Self-Distillation for Multi-Token Prediction

Anonymous ACL submission

Abstract

As Large Language Models (LLMs) continue to scale, inference efficiency has emerged as a critical bottleneck. Multi-Token Prediction (MTP) offers a promising avenue to accelerate LLM inference by predicting multiple future tokens simultaneously. Wherein, DeepSeek MTP has gradually become an indispensable component in industrial LLMs. However, it still faces two challenges: limited accept rates of MTP heads, and difficulties in jointly training multiple MTP heads. To address these challenges, we propose **MTP-D**, a simple yet effective self-distillation method that substantially boosts MTP head accept rates while maximally preserving main-head performance. We further introduce a **looped extension strategy** for MTP-D, enabling effective and economical extension to multiple MTP heads and yielding significant additional inference speedup (up to $4.55\times$). Beyond these contributions, we conduct systematic experiments across seven benchmarks to explore and validate key insights into distillation strategies and the scalability potential of DeepSeek MTP. Results demonstrate that MTP-D and the grouped extension strategy consistently enhance MTP head performance and inference efficiency, facilitating the practical usage of MTP in LLMs.

1 Introduction

Large Language Models (LLMs) have demonstrated strong performance across diverse tasks (Guo et al., 2025; Kimi et al., 2025; Qwen, 2025). As task complexity and model scale continue to grow, inference efficiency has become increasingly critical, especially in agentic applications with long, multi-step interactions (Yao et al., 2022). However, most LLMs rely on the Next-Token Prediction (NTP) paradigm, which performs autoregressive, token-by-token generation and inherently incurs high latency and computational cost, particularly for long context (Mehra et al., 2025; DeepSeek-AI, 2026).

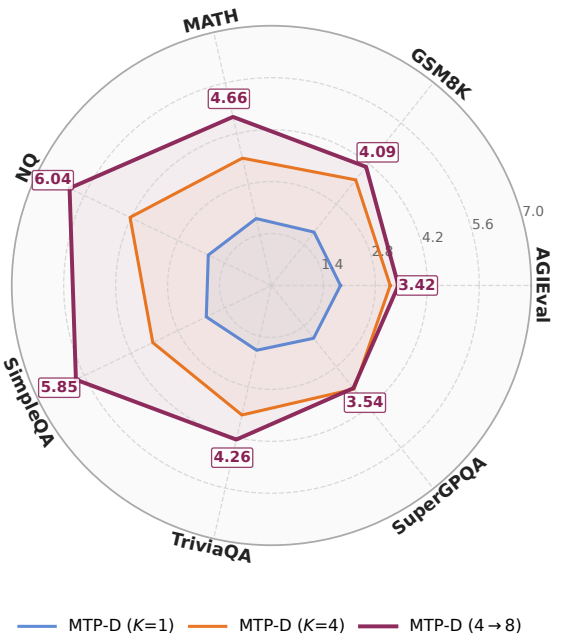


Figure 1: Speedup of MTP-D and its looped extension strategy across benchmarks from diverse domains.

Multi-Token Prediction (MTP) has been proposed as an effective approach to alleviate this inefficiency (Gloeckle et al., 2024). Unlike NTP, MTP equips LLMs with multiple prediction heads to simultaneously predict several future tokens, enabling significant inference acceleration. This design has been widely adopted in popular industrial LLMs (Xiaomi et al., 2025; Qwen, 2025; Zeng et al., 2026). In general, higher accept rates and greater scalability of MTP heads translate into more substantial inference speedups.

The cascaded MTP architecture of DeepSeek-V3 (Deepseek et al., 2024) effectively improves the performance of MTP heads. However, it still faces several challenges for subsequent work: (a) **the limited accept rates of MTP heads**, which could lead to the exponential decline of the cumulative accept rate in speculative decoding, thereby

undermining practical inference speedup. (b) the difficulty of jointly training multiple MTP heads. Given the constraints on training efficiency and the risk of degrading the main head's performance for industrial LLMs, the number of MTP heads that can be jointly trained is typically limited, **highlighting the urgent need for an efficient MTP head extension strategy**.

To address these challenges, we propose **MTP-D**, a simple yet effective self-distillation method for pre-training. MTP-D applies gradient-detached, Top-N-logits distillation from the main head to MTP heads as auxiliary training losses, along with a looped extension strategy via economical continued pre-training. Specifically, the Top-N logits of the main head are used to supervise MTP head training, which naturally aligns with the original design intent of MTP. Furthermore, we introduce a **looped MTP extension strategy** that treats trained MTP heads as a group, iteratively extends them group by group as initialization for new MTP heads, and updates them via continued pre-training. By exploiting intra-group correlations among MTP heads and the distributional consistency induced by distillation, this strategy efficiently extends MTP heads with substantially fewer tokens, achieving significant inference speedups.

Experimental results demonstrate that MTP-D achieves strong distributional consistency with the main head, effectively improving MTP head accept rates and facilitating subsequent head extension. Specifically, with 4 MTP heads, MTP-D yields an average improvement of 3.30% in accept rate and 11.42% in inference speedup. Moreover, our grouped extension strategy enables cost-efficient scaling of MTP heads from 4 to 8, achieving a further speedup of up to 4.55×. In addition, we uncover and validate several interesting insights regarding the scalability of MTP. Our key contributions are summarized as follows:

1. We propose MTP-D, a novel self-distillation framework that improves MTP head accept rates by enforcing strong distributional consistency with the main head.

2. We introduce a looped extension strategy, enabling cost-efficient extension of trained MTP heads via continue pre-training, yielding up to $4.55\times$ speedup under 8 speculative tokens.

3. Extensive experiments across seven benchmarks demonstrate that MTP-D significantly improves MTP head accept rates and inference speedup, offering practical insights into the deployment of MTP in LLMs.

2 Preliminary

Next-Token Prediction is constrained by its training paradigm, which substantially limits both the sample efficiency during pretraining and the inference efficiency of LLMs. Recent studies show that Multi-Token Prediction alleviates these limitations by employing N independent output heads to simultaneously predict the next N tokens, thereby providing richer supervisory signals and significantly improving sample efficiency (Qi et al., 2020). Furthermore, combined with speculative decoding, MTP can dramatically accelerate LLM inference (Leviathan et al., 2023; Chen et al., 2023; Gloeckle et al., 2024; Li et al., 2024, 2026).

DeepSeek-V3 (Deepseek et al., 2024) advances MTP by adopting a cascaded architecture that explicitly models inter-token dependencies while preserving the complete causal chain and autoregressive nature. This architecture has been widely integrated into prominent industrial LLMs, such as MiMo, GLM, Kimi, LongCat, and Qwen3-Next (Gloeckle et al., 2024; Xiaomi et al., 2025; Zeng et al., 2026; Team et al., 2026; LongCat et al., 2025; Qwen, 2025). Specifically, the loss function of DeepSeek MTP is formulated as follows:

$$\begin{aligned}\mathcal{L}^{\mathrm{CE}}_{\mathrm{mtp}} &= \sum_{k=1}^{K} \alpha_k \, \mathcal{L}^{\mathrm{CE}}_{\mathrm{mtp_k}} \\ &= \sum_{k=1}^{K} \alpha_k \, \mathrm{CE}\Big(\hat{\mathbf{P}}^{k}_{k+1:T+1},\, \mathbf{t}_{k+1:T+1}\Big)\end{aligned} \tag{1}$$

where $\mathrm{CE}(\cdot)$ represents the cross-entropy loss, α_k denotes the weight coefficient of the k-th MTP CE loss term, and $\mathbf{t}$ and $\hat{\mathbf{P}}$ denote the ground-truth token sequence and its corresponding predicted probability distribution, respectively.

3 Method

Multi-Token Prediction has been widely proven effective for accelerating LLM inference, where the degree of speedup is primarily determined by the accept rate of speculative tokens generated by MTP heads. We introduce MTP-D, a self-distillation approach that aligns the output distributions of MTP heads with the main head during pre-training, and further propose a looped extension strategy for ef-

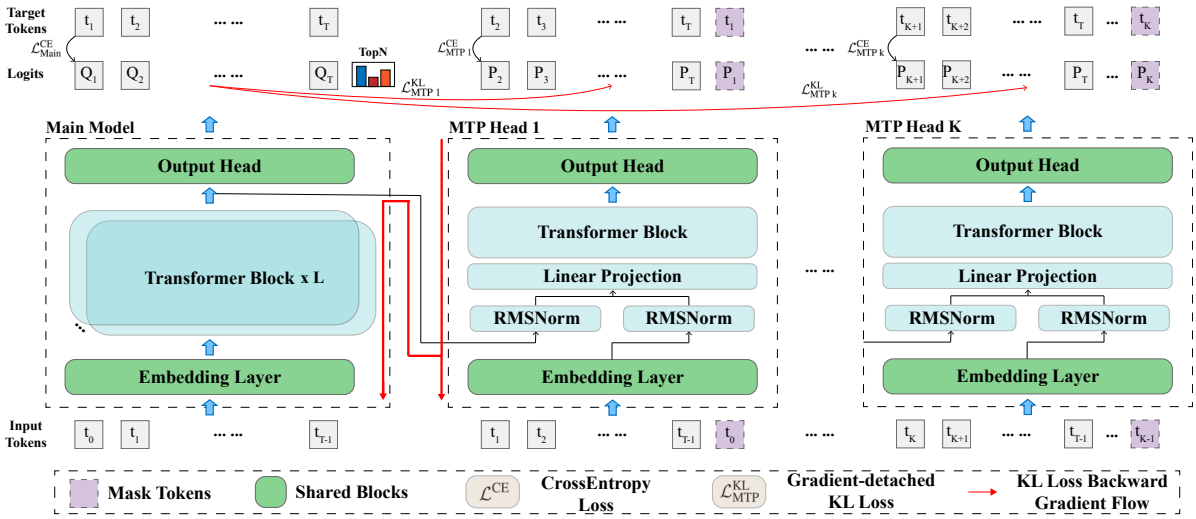


Figure 2: Overview of the gradient-detached, Top$N$-logits-selected self-distillation method, MTP-D.

ficient extension of MTP heads via continued pre-training.

3.1 Self-Distillation for MTP in Pre-Training

As shown in Figure 8, MTP heads incur substantially higher training losses than the main head, with losses increasing monotonically with the MTP index. To enhance MTP head performance, we propose a gradient-detached, Top$N$-logits-selected self-distillation method (Figure 2). This method introduces an auxiliary distillation loss that aligns the output logit distributions of MTP heads with those of the main head, enforcing distributional consistency and thereby significantly improving MTP head performance.

Specifically, building upon the MTP loss $\mathcal{L}^{\text{CE}}_{\text{mtp}}$ in Eq. (1), we introduce a unidirectional Kullback-Leibler (KL) divergence loss (Kullback and Leibler, 1951) as a self-distillation objective, transferring knowledge from the Top$N$ logits of the main head to the corresponding positions of the MTP heads:

$$\begin{aligned}\mathcal{L}^{\text{KL}}_{\text{mtp}} &= \sum_{k=1}^{K} \beta_k \, \mathcal{L}^{\text{KL}}_{\text{mtp}_k} \\ &= \sum_{k=1}^{K} \beta_k \, \text{KL}\Big(\tilde{\mathbf{P}}^{k}_{k+1:T+1}, \, \text{sg}(\tilde{\mathbf{Q}})_{k+1:T+1}\Big)\end{aligned} \tag{2}$$

where,

$$\mathcal{I}^{\text{N}}_t = \text{TopK}\left(\hat{\mathbf{Q}}_t, N\right) \tag{3}$$

$$\tilde{\mathbf{Q}}_{k+1:T+1} = \sigma\left(\hat{\mathbf{Q}}_{k+1:T+1}\left[\ldots, \mathcal{I}^{N}_t\right]\right) \tag{4}$$

$$\tilde{\mathbf{P}}^{k}_{k+1:T+1} = \log\left(\sigma\left(\hat{\mathbf{P}}^{k}_{k+1:T+1}\left[\ldots, \mathcal{I}^{N}_t\right]\right)\right) \tag{5}$$

Here, $\mathcal{I}^{N}_t$ denotes the index set of the Top$N$ elements of the main head logits $\hat{\mathbf{Q}}_t$ along the vocabulary dimension V for the t-th token. $\sigma(\cdot)$ and $\log(\sigma(\cdot))$ denote the softmax and log-softmax functions, respectively. $\text{KL}(\cdot)$ denotes the KL divergence, β_k is the weighting coefficient for the k-th MTP KL loss term, and $\text{sg}(\cdot)$ denotes the stop-gradient operation.

Our method is built on two key designs: gradient-detached self-distillation and Top$N$ logit selection, as elaborated below.

Gradient-detached self-distillation. Empirically, we find that naively applying self-distillation causes a significant performance drop in the main head. To mitigate this issue, we apply a stop-gradient operation to the main head logits $\hat{\mathbf{Q}}$ in $\mathcal{L}^{\text{KL}}_{\text{mtp}_k}$, preventing gradients from propagating back through $\hat{\mathbf{Q}}$. As illustrated by the red path in Figure 2, $\mathcal{L}^{\text{KL}}_{\text{mtp}_k}$ propagates gradients exclusively through $\hat{\mathbf{P}}$ during the backward pass, following the same backward path as the MTP cross-entropy loss $\mathcal{L}^{\text{CE}}_{\text{mtp}_k}$. Consequently, with appropriate weighting coefficients α_k for $\mathcal{L}^{\text{CE}}_{\text{mtp}_k}$ and β_k for $\mathcal{L}^{\text{KL}}_{\text{mtp}_k}$, our method maximizes MTP head performance while preserving comparable main model performance.

Top$N$-selected logits. Modern LLMs often employ extremely large vocabularies (*e.g.*, 122,880 in our setting), making full-vocabulary self-distillation computationally expensive. As shown in Figure 6, main-head logits follow a long-tailed distribution after softmax, with most token probabilities approaching zero. In practice, the cumulative probability already reaches 0.9952 for Top$N = 10{,}000$, which suffices for effective self-

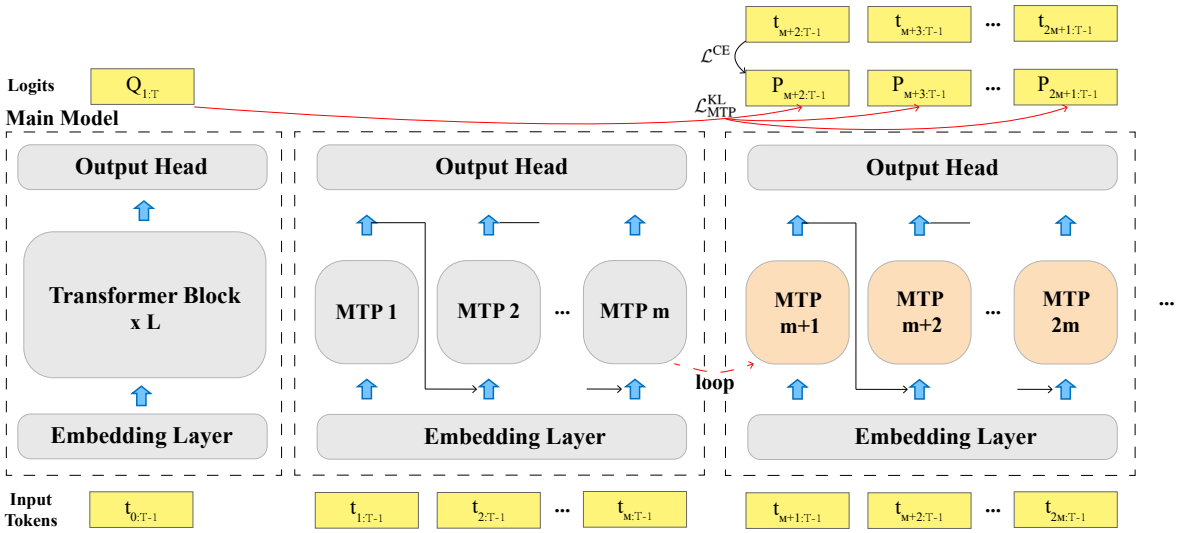


Figure 3: Illustration of the training strategy for looped extension of MTP-D. The gray blocks represent the frozen main model and the trained MTP heads from 1 to m. The weights of the MTP heads from 1 to m are copied to initialize the MTP heads from $m+1$ to $2m$. The orange blocks denote the trainable MTP heads.

distillation. Distilling over all logits therefore introduces redundancy, high memory consumption, numerical instability, and weak supervision from near-zero-probability tokens, as illustrated in Figure 7. Based on our analyses and ablation studies, we set Top$N$ = 10,000, which ensures both efficient and stable self-distillation. Further details are provided in Appendix C.

Other explorations. We further investigate several self-distillation strategies for MTP, including: (1) variants of the KL divergence, *i.e.*, forward, reverse, and symmetric KL; (2) different Top$N$ logit selection strategies, *i.e.*, the choice of N and selecting Top$N$ from $\hat{\mathbf{Q}}_t$ only versus the union of Top$N$ from both $\hat{\mathbf{Q}}_t$ and $\hat{\mathbf{P}}_t$; (3) dynamic scheduling of α_k and β_k during training; Furthermore, we conduct a fine-grained hyperparameter search for MTP-D to offer practical guidance for followed works. A detailed analysis is provided in Section 4.4.

Final training objective. The final loss for MTP heads combines two components: $\mathcal{L}_{\text{mtp}}^{\text{CE}}$ (Eq. 1) and $\mathcal{L}_{\text{mtp}}^{\text{KL}}$ (Eq. 2). Specifically, $\mathcal{L}_{\text{mtp}}^{\text{CE}}$ supervises MTP heads with ground-truth tokens, ensuring basic prediction correctness, especially in the early stages of pre-training. $\mathcal{L}_{\text{mtp}}^{\text{KL}}$ transfers higher-level semantic knowledge from the main head to the MTP heads via distillation, while aligning their output distributions with that of the main head. Together, these two objectives ensure stable and effective pre-training: $\mathcal{L}_{\text{mtp}} = \mathcal{L}_{\text{mtp}}^{\text{CE}} + \mathcal{L}_{\text{mtp}}^{\text{KL}}$.

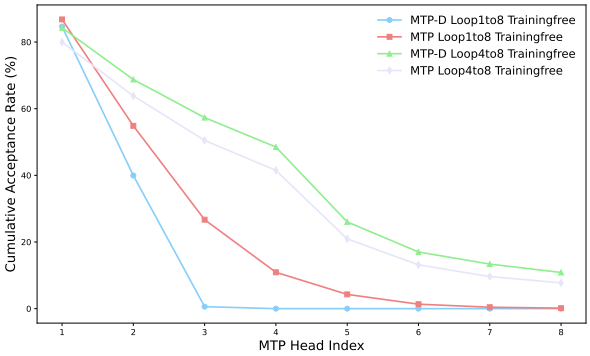
3.2 Looped Extension Strategy for MTP Heads in Continue Pre-Training

Our MTP-D achieves higher accept rates through distillation, thereby establishing a solid foundation for the extension of the MTP heads.

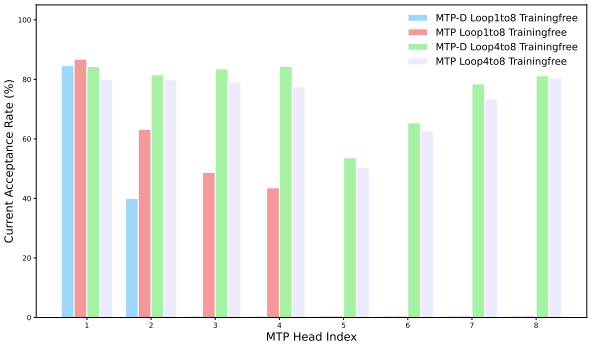
As shown in Figure 2, the DeepSeek MTP architecture exhibits inherent structural consistency and input-output similarity between heads, which provides a natural basis for the extension of the MTP heads via looped extension. Specifically, as illustrated in Figure 3, **looped extension** refers to an operation where a previously trained set of m MTP heads is used to initialize the next set of m heads, followed by continued pre-training on the expanded configuration. By iteratively repeating this process, MTP heads can be progressively extended in a cost-efficient manner.

Training-Free Explorations. To validate this insight, we conduct a set of pilot experiments under a training-free setting, where both the DeepSeek MTP and MTP-D are extended to 8 MTP heads. As illustrated in Figure 4, several key observations emerge:

First, MTP heads at loop connection points exhibit a noticeable drop in accept rate, though it remains at an acceptable level. As the number of MTP heads increases, accept rates gradually recover and converge toward those of the previously trained heads. These observations suggest that the cascaded architecture of DeepSeek MTP, with its inherent structural consistency and input-output similarity, naturally supports scalability.



(a) Cumulative accept rates for AGIEval.



(b) Accept rates for AGIEval.

Figure 4: Accept rate and cumulative accept rate of MTP heads for different models under a training-free setting. Using the AGIEval benchmark as an example, (a) shows the cumulative accept rate as the loop is extended to 8 MTP heads for different models, while (b) presents the accept rate of each MTP head.

Compared to DeepSeek MTP, MTP-D exhibits substantially better scalability. Under the 1-to-8 loop setting, the cumulative accept rate of DeepSeek MTP drops to merely 0.6% at MTP head 3, whereas MTP-D maintains 26.70%, enabling further scaling. This improvement stems from the stronger distributional consistency between MTP heads and the main head enforced by distillation, which significantly enhances scalability. A detailed analysis of training-free looped MTP is provided in Appendix F.

Looped Extension of MTP-D. Looped extension inherently preserves the correlations among intra-group MTP heads. As shown in Figure 3, continued pre-training is performed with the same self-distillation strategy, which maintains distributional consistency across all heads. Furthermore, to avoid degrading main head performance and to reduce training cost, both the main model and the previously trained MTP heads are kept frozen during continued pre-training.

4 Experiments

4.1 Experimental Setup

Model configuration and training details. MTP-D was implemented on both a 2B Dense model and an N10BA1B MoE LLM. All experiments were conducted on the FineWeb-Edu-350BT pre-training dataset (Penedo et al., 2024), with MTP-D pre-training consuming 350B tokens and looped extension continued pre-training consuming 70B tokens. All experiments were carried out on NVIDIA H20 GPUs using the TorchTitan pre-training framework. Further details are provided in Appendix B.

Evaluation benchmarks. We evaluate our MTP-D on a diverse set of widely used pre-training benchmarks spanning multiple domains, including AGIEval (Zhong et al., 2024) for human-centric standardized examinations; GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) for mathematical reasoning; NaturalQuestions (Kwiatkowski et al., 2019) for knowledge-intensive question answering; SimpleQA (Wei et al., 2024) for short fact-seeking queries; SuperGPQA (Du et al., 2025) for graduate-level knowledge and reasoning; and TriviaQA (Joshi et al., 2017) for reading comprehension.

Speculative decoding and metrics. We adopt a lossless speculative decoding strategy (Cai et al., 2024; Xia et al., 2023, 2024), which guaranties that the outputs of the MTP-augmented model are exactly identical to those of the main model alone, as detailed in Algorithm 1.

We report the following metrics: (1) *Accuracy*: the average accuracy of the main model across benchmarks; (2) *Accept Rate (AR)*: the fraction of draft tokens generated by each MTP head that are accepted during verification, relative to the total number of tokens verified for that head; (3) *Cumulative Accept Rate (CAR)*: the fraction of draft tokens accepted for each MTP head relative to the total number of tokens generated by that head; (4) *Speedup Ratio*: the relative inference speedup over the baseline, *i.e.*, NTP; and (5) *Accepted Length* (τ): the average number of draft tokens accepted per decoding step. More details are provided in Appendix D.

4.2 Main Experiments

We conduct a comprehensive evaluation of DeepSeek MTP and our MTP-D on 2B Dense and 10B MoE LLMs across multiple pre-training benchmarks. The evaluation metrics include average

K	Model	Method	General	Math		Knowledge			STEM	
			AGIEval	**GSM8K**	**MATH**	**NQ**	**SimpleQA**	**TriviaQA**	**SuperGPQA**	**Avg.**
			τ / Speedup	τ / Speedup	τ / Speedup	τ / Speedup	τ / Speedup	τ / Speedup	τ / Speedup	τ / Speedup
		Joint Training								
1	2B Dense	MTP	1.86 / 1.65x	1.85 / 1.73x	1.86 / 1.75x	1.91 / 1.69x	1.94 / 1.63x	1.82 / 1.61x	1.85 / 1.79x	1.87 / 1.69x
		MTP-D	**1.89 / 1.86x**	**1.89 / 1.84x**	**1.90 / 1.85x**	**1.94 / 1.89x**	**1.96 / 1.95x**	**1.87 / 1.79x**	**1.88 / 1.82x**	**1.90 / 1.86x**
	10B MoE	MTP	1.85 / 1.70x	1.85 / 1.79x	1.85 / 1.67x	1.91 / 1.71x	1.91 / 1.64x	1.80 / 1.69x	1.83 / 1.72x	1.86 / 1.70x
		MTP-D	**1.91 / 1.84x**	**1.89 / 1.84x**	**1.88 / 1.75x**	**1.94 / 1.84x**	**1.94 / 1.75x**	**1.86 / 1.82x**	**1.87 / 1.85x**	**1.90 / 1.81x**
4	2B Dense	MTP	3.48 / 2.82x	4.16 / 3.16x	3.37 / 3.28x	4.22 / 3.66x	3.55 / 3.41x	3.46 / 3.43x	3.56 / 2.95x	3.69 / 3.24x
		MTP-D	**3.72 / 3.20x**	**4.27 / 3.64x**	**3.57 / 3.52x**	**4.28 / 4.23x**	**3.60 / 3.55x**	**3.63 / 3.58x**	**3.76 / 3.57x**	**3.83 / 3.61x**
	10B MoE	MTP	3.57 / 2.85x	4.14 / 3.11x	3.30 / 2.80x	3.92 / 3.46x	3.46 / 3.11x	3.25 / 3.20x	3.44 / 2.84x	3.58 / 3.05x
		MTP-D	**3.72 / 3.23x**	**4.24 / 3.76x**	**3.35 / 3.30x**	**4.08 / 4.03x**	**3.69 / 3.64x**	**3.50 / 3.45x**	**3.76 / 3.17x**	**3.76 / 3.51x**
		MTP Extension								
8	2B Dense	MTP (1→8)†	2.25 / 1.80x	2.27 / 2.09x	2.16 / 2.01x	2.43 / 2.28x	2.50 / 2.35x	2.26 / 2.11x	2.24 / 1.97x	2.30 / 2.09x
		MTP (4→8)†	3.87 / 2.85x	3.96 / 3.47x	3.86 / 3.71x	5.36 / 5.21x	5.36 / 4.78x	3.83 / 3.68x	3.83 / 3.29x	4.30 / 3.86x
		MTP-D (1→8)†	2.85 / 2.21x	2.76 / 2.51x	2.92 / 2.77x	3.29 / 3.14x	3.29 / 3.11x	2.76 / 2.61x	2.78 / 2.29x	2.95 / 2.66x
		MTP-D (4→8)†	**4.26 / 3.06x**	**4.10 / 3.56x**	**4.28 / 4.13x**	**5.56 / 5.41x**	**5.66 / 5.04x**	**4.18 / 4.03x**	**4.14 / 3.37x**	**4.60 / 4.09x**
		MTP (1→8)	3.98 / 2.91x	3.80 / 3.52x	3.86 / 3.71x	5.07 / 4.92x	5.82 / 5.61x	3.65 / 3.50x	3.95 / 3.10x	4.30 / 3.90x
		MTP (4→8)	4.25 / 3.09x	4.38 / 3.86x	4.45 / 4.30x	5.86 / 5.71x	5.99 / 5.49x	4.02 / 3.87x	4.22 / 3.49x	4.74 / 4.26x
		MTP-D (1→8)	4.41 / 3.24x	4.18 / 3.80x	4.60 / 4.45x	5.76 / 5.61x	6.29 / 5.71x	3.97 / 3.82x	4.18 / 3.34x	4.77 / 4.28x
		MTP-D (4→8)	**4.69 / 3.42x**	**4.47 / 4.09x**	**4.81 / 4.66x**	**6.19 / 6.04x**	**6.31 / 5.85x**	**4.41 / 4.26x**	**4.51 / 3.54x**	**5.06 / 4.55x**
		MTP-D (4→8, 350B)	4.70 / 3.43x	4.50 / 4.12x	4.84 / 4.69x	6.20 / 6.05x	6.31 / 5.83x	4.43 / 4.28x	4.52 / 3.55x	5.07 / 4.56x
16	2B Dense	MTP-D (1→16)†	2.81 / 1.75x	2.72 / 2.00x	2.86 / 2.53x	3.24 / 3.09x	3.24 / 2.86x	2.75 / 2.60x	2.75 / 1.88x	2.91 / 2.39x
		MTP-D (4→16)†	**4.25 / 2.49x**	**4.13 / 2.96x**	**4.31 / 3.67x**	**5.80 / 5.65x**	**5.29 / 4.57x**	**4.13 / 3.98x**	**4.16 / 2.68x**	**4.58 / 3.71x**
		MTP-D (1→16)	4.75 / 2.53x	4.50 / 2.94x	5.10 / 3.77x	6.93 / 6.06x	6.93 / 4.63x	4.14 / 3.99x	4.42 / 2.81x	5.25 / 3.82x
		MTP-D (4→16)	**5.16 / 2.92x**	**4.81 / 3.48x**	**5.27 / 4.47x**	**7.70 / 7.55x**	**7.40 / 6.33x**	**4.65 / 4.50x**	**4.83 / 2.95x**	**5.69 / 4.60x**

Table 1: Accept length and speedup of MTP joint training and extension. *Joint Training*: MTP and MTP-D with $K \in \{1, 4\}$ on 2B Dense and 10B MoE. *MTP Extension*: $A \rightarrow B$ extends a $K{=}A$-head checkpoint to $K{=}B$ via continue Pre-training or training-free inference (†). **Bold**: best per group. For MTP extension, MTP-D (4→8, 350B) uses 350B tokens and others use 70B tokens.

main-head accuracy, accept rate (AR), cumulative accept rate (CAR) of each MTP head, as well as speedup ratio and accept length (τ).

MTP-D substantially improves accept length and speedup ratio while maintaining comparable main-head performance. As shown in Table 1, MTP-D consistently improves the accept length (τ) and speedup ratio of MTP heads on both the 2B Dense and 10B MoE models, demonstrating the effectiveness of our self-distillation strategy. Specifically, in terms of τ, with 1 MTP head, MTP-D achieves gains of +1.81% on Dense and +2.23% on MoE; with 4 MTP heads, the gains increase to +4.08% on Dense and +5.11% on MoE, indicating that the benefit of distillation becomes more pronounced as the number of MTP heads increases and model sizes scale up. As shown in Table 7, MTP-D maintains comparable main-head accuracy to DeepSeek MTP, with both surpassing the NTP baseline.

Furthermore, Table 6 reports the complete acceptance rate metrics (CAR/AR) for both MTP and MTP-D. For $K = 1$, MTP-D achieves an average improvement of $+4.12\%$. For $K = 4$, MTP-D achieves an average improvement of $+5.1\%$, with the advantage growing progressively as H increases. All MTP heads benefit from our self distillation.

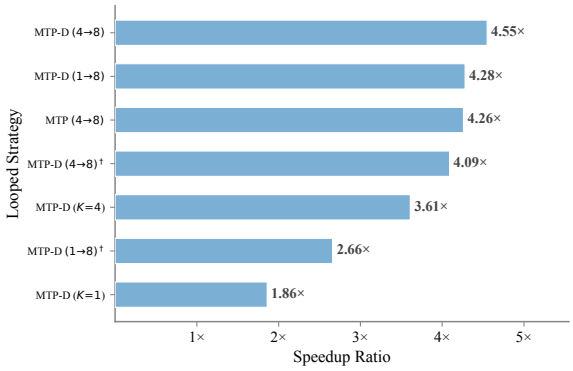


Figure 5: Comparison of performance across different looped extension strategies with up to 8 loops.

4.3 Results of Looped MTP Extension

In this section, we investigate the scalability of looped extension and the performance upper bounds of models with varying numbers of MTP heads. As shown in Table 1, we report detailed re-

sults for both DeepSeek MTP and MTP-D extended to 8 and 16 MTP heads ($K = 8$ and $K = 16$) via either continued pre-training or training-free looping. Additionally, Appendix F presents the CAR curves for MTP heads looped up to 8 (Figure 12) and up to 16 (Figure 13) under both training-free and looped extension settings. From these results, we draw the following insights:

Insight 1: Cascaded MTP is inherently scalable due to its structural consistency across heads. As shown in Figures 11, when extending from 4 to 8 MTP heads in a training-free manner, the accept rate drops primarily at the loop junction (*e.g.*, from 80% to 50%), yet remains at an acceptable level, while subsequent heads within the group gradually recover to levels comparable to those of the trained heads.

Insight 2: MTP-D exhibits superior scalability compared to DeepSeek MTP. As shown in Figure 10, under training-free extension from 1 to 8 loops, DeepSeek MTP undergoes rapid performance collapse, whereas MTP-D maintains substantially higher CAR throughout. This suggests that distillation enhances the distributional consistency between MTP heads and the main head, thereby better facilitating subsequent MTP head extension. As further evidenced in Table 1, MTP-D (4→8) substantially outperforms MTP (4→8) in both accept length and speedup ratio.

Insight 3: Grouped MTP heads exhibit stronger loop scalability than a single MTP head. As shown in Table 1, comparing MTP-D (1→8) and MTP-D (4→8) reveals that grouped extension consistently achieves higher accept length and speedup ratio, indicating that intra-group correlations are preserved during loop extension.

Insight 4: A limited amount of data suffices for looped MTP extension. For MTP-D (4→8), scaling the continued pre-training data from 70B to 350B tokens yields only marginal gains in accept length and speedup ratio.

Insight 5: Inference speedup peaks at a practically optimal number of MTP heads. As shown in Table 1, while MTP-D (4→16) still achieves a notable improvement in accept length over MTP-D (4→8) (+12.45%), the speedup ratio no longer increases. This is because as the number of MTP heads grows, the computational overhead of the draft heads becomes the bottleneck, offsetting the gains from higher token acceptance.

Strategy		Acc↑	MTP Loss↓	Accept Length↑	Speedup↑
NTP (baseline)		11.06	–	1	1×
MTP-D		**11.68**	**2.609**	**1.90**	**1.86×**
no detach		<u>10.19</u>	–	–	–
Top$N$	10000×2	11.47	2.613	1.90	1.86×
	1000	11.47	2.618	1.87	1.83×
	1	11.27	2.704	1.80	1.76×
β_k	1.5	11.04	2.620	1.85	1.82×
	0.5	11.65	2.632	1.83	1.80×
	0.3	11.57	2.654	1.82	1.79×
	0.1	11.69	2.684	1.80	1.77×
KL	reverse	11.62	2.630	1.84	1.81×
	hybrid	11.48	2.654	1.81	1.79×

Table 2: Ablation results for MTP-D with 1 MTP head. The MTP-D configuration is: `detach`, Top$N = 10{,}000$, $\beta_k = 1.0$, forward KL divergence (bold). The <u>no detach</u> entry significantly degrades main-head accuracy and is excluded from speculative decoding evaluation.

4.4 Ablation Study

We conduct comprehensive ablation experiments to validate the contribution of each component in MTP-D, using average main-head accuracy (Acc), MTP head loss, MTP head accept length (τ), and speedup ratio as evaluation metrics.

Taking a single MTP head for example, we ablate four key components of MTP-D: `detach`, Top$N$, β_k, and the KL divergence variant. The default configuration is: `detach` enabled, Top$N = 10{,}000$, $\beta_k = 1.0$, and forward KL. Analysis is based on the loss curves in Figure 9 and the benchmark results in Table 2.

First, removing `detach` substantially increases the main-head loss (+0.079) and degrades its performance (-1.49%), demonstrating that the stop-gradient operation on the main-head logits $\hat{\mathbf{Q}}$ effectively prevents gradient backpropagation through $\hat{\mathbf{Q}}$, thereby mitigating interference to the main head.

Next, we examine the effect of Top$N$ along the vocabulary dimension V of $\hat{\mathbf{Q}}$. Due to the long-tailed distribution in the vocabulary space (Figure 6), distilling over all logits introduces computational redundancy, excessive memory consumption, and numerical instability in logarithmic operations, potentially causing gradient oscillation or vanishing (Figure 7). As Top$N$ increases, main-head performance remains largely stable, while MTP head loss decreases significantly until Top$N$ exceeds a threshold, beyond which further gains

plateau. Here, "10000×2" denotes the union of the Top$N$ sets from $\hat{\mathbf{Q}}$ and $\hat{\mathbf{P}}$. Balancing training efficiency and MTP head performance, we select Top$N$ = 10,000.

The weighting factor β_k is another critical design choice. An appropriate value of β_k in the range $[0.1, 1.0]$ improves both main-head performance and MTP head accept rate. However, an excessively large β_k (*e.g.*, $\beta_k = 1.5$) leads to a notable increase in main-head loss (+0.028) and a corresponding degradation in main-head accuracy (11.04), highlighting the importance of careful tuning. Finally, we ablate the KL divergence variant (details in Appendix E) and find that forward KL achieves the best trade-off between training loss and evaluation performance.

4.5 Other Comparison

In this work, we focus on the joint pre-training with extension paradigm, which is increasingly adopted in industrial-scale LLMs (*e.g.*, DeepSeek (DeepSeek-AI, 2026), GLM (Zeng et al., 2026), Kimi (Team et al., 2026) and MiMo (Xiaomi et al., 2025; Xiao et al., 2026)), and propose effective strategies for MTP head training and extension. Further details are provided in Appendix G.

To further validate the advantage of this paradigm, we compare MTP-D against Eagle3 (Li et al., 2026) under matched parameters and training data. As shown in Table 3, MTP-D achieves substantially higher accept length and speedup ratio across all settings.

Method	**K**	**Avg. τ**	**Avg. Speedup**
MTP	1	1.87	1.69×
	4	3.69	3.24×
	8	4.74	4.26×
MTP-D (Ours)	1	**1.90**	**1.86**×
	4	**3.83**	**3.61**×
	8	**5.06**	**4.55**×
Eagle3	1	1.83	1.65×
	4	3.00	2.71×
	8	4.52	4.07×

Table 3: Comparison of MTP, MTP-D, and Eagle3 on the 2B Dense model. For $K = 8$, the results of MTP and MTP-D are obtained via looped extension.

Furthermore, Appendix H presents a detailed experimental validation of MTP-D under production-scale data, models, and infrastructure.

5 Related Work

MTP. The MTP paradigm has been widely studied in both research and industrial LLMs, as it provides rich supervision, improves sample efficiency, and accelerates inference. The classical MTP architecture was introduced by the Meta team (Gloeckle et al., 2024), and DeepSeek-V3 (Deepseek et al., 2024) further proposed the cascaded DeepSeek MTP, which has seen broad adoption. Several studies have explored MTP's mechanisms and applications in LLMs, including adaptation to small models (Aynetdinov and Akbik, 2025), strategies for parallel token prediction (Mehra et al., 2025), and its impact on relational learning (Zhong et al., 2025). Moreover, other works investigate speculative decoding (Cai et al., 2025; Li et al., 2024, 2026; Chen et al., 2026).

Distillation in LLMs. Supervised knowledge distillation (Buciluǎ et al., 2006; Hinton et al., 2015) is a classical technique successfully applied to auto-regressive models (Sanh et al., 2019) and has become central in industrial LLMs, particularly during post-training (Liu et al., 2025b; Agarwal et al., 2024; Gu et al., 2023; Wen et al., 2023). For instance, DeepSeek R1 (Guo et al., 2025) demonstrated that distilling chain-of-thought data enhances reasoning, making distillation a key part of LLM pipelines. It has also been explored in pre-training, with its scaling laws systematically analyzed (Busbridge et al., 2025). Inspired by these advances, we adapt distillation to MTP, where the main head serves as teacher and MTP heads as students, enabling natural self-distillation.

6 Conclusion

In this work, we propose MTP-D, a self-distillation framework for MTP pre-training, together with a looped extension strategy for continued pre-training. MTP-D consistently improves MTP head accept length and speedup ratio across different LLM backbones while maintaining comparable main-head performance. Combined with the looped extension strategy, MTP-D scales up to 8 MTP heads, achieving a further inference speedup of up to 4.55×. Extensive experiments demonstrate the effectiveness and generality of our approach. The proposed methods and associated insights offer valuable guidance for improving the pre-training and inference efficiency of future LLMs with MTP, and provide practical inspiration for broader applications of MTP.

Limitations

This work focuses on self-distillation for multi-token prediction during pre-training. Future work should explore its adaptation to post-training and investigate the scalability of MTP-D throughout the full LLM production pipeline. Additionally, we plan to explore advanced MTP techniques, such as shared KV cache MTP and parallel MTP (*e.g.*, Dflash), and examine their impact on LLM pre-training. Due to infrastructure and resource constraints, inference evaluation was not conducted on industrial inference frameworks such as vLLM or SGLang and we will progressively address this in future work. We hope this work inspires further validation of MTP-D at industrial scale across the entire training pipeline.

References

Kingma DP Ba J Adam and 1 others. 2014. A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 1412(6).

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *The twelfth international conference on learning representations*.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Ansar Aynetdinov and Alan Akbik. 2025. Pre-training curriculum for multi-token prediction in language models. *arXiv preprint arXiv:2505.22757*.

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

Dan Busbridge, Amitis Shidani, Floris Weers, Jason Ramapuram, Etai Littwin, and Russ Webb. 2025. Distillation scaling laws. *arXiv preprint arXiv:2502.08606*.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D Lee, Deming Chen, and Tri Dao. 2024. Medusa: Simple llm inference acceleration framework with multiple decoding heads. *arXiv preprint arXiv:2401.10774*.

Yuxuan Cai, Xiaozhuan Liang, Xinghua Wang, Jin Ma, Haijin Liang, Jinwen Luo, Xinyu Zuo, Lisheng Duan, Yuyang Yin, and Xi Chen. 2025. Fastmtp: Accelerating llm inference with enhanced multi-token prediction. *arXiv preprint arXiv:2509.18362*.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. 2023. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*.

Jian Chen, Yesheng Liang, and Zhijian Liu. 2026. Dflash: Block diffusion for flash speculative decoding. *arXiv preprint arXiv:2602.06036*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Aixin Deepseek, Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

DeepSeek-AI. 2026. Deepseek-v4: Towards highly efficient million-token context intelligence.

Xinrun Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, King Zhu, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, and 1 others. 2025. Supergpqa: Scaling llm evaluation across 285 graduate disciplines. *arXiv preprint arXiv:2502.14739*.

Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. 2024. Better & faster large language models via multi-token prediction. *arXiv preprint arXiv:2404.19737*.

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

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

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. *arXiv preprint arXiv:2103.03874*.

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

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, and 1 others. 2024. Minicpm: Unveiling the potential of small language models

with scalable training strategies. *arXiv preprint arXiv:2404.06395*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Kimi, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, and 1 others. 2025. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, and 1 others. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024. Eagle: Speculative sampling requires rethinking feature uncertainty. *arXiv preprint arXiv:2401.15077*.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2026. Eagle-3: Scaling up inference acceleration of large language models via training-time test. *Advances in Neural Information Processing Systems*, 38:136737–136756.

Wanchao Liang, Tianyu Liu, Less Wright, Will Constable, Andrew Gu, Chien-Chin Huang, Iris Zhang, Wei Feng, Howard Huang, Junjie Wang, and 1 others. 2024. Torchtitan: One-stop pytorch native solution for production ready llm pre-training. *arXiv preprint arXiv:2410.06511*.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, and 1 others. 2025a. Muon is scalable for llm training. *arXiv preprint arXiv:2502.16982*.

Kaiyuan Liu, Shaotian Yan, Rui Miao, Bing Wang, Chen Shen, Jun Zhang, and Jieping Ye. 2025b. Where did this sentence come from? tracing provenance in llm reasoning distillation. *arXiv preprint arXiv:2512.20908*.

Meituan LongCat, Bei Li, Bingye Lei, Bo Wang, Bolin Rong, Chao Wang, Chao Zhang, Chen Gao, Chen Zhang, Cheng Sun, and 1 others. 2025. Longcat-flash technical report. *arXiv preprint arXiv:2509.01322*.

Somesh Mehra, Javier Alonso Garcia, and Lukas Mauch. 2025. On multi-token prediction for efficient llm inference. *arXiv preprint arXiv:2502.09419*.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin A Raffel, Leandro Von Werra, Thomas Wolf, and 1 others. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*, 37:30811–30849.

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training. *arXiv preprint arXiv:2001.04063*.

Qwen. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

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*.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, SH Cai, Yuan Cao, Y Charles, HS Che, Cheng Chen, Guanduo Chen, and 1 others. 2026. Kimi k2. 5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*.

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. 2024. Measuring short-form factuality in large language models. *arXiv preprint arXiv:2411.04368*.

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

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 the Association for Computational Linguistics: EMNLP 2023*, pages 3909–3925.

Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhifang Sui. 2024. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. *arXiv preprint arXiv:2401.07851*.

Bangjun Xiao, Bingquan Xia, Bo Yang, Bofei Gao, Bowen Shen, Chen Zhang, Chenhong He, Chiheng Lou, Fuli Luo, Gang Wang, and 1 others. 2026. Mimo-v2-flash technical report. *arXiv preprint arXiv:2601.02780*.

LLM Xiaomi, Bingquan Xia, Bowen Shen, Dawei Zhu, Di Zhang, Gang Wang, Hailin Zhang, Huaqiu Liu, Jiebao Xiao, Jinhao Dong, and 1 others. 2025. Mimo: Unlocking the reasoning potential of language model–from pretraining to posttraining. *arXiv preprint arXiv:2505.07608*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, and 1 others. 2026. Glm-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*.

Qimin Zhong, Hao Liao, Siwei Wang, Mingyang Zhou, Xiaoqun Wu, Rui Mao, and Wei Chen. 2025. Understanding and enhancing the planning capability of language models via multi-token prediction. *arXiv preprint arXiv:2509.23186*.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2024. Agieval: A human-centric benchmark for evaluating foundation models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2299–2314.

A The Use of LLMs

In this paper, we leveraged LLMs to support and refine the writing. Specifically, LLMs were used for grammar and spelling correction, as well as polishing linguistic expressions to enhance clarity and readability.

B Model Configurations and Training Details

Table 4 summarizes the configurations of the two models used in our experiments: a 2B Dense model and an N10B A1B MoE model. The 2B Dense model comprises 32 decoder layers and K MTP heads. The N10B A1B MoE model consists of 22 decoder layers (1 dense "af" layer and 21 MoE "ae" layers) and K MTP heads. Both models employ grouped query attention (GQA) with a vocabulary size of 122,880. The number of MTP head layers (`mtp_head_layers`) is set to 1. Following the DeepSeek MTP architecture, the MTP heads share the embedding layer and output projection with the main head. Notably, for both models, each MTP head adopts a single dense transformer layer rather than a MoE layer.

Table 5 reports the training hyperparameters for MTP-D across both the pre-training and continued pre-training stages. The two stages differ primarily in warmup strategy and data size: continued pre-training disables learning rate warmup and trains on 70B tokens. The MTP loss coefficient α_k is set to 0.3 following DeepSeek-V3 (Deepseek et al., 2024). The distillation coefficient β_k is determined via fine-grained search, yielding $\beta_k = 1.0$ for single-MTP models ($K = 1$) and $\beta_k = 0.5$ for four-MTP models ($K = 4$).

Specifically, all models were trained using the AdamW optimizer (Adam et al., 2014) with $(\beta_1, \beta_2) = (0.9, 0.95)$ and $\epsilon = 1 \times 10^{-8}$, along with a weight decay of 0.1 and gradient clipping with a maximum norm of 1.0. The learning rate followed the Warmup-Stable-Decay (WSD) schedule (Hu et al., 2024), with a peak value of 3×10^{-4} and a minimum of 0. The sequence length was set to 4,096 tokens with a batch size of 8,192 tokens. RMSNorm was used for layer normalization, and rotary position embeddings (RoPE) (Su et al., 2024) were applied. For MTP-D pre-training, the warmup was set to 1,000 steps over 350B tokens; for looped extension, warmup was disabled and training ran for 70B tokens. All experiments were conducted on the FineWeb-Edu-350BT dataset, a randomly sampled subset of approximately 350B tokens from the FineWeb-Edu corpus (Penedo et al., 2024), using the TorchTitan pre-training framework (Liang et al., 2024) on NVIDIA H20 GPUs (98 GB).

Table 4: Detailed description of model configurations.

Hyperparameter	2B Dense	N10BA1B MoE
dim	2048	1536
n_heads	16	32
n_kv_heads	4	4
head_dim	128	128
embedding_tie	False	False
qk_norm	True	True
ffn_hidden_dim	6144	6912
max_seq_len	4096	4096
rope_theta	10000.0	10000.0
norm_eps	1e-5	1e-5
decoder_layers	32 af	af + 21 ae
expert_hidden_dim	-	768
num_experts	-	128
use_shared_expert	-	True
num_shared_experts	-	1
top_k	-	8
use_grouped_mm	-	True
initializer_range	-	0.02
num_mtp_heads	K	K
mtp_head_layers	1	1
mtp_dense_or_moe	dense	dense

Table 5: Training hyperparameter details for MTP-D and looped extension

Hyperparameter	MTP-D	Looped extension
Optimizer	AdamW	
Adam (β_1, β_2)	(0.9, 0.95)	
Adam ϵ	1×10^{-8}	
Weight decay	0.1	
Clip grad norm	1.0	
Max lr	3.0×10^{-4}	
Min lr	0	
Lr decay	Cosine	
Decay rate	10%	
Sequence length	4096	
Batch size	8192	
Normalization	RMSNorm	
Vocabulary size	122880	
Positional encoding	ROPE	
Warmup steps	**1000**	**0**
Data Size	**350B**	**70B**
α_k	0.3	
β_k	1.0 for K=1 0.5 for K=4	

C Analysis of Logits Probability Distribution

As illustrated in Figure 6, the main-head logits for the t-th token exhibit a long-tailed distribution after softmax, with most candidate tokens as-

signed near-zero probabilities. Specifically, the cumulative probability reaches 0.9952 for Top$N$ = 10,000, 0.8341 for Top$N$ = 1,000, 0.5338 for Top$N$ = 100, 0.2395 for Top$N$ = 10, and 0.0961 for Top$N$ = 1.

As shown in Figure 7, distilling over the full vocabulary leads to training instability and loss divergence. We attribute this to two factors: (1) the sheer number of near-zero logits introduces severe numerical instability in the log-softmax computation, causing gradient explosion or vanishing; and (2) the overwhelming proportion of low-probability tokens produces noisy and uninformative gradients that dominate the distillation signal, suppressing effective learning from the high-probability tokens that matter most. Furthermore, once Top$N$ exceeds a certain threshold, the cumulative probability of the selected tokens approaches 1, providing a sufficiently complete distribution for self-distillation. Based on these observations, we select Top$N$ = 10,000 as it achieves the optimal balance between training stability, computational efficiency, and distillation quality.

D Speculative Decoding

To ensure that the inference outputs of the MTP-augmented model are exactly identical to those of the main model alone, we adopt a lossless speculative decoding strategy. To guarantee robustness and reproducibility of inference across all experiments, inference is performed in a single-batch setup with greedy decoding applied to all samples.

We report the following metrics for MTP evaluation: (1) *Accept Rate (AR)*: the fraction of draft tokens generated by each MTP head that are accepted during verification, relative to the total number of tokens verified for that head; (2) *Cumulative Accept Rate (CAR)*: the fraction of accepted draft tokens for each MTP head relative to the total number of decoding steps; (3) *Speedup Ratio*: the relative inference speedup over the NTP baseline; and (4) *Accepted Length (τ)*: the average number of tokens generated per decoding step, including both the main head token and accepted draft tokens.

$$\mathrm{AR}_j = \frac{\sum_{s=1}^{S} A_j^{(s)}}{\sum_{s=1}^{S} C_j^{\mathrm{cmp},(s)}}, \qquad j = 1, \ldots, K. \quad (6)$$

$$\mathrm{CAR}_j = \frac{\sum_{s=1}^{S} A_j^{(s)}}{\sum_{s=1}^{S} C_{\mathrm{step}}^{(s)}}, \qquad j = 1, \ldots, K. \quad (7)$$

$$\tau = 1 + \sum_{j=1}^{K} \mathrm{CAR}_j, \qquad j = 1, \ldots, K. \quad (8)$$

$$\text{Speedup} = \frac{\sum_{s=1}^{S} t_{\mathrm{NTP}}^{(s)}}{\sum_{s=1}^{S} t_{\mathrm{MTP}}^{(s)}}. \quad (9)$$

Here, S denotes the total number of decoding steps across the benchmark, and K denotes the number of MTP heads. $A_j^{(s)}$ is the number of draft tokens generated by the j-th MTP head that are accepted by the main head at step s; $C_j^{\mathrm{cmp},(s)}$ is the number of tokens from the j-th MTP head verified by the main head at step s; and $C_{\mathrm{step}}^{(s)}$ is the total number of decoding steps at step s, which is identical across all MTP heads.

E Detailed Analysis for MTP-D

E.1 Variants of KL Loss

The reverse KL divergence, defined as $\mathrm{KL}(q\|p) = \mathbb{E}_{\mathbf{x}\sim q}\left[\log \frac{q(\mathbf{x})}{p(\mathbf{x})}\right]$, encourages the model distribution q to concentrate on the modes of the target distribution p. The corresponding loss is formulated as:

$$\mathcal{L}_{\mathrm{mtp}_k}^{\text{KL-R}} = \mathrm{KL}\Big(\mathrm{sg}(\bar{\mathbf{Q}})_{k+1:T+1},\, \bar{\mathbf{P}}_{k+1:T+1}^{k}\Big) \quad (10)$$

$$\bar{\mathbf{Q}}_{k+1:T+1}^{k} = \log\left(\sigma\left(\hat{\mathbf{Q}}_{k+1:T+1}^{k}\,[\cdots]\right)\right) \quad (11)$$

$$\bar{\mathbf{P}}_{k+1:T+1} = \sigma\left(\hat{\mathbf{P}}_{k+1:T+1}\,[\cdots]\right) \quad (12)$$

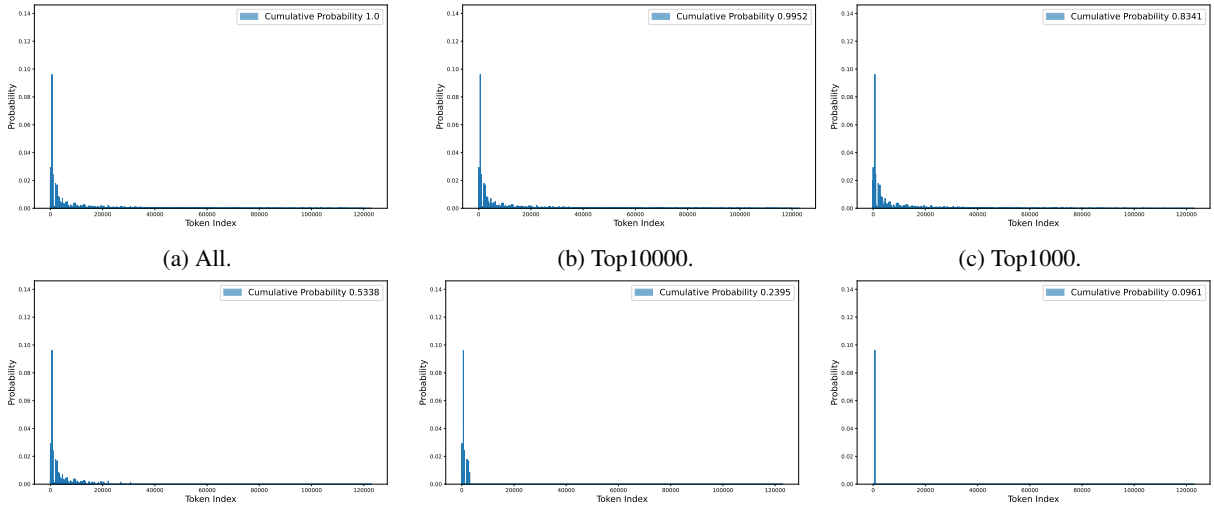
The hybrid KL loss is defined as a weighted combination of the forward and reverse KL divergences:

$$\mathcal{L}_{\mathrm{mtp}_k}^{\text{KL-H}} = \lambda\, \mathcal{L}_{\mathrm{mtp}_k}^{\mathrm{KL}} + (1-\lambda)\, \mathcal{L}_{\mathrm{mtp}_k}^{\text{KL-R}}, \quad \lambda \in [0,1]. \quad (13)$$

E.2 Details of Main Experiments

Table 7 reports the average main-head accuracy across multiple benchmarks for the main experiments. The results show that for both the 2B Dense and 10B MoE models, the main-head average accuracy of both MTP-D and MTP consistently surpasses the NTP baseline. Furthermore, when $K = 1$, MTP-D outperforms MTP, while at $K = 4$, the two methods maintain comparable performance.

Additionally, Table 6 reports the detailed per-head AR and CAR metrics.



(a) All. (b) Top10000. (c) Top1000.

(d) Top100. (e) Top10. (f) Top1.

Figure 6: Illustration of the probability distributions of the main head logits for the t-th token under different Top$N$ settings.

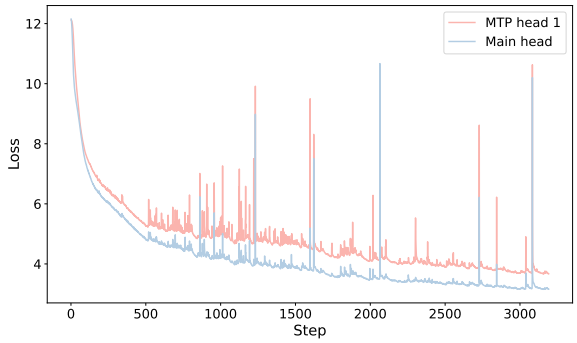


Figure 7: Training loss curves for the 2B Dense model with 1 head using the whole vocabulary.

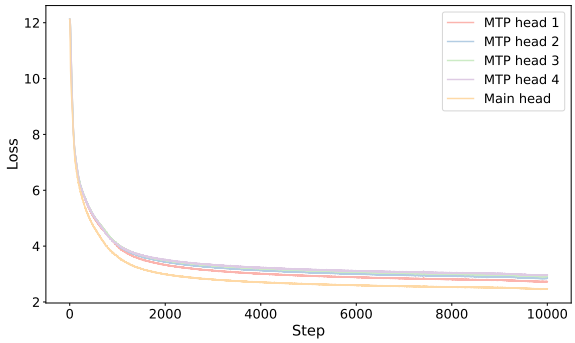


Figure 8: Illustration of the training loss curves for a 2B dense LLM equipped with 4 heads. The final losses of the main head and MTP heads 1 to 4 are 2.47 and [2.73, 2.85, 2.92, 2.96], respectively.

E.3 Details of Ablation Experiments

In this subsection, we provide a detailed analysis of the ablation experiments for MTP-D with 1 and 4 MTP heads.

E.3.1 1 MTP head

Loss serves as an important metric for pre-training evaluation. Figure 9 presents the final loss comparison across different variants of MTP-D with a single MTP head ($K = 1$), covering both main-head and MTP-head losses. While maintaining comparable main-head loss, MTP-D achieves substantially lower MTP-head loss than all other variants.

E.3.2 4 MTP heads

We provide additional ablation experiments for MTP-D with 4 heads, analyzing the multi-level distillation strategies and the corresponding β_k values.

For MTP-D with 4 heads, the hyperparameters are set to $\beta_k = 0.5$ with main-to-MTP distillation. We conduct a hyperparameter study on β_k, and as shown in Table 8, $\beta_k = 0.5$ achieves the best trade-off between main-head accuracy and MTP head loss, and is therefore selected as the default setting.

F Detailed Analysis of Looped Extension

In this section, we provide a detailed analysis of accept rates and cumulative accept rates across multiple pre-training benchmarks for looped MTP extension, under both training-free and continued pre-training settings.

				General	Math		Knowledge			STEM
K	Model	Method	H	AGIEval	GSM8K	MATH	NQ	SimpleQA	TriviaQA	SuperGPQA
1	2B Dense	MTP	1	85.75	84.86	85.66	90.91	94.28	82.32	85.10
		MTP-D	1	**88.98**	**88.54**	**89.58**	**94.40**	**96.00**	**86.78**	**87.93**
	10B MoE	MTP	1	85.41	85.24	85.25	91.34	90.65	80.18	82.51
		MTP-D	1	**90.98**	**89.36**	**87.62**	**93.80**	**93.99**	**86.27**	**87.38**
4	2B Dense	MTP	1	81.88 / 81.88	90.32 / 90.32	81.27 / 81.27	89.23 / 89.23	85.17 / 85.17	82.09 / 82.09	82.62 / 82.62
			2	66.81 / 81.59	82.47 / 91.31	64.63 / 79.52	82.55 / 92.52	68.07 / 79.91	66.84 / 81.42	68.26 / 82.62
			3	54.28 / 81.24	74.40 / 90.21	50.83 / 78.65	77.14 / 93.45	55.74 / 81.89	53.39 / 79.89	56.65 / 82.99
			4	45.47 / 83.77	68.52 / 92.09	40.27 / 79.23	73.51 / 95.30	45.60 / 81.82	43.75 / 81.94	48.62 / 85.83
		MTP-D	1	**85.86** / 85.86	**91.69** / 91.69	**84.73** / 84.73	**90.77** / 90.77	**85.20** / 85.20	**84.71** / 84.71	**86.20** / 86.20
			2	71.96 / 83.81	84.63 / 92.30	69.46 / 81.98	83.98 / 92.52	70.97 / 83.29	71.16 / 84.00	71.96 / 83.48
			3	61.25 / 85.12	78.33 / 92.55	56.75 / 81.70	78.47 / 93.44	57.57 / 81.12	58.57 / 82.31	62.57 / 86.95
			4	**52.96** / 86.46	**72.76** / 92.89	**46.42** / 81.81	**74.78** / 95.30	**46.27** / 80.38	**48.52** / 82.85	**54.98** / 87.87
	10B MoE	MTP	1	82.38 / 82.38	89.63 / 89.63	80.54 / 80.54	82.81 / 82.81	83.64 / 83.64	78.06 / 78.06	80.81 / 80.81
			2	68.11 / 82.67	81.39 / 90.80	63.11 / 78.35	74.54 / 90.02	67.29 / 80.45	61.40 / 78.66	64.48 / 79.79
			3	57.26 / 84.07	74.31 / 91.30	48.58 / 76.98	68.99 / 92.56	52.80 / 78.47	47.66 / 77.62	53.34 / 82.72
			4	48.82 / 85.27	68.22 / 91.81	37.85 / 77.92	65.31 / 94.67	41.87 / 79.30	37.45 / 78.58	45.22 / 84.78
		MTP-D	1	**85.67** / 85.67	**91.41** / 91.41	**82.22** / 82.22	**86.98** / 86.98	**86.40** / 86.40	**82.46** / 82.46	**85.92** / 85.92
			2	71.93 / 83.97	83.84 / 91.72	65.10 / 79.18	79.38 / 91.26	73.20 / 84.72	68.42 / 82.98	72.79 / 84.72
			3	61.55 / 85.58	76.97 / 91.82	49.75 / 76.42	73.24 / 92.26	60.36 / 82.45	55.04 / 80.45	62.88 / 86.38
			4	**52.47** / 85.25	**71.47** / 92.86	**37.89** / 76.16	**68.05** / 92.92	**49.18** / 81.48	**44.51** / 80.87	**54.28** / 86.32

Table 6: Cumulative acceptance rate and acceptance rate of MTP heads for the 2B Dense and 10B MoE models across multiple benchmarks. Here, H denotes the index of the MTP head. For the $K = 1$ group, the CAR is identical to the AR. For the $K = 4$ group, each cell reports CAR/AR (%) values. The best CARs of different MTP heads are highlighted in **bold**.

K	Model	Method (MTP)	Mean
	2B Dense	NTP (baseline)	11.06
	10B MoE	NTP (baseline)	13.49
1	2B Dense	MTP	11.28
		MTP-D	11.68
	10B MoE	MTP	13.75
		MTP-D	14.15
4	2B Dense	MTP	11.06
		MTP-D	11.13
	10B MoE	MTP	13.80
		MTP-D	13.79

Table 7: Mean accuracy of the main head averaged over seven benchmarks for the 2B Dense and 10B MoE models.

Strategy	Acc↑	MTP Loss↓			
		1	2	3	4
MTP-D	**11.13**	**2.671**	**2.777**	**2.839**	**2.887**
β_k 1.0	10.93	2.690	2.784	2.844	2.885
β_k 0.3	11.25	2.687	2.797	2.863	2.909
β_k 0.1	11.40	2.716	2.833	2.901	2.945

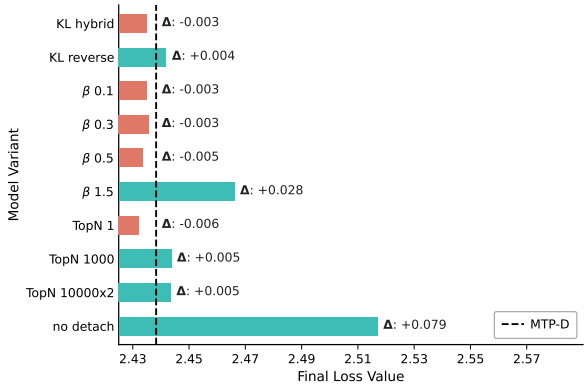
Table 8: Ablation results for MTP-D with 4 MTP head. The MTP-D configuration is: `detach`, Top$N = 10{,}000$, $\beta_k = 0.5$, forward KL divergence (bold).

F.1 Training-Free Looped Extension

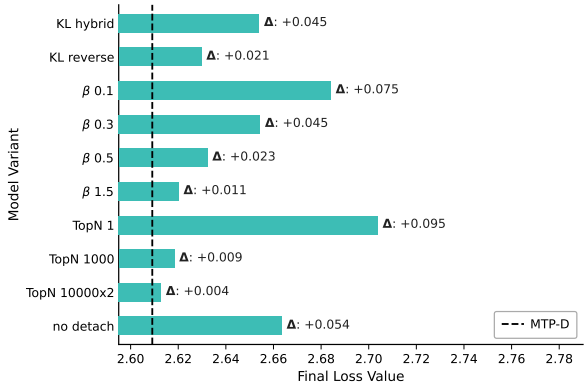
Figures 10 and 11 show the CAR and AR of MTP heads looped up to 8 under the training-free setting.

First, MTP heads at loop connection points exhibit a noticeable drop in accept rate compared to other heads. Taking AGIEval as an example, the accept rate of the 4-to-8 looped MTP drops sharply from approximately 80% at MTP head 4 to around 50% at MTP head 5. Despite this degradation, the accept rates remain at an acceptable level. Moreover, as the MTP heads scale from 5 to 8, their accept rates gradually recover and approach those of the originally trained heads 1 to 4. A similar trend is observed for the 1-to-8 looped MTP. These results indicate that the cascaded architecture of DeepSeek MTP, with its inherent structural consistency and input-output similarity, naturally supports scalability.

Furthermore, comparing MTP-D and DeepSeek MTP under the 1-to-8 training-free loop setting reveals that MTP heads trained with self-distillation exhibit substantially superior scalability. As shown in Figure 4(a), the cumulative accept rate of DeepSeek MTP collapses to merely 0.6% at MTP head 3, whereas MTP-D maintains 26.70%, en-



(a) Loss comparison of main head.



(b) Loss comparison of MTP head.

Figure 9: Ablation experiments for MTP-D and its variants with loss as metrics. Orange indicates losses better than MTP-D, while green indicates worse performance.

abling further scaling to a larger number of MTP heads. These results demonstrate that MTP-D significantly enhances distributional consistency between MTP heads and the main head, thereby endowing it with markedly improved scalability.

F.2 Continued Pre-trained Looped Extension

Figures 12 and 13 present the CAR curves for looped extension up to 8 and 16 MTP heads, respectively, under the continued pre-training setting.

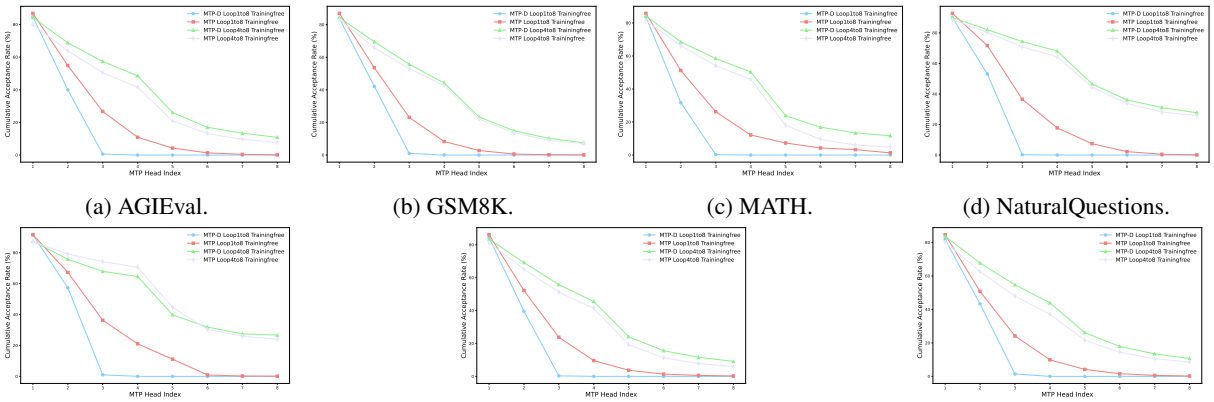
G Other Comparison

Existing speculative decoding methods can be broadly categorized into two paradigms: (1) *joint pre-training with extension*, where MTP draft heads are co-trained with the main model from scratch and subsequently extended to more heads; and (2) *post-hoc draft model training*, where a separate lightweight draft model (*e.g.*, Eagle3 (Li et al., 2026), Medusa (Cai et al., 2024)) is trained after the target LLM has been fully trained. While post-hoc methods offer flexibility for off-the-shelf LLMs, they tend to yield lower accept rates under the same parameter budget. In this work, we focus on the joint pre-training with extension paradigm, which is increasingly adopted in industrial-scale LLMs (*e.g.*, DeepSeek (DeepSeek-AI, 2026), GLM (Zeng et al., 2026), and MiMo (Xiaomi et al., 2025; Xiao et al., 2026)), and propose effective strategies for MTP head training and extension.

H Production-Scale Experimental Validation

To further validate the effectiveness of MTP-D, we apply our self-distillation method to the MTP extension stage of a production-scale pre-trained model with 1 MTP head, under production-level data and training infrastructure. The model is a 7B MoE architecture with the Muon optimizer (Liu et al., 2025a) and MLA attention (Deepseek et al., 2024). Using the vLLM inference framework (Kwon et al., 2023), we evaluate on three categories of benchmarks: Math (GSM8K (Cobbe et al., 2021)), Code (MBPP (Austin et al., 2021)), and Knowledge (SimpleQA (Wei et al., 2024)). The results are reported in Tables 9, 10, and 11.

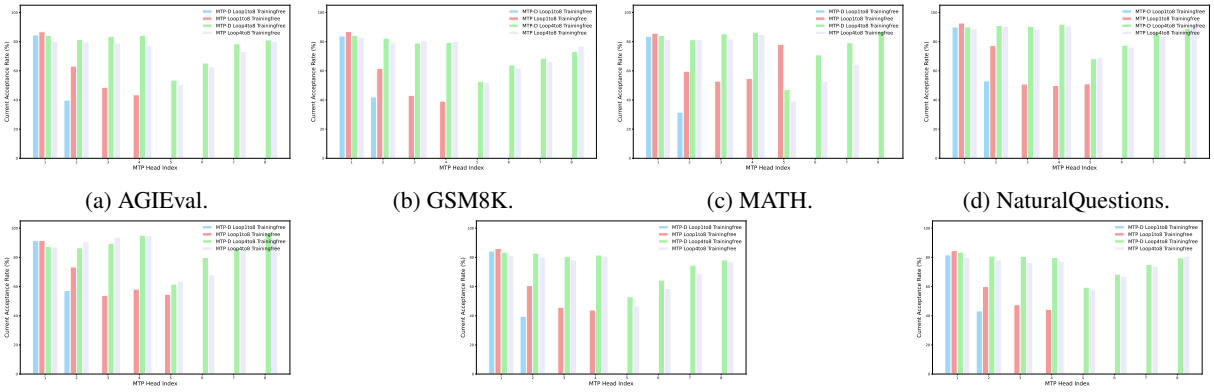
Overall, 4MTP-D achieves the best throughput across all three benchmarks, achieving accept lengths of 3.97, 4.02, and 3.45 on GSM8K, MBPP+, and SimpleQA, respectively. Specifically, compared to 4MTP, 4MTP-D improves tok/s, speedup, and acceptance length by 2.8%, 2.9%, and 0.9% on average, respectively. Furthermore, progressive MTP head extension yields consistent gains across all metrics, with speedup, acceptance length, and tok/s increasing monotonically from Baseline $\rightarrow$ 1MTP-D ($+0.75\times$, $+0.93$, $+24.2$) $\rightarrow$ 2MTP-D ($+1.15\times$, $+1.67$, $+37.2$) $\rightarrow$ 3MTP-D ($+1.40\times$, $+2.31$, $+45.1$) $\rightarrow$ 4MTP-D ($+1.73\times$, $+2.81$, $+55.7$). These results demonstrate that the combination of MTP-head scaling and self-distillation remains effective under production-scale models and deployment environments.



(a) AGIEval. (b) GSM8K. (c) MATH. (d) NaturalQuestions.

(e) SimpleQA. (f) SuperGPQA. (g) TriviaQA.

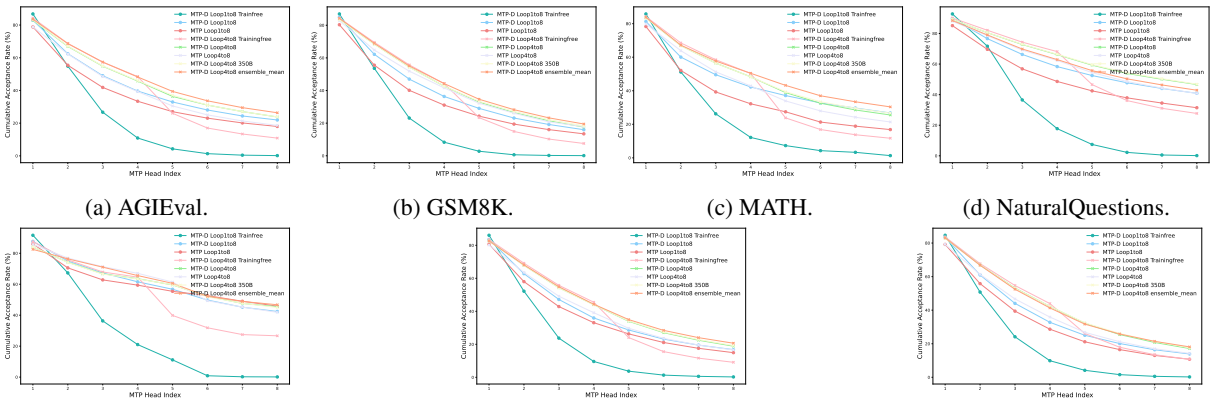
Figure 10: Cumulative accept rates on multiple pretraining benchmarks with the training-free looped MTP scaled up to 8.



(a) AGIEval. (b) GSM8K. (c) MATH. (d) NaturalQuestions.

(e) SimpleQA. (f) SuperGPQA. (g) TriviaQA.

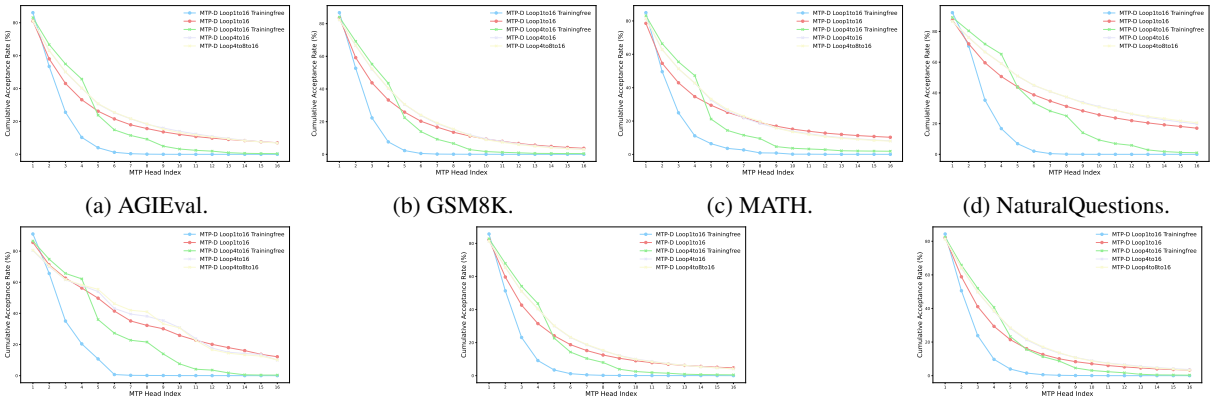
Figure 11: Accept rates on multiple pretraining benchmarks with the training-free looped MTP scaled up to 8.



(a) AGIEval. (b) GSM8K. (c) MATH. (d) NaturalQuestions.

(e) SimpleQA. (f) SuperGPQA. (g) TriviaQA.

Figure 12: Cumulative accept rates on multiple pretraining benchmarks with the looped MTP scaled up to 8.



(a) AGIEval. (b) GSM8K. (c) MATH. (d) NaturalQuestions.

(e) SimpleQA. (f) SuperGPQA. (g) TriviaQA.

Figure 13: Cumulative accept rates on multiple pretraining benchmarks with the looped MTP scaled up to 16.

Method	Heads	tok/s	Speedup	s/sample	Accept Length	Acceptance Rate (CAR / AR, %)				Draft%
						H_0	H_1	H_2	H_3	
Baseline	–	31.0	1.000×	8.287	–	–	–	–	–	–
1MTP-D	1	54.0	1.743×	4.790	1.934	93.4 / 93.4	–	–	–	7.2
2MTP-D	2	65.1	2.099×	3.967	2.695	92.5 / 92.4	77.0 / 83.2	–	–	17.1
3MTP-D_{sh}	3	73.9	2.383×	3.498	3.367	91.8 / 91.8	78.7 / 85.6	66.1 / 83.3	–	24.0
3MTP-D	3	75.5	2.436×	3.443	3.411	92.6 / 92.6	80.1 / 86.4	68.4 / 84.7	–	24.6
4MTP	4	91.0	2.930×	3.169	3.910	92.1 / 92.1	77.7 / 84.3	65.4 / 83.3	55.9 / 83.6	29.5
4MTP-D	**4**	**94.2**	**3.039×**	**2.754**	**3.969**	92.5 / 92.5	78.8 / 85.1	67.7 / 85.0	57.9 / 83.5	29.2

Table 9: Speculative decoding results on **GSM8K** (`max_tokens` = 256). "Accept Length" = average number of tokens accepted per decoding step. Acceptance-rate columns show CAR / AR rates per draft head H_i. 3MTP-D_{sh} shares parameters across heads; 4MTP uses standard MTP without distillation. **Bold**: best.

Method	Heads	tok/s	Speedup	s/sample	Accept Length	Acceptance Rate (CAR / AR, %)				Draft%
						H_0	H_1	H_2	H_3	
Baseline	–	33.3	1.000×	15.425	–	–	–	–	–	–
1MTP-D	1	60.5	1.818×	8.507	1.975	97.5 / 97.5	–	–	–	6.4
2MTP-D	2	76.9	2.310×	6.692	2.794	97.3 / 97.3	82.1 / 84.3	–	–	16.2
3MTP-D_{sh}	3	77.6	2.330×	6.692	3.450	97.2 / 97.2	84.3 / 86.6	63.5 / 74.4	–	24.1
3MTP-D	3	85.8	2.576×	6.040	3.495	97.3 / 97.3	85.8 / 88.2	66.4 / 76.5	–	23.9
4MTP	4	89.8	2.697×	5.863	3.987	96.8 / 96.8	84.0 / 86.7	64.3 / 75.5	53.5 / 78.4	29.5
4MTP-D	**4**	**93.2**	**2.800×**	**5.648**	**4.021**	97.1 / 97.1	84.7 / 87.2	66.0 / 76.8	54.4 / 77.9	29.4

Table 10: Speculative decoding results on **MBPP+** (`max_tokens` = 512). Column descriptions follow Table 9. **Bold**: best throughput.

Method	Heads	tok/s	Speedup	s/sample	Accept Length	Acceptance Rate (CAR / AR, %)				Draft%
						H_0	H_1	H_2	H_3	
Baseline	–	32.8	1.000×	1.962	–	–	–	–	–	–
1MTP-D	1	55.3	1.688×	1.168	1.876	87.6 / 87.6	–	–	–	6.6
2MTP-D	2	66.9	2.042×	0.963	2.520	86.7 / 86.7	65.3 / 75.2	–	–	15.8
3MTP-D_{sh}	3	61.0	1.863×	1.063	3.010	84.1 / 84.1	65.7 / 77.8	51.3 / 77.9	–	24.5
3MTP-D	3	71.2	2.174×	0.909	3.035	84.4 / 84.4	66.5 / 78.6	52.6 / 78.9	–	23.4
4MTP	4	76.0	2.319×	0.919	3.430	82.3 / 82.3	66.0 / 79.8	51.9 / 78.6	42.8 / 81.9	29.2
4MTP-D	**4**	**76.8**	**2.344×**	**0.847**	**3.445**	82.3 / 82.3	65.6 / 79.5	53.3 / 81.1	43.4 / 80.9	29.0

Table 11: Speculative decoding results on **SimpleQA** (`max_tokens` = 64). Column descriptions follow Table 9. **Bold**: best throughput.

Algorithm 1 Lossless Speculative Decoding

Input: Initial prompt sequence $\mathbf{x}$, Main model f, MTP heads $\{g_k\}_{k=1}^{K}$, Max new tokens N

Output: Generated sequence $\mathbf{x}$

```
L_prompt ← |x|    L_gen ← 0
C_step ← 0
for j = 1 to K do
    A_j ← 0  C_j^cmp ← 0
end
while L_gen < N do
    // Verification and Correction
    (P, h) ← f(x)  is_verified ← True
    if L_gen > 0 then
        C_step ← C_step + 1  m ← 0
        for j = 1 to K do
            i ← K − j + 1  C_j^cmp ← C_j^cmp + 1
            x̂ ← Sample(P_{|x|−i})
            if x̂ ≠ x_{|x|−i+1} then
                x ← x_{1:|x|−i} ‖ (x̂)
                h ← h_{1:|x|−i}
                is_verified ← False
                break
            end
            else
                m ← m + 1
            end
        end
        if m > 0 then
            for t = 1 to m do
                A_t ← A_t + 1
            end
        end
    end
    if EOS ∈ x then
        break
    end
    // Speculative Expansion
    if is_verified then
        x_{|x|+1} ← Sample(P_{|x|})
        x ← x ‖ (x_{|x|+1})
    end
    for k = 1 to K do
        (P̂_MTP, h) ← g_k(x, h)
        x_{|x|+1} ← Sample(P̂_MTP)
        x ← x ‖ (x_{|x|+1})
    end
    L_gen ← |x| − L_prompt
end
return x
```