde{0}.6$ pp. Concretely, EM remains below 2 % in all conditions (1.60 % $\rightarrow$ 1.00 %), which is typical for open-ended generation. Nevertheless, the model retains 21.04 ROUGE-L and 0.832 BERTScore-F1, only -2 pt and -0.03 pt relative to the baseline (23.06 / 0.861). Hence, the filtered student sacrifices a small amount of token-level overlap while maintaining almost the same contextual semantic fidelity-a desirable trade-off for applications valuing paraphrastic variety.

Filtering Schedule Study . Table 5 (rows TA-A $\rightarrow$ TA-B $\rightarrow$ TA-C) isolates the impact of increasing distributional sparsity: A step from k = 200/0.90 $\rightarrow$ 100/0.95 reduces EM by a further 0.4 pp but incurs almost 3× larger ROUGE-L loss (-6.92 vs. -2.02) and doubles the semantic degradation (-5.89

vs. -2.96). The trend continues when the *top_k* constraint is removed (*top_p* = 0.98), confirming that sparsity yields diminishing returns: beyond a moderate threshold, additional pruning erodes lexical recall far faster than it improves diversity, while semantic quality plateaus. These observations motivate selecting $k{\approx}200$, $p{\approx}0.90$ as a sweet-spot that preserves meaning yet meaningfully lowers the strict copy ratio.

top_k/top_p	EM (pp)	ROUGE-L	BERTScore
200 / 0.90	-0.60	-2.02	-2.96
100 / 0.95	-1.00	-6.92	-5.89
- / 0.98	-0.40	-9.21	-5.32

Table 5: Metrics under Varying TA-Filtering Strengths (relative to PromptKD baseline)

Actionable Recommendations. 1) Use the 200/0.90 configuration as a default when semantic fidelity is paramount but verbatim overlap must be curtailed. 2) If higher ROUGE is required, re-introduce a light min-length constraint or combine the KL objective with a small cross-entropy term to recover n-gram recall without affecting semantic similarity.

Summary. Among the TA-Filtering variants, the configuration $(k = 200,\ p = 0.90)$ yielded the most stable performance. It achieved the highest token-level overlap (ROUGE-L = 21.04) and the strongest semantic similarity (BERTScore F1 = 83.23), indicating that a moderately conservative mask minimizes information loss while still enhancing distillation.

In contrast, a more aggressive mask $(top\text{-}k = 100,\ top\text{-}p = 0.95)$ pruned away informative logits along with noise, resulting in a significant drop in all metrics. Conversely, an overly loose setting $(top\text{-}k = \text{None},\ top\text{-}p = 0.98)$ retained low-probability noise tokens, degrading the teacher signal and reducing downstream quality.

Although all three TA-Filtering settings underperformed the original PromptKD baseline, the performance gap appears to stem from an imbalance between noise removal and information retention. These findings suggest that a carefully tuned $(top\_k,\ top\_p)$ pair—or an adaptive sparsity schedule—has the potential to surpass the baseline, offering a practical avenue for further gains.

6 Conclusion

In this study, we explored knowledge distillation strategies for generative language models, focusing on blending-based and uncertainty-aware methods. Our experiments show that Soft Blending and Dynamic Blending consistently outperform the PromptKD baseline in ROUGE-L and exact match accuracy, while preserving comparable semantic fidelity (BERTScore). Dynamic Blending is especially notable for its ability to adapt blending ratios based on contextual uncertainty, offering a flexible architecture well-suited to diverse NLP tasks.

In contrast, entropy-based loss weighting underperforms across all metrics, suggesting that naïve token-level uncertainty modeling may suppress useful gradients or overfit to confident predictions. Meanwhile, TA Filtering, particularly with a conservative top-k and top-p setting, reduces verbatim copying without sacrificing semantic alignment—making it ideal for applications where paraphrastic diversity is preferred.

These results highlight the value of moving beyond static teacher imitation. The adaptability of Dynamic Blending and the selective control of TA Filtering point to promising directions for uncertainty-aware distillation frameworks. On the other hand, the shortcomings of entropy-based weighting underscore the need for more nuanced uncertainty modeling.

Overall, this work validates the feasibility of designing distillation strategies that balance structural adaptability with semantic preservation. As future work, we plan to extend these approaches to larger models and tasks, refine uncertainty measures, and explore hybrid methods that unify blending and filtering. This study lays the groundwork for more robust, interpretable, and controllable training pipelines in generative NLP.

References

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

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

[3] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

[4] Wenchong He, Zhe Jiang, Tingsong Xiao, Zelin Xu, and Yukun Li. A survey on uncertainty quantification methods for deep learning. *arXiv preprint arXiv:2302.13425*, 2024.

[5] Peyman Passban, Yimeng Wu, Mehdi Rezagholizadeh, and Qun Liu. Alp-kd: Attention-based layer projection for knowledge distillation. *arXiv preprint arXiv:2012.14022*, 2020.

[6] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021.

Team Contribution
Hyeonwoo Baek: conducted loss weighting experiments and compiled the results into a comprehensive report.
Chanhong Min: Conceived the main idea of the proposed method and developed the TA Filtering implementation.
Hyolim Son: designed the baseline model and TA Filtering method with altering the values of top$_k andtop_p$
$SeunghyunLee$: $designedtheSoftBlendingmethodvia\alpha$-weighted teacher-student logit interpolation under prompt conditioning.