Decoding with Limited Teacher Supervision Requires Understanding When to Trust the Teacher

Hyunjong Ok, Jegwang Ryu, Jaeho Lee*
Pohang University of Science and Technology (POSTECH)
hyunjong.ok@gmail.com, {jegwang.ryu, jaeho.lee}@postech.ac.kr

Abstract

How can small-scale large language models (LLMs) efficiently utilize the supervision of LLMs to improve their generative quality? This question has been well studied in scenarios where there is no restriction on the number of LLM supervisions one can use, giving birth to many decoding algorithms that utilize supervision without further training. However, it is still unclear what is an effective strategy under the *limited supervision* scenario, where we assume that no more than a few tokens can be generated by LLMs. To this end, we develop an algorithm to effectively aggregate the small-scale LLM and LLM predictions on initial tokens so that the generated tokens can more accurately condition the subsequent token generation by small-scale LLM only. Critically, we find that it is essential to adaptively overtrust or disregard the LLM prediction based on the confidence of the small-scale LLM. Through our experiments on a wide range of models and datasets, we demonstrate that our method provides a consistent improvement over conventional decoding strategies.

Code: https://github.com/HJ-Ok/DecLimSup

1 Introduction

Large language models (LLMs) have demonstrated their tremendous capability to generate human-like text sentences that convey rich knowledge in various problem domains (Achiam et al., 2023; Bai et al., 2022; Team et al., 2023; Touvron et al., 2023). However, due to their gigantic model scale and autoregressive nature based on next-token generation, LLMs often suffer from having a significantly high latency (Xu and McAuley, 2023).

Small-scale LLMs, called sLLMs, have thus garnered much attention (Gunasekar et al., 2023; Li et al., 2023b; Abdin et al., 2024). sLLMs can run much faster than LLMs, making them a promising alternative to LLMs for applications that require agile responses or on-device inference. However, sLLMs tend to perform clearly worse than their larger counterparts, especially for tasks that require in-depth reasoning (Biderman et al., 2023).

If we have both LLM and sLLM available, can we use both models to enjoy the quality of LLM and the speed of sLLM? A recent line of works shows that this is possible, even without any further training, by having two models collaborate on the decoding procedure: In speculative decoding (Leviathan et al., 2023; Xia et al., 2023), the candidate text is generated rapidly by the sLLM, which is then verified by the LLM to ensure its correctness. This method admits parallel inference, enabling a rapid generation of LLM-level responses. However, this approach requires a lot of memory to load multiple instances of LLMs and sLLMs, and thus, it is very difficult to use on edge devices.

In this paper, we consider an alternative decoding scenario with *limited LLM supervision*. That is, we assume that one primarily decodes with sLLM (the student) but can utilize a very limited number of supervisions from the LLM (the teacher). The supervision may be as scarce as generating a single token, incorporating the scenario where the LLM is remotely located, *e.g.*, at a central server. Critically, we no longer restrict the model to generate the exact same outcome as the LLM, but simply aim to maximize the generative quality itself.

A natural approach is to use two models to predict the same token and aggregate their predictions to generate a better token. Such an approach has been recently studied by Li et al. (2023a), without any limitation on the number of LLM supervision. Here, it turns out that an effective strategy is to *overtrust* the teacher and negatively account for the student. Precisely, one takes a weighted sum of teacher and student logits with weights $1 + \alpha$ and $-\alpha$, respectively. It has been observed that a single positive $\alpha = 0.5$ works consistently well

* corresponding author

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 12460–12476
November 12-16, 2024 ©2024 Association for Computational Linguistics

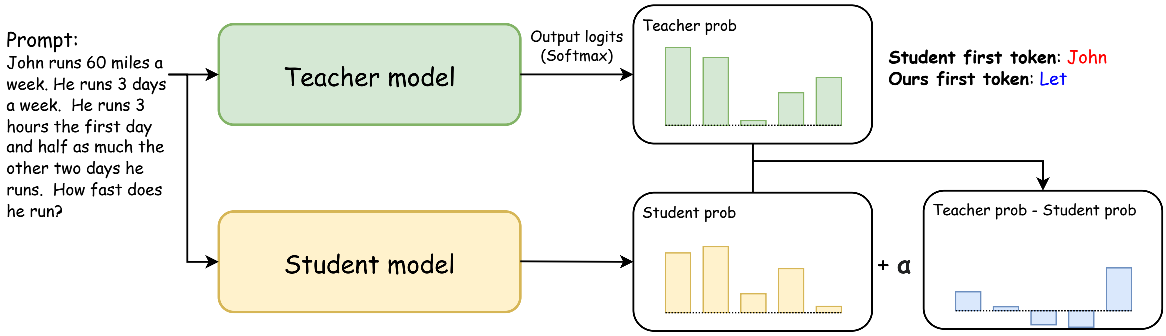

Figure 1: Illustration of our methodology. As the parameter α increases, the methodology leverages a more significant disparity in knowledge between the teacher and the student models. The example shows a modification in the initial generated word from 'John' to 'Let,' which allows the sentence to generate the correct answer when subsequently generated with the student model.

over diverse scenarios (O'Brien and Lewis, 2023).

Under the limited supervision scenario, however, we make a critical observation that overtrusting the teacher no longer continues to be the dominant strategy. Intriguingly, we find that overtrusting the student works much better under certain setups, enjoying a better alignment with subsequently generated tokens where no supervision is available. In fact, our empirical analysis reveals that *who should we overtrust by how much* highly varies over the choice of models and tasks, and even for each datum. In other words, for supervision-limited scenarios, we are in desperate need of a mechanism to determine whom to overtrust by how much.

To this end, we develop an algorithm to utilize the LLM supervision in an adaptive manner to improve sLLM performance. In particular, we identify that the entropy (*i.e.*, confidence) of the generated tokens is highly correlated with whether one should trust the teacher or the student. If the student's generated token has high or low entropy, overtrusting the student works better. This implies if the student is confident, the teacher's information could be the noise, and if the student ponders generating a token (*i.e.*, high entropy), the teacher's information can cause confusion. Our method predicts the optimal α on a per-datum basis, consistently improving the predictive performance of sLLM over a wide range of models and tasks.

Our key contributions are threefold:

- We formulate and initiate research toward sLLM decoding with limited LLM supervision, which bears much practical importance.
- We discover that, with limited supervision, the conventional strategies of overtrusting LLMs are largely suboptimal.
- We propose a novel entropy-based mechanism to determine who to overtrust by how much among sLLM and LLM and demonstrate its effectiveness on a wide range of setups.

2 Framework

We now formally describe the problem of decoding with limited teacher supervision and the overtrust-based framework to aggregate teacher and student predictions. Similar to recent works (Leviathan et al., 2023; Li et al., 2023a) and unlike distillation (Hinton et al., 2015), we do not assume that we can train the sLLM further.

We consider a setup where we have two models available: The teacher LLM and the student sLLM. Given some input prompt, we assume that we can invoke the teacher up to N times and the student unlimited times to generate subsequent tokens.

For decoding, we consider aggregating the predictions of the teacher and student at the token-level. Concretely, let $f_s(x)$, $f_t(x)$ be the prediction logits of the student and teacher. Then, we consider a prediction based on the aggregated softmax

$$S_\alpha = \sigma(f_s(x)) + \alpha(\sigma(f_t(x)) - \sigma(f_s(x))) \quad (1)$$

where σ denotes the softmax function and $\alpha \in \mathbb{R}$ is a tunable parameter that determines which model should be trusted. If $\alpha = 1$, we are following the teacher's prediction, and if $\alpha = 0$, we are using the student prediction. Contrastive decoding (Li et al., 2023a) uses $\alpha > 1$, which is to overtrust the teacher and disregard the student.[1] In any case, the token is generated as an output achieving the maximum aggregated softmax S_α (Fig. 1).

[1] Note that the contrastive decoding actually combines logits instead of softmax. We use softmax for our cases since it empirically works better.

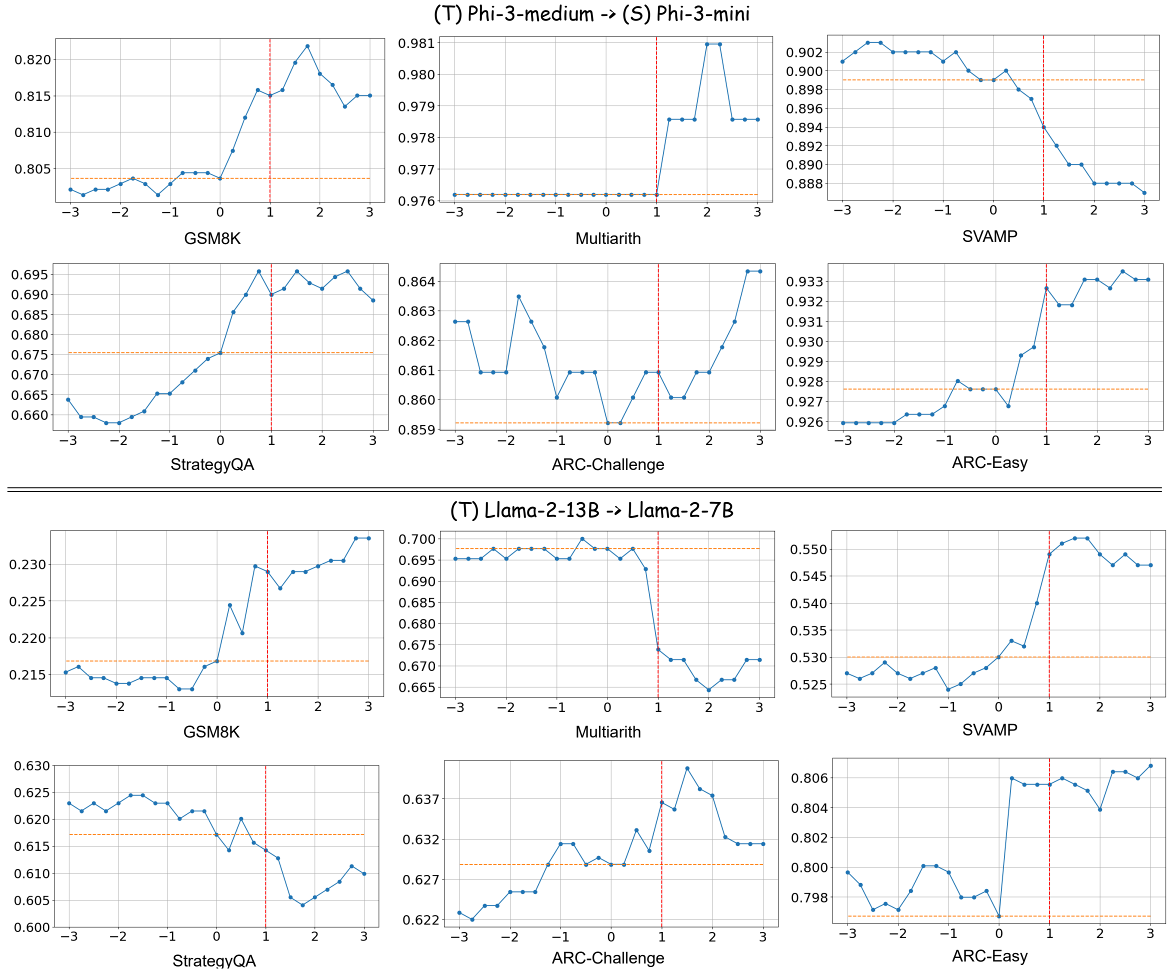

Figure 2: Visualization of accuracy as a function of α. The red dashed line indicates α=1, and the orange line represents the student model's baseline performance.

As the number of teacher supervisions is limited, we make combined predictions (eq. 1) only for the first N tokens. That is, the text consists of

$$(\underbrace{p_1, \cdots, p_m}_{\text{given prompt}}, \underbrace{t_1, \cdots, t_N}_{\text{LLM + sLLM}}, \underbrace{t_{N+1}, \cdots}_{\text{sLLM only}}). \qquad (2)$$

3 Method

Given this framework, we identify two core algorithmic questions. First, what is the optimal value of α? Second, given some prompts, how can we determine whether we should use the teacher's knowledge or not, specifically for the datum?

3.1 Key Observations

We now conduct a systematic empirical analysis to answer these questions.

(1) Strange case of $N = 1$. Under a wide range of setups, we have empirically evaluated the case of $N = 1$ with varying *trust hyperparameter* α (Fig. 2). We make two intriguing observations.

- Even a *single-token supervision* from LLM can ***boost the accuracy substantially***; in Phi3-mini evaluated on the StrategyQA dataset, the accuracy increases by 2%p.

- The *optimal α significantly differs* from task to task and model to model. Surprisingly, there are certain cases where the optimal α is smaller than zero, *i.e.*, overtrusting the student works better. Detailed case studies are in Appendix A. This contrasts with the case of unlimited supervision, where $\alpha = 1.5$ works consistently well (O'Brien and Lewis, 2023).

Summing up these observations, we conclude that we are in need of a good mechanism to predict the optimal α rather than using a fixed value.

(2) Entropy and supervision. We also analyze the relationship between the prediction entropy of the student and the effect of teacher supervision (Fig. 3), where we use the optimal α approximated

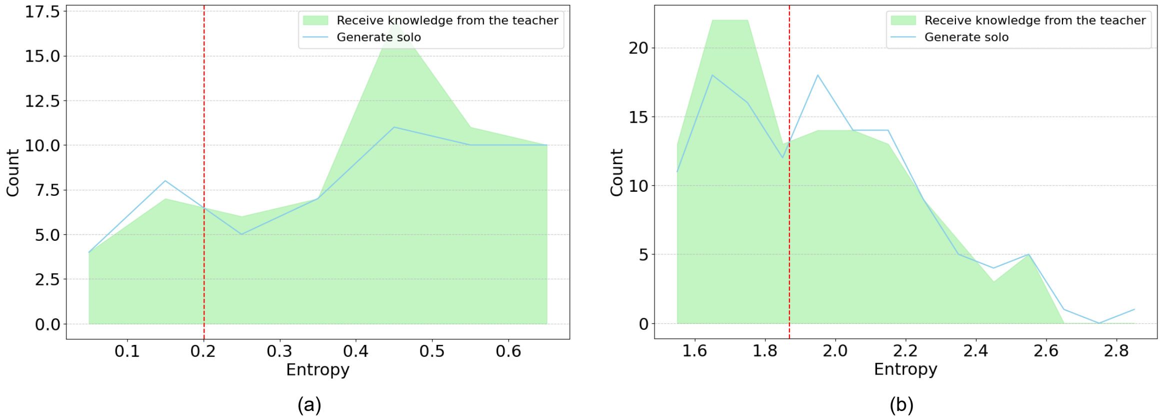

Figure 3: Visualization of the number of correct answers performed by “Receive knowledge from the teacher” and “Generate solo” based on student entropy values utilizing Llama-2 on GSM8K. (a) represents cases with low entropy, while (b) shows cases with high entropy. The red dashed line indicates the threshold beyond which “Generate solo” demonstrates superior performance.

from the previous analysis.

We observe that there exists certain *interval of entropy* that whenever the student prediction entropy lies inside the interval, aggregating teacher predictions are likely to boost performance. If the student is extremely confident, the teacher’s prediction will only add noise, leading to a lower accuracy. If the student is extremely unsure, aggregating teacher knowledge may disrupt the student’s careful consideration and degrade the performance.

3.2 Algorithmic implications

Motivated by these observations, we develop two mechanisms to best incorporate the limited supervision from the teacher LLM.

Predicting optimal α for each datum. To avoid performing an extensive search for the optimal α for each data point, we propose to train a predictor that estimates the optimal α. More specifically, the model predicts α based on the teacher and student logits given as inputs (Fig. 4). We compare two different models, XGBoost and DNN.

Entropy-based knowledge injection. We determine whether to utilize the teacher supervision or not based on an interval-based classifier using the student classification entropy (Algorithm 1).

4 Experimental Setup

To demonstrate the superiority of our method, we evaluate our method on two types of tasks: classification tasks over multiple domains (section 4.1) and various LLM benchmarks (section 4.2). In section 4.3, we describe the implementation details of our algorithm. Any detail that does not appear in this section is given in Appendix B.

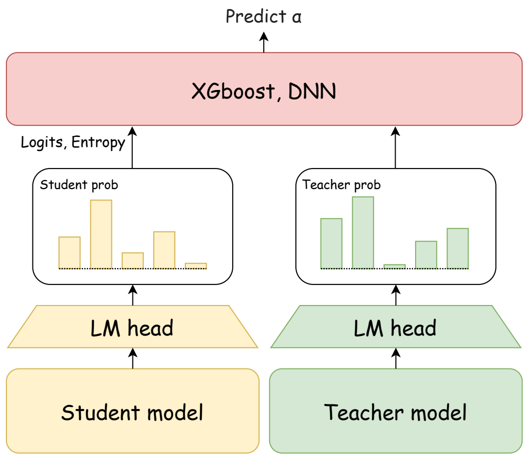

Figure 4: Illustration of the optimal α predict module.

4.1 Classification Task

For the classification task, we first fine-tuned each student and teacher model. Then, we predict the class based on this equation 1 in inference time.

Datasets. For the classification task, we use three datasets: For image, we use the CIFAR-100 (Krizhevsky et al., 2009); for audio, we use the ESC-50 (Piczak, 2015); for text, we use the MNLI (Williams et al., 2018) dataset.

Models. For CIFAR-100, we use the DeiT (Touvron et al., 2021). For ESC-50, we use the AST (Gong et al., 2021). For MNLI, we use the DeBERTa (He et al., 2020). Details about teacher

Algorithm 1 Entropy-based knowledge injection

Require: Entropy of the generated token E, two thresholds T_1 and T_2 ($T_1 < T_2$)
  if $T_1 < E < T_2$ **then**
    Receive knowledge from the teacher
  else if $E \leq T_1$ or $T_2 \leq E$ **then**
    Generate solo
  end if

and student models exist in Appendix B.1.

Other details. For CIFAR-100 and MNLI, we report the average score from 5 runs with different random seeds for each setting. Experiments are done on both Intel Gaudi 2 or NVIDIA A6000, and we report the cross-validation score for ESC-50 following Chen et al. (2022).

4.2 LLM Benchmark

Datasets. For LLM benchmarks, we evaluate a total of six different tasks. In particular, we use the GSM8K (Cobbe et al., 2021), Multi-arith (Koncel-Kedziorski et al., 2016), SVAMP (Patel et al., 2021), partition of MATH (MATH-Easy), ARC (Clark et al., 2018), and StrategyQA (Geva et al., 2021) datasets. Further details are available in the Appendix B.3.

Models. We have tested over a total of four different families of models: Phi-3 (Abdin et al., 2024), Llama-2 (Touvron et al., 2023), Vicuna (Zheng et al., 2024), and Qwen (Bai et al., 2023). Detailed information on teacher and student models exists in Appendix B.1.

Decoding. To confirm the exact performance measurement, we do not apply sampling algorithms (Fan et al., 2018; Holtzman et al., 2020) during the decoding process.

Prompt. We have provided few-shot prompts, with the number of shots ranging from 2 to 8. The prompts end with the phrase 'the answer is.' Then, we extract that span and compare it with the ground truth answer. More detailed information about the prompts is given in Appendix D.

Hardware. The experiments have mainly taken place on the Intel Gaudi 2, NVIDIA A100, and H100. The results have been cross-validated over devices. The model inference has been performed using the bfloat16 format.

Methods	CIFAR-100	ESC-50	MNLI-m	MNLI-mm
Student	82.71 ± 0.24	77.80 ± 2.75	90.96 ± 0.07	90.84 ± 0.11
Teacher	89.47 ± 0.16	94.55 ± 0.84	91.60 ± 0.06	91.58 ± 0.13
Ours	$\mathbf{89.75} \pm 0.15$	$\mathbf{94.75} \pm 0.98$	$\mathbf{91.91} \pm 0.10$	$\mathbf{91.99} \pm 0.19$

Table 1: Experimental results on the classification task

Student Model	Student acc.	Teacher acc.	Ours acc.	Optimal α
Distill-BERT	82.08 ± 0.15	91.60 ± 0.06	91.75 ± 0.04	0.75 ± 0.13
DeBERTa-v3-small	87.76 ± 0.11	91.60 ± 0.06	91.81 ± 0.12	0.73 ± 0.12
DeBERTa-large	90.96 ± 0.07	91.60 ± 0.06	91.91 ± 0.10	0.54 ± 0.03

Table 2: Experimental results on different students for optimal α ablation study on MNLI matched dataset.

Other details. We search for the optimal α by exploring the range from 3 to -3 with 0.25 intervals. Detailed information about the evaluation strategy of each benchmark is in Appendix B.3.2.

4.3 Implementing our Algorithm

Our model, designed to predict the optimal α, is initially conducted in inference across α ranging from 3 to -1 in intervals of 0.25 to make the training dataset. The dataset is constructed by labeling αs that correctly predict the answer as one and those that do not as zero. Then, we train the model to perform multi-label binary classification. Our implementation uses logits from both student and teacher models as inputs. Additionally, we have incorporated the entropy of each logit as further input. The α with the highest confidence is selected as the output during inference.

To demonstrate the superiority of our methodology, we mainly experiment with Phi-3, which shows the best performance on the GSM8K dataset. Since GSM8K is absent in the validation set, we split the training set into five cross-validation folds. Experiments are done on a single GeForce RTX 4090 GPU, and we conduct experiments using random seeds and report the average test score across these five folds.

5 Results

5.1 Classification task

We assess experimental results across various classification tasks within diverse domains, as detailed in Table 1. Our method consistently outperforms the teacher model across all classification tasks, indicating significant enhancements in model performance. Moreover, we analyze our method in the MNLI dataset using student models of varied performance and size, illustrated in Table 2. The table shows that even if a student model performs significantly worse than the teacher model, an appropriate

Models	GSM8K	Multiarith	SVAMP	MATH-Easy	StrategyQA	ARC-Challenge	ARC-Easy
Phi3-mini	80.36	97.62	89.90	63.51	67.54	85.92	92.76
w/ reference ($\alpha = 1$)	81.50	97.62	89.40	64.39	69.00	86.09	93.27
w/ $\alpha = 1.5$ (O'Brien and Lewis, 2023)	81.96	97.86	89.00	65.09	69.58	86.01	93.18
w/ optimal α	82.18	**98.10**	90.30	65.09	69.58	86.43	93.35
+ entropy sweet spot	**82.34**	**98.10**	**90.50**	**65.61**	**70.01**	**86.60**	**93.56**
Phi3-medium	90.83	98.33	92.60	72.81	77.00	89.42	95.33
Llama-2-7B	21.68	69.76	53.00	9.65	61.72	62.88	79.67
w/ reference ($\alpha = 1$)	22.90	67.38	54.90	9.83	61.43	63.65	80.56
w/ $\alpha = 1.5$ (O'Brien and Lewis, 2023)	22.90	67.14	55.20	9.65	60.55	64.08	80.56
w/ optimal α	23.35	70.00	55.20	10.00	62.45	64.08	80.68
+ entropy sweet spot	**24.03**	**70.48**	**56.10**	**10.53**	**62.59**	**64.59**	**80.81**
Llama-2-13B	37.38	86.67	60.8	12.63	67.69	67.15	80.51
Vicuna-7B	19.56	61.67	45.00	7.90	65.36	62.88	80.60
w/ reference ($\alpha = 1$)	21.08	60.71	45.80	7.90	64.77	62.29	80.77
w/ $\alpha = 1.5$ (O'Brien and Lewis, 2023)	20.32	60.48	46.30	7.54	65.50	62.12	80.35
w/ optimal α	21.08	62.38	46.60	8.42	65.65	63.91	80.98
+ entropy sweet spot	**21.38**	**63.81**	**46.90**	**8.77**	**66.23**	**64.42**	**81.36**
Vicuna-13B	36.01	85.71	55.8	12.11	66.08	69.54	83.46
Qwen-1.8B (4B on MATH-Easy)	35.03	83.10	34.50	27.54	58.22	50.00	70.41
w/ reference ($\alpha = 1$)	35.10	83.81	35.80	27.89	59.97	48.38	70.20
w/ $\alpha = 1.5$ (O'Brien and Lewis, 2023)	34.72	83.57	36.30	28.42	60.12	48.21	70.12
w/ optimal α	35.41	83.81	**37.10**	28.77	60.41	50.94	70.41
+ entropy sweet spot	**36.09**	**85.95**	**37.10**	**29.82**	**61.14**	**51.28**	**71.04**
Qwen-4B (7B on MATH-Easy & StrategyQA)	46.32	91.91	60.5	30.70	67.83	65.70	78.91

Table 3: Experiment results on various LLM benchmarks. The results are in case only the first generated token received knowledge from the teacher model. 'entropy sweet spot' is a method using Algorithm 1.

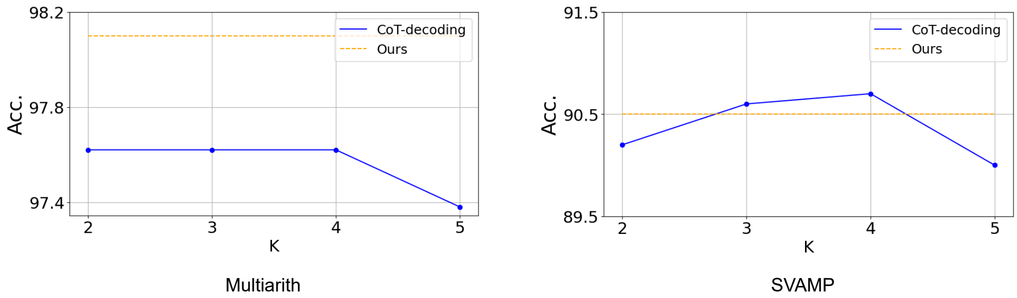

Figure 5: Results of comparison of our method with CoT-decoding using the Phi-3. K denotes counts of exploring paths starting from top-k.

mixture of their knowledge through an optimal α value can surpass the teacher's performance. Additionally, our results show that as the performance gap between the student and teacher narrows, the optimal α value converges to 0.5. This indicates a simple weighted average ensemble, as when there is minimal difference in knowledge between the students and teachers.

5.2 LLM benchmark

Overall results. We conduct vast experiments to evaluate the effectiveness of our proposed method across various LLMs and diverse benchmarks. The results of these experiments are in Table 3. Our approach is applied solely to the first token generated. As demonstrated by the results, directly utilizing the logits from a teacher model and utilizing the previous method (O'Brien and Lewis, 2023) can enhance performance; however, it is slight and occasionally leads to degraded performance. By optimizing the α parameter, we consistently achieve performance improvements over them across all models. Furthermore, we observe a significant enhancement in model performance by determining whether to employ our decoding method based on the entropy value.

Comparision with CoT-decoding. As the concept to reasonably determine the first token, our method is related to CoT-decoding (Wang and Zhou, 2024), a methodology that begins by selecting the top-k tokens from the initial token and subsequently generating sentences from each. The most confident response is then chosen as the final output. We compare our method to CoT-decoding using SVAMP and Multiarith, which do not require elaborate prompting, focusing on performance and

Models	GSM8K	Multiarith	SVAMP	MATH-Easy	StrategyQA	ARC-Challenge	ARC-Easy
Phi3-mini	80.36	97.62	89.90	63.51	67.54	85.92	92.76
N = 3	82.79	**98.10**	90.10	**66.67**	69.87	86.60	93.48
N = 5	82.34	**98.10**	90.10	66.32	70.31	**86.69**	**93.69**
N = 10	**83.32**	**98.10**	**90.30**	**66.67**	**72.20**	**86.69**	93.43
Llama-2-7B	21.68	69.76	53.00	9.65	61.72	62.88	79.67
N = 3	23.50	70.48	56.70	9.83	63.17	63.48	80.39
N = 5	23.43	**70.95**	**56.90**	**10.18**	63.46	**64.51**	**80.77**
N = 10	**24.56**	**70.95**	56.60	10.00	**64.05**	64.42	80.56
Vicuna-7B	19.56	61.67	45.00	7.90	65.36	62.88	80.60
N = 3	20.85	63.10	45.70	9.83	66.23	64.76	81.19
N = 5	20.70	64.05	46.20	**10.00**	**66.67**	64.68	**81.23**
N = 10	**21.91**	**64.76**	**47.00**	9.30	66.52	**65.78**	81.02
Qwen-1.8B (4B on MATH-Easy)	35.03	83.10	34.50	27.54	58.22	50.00	70.41
N = 3	**35.71**	83.81	39.30	29.47	**62.45**	51.45	70.79
N = 5	35.18	**84.05**	41.20	30.35	61.86	52.90	70.83
N = 10	35.63	83.57	**43.20**	**31.23**	61.72	**54.18**	**72.14**

Table 4: Experiment results on various LLM benchmarks with diverse N. We use optimal α for all experiments.

Methods	Seconds / Sentences		
	Total	Phi-3-medium	Phi-3-mini
Student Greedy	2.52	-	2.52
Teacher Greedy	5.20	5.20	-
CoT-decoding (K=3)	8.15	-	8.15
CoT-decoding (K=5)	14.65	-	14.65
Ours	2.63	0.08	2.55

Table 5: Comparative speed analysis of our method against CoT-Decoding. Our approach requires one additional token computation for teacher per sentence.

Methods	TOTAL FLOPs (TFLOPS)		
	Total	Phi-3-medium	Phi-3-mini
Speculative decoding	1309.32	1074.86	234.46
Ours	273.52	7.65	265.87

Table 6: Comparing FLOPs of our method against speculative decoding. For speculative decoding, we set the number of tokens to verify as four.

speed. The performance results are in Fig. 5, and the speed comparison is in Table 5. The performance of our method is competitive with CoT-decoding and even surpasses it in Multiarith. The speed calculated on the A100 GPU shows that our method is faster than CoT-decoding. Time cost in DNN is negligible in that it is smaller than 0.001 seconds per sentence.

Comparision with Speculative decoding. Our method shares similarities with speculative decoding in designing an efficient collaboration between small language models and LLMs. However, unlike speculative decoding, we implement a decoding strategy that operates under limited LLM supervision, significantly containing less computational cost. As illustrated in Table 6, the FLOPs comparison between speculative decoding and our proposed method evidences a substantial reduction in computational costs.

Models	GSM8K
Phi3-mini	80.36
w/ entropy sweet spot	82.34
w/ XGboost	81.44 ± 0.51
w/ DNN	**82.44** $\pm$ 0.22

Table 7: Experiment results on optimal α predictor.

Various tokens setting. We define the extent of collaborative decoding between the student and teacher models by notating 'N,' the number of tokens for collaboration. Then, we conduct diverse N values to assess the impact on performance, and the results are in Table 4. Moreover, we conduct our method on all tokens reported in Appendix C.3.

5.3 Optimal α prediction model

We evaluated our optimal α prediction model using the Phi-3 on the GSM8K, and the results in Table 7. While XGBoost failed to classify effectively, the DNN exceeded the performance of the entropy sweet spot. Demonstrates that the model can autonomously predict the appropriate knowledge without needing hand-crafted tuning processes. Additionally, we conducted more experiments for checking generalizability, such as results on unseen benchmarks and joint training, as detailed in Appendix C.1.

5.4 Case study

Table 8 provides example cases where only the first generated token receives knowledge from a teacher. As follows, receiving initial guidance from the teacher encourages the student to generate correct

[GSM8K] *John runs 60 miles a week. He runs 3 days a week. He runs 3 hours the first day and half as much the other two days he runs. How fast does he run?*
Ground truth answer: 10
Generate solo: John runs 3 hours the first day, so he runs 3/3 = 1 hour per day for the other two days. He runs a total of 3 + 1 + 1 = 5 hours a week. He runs 60 miles a week, so his speed is 60 miles / 5 hours = 12 miles per hour. The answer is 12.
Receiving knowledge from the teacher (w/ α = 1.5): Let's break this down step by step.
1. John runs 3 hours on the first day.
2. He runs half as much time on the other two days, so that's 3 / 2 = 1.5 hours per day.
3. In total, he runs for 3 + 1.5 + 1.5 = 6 hours a week.
4. He runs 60 miles a week, so to find his speed, we divide the total distance by the total time: 60 miles / 6 hours = 10 miles per hour.
John runs at a speed of 10 miles per hour. The answer is 10.
Receiving knowledge from the teacher (w/ optimal α): Let's break this down step by step.
1. John runs 3 hours on the first day.
2. He runs half as much time on the other two days, so that's 3 / 2 = 1.5 hours per day.
3. In total, he runs for 3 + 1.5 + 1.5 = 6 hours a week.
4. He runs 60 miles a week, so to find his speed, we divide the total distance by the total time: 60 miles / 6 hours = 10 miles per hour.
John runs at a speed of 10 miles per hour. The answer is 10.
Teacher generation: First, we need to find out how many hours John runs in total. He runs 3 hours the first day and half as much the other two days. So, he runs 3 + (3/2) + (3/2) = 3 + 1.5 + 1.5 = 6 hours in total. Now, we know he runs 60 miles in 6 hours. To find his speed, we divide the distance by the time. So, he runs 60 miles / 6 hours = 10 miles per hour. The answer is 10.

[StrategyQA] *Is Argon near Neon on the periodic table of elements?*
Ground truth answer: True (yes)
Generate solo: Argon is in period 3 and Neon is in period 2. Periods are horizontal rows on the periodic table. Thus, Argon is not near Neon on the periodic table. So the answer is no.
Receiving knowledge from the teacher (w/ α = 1.5): Argon is in period 3 and Neon is in period 2. Periods are horizontal rows on the periodic table. Thus, Argon is not near Neon on the periodic table. So the answer is no.
Receiving knowledge from the teacher (w/ optimal α): Yes. Argon is the 18th element on the periodic table. Neon is the 10th element on the periodic table. Thus, Argon is near Neon on the periodic table of elements. So the answer is yes.
Teacher generation: Argon and Neon are both noble gases. Noble gases are in the same column on the periodic table. Thus, Argon is near Neon on the periodic table. So the answer is yes.

Table 8: Detailed results showing how our method brought performance gain in LLM benchmarks utilizing Phi-3 model. Each generated tokens all in green; if the answer is correct we highlighted in blue, else red.

solutions. A detailed analysis of the first sample from the GSM8K dataset reveals that 'Receiving knowledge from the teacher' facilitates the subsequent student generation process by dramatically deviating from the initial tokens of both the student and the teacher. Notably, in the second example, our method begins by generating the correct answer "Yes," reflecting the fact that the teacher has a much higher probability of generating "Yes" than the student; this results in better conditioning of the subsequent tokens, leading to correct reasoning as a result. This example suggests why an α of 1.5 is not optimal, commonly used in previous research (O'Brien and Lewis, 2023).

6 Related work

Efficient decoding strategy. Speculative decoding (Xia et al., 2023; Leviathan et al., 2023; Zhou et al., 2024) enables faster generation by having the draft model generate first, which the target model then verifies in parallel, ensuring distributions identical to those generated by the target model. On the other hand, our method aims to efficiently enhance the student model's performance by utilizing only a few tokens generated by the teacher.

Logit arithmetic. Recent works suggest using arithmetic to harness the capabilities of two or more language models during decoding. Contrastive decoding (Li et al., 2023a; O'Brien and Lewis, 2023) enhances LLM generation quality by subtracting the log probabilities of a smaller model from those of the LLM. SafeDecoding (Xu et al., 2024) mitigates jailbreak attacks by amplifying the probabilities of tokens aligned with human values, using the distribution difference between the safety-enhanced expert model and the original model. Other analogous studies utilize differences in logits over vocabulary distributions between sLLMs to influence the output of LLM. DExperts (Liu et al., 2021) suggests a decoding strategy to reduce undesirable outputs of target LLM by leveraging "expert" LMs and "anti-experts" LMs. Using a similar equation, Emulator fine-tuning (Mitchell et al., 2024) emulates instruction-tuning large pre-trained models by ensembling them with small fine-tuned models. (Zhao et al., 2024) uses a small unsafe model to manipulate the output distribution of a larger model to induce jailbreaking. Finally, proxy-tuning (Liu et al., 2024) utilizes the difference in output logits between tuned and untuned small LMs to shift the predictions of the untuned large LMs. Unlike previous methods, our approach involves searching for the optimal mixing coefficient α across a broader range. We discover that the optimal point may differ from those used in previous studies. Our work also employs an entropy-based method and a model-driven approach to enhance the general reasoning ability of LMs.

Importance of the first token. The chain-of-thought decoding (Wang and Zhou, 2024) shows that the first token that LLMs produce have a significant impact on the quality of the entire response in the reasoning task. Exploring paths starting from top-k alternative tokens enables chain-of-thought reasoning without prompts. Similarly, our study also shows that variations in the initial token can greatly impact the model's performance. However, our method only requires a single token from the teacher model, eliminating the need for additional computation to produce all possible k outcomes.

7 Conclusion

In this paper, we have formulated the problem of sLLM decoding with a limited LLM supervision. Through our study, we have unveiled that the optimal combination of sLLM and LLM predictions may significantly depend on the considered task, models, and even datum. Understanding when and why such discrepancies happen is an important future question that needs to be addressed. We believe that our entropy-based perspective will help provide a strong starting point for this pursuit.

Limitations

A notable limitation of our method is that, in its current form, it is difficult to incorporate the predictions of the teacher that has a different embedding space, *e.g.*, using a different tokenizer. Another limitation is that we rely on a single feature, namely the prediction entropy, to determine how to aggregate the predictions. A more in-depth analysis of what other features one can utilize is much needed.

Ethics statement

All experimental results we provide in this paper is based on publicly available datasets and open-source models.

Acknowledgments

This research was supported in part by the NAVER-Intel Co-Lab, and in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2024-00453301). The work was conducted by POSTECH and reviewed by both NAVER and Intel.

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

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*.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, Usvsn Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *Proceedings of the International Conference on Machine Learning*.

Ke Chen, Xingjian Du, Bilei Zhu, Zejun Ma, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. 2022. Hts-at: A hierarchical token-semantic audio transformer for sound classification and detection. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 646–650. IEEE.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

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

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Yuan Gong, Yu-An Chung, and James Glass. 2021. AST: Audio Spectrogram Transformer. In *Proc. Interspeech 2021*, pages 571–575.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. 2023. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. In *Advances in Neural Information Processing Systems*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint 1503.02531v1*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *Proceedings of the International Conference on Machine Learning*.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori Hashimoto, Luke Zettlemoyer, and Mike Lewis. 2023a. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023b. Textbooks are all you need ii: phi-1.5 technical report. *arXiv preprint arXiv:2309.05463*.

Alisa Liu, Xiaochuang Han, Yizhong Wang, Yulia Tsvetkov, Yejin Choi, and Noah A. Smith. 2024. Tuning language models by proxy. *arXiv preprint 2401.08565v3*.

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. 2021. DExperts: Decoding-time controlled text generation with experts and anti-experts. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

I. Loshchilov. and . Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Eric Mitchell, Rafael Rafailov, Archit Sharma, Chelsea Finn, and Christopher D Manning. 2024. An emulator for fine-tuning large language models using small language models. In *International Conference on Learning Representations*.

Sean O'Brien and Mike Lewis. 2023. Contrastive decoding improves reasoning in large language models. *arXiv preprint 2309.09117*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094.

Karol J Piczak. 2015. Esc: Dataset for environmental sound classification. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 1015–1018.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. 2021. Training data-efficient image transformers & distillation through attention. In *International* *conference on machine learning*, pages 10347–10357. PMLR.

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*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *EMNLP 2018*, page 353.

Xuezhi Wang and Denny Zhou. 2024. Chain-of-thought reasoning without prompting. *arXiv preprint 2402.10200v1*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of NAACL-HLT*, pages 1112–1122.

Heming Xia, Tao Ge, Peiyi Wang, Si-Qing Chen, Furu Wei, and Zhifang Sui. 2023. Speculative decoding: Exploiting speculative execution for accelerating seq2seq generation. In *Findings of EMNLP*.

Canwen Xu and Julian McAuley. 2023. A survey on model compression and acceleration for pretrained language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Jinyuan Jia, Bill Yuchen Lin, and Radha Poovendran. 2024. Safedecoding: Defending against jailbreak attacks via safety-aware decoding. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Xuandong Zhao, Xianjun Yang, Tianyu Pang, Chao Du, Lei Li, Yu-Xiang Wang, and William Yang Wang. 2024. Weak-to-strong jailbreaking on large language models. *arXiv preprint arXiv:2401.17256*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2024. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.

Yongchao Zhou, Kaifeng Lyu, Ankit Singh Rawat, Aditya Krishna Menon, Afshin Rostamizadeh, Sanjiv Kumar, Jean-François Kagy, and Rishabh Agarwal. 2024. Distillspec: Improving speculative decoding via knowledge distillation. In *International Conference on Learning Representations*.

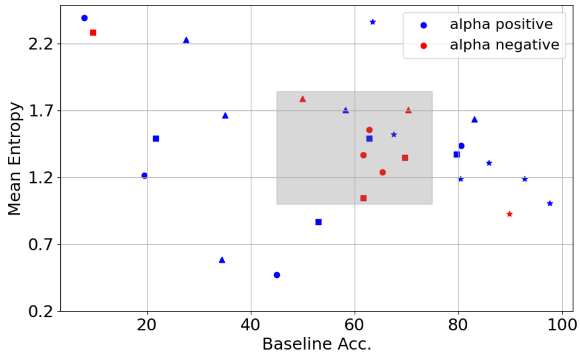

Figure 6: Illustration of the correlation between performance, mean entropy, and the sign of optimal α. Different geometric shapes represent distinct models.

A Detailed empirical study in optimal α cases

We explore the relationship between the signs of the α (positive/negative) and their baseline performance and entropy of the first token, as depicted in Fig. 6. As illustrated, the α values are predominantly positive when the baseline accuracy and entropy are either high or low. However, α values in the intermediate performance and entropy, which in the gray zone of Fig. 6 generally exhibit negative values—three instances where positive α values in the gray zone own significant performance disparities between the teacher and student models. Furthermore, our observations reveal that high entropy does not always correlate with performance, suggesting that while high entropy can indicate challenging problem instances, it may also reflect considerate thinking by the model, and teachers' knowledge can be confused. These findings support our entropy-based Algorithm 1.

B Detailed informations

To demonstrate the superiority of our method, we conduct experiments with various models and datasets. The details of the model, hyper-parameter settings, and datasets are below.

B.1 Model information

The detailed model information for students and teachers is in Table 11. In the exceptional case of StrategyQA, we utilize the Qwen model with a 7B model as the teacher reasons that the 4B model exhibited inferior performance compared to the 1.8B model. On MATH-easy, as the accuracy is 0 in the 1.8B Qwen model, we utilize student a 4B model and 7B for the teacher. For the optimal α prediction model, we make a simple 5-layer DNN.

Datasets	Hyper-parameters		
	epochs	batch size	lr
CIFAR-100	20	128	5×10^{-4}
ESC-50	10	32	1×10^{-4}
MNLI Teacher model†	3	64	3×10^{-6}
MNLI Student model†	3	64	1×10^{-5}

Table 9: Hyper-parameter settings on classification task. The same parameter settings are used for both students and teachers. The parameter setting marked with † are from He et al. (2020)

Models	Hyper-parameters			
	epochs	batch size	lr	max depth
XGboost	-	-	3×10^{-1}	6
DNN	5	1024	5×10^{-7}	-

Table 10: Hyper-parameter settings on optimal α predict model.

B.2 Hyper-parameter settings

We conduct our experiments with hyper-parameter settings as outlined in Table 9 for classification and Table 10 optimal α prediction model. We utilize AdamW (Loshchilov. and Hutter, 2019) optimizer for classification task and for optimal α predict DNN. We performed a grid search for the learning rate, batch size, and weight decay to find an optimal value.

B.3 Dataset statistics

We conduct experiments on a range of datasets to demonstrate the superiority of our method for various domain classification tasks and LLM benchmarks. The overall dataset statistics are illustrated in Table 12, and each task description is below.

B.3.1 Classification task

CIFAR-100. Popular datasets for image classification. It consists of 32 × 32-color images, 50,000 for the train set and 10,000 for the test set with 100 classes.

ESC-50. The audio classification dataset comprises 2,000 environmental sound recordings, each lasting 5 seconds. These recordings are annotated across 50 classes, each allocated a single class label.

MNLI. The Nature Language Inference dataset is part of the GLUE benchmark (Wang et al., 2018). It consists of 431,993 data with three classes, and the test set consists of two distinct parts. The 'matched'

	Classification task	
Task	Student	Teacher
CIFAR-100	facebook/deit-base-patch16-224	facebook/deit-tiny-patch16-224
ESC-50	Only three layers of the Teacher	MIT/ast-finetuned-audioset-16-16-0.442
MNLI	microsoft/deberta-large-mnli	microsoft/deberta-v2-xxlarge-mnli
	LLM benchmarks	
Models	Student	Teacher
Llama-2	meta-llama/Llama-2-7b-chat-hf	meta-llama/Llama-2-13b-chat-hf
Vicuna-v1.5	lmsys/vicuna-7b-v1.5	lmsys/vicuna-13b-v1.5
Qwen1.5	Qwen/Qwen1.5-1.8B-Chat	Qwen/Qwen1.5-4B-Chat
Phi-3	microsoft/Phi-3-mini-4k-instruct	microsoft/Phi-3-medium-4k-instruct

Table 11: Teacher & Student model information.

		Classification tasks		
Datasets	#Total	#Train	#Dev	#Test
CIFAR-100	60,000	50,000	-	10,000
ESC-50	2,000	2,000	-	-
MNLI-matched	431,993	392,702	9,815	9,796
MNLI-mismatched	-	-	9,832	9,847
		LLM benchmarks		
Datasets	#Total	#Train	#Dev	#Test
GSM8K	8,792	7,473	-	1,319
Multiarith	600	420	-	180
SVAMP	1,000	700	-	300
MATH-easy	966	570	-	396
StrategyQA	2,290	1,603	-	687
ARC-challenge	2,590	1,119	299	1,172
ARC-easy	5,197	2,251	570	2,376

Table 12: Dataset statistics information.

is from the same source as the training dataset, and the ‘mismatched’ is from a different domain.

B.3.2 LLM benchmarks

GSM8K. A dataset of 8,792 linguistically diverse math word problems for grade school students, crafted by human problem writers (Cobbe et al., 2021). It is intended for multi-step mathematical reasoning, with problems requiring between 2 to 8 steps to solve. Solutions mainly involve performing a series of basic arithmetic operations to arrive at the final answer. We use test set for evaluation in this paper.
Multiarith. A collection of multi-step arithmetic problems from MAWPS (Koncel-Kedziorski et al., 2016). These problems require several reasoning steps involving simple arithmetic calculations to obtain the final answer. Because of the low resource test data, we use train set.
SVAMP. A challenge set for elementary-level math word problems. Each problem consists of a brief natural language narrative that describes a scenario and asks a question about certain unknown quantities. We use both train and test set for evaluation.
MATH. Dataset that consists of 12,500 problems from high school math competitions (Hendrycks et al., 2021). The problems are challenging because they require a higher level of step-by-step mathematical problem-solving skills to solve. Cause many models have poor performance, we predicted some easy tasks we defined as MATH-Easy, which contains the train set with level 1 and 3 algebra parts.
StrategyQA focuses on open-domain 2,780 questions that require models to deduce a multi-hop strategy to answer. We use test set in this paper.
ARC. The AI2’s Reasoning Challenge (ARC) dataset designed for multiple-choice question-answering, encompasses science exam questions ranging from grades 3 to 9. It is partitioned into Easy and Challenge sections, with the latter comprising more complex questions that demand reasoning skills. We use test set in this paper.

C Additional results

C.1 More experiments of optimal α predict model

Through additional empirical analysis, we find that alpha prediction models are generalizable, at least to some extent. We perform two sets of experiments. As shown in Table 13, we trained the alpha predictor on GSM8K and evaluated it on Multiarith

Models	Multiarith	MATH-Easy
Phi3-mini	97.62	63.51
w/ reference (α = 1)	97.62	64.39
w/ optimal α	98.10	65.09
w/ entropy sweet spot	98.10	65.61
w/ DNN (Unseen)	97.95 $\pm$ 0.13	65.86 $\pm$ 0.50

Table 13: Experiment results on unseen benchmarks. We train with GSM8K and evaluate with Multiarith and MATH-Easy.

Models	GSM8K	ARC-Challenge	ARC-Easy
Phi3-mini	80.36	85.92	92.76
w/ reference (α = 1)	81.50	86.09	93.27
w/ optimal α	82.18	86.43	93.35
w/ entropy sweet spot	82.34	86.60	93.56
w/ DNN	81.91 $\pm$ 0.20	86.47 $\pm$ 0.20	93.29 $\pm$ 0.09

Table 14: Experiment results on joint training. We trained with the GSM8K, ARC-Challenge, and ARC-Easy train set and evaluated on each benchmark's test set.

and MATH-Easy (all math datasets). We observe that the predictor achieves a close performance to the "entropy sweet spot," demonstrating effective generalizability to unseen data. As illustrated in Table 14, we trained the alpha predictor on a joint set of GSM8K, ARC-Challenge, and ARC-Easy and predicted on each dataset. Again, we observe that the predictor performs well on each task, indicating that one can train an adequate alpha predictor by combining data from various sources.

C.2 The rank of a new word in the original distribution

Fig. 7 shows the frequency of the ranks of new tokens generated with supervision from the teacher model within the distribution of tokens predicted by the student model, according to α. As evident from the results, as α moves away from zero, our algorithm selects more low-ranked words. This indicates that effectively selecting new words that lead to coherent sentences through extrapolation may result in dramatic performance gains.

C.3 Don't Trust Your Student

Based on the transformation of equation 1 into equation 3, extending our decoding method between the student and teacher models to all tokens and setting α higher than 1 can be considered deducting the student's logits from the teachers'. This operation disregards the knowledge represented solely by the student's logits, not trusting the student's knowledge. We term this methodology 'Don't Trust Your Student'(DTYS) and apply it to GSM8K. The results, as shown in Table 15, indicate an overall performance improvement over the teacher model baseline, demonstrating that DTYS is an effective strategy for leveraging a less capable model to enhance the performance of a more qualified one.

Models	GSM8K
Phi3-mini (Student)	80.36
Phi3-medium (Teacher)	90.83
DTYS	**91.51**
Llama2-7B (Student)	21.68
Llama2-13B (Teacher)	37.38
DTYS	**38.74**
Vicuna-7B (Student)	19.56
Vicuna-13B (Teacher)	36.01
DTYS	**37.53**
Qwen-1.8B (Student)	35.03
Qwen-4B (Teacher)	46.32
DTYS	**46.78**

Table 15: Experiment results on Don't trust your student.

$$S'_\alpha = \alpha \cdot \sigma(f_t(x)) - (\alpha - 1) \cdot \sigma(f_s(x)) \quad (3)$$

D Full Prompts of each benchmark

For the GSM8K, Multiarith, SVAMP, and StrategyQA, we refer to the chain-of-thought prompt used in (Wei et al., 2022). We use 2 shots for the GSM8K, 6 shots for StrategyQA, and 8 shots for the rest. The full prompts used in the MATH and ARC benchmarks are shown in Table 16 and Table 17, respectively. Due to space constraints in the paper, we omit line breaks between each sentence in MATH and between questions and options in ARC.

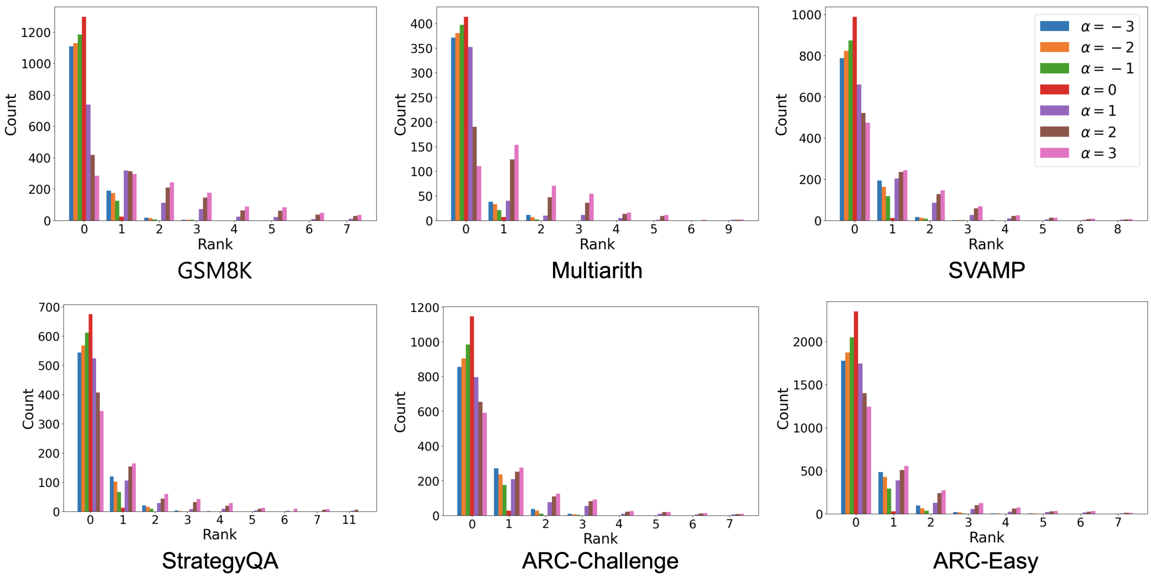

Figure 7: The frequency of rankings of new words in the evaluation of the Phi3-mini.

Prompt for MATH

Question: What is the largest value of x such that the expression \[\dfrac{x+1}{8x^2−65x+8}\] is not defined?
Let's think step by step. In this particular case, the fraction will be undefined only if its denominator is equal to zero. Because of this, we can ignore the numerator. We start by setting the binomial in the denominator equal to 0: \begin{align∗} 8x^2−65x+8=0 \\\Rightarrow\qquad (8x−1)(x−8)=0 \end{align∗} We find that the two possible values for x are $\frac18$ and 8. Since the question asks for the largest value, the final solution is $\boxed{8}$. The answer is 8
Question: BoatWorks built 3 canoes in January of this year and then each subsequent calendar month they built twice the number of canoes they had built the previous month. How many total canoes were built by BoatWorks by the end of March of this year?
Let's think step by step. The number of boats built is $3+3\cdot2+3\cdot2^2 = 3+6+12 = \boxed{21}$. The answer is 21
Question: If $5a+2b=0$ and a is two less than b, what is $7b$?
Let's think step by step. First we begin by solving the system of equations \begin{align∗} 5a+2b&=0, \\ b−2&=a. \end{align∗} Making the substitution for a from the second equation to the first, we get $5(b−2)+2b=0$, which simplifies to $7b−10=0$. Solving for b, we find that $b=\frac{10}{7}$. Hence $7b=7\cdot \frac{10}{7}=\boxed{10}$. The answer is 10
Question: The difference between two numbers is 9, and the sum of the squares of each number is 153. What is the value of the product of the two numbers?
Let's think step by step. Call the first number x and the second number y. Without loss of generality, assume $x > y$. We can represent the information given in the problem with the following system of linear equations: \begin{align∗} x − y &= 9\\ x^2 + y^2 &= 153 \end{align∗} Solving for x in the first equation and substituting into the second yields $(9+y)^2 + y^2 = 153$, or $2y^2 + 18y − 72 = 0$. Canceling a 2 gives $y^2 + 9y − 36 = 0$, which factors into $(y+12)(y−3)$. Thus, $y = 3$ and $x = 12$. So, $x \cdot y = \boxed{36}$. The answer is 36
Question: Simplify $\frac{1}{1+\sqrt{2}}\cdot\frac{1}{1−\sqrt{2}}$.
Let's think step by step. Multiplying the numerators simply yields 1. Multiplying the denominators gives $1+\sqrt{2} − \sqrt{2} −2 = 1 − 2 = −1$. So, the answer is $\frac{1}{−1} = \boxed{−1}$. The answer is −1
Question: Four people can mow a lawn in 6 hours. How many more people will be needed to mow the lawn in 4 hours, assuming each person mows at the same rate?
Let's think step by step. The number of people mowing and the time required to mow are inversely proportional. Letting n be the number of people and t be the amount of time, we have $nt = (4)(6)= 24$ because 4 people can mow a lawn in 6 hours. If m people can mow the lawn in 4 hours, then we must have $m(4) = 24$, so $m=6$. Therefore, we need $6−4 = \boxed{2}$ more people to complete the job in 4 hours. The answer is 2
Question: BoatsRUs built 7 canoes in January of this year and then each subsequent calendar month they built twice the number of canoes they had built the previous month. How many total canoes were built by BoatsRUs by the end of May of this year?
Let's think step by step. The numbers of canoes built by BoatsRUs each month form a geometric sequence: 7, 14, 28, 56, 112. The first term is 7 and the common ratio is 2, so the sum of these terms is $\frac{7(2^5−1)}{2−1} = \boxed{217}$. The answer is 217
Question: Find the coefficient of the x^2 term in the expansion of the product $(ax^3 + 3x^2 − 2x)(bx^2 − 7x − 4)$.
Let's think step by step. We only need to worry about the terms that multiply to have a degree of 2. This would be given by the product of the terms $3x^2$ and $−4$ as well as the product of the terms $−2x$ and $−7x$. Thus, $$(3x^2) \times (−4) + (−2x) \times (−7x) = −12x^2 + 14x^2 = 2x^2,$$and the coefficient is $\boxed{2}$. The answer is 2

Table 16: Few-shot exemplars for full chain of thought prompt for MATH

Prompt for ARC

Q: George wants to warm his hands quickly by rubbing them. Which skin surface will produce the most heat? (A) dry palms (B) wet palms (C) palms covered with oil (D) palms covered with lotion
A: Let's think step by step. Friction produces heat. Dry skin has a higher coefficient of friction than wet or oily skin. Therefore, dry palms will produce the most heat when rubbed together. Therefore, the answer is (A).
Q: Which of the following statements best explains why magnets usually stick to a refrigerator door? (A) The refrigerator door is smooth. (B) The refrigerator door contains iron. (C) The refrigerator door is a good conductor. (D) The refrigerator door has electric wires in it.
A: Let's think step by step. Magnets are attracted to ferrous materials, which contain iron. The refrigerator door is usually made of steel, which is an alloy of iron and carbon. The presence of iron in the refrigerator door allows the magnet to stick to it.
Therefore, the answer is (B).
Q: A fold observed in layers of sedimentary rock most likely resulted from the (A) cooling of flowing magma. (B) converging of crustal plates. (C) deposition of river sediments. (D) solution of carbonate minerals.
A: Let's think step by step. Sedimentary rocks are formed from the accumulation and compaction of sediments, such as sand, mud, and organic matter. When these sediments are deposited, they are typically laid down in horizontal layers. If these layers are later subjected to compressional forces, such as those that occur when two crustal plates converge, they can be folded. The folding of the layers is a result of the deformation of the rock under the influence of stress. The other options (A, C, and D) are less likely to cause folding in sedimentary rocks. Cooling of flowing magma (A) can cause the formation of igneous rocks, but it does not typically cause folding in sedimentary rocks. Deposition of river sediments (C) can result in the formation of sedimentary layers, but it does not typically cause folding. Solution of carbonate minerals (D) can cause the formation of caves and other karst features, but it does not typically cause folding in sedimentary rocks. Therefore, The answer is (B).
Q: Which of these do scientists offer as the most recent explanation as to why many plants and animals died out at the end of the Mesozoic era? (A) worldwide disease (B) global mountain building (C) rise of mammals that preyed upon plants and animals (D) impact of an asteroid created dust that blocked the sunlight
A: Let's think step by step. The question asks for the most recent explanation for the mass extinction at the end of the Mesozoic era. Answer choice (A), worldwide disease, is not a widely accepted explanation for the mass extinction. Answer choice (B), global mountain building, is not a widely accepted explanation for the mass extinction. Answer choice (C), rise of mammals that preyed upon plants and animals, is not a widely accepted explanation for the mass extinction. Answer choice (D), impact of an asteroid created dust that blocked the sunlight, is the most widely accepted explanation for the mass extinction. Therefore, the answer is (D).
Q: A boat is acted on by a river current flowing north and by wind blowing on its sails. The boat travels northeast. In which direction is the wind most likely applying force to the sails of the boat? (A) west (B) east (C) north (D) south
A: Let's think step by step. The boat is traveling northeast, which means it is moving both north and east. The river current is flowing north, so it is not causing the boat to move east. Therefore, the wind must be blowing on the sails of the boat in a direction that causes the boat to move east. The only option that satisfies this requirement is (B) east. Therefore, the answer is (B).

Table 17: Few-shot exemplars for full chain of thought prompt for ARC.