Think in Sentences: Explicit Sentence Boundaries Enhance Language Model's Capabilities

Zhichen Liu, Yongyuan Li, Yang Xu*
Department of Computer Science and Engineering
Southern University of Science and Technology
liuzc2024@mail.sustech.edu.cn, xuyang@sustech.edu.cn

Abstract

Researchers have explored different ways to improve large language models (LLMs)' capabilities via dummy token insertion in contexts. However, existing works focus solely on the dummy tokens themselves, but fail to leverage the inherent sentence-level structure of natural language. This is a critical oversight, as LLMs acquire linguistic capabilities through exposure to human-generated texts, which are inherently structured at the sentence level. Motivated by this gap, we propose an approach that inserts delimiters at sentence boundaries in LLM inputs, which not only integrates dummy tokens into the context, but also facilitates LLMs with sentence-by-sentence processing behavior during reasoning. Two concrete methods: (1). In-context learning and (2). Supervised fine-tuning are experimented using 7B models to 600B Deepseek-V3. Our results demonstrate consistent improvements across various tasks, with notable gains of up to 7.7% on GSM8k and 12.5% on DROP. Furthermore, the fine-tuned LLMs can incorporate sentence awareness evidenced by their internal representations. Our work establishes a simple yet effective technique[1] for enhancing LLM's capabilities, offering promising directions for cognitive-inspired LLM enhancement paradigm.

1 Introduction

Sentence-level structure has long been a cornerstone of early neural language models: Skip-thought vectors (Kiros et al., 2015) were trained to reconstruct neighboring sentences, while BERT's next-sentence prediction task (Devlin et al., 2019) proved indispensable for downstream performance by encoding inter-sentence coherence. Yet with the rise of large language models (LLMs), whose success stems primarily from scaling pretraining on massive unstructured text, sentence boundaries have been increasingly sidelined, treated as indistinguishable from ordinary tokens in the token-by-token processing pipeline. This oversight is striking: human language generation relies on incremental, sentence-by-sentence cognition, but LLMs learn from the continuous text that results from this process, creating an inherent misalignment between human cognitive mechanisms and model input processing.

Against this backdrop, we argue that re-emphasizing sentence-level information offers a largely untapped avenue to enhance LLMs, especially for "free-lunch" (cost-neutral) improvements. Since GPT series (Brown et al., 2020; Ouyang et al., 2022) established modern LLM training paradigms, efforts to improve performance have followed two main paths: training-time scaling (e.g., scaling laws for model/data size (Kaplan et al., 2020; Hoffmann et al., 2022; Chowdhery et al., 2022; Touvron et al., 2023)) and test-time scaling (e.g., instruction for thinking step-by-step (Wei et al., 2022; Yao et al., 2023), or reinforcement learning (RL) for self-reflection (Renze and Guven, 2024; Qi et al., 2024; Zhang et al., 2024)). However, these methods incur substantial costs: training-time scaling demands massive compute or data, while test-time scaling increases inference latency and token consumption.

To address this, recent work (Goyal et al., 2024) proposed inserting special "pause" tokens into contexts as a free-lunch alternative, obtaining performance gains without extra costs. Yet this approach suffers from limited robustness and generality: dummy token placement lacks linguistic priors, requiring manual tuning across tasks and does not leverage the inherent structure of human language. This gap raises our research question: **Can we design an effective strategy that harnesses sentence-level linguistic priors to robustly enhance LLM performance?**

* Corresponding author.
[1] A demonstrative code repository is provided: https://github.com/CLCS-SUSTech/think-in-sentence.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2275–2288
July 2-7, 2026

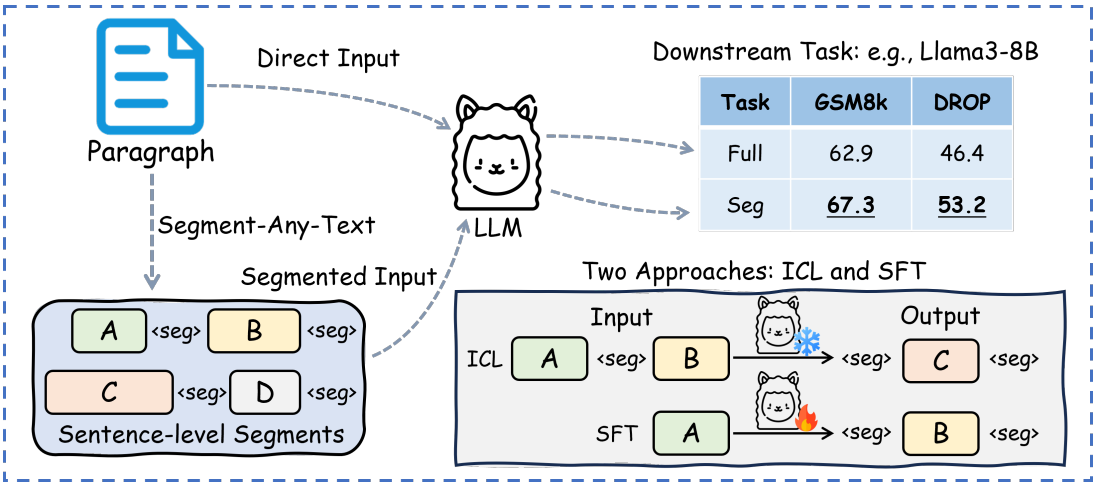

Figure 1: **Overview of Sentence-Level Inference:** We insert delimiters at sentence boundaries to enable LLMs to "pause and integrate context" during inference. Two approaches are proposed: (1) In-Context Learning (ICL): LLMs infer with delimiter placement from exemplars in long contexts; (2) Supervised Fine-Tuning (SFT): LLMs learn sentence-segmented patterns via delimiter-inserted training data. For Llama3-8B-Instruct, this approach improves performance by ∼4.4% on GSM8k and ∼6.8% on DROP over unsegmented inputs.

1.1 Main Contributions

We introduce a *sentence-level inference* paradigm that accentuates sentence boundaries via task-agnostic delimiters, bridging the gap between LLMs' token-by-token processing and the more human-like sentence-by-sentence cognition process. Our key contributions are threefold:

Paradigm Innovation: Unlike explicit reasoning prompts (e.g., CoT), we implicitly enhance inference by inserting delimiters at sentence boundaries. These delimiters act as "inference anchors" – not mere grammatical markers – to trigger a "context integration → next-step planning" cycle at the end of each sentence, thereby simulating human post-sentence reflection.

Dual Implementation: We propose two complementary methods to instantiate this paradigm: (a) ICL, where LLMs learn delimiter placement from contextual exemplars (suited for long-input scenarios); (b) SFT, where models are fine-tuned on delimiter-inserted, sentence-segmented data (for short-input tasks). Both methods require minimal overhead, qualifying as free-lunch strategies.

Empirical and Mechanistic Insights: Across model scales (7B to 600B), our methods yield consistent downstream gains (e.g., ∼7.7% on GSM8k, ∼12.5% on DROP). Ablations reveal that: (i) structured delimiters outperform arbitrary tokens for ICL; (ii) sentence-level segmentation is the optimal granularity; (iii) gains arise from synergy between LLMs' Chain-of-Thought reasoning and sentence-level inference. We further validate mechanisms via attention map visualization, showing that delimiters capture more information than normal tokens.

1.2 Related Works

Test-Time Scaling for LLMs Test-time scaling aims to improve performance by extending inference "thinking time." CoT (Wei et al., 2022) and ToT (Yao et al., 2023) use instruction prompts to elicit step-by-step reasoning, while follow-ups add self-verification (Renze and Guven, 2024) or RL-driven search (e.g., MCTS (Qi et al., 2024; Zhang et al., 2024)) to explore solution spaces. RL has also been applied to training (e.g., DeepSeek R1 (DeepSeek-AI et al., 2025a), Kimi K1.5 (Team et al., 2025)) to teach self-exploration. While effective, these methods drastically increase inference latency and token costs, limiting deployment.

Pause/Dummy Token Strategies Goyal et al. (2024) pioneered cost-neutral test-time scaling via inserting pause tokens, showing gains in pretraining/fine-tuning for 1B-scale models. However, their approach has critical limitations: (i) no validation on large-scale LLMs (≥7B parameters); (ii) token count requires task-specific manual tuning; (iii) lack of linguistic priors leads to limited robustness across tasks.

Sentence-Level Granularity in LLMs Recent work has revisited sentence-level structure for LLMs, though with different goals. Qiu et al.

(2025) proposed a sentence-level reward model that outperforms token/response-level alternatives for alignment. Zheng et al. (2025) replaced GRPO's (Shao et al., 2024) token-level objective with sequence-level optimization, improving stability. These works validate the value of sentence-level paradigm in their objectives, while our work targets inference-time, free-lunch performance gains via sentence-level inference.

Beyond sentence-level boundaries, recent studies have also explored incorporating finer-grained syntactic and semantic structures into prompt engineering. For instance, leveraging syntax trees has shown benefits in specific structured tasks like aspect-based sentiment analysis (Labate and Cozman, 2024) and semantic infusing (Yin et al., 2024); however, extending such complex syntactic augmentations to general-purpose reasoning scenarios remains an open and promising direction.

2 Method

Our central hypothesis is that by explicitly modeling sentence boundaries, we can induce a more structured, sentence-by-sentence reasoning process in LLMs, thereby enhancing their performance on complex downstream tasks. To this end, we reformulate the standard language modeling objective to incorporate sentence-level structural information. We introduce a special delimiter token, denoted as "x_{seg}", which is inserted at the end of each sentence. This transforms a text sequence T:

$$T = [t_1, t_2, t_3, ..., t_n] \tag{1}$$

into a structurally-annotated sequence S:

$$S = [s_1, x_{seg}, s_2, x_{seg}, ..., s_n, x_{seg}] \tag{2}$$

Here, each s_i represents a sentence from the original text T, consisting of multiple tokens t. Consequently, the model's objective is no longer limited to predicting the next token in a flat sequence; it further entails learning the optimal timing to generate the delimiter "x_{seg}". In doing so, the model performs implicit sentence segmentation as part of its generative objective. Despite simplicity, this modification effectively encourages the model to better recognize and leverage sentence-level semantics. We explore two primary strategies to implement this capability in LLMs: In-Context Learning and Supervised Fine-Tuning.

2.1 Sentence-Aware Prompting via In-Context Learning

In-Context Learning (ICL) offers a lightweight, inference-time approach to elicit desired behaviors from LLMs without updating the model weights. We use ICL to guide the model to adopt a sentence-delimited generation style. This is achieved by including few-shot examples in the prompt, where each sentence within the demonstration is explicitly terminated by the predefined delimiter. The model is then tasked with completing the final, incomplete example. The generation process follows the standard auto-regressive objective, but the context primes the model to continue the observed pattern:

$$y_t = \underset{y}{\text{argmax}}\ P(y|C_{\text{few-shot}}, Q, y_{<t}; \theta) \tag{3}$$

where $C_{\text{few-shot}}$ is the context containing sentence-delimited examples, Q is the user's query, and θ represents the frozen model parameters. According to Dong et al. (2024), the model learns from analogy to structure the intermediate reasoning and the output in a sentence-by-sentence manner. As validated in experiments in Section 3, this ICL-enabled structured generation process leads to stable performance gains. However, the efficacy of ICL is contingent on the availability of sufficient context length for demonstrations, limiting its applicability in zero-shot or context-constrained scenarios.

2.2 Internalizing Sentence Structure via Supervised Fine-Tuning

To overcome the limitations of ICL and to build a more robust, inherently sentence-aware model, we propose a Supervised Fine-Tuning (SFT) strategy. This approach aims to internalize the sentence-level structural prior directly into the model's parameters, making the behavior more *intrinsic* rather than context-dependent.

First, we curate a fine-tuning dataset by systematically preprocessing a collection of large-scale text corpora, with delimiters inserted at every sentence boundary. Then we fine-tune the language model on this modified dataset using the standard causal language modeling (CLM) objective. The loss function is rewritten to reflect the sentence-level training objective as follows:

$$\mathcal{L}_{SFT}(\theta) = \sum_{s' \in S}^{S} \sum_{i=1}^{|s'|} \log P(t_i | t_{<i}; \theta) \tag{4}$$
$$\text{where } s' = [s, x_{seg}] \text{ and } t_{|s'|} = x_{seg}$$

Through the training process, the model learns to predict sentence boundaries, which it integrates as a fundamental component of language generation. For implementation, we add the delimiter as a special token into the tokenizer, thereby introducing new embeddings and LM head weights. Compared to ICL, the SFT approach yields a model that natively generates sentence-delimited text, making it more effective for zero-shot applications and better aligned with real-world deployment scenarios where concise prompts are preferred.

3 Experiments

We conduct a comprehensive suite of experiments to validate our central hypothesis: inducing sentence-level awareness in LLMs enhances their reasoning capabilities. We aim to answer two concrete research questions:

1. **RQ1:** Does prompting with sentence delimiters during inference (i.e., the ICL approach) improve performance on reasoning tasks across various model scales?

2. **RQ2:** Can such sentence-aware behavior be permanently internalized via fine-tuning (i.e., the SFT approach), and how does this compare to standard fine-tuning and other methods?

3.1 Experiment Setup

Models. Our experiments span various sizes of LLMs. For ICL, we evaluate open-source LLMs including **LLaMA3-8B-Instruct** (Grattafiori et al., 2024) and **Qwen2-7B-Instruct** (Yang et al., 2024), a larger LLM **Qwen2.5-72B-Instruct** (Qwen et al., 2025), and a SOTA LLM, **DeepSeek-V3** (DeepSeek-AI et al., 2025b), via its API[2]. For SFT, we perform full-parameter fine-tuning on **LLaMA3-8B-Base** using 8×NVIDIA L40 GPUs.

Datasets and Tasks. We use a diverse suite of benchmarks targeting on different reasoning types:

- **Mathematical Reasoning:** GSM8k (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021b).
- **Reading Comprehension:** DROP (Dua et al., 2019), which requires reasoning over paragraphs.
- **General Knowledge Understanding:** MMLU (Hendrycks et al., 2021a) and its more challenging successor, MMLU-Pro (Wang et al., 2024).

[2]https://api-docs.deepseek.com/

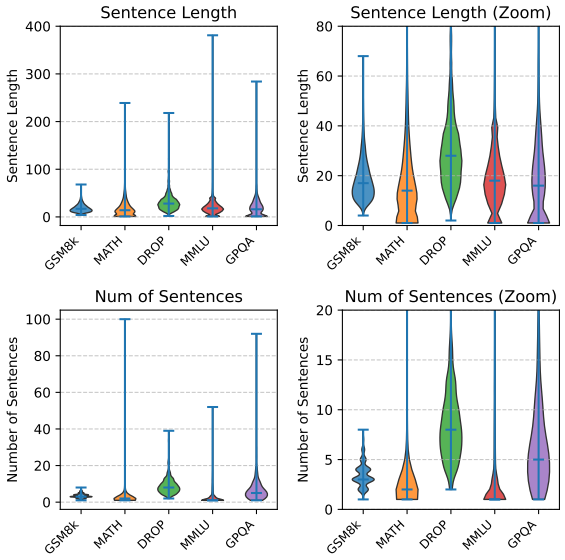

Figure 2: The distributions of sentence lengths and number of sentences for each dataset. The left column figures are the origin distribution, and the right column figures are zoomed-in views. Horizontal bars indicate medians and extrema. Sentence lengths are counted by number of tokens, from the Llama3 tokenizer.

- **Expert-Level QA:** GPQA (Rein et al., 2024), a dataset of graduate-level questions.
- **Code Generation:** HumanEval (Chen et al., 2021) for Python code synthesis.

For SFT, we use a curated subset of the TULU3 dataset (Lambert et al., 2025), from which we exclude safety, multilingual, and table-related data, to focus on general instruction following. Figure 2 shows a statistical overview of sentence counts and lengths for the five datasets.

Implementation Details. For the purpose of identifying sentence boundaries, we apply the SAT-12L-SM model (Frohmann et al., 2024), a state-of-the-art sentence segmentation tool, to preprocess all text data, which return sentence boundaries as token positions. Detailed usage see Appendix C. Then we insert the delimiter token "x_{seg}" at these boundaries. For SFT, delimiter is added as a new token to the tokenizer, whose corresponding embeddings are learned during training. The evaluation protocols, including few-shot settings for Chain-of-Thought (CoT) prompting, are detailed in Appendix A. Unless otherwise specified, all results are reported using exact match accuracy, with Pass@1 for HumanEval. To ensure a fair comparison, chat templates are disabled for all local evaluations.

Dataset	Qwen2-7B-Inst			Llama3-8B-Inst			Qwen2.5-72B-Inst			Deepseek-V3		
	base	seg	Δ	base	seg	Δ	base	seg	Δ	base	seg	Δ
MMLU	64.43	69.96	+5.53%↑	62.89	67.28	+4.39%↑	86.64	86.40	-0.24%↓	74.04	74.82	+0.78%↑
GSM8k	73.92	81.65	+7.73%↑	75.51	78.01	+2.5%↑	90.14	91.96	+1.82%↑	95.00	95.30	+0.3%↑
MATH	53.33	54.30	+0.97%↑	32.60	32.26	-0.34%↓	73.04	75.78	+2.74%↑	89.40	90.60	+1.2%↑
DROP	38.14	50.64	+12.50%↑	46.39	53.16	+6.77%↑	58.74	60.38	+1.64%↑	75.10	79.10	+4%↑

Table 1: In-Context Learning results. We compare the performance of vanilla inference (base) against ICL (seg), delimiter here is "<seg>". Δ denotes the absolute improvement. Our method yields consistent gains across models and tasks, with particularly strong improvements on smaller models and in reading comprehension task.

	MMLU	GSM8k	MATH	DROP	MMLU-pro	GPQA	HumanEval
Std-FT	59.02	72.48	30.86	48.50	34.25	26.93	56.71
Pause-FT	56.11	**75.44**	**33.50**	**55.97**	35.71	24.16	-
Seg-FT	**60.13**	74.91	31.58	54.26	**40.71**	**27.43**	**62.80**

Table 2: Supervised Fine-Tuning results on LLaMA3-8B-Base. Our method (Seg-FT) is compared against standard fine-tuning (Std-FT) and pause-token fine-tuning (Pause-FT). Best performance is in **bold**, and results outperforming the Std-FT baseline are underlined. Our approach demonstrates superior robustness and generalization.

Baselines. For ICL, the main baseline is the vanilla performance of each model without inserting delimiters. For SFT, our method is to fine-tune a Llama3-8B-Base model on the curated TULU3 dataset with delimiters inserted, which we indicated **Seg-FT**. It is compared with two baselines: Std-FT, a standard fine-tuning baseline, which fine-tunes the same model on the original TULU3 subset *without* inserting delimiters; Pause-FT, a pause-token fine-tuning baseline, which fine-tunes the same model following the settings of `StdPT_PauseFT` in Goyal et al. (2024), with 10 pause tokens inserted in both training and inference stage.

3.2 Results Analysis

3.2.1 RQ1: ICL Boosts Reasoning

As shown in Table 1, inference with sentence-delimited prompts consistently improves performances across nearly all configurations.

Key Observation 1: Smaller models benefit disproportionately. The 7B-level LLMs (Qwen2-7B, LLaMA3-8B) exhibit the most significant gains, such as a +7.73% on GSM8k for Qwen2-7B and +5.53% on MMLU. This suggests that explicit structural guidance is particularly effective for LLMs with less capacity, helping them organize their reasoning process more effectively. For larger, more capable LLMs (such as Qwen2.5-72B and DeepSeek-V3), the improvements are more modest but still present (smaller in MMLU but larger in MATH and DROP), indicating that even powerful LLMs can benefit from our sentence delimiters-inserted prompting.

Key Observation 2: Performance gains correlate with task types. The most dramatic improvement is observed on DROP (+12.5% for Qwen2-7B), a reading comprehension task that requires tracking information across multiple sentences within a context. A reasonable explanation is that by explicitly segmenting sentences, it enable the LLM to process individual facts encoded in separate sentences more effectively, and better understand their relationships, which is important for this type of task.

3.2.2 RQ2: SFT Internalizes Robust Sentence Awareness

Table 2 shows the results of the SFT approach, yielding several interesting insights. Our method (Seg-FT) has overall better performance than the baselines (Std-FT and Pause-FT).

Key Observation 3: Sentence-based SFT is more robust than pause-based SFT. Our method (Seg-FT) consistently outperforms the Std-FT baseline across all seven benchmarks. In contrast, Pause-FT, while staying strong on procedural tasks like GSM8k and MATH, suffers from performance degradation in knowledge-intensive QA tasks like MMLU and GPQA. This suggests that while simply "pausing" can aid methodical computation, it may disrupt the model's access to or reasoning over its stored knowledge. Our method, by encap-

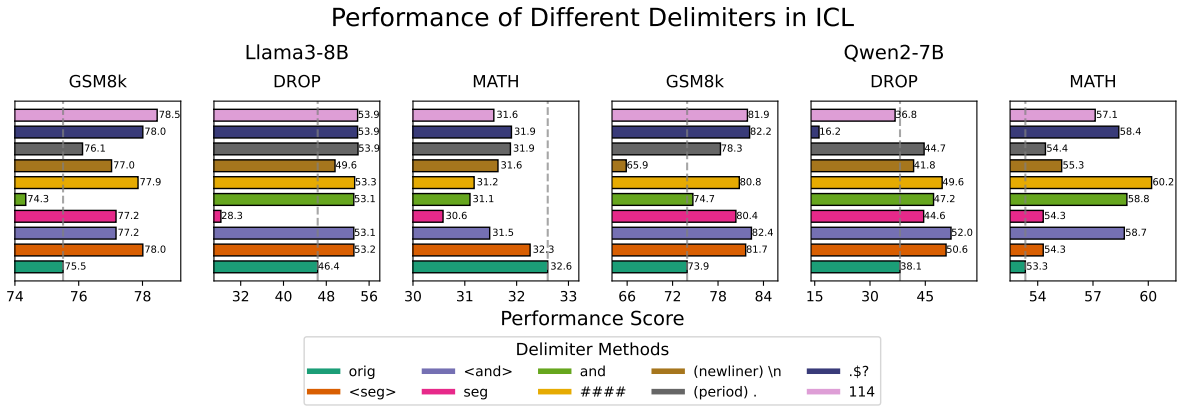

Figure 3: Performance of different delimiter choices in ICL across three datasets. More structured delimiters could consistently yield a better performance, demonstrating the value of a clear, non-semantic structural signal. "orig." denotes the baseline without any delimiters.

sulating the generation process into meaningful linguistic units (sentences), seems to provide a more robust and universally beneficial structural prior.

Surprising Observation: Sentence awareness generalizes to code. A striking result is the +6.09% absolute improvement on HumanEval. During inference, we observed that the Seg-FT model is able to insert delimiters within codes. As there exhibits some similar patterns between human language and python code, for example, using newliner as delimiters, it enables the model to learn from the commonalities between the two, thereby acquiring the ability to generalize the segmentation of natural language to code.

4 Ablation Studies and Analysis

To analyze what factors contribute to our method's success, we conduct a series of targeted ablation studies. These experiments are designed to answer three fundamental questions: (1) What properties make an effective delimiter? (2) Is sentence-level segmentation truly the optimal strategy for placing these delimiters? (3) What are the underlying mechanisms of delimiters enhancing model performances?

4.1 On the Importance of a Clear Structural Signal: Delimiter Choice

In general, we find that the choice of delimiter is non-trivial, and its form and semantics can influence how the model interprets it. We hypothesize that an ideal delimiter should function as a pure structural marker, which is irrelevant of the semantic content of the text. To test this hypothesis, we evaluate a spectrum of delimiters under the ICL setting: syntactically distinct tokens ["<seg>", "<and>", "####"] (structured), common words ["seg", "and"] (semantic), punctuation used in human text ["\n", "."] (delimiters in natural language), a numeric token ["114"] and a meaningless symbol string [".&?"] (arbitrary).

As illustrated in Figure 3, our hypothesis is supported by the results. Structured delimiters consistently achieve the highest performance, which are the only delimiters that outperform baseline in all tasks. In contrast, semantic delimiters like "and" and "seg" often perform worse. This is presumably due to the semantic ambiguity they create, which force the model to disambiguate whether the token is a structural marker or part of the content. Arbitrary and natural delimiters show mixed results; while they outperform the baseline in some cases, the effect is inconsistent. It confirms that the performance gain does not stem from any specific semantic meaning, but rather from the introduction of a regular, discernible pattern. The advantage of structured tokens like "<seg>" resides in their function to provide a less ambiguous signal of sentence boundaries – this enables the model to decouple structural processing from semantic reasoning.

4.2 On the Optimality of Granularity: Sentence vs. Alternative Segmentations

Having established the role of the delimiter's form, we now investigate its placement. Is segmentation at the sentence level inherently better than other granularities? We explore two alternatives: fixed-length chunking and random placement.

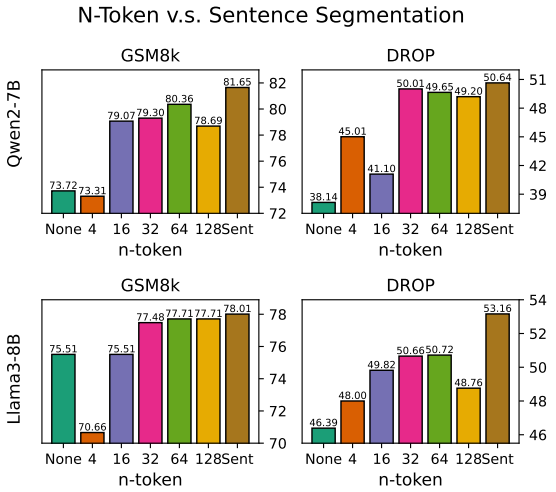

Figure 4: Sentence segmentation (Sent) vs. fixed n-token chunking. Sentence-level segmentation consistently outperforms fixed-chunking strategies, whose effectiveness decrease when the chunk size (n) is either too large or too small, only peaking when n is close to the majority sentence length.

Comparison with Fixed-Length Chunking. We replace sentence segmentation with a simple heuristic: inserting a delimiter every n tokens. Figure 4 reveals a clear pattern: as n increases, performance rises first, then falls. Very fine-grained chunking (e.g., $n = 4, 8$) is detrimental, as it fragments coherent semantic units within sentences. At the other end, very coarse-grained chunking (e.g., $n = 128$) makes the structural signals too sparse to effectively guide step-by-step reasoning. The optimal performance is achieved within the range $n \in [32, 64]$, which covers the typical sentence lengths in our test data (see Figure 2). This strongly suggests that sentence is the "natural" unit of model reasoning: it balances between semantic integrity and the structural guidance function, which is a perfect analogy to how human process information, e.g., cognitive chunking[3].

Comparison with Random Placement. To isolate the effect of delimiter positioning from the mere presence of additional tokens, we conducted a control experiment. For each input, we inserted the same number of delimiters as in sentence segmentation, but placed them at random positions. Results in Figure 5 show that even random insertion yields a modest improvement over the baseline. This indicates what we term a minor "dummy token" effect: any regular interruption can slightly

[3] https://dictionary.apa.org/chunking

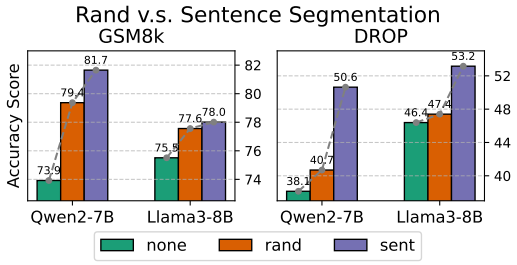

Figure 5: Sentence-level vs. random delimiter placement. Meaningful placement at sentence boundaries contributes more to the performance gains, far surpassing the minor effect of random insertions.

alter the model's processing. However, sentence-level placement consistently and significantly outperforms random placement. Therefore, we can conclude that the performance gain is not an artifact of adding extra tokens randomly, but is largely driven by placing delimiters at sentence boundaries – positions that are meaningful and aligned with linguistic structure.

4.3 Probing the Mechanism: Reasoning and Attention

Why does sentence-level segmentation work so effectively? We investigate the mechanism from two perspectives: its role in the reasoning process and its effect on the model's attention patterns.

Enhancing Deliberative Reasoning. We hypothesize that our method primarily benefits multi-step, deliberative reasoning rather than direct knowledge recall. To test this, we evaluate our fine-tuned model (Seg-FT and Std-FT) on MMLU using two zero-shot evaluation protocols: (1) Prob-based, which measures the model's immediate likelihood of the correct answer token, thereby probing knowledge recall; and (2) CoT-based, which prompts the model to generate a reasoning chain before the answer, hence probing deliberative reasoning.

	Std-FT	Seg-FT	Improvement
Prob	61.90	61.19	-0.71%
CoT	59.02	60.13	+1.12%

Table 3: MMLU zero-shot performance of SFT models under two evaluation protocols. The benefits of our method manifest exclusively in the CoT setting, highlighting its role in enhancing deliberative reasoning.

Table 3 shows a clear divergence. In the Prob-based setting, our method provides no benefit and

even causes a slight degradation. However, in the CoT setting, it yields a clear improvement of +1.12%. This result suggests that sentence-level delimiters do not simply improve the model's capabilities in retrieving static knowledge. Instead, the primary improvements are related to the dynamic, step-by-step reasoning process.

Attention as an Explanatory Lens. To visualize the mechanism in terms of internal representations, we analyze the model's attention patterns. Examples of attention heatmaps (see Appendix F) show that delimiter tokens act as focal points, drawing significant attention from subsequent tokens within the sequence.

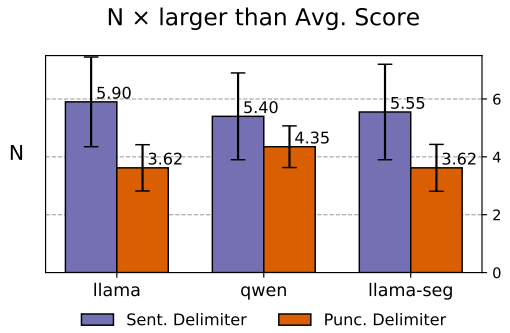

Figure 6: Relative attention scores for different delimiter types on the GSM8k dataset. Our delimiter (Sent. delimiter) receives significantly higher attention than both the sentence average (N× larger than avg.) and traditional punctuation delimiters (punc. delimiter).

For quantitative analysis, we compute the average attention paid to delimiter tokens by the final token of each sentence, and compare it against the attention paid to other tokens. As shown in Figure 6, our special delimiter (sent. delimiter) receives substantially higher attention than other tokens on average. Interestingly, it attracts significantly more attention than natural punctuations (punc. delimiter) like periods or newlines. It indicates that the model has learned to treat the delimiter token as a more reliable "signpost" for demarcating the units of thought, compared to natural punctuation – which is ambiguous and semantically overloaded. These delimiters thus function effectively as structural anchors, which the model can leverage to organize information flow during inference.

5 Conclusions

In this study we explore how explicitly modeling sentence structure in input can serve as a scaffold for enhancing the reasoning capabilities of Large Language Models in depth. We introduce a simple yet effective paradigm: teaching models to generate explicit boundary delimiters via in-context learning or fine-tuning. We validate the proposed methods through experiments on two directions: a lightweight, inference-time In-Context Learning strategy; and a more robust Supervised Fine-Tuning method that internalizes prior knowledge on sentence structures directly into the model's parameters.

Our experiments are comprehensive in terms of model size, spanning from 7B to over 600B parameters, revealing consistent and significant performance gains across a diverse suite of reasoning benchmarks, including improvements of up to 7.7% on GSM8k and 12.5% on DROP. Our ablation studies further shed light on three key findings: (1) structurally distinct, non-semantic delimiters yield best effectiveness; (2) sentence is the optimal granularity for segmentation, outperforming both finer and coarser chunking strategies; and (3) the primary mechanism underlying the improvement is in facilitating of deliberative, step-by-step reasoning, a conclusion supported by both comparative analysis and attention visualization.

Beyond improving downstream task performance, our work also introduces a novel approach to structured text generation. By training LLMs to natively generate sentence-delimited output, we eliminate the computational overhead of post-hoc segmentation–a common requirement in applications like text-to-speech, retrieval-augmented generation, and controllable decoding. Therefore, this study validates a feasible pathway towards more efficient, structurally-aware, and capable language models, laying the ground for potential future explorations in cognitive-inspired LLM architectures.

Looking forward, we outline several promising research avenues for future research. Extending our SFT approach to the pre-training stage could potentially instill sentence awareness as a basic capability in foundation models. Furthermore, exploring the applicability of this method to low-resource languages and specialized domains (e.g., legal or medical texts) will be critical for assessing its universality. Finally, enabling models to perform self-segmentation has the potential to yield more adap-

tive and resource-efficient implementations.

6 Limitations

While our findings are promising, this study has several limitations that represent important directions for future work.

Generalization of Segmentation Methods. Our experiments primarily rely on a state-of-the-art neural sentence segmenter (SaT). The robustness of our approach when using alternative segmentation methods, such as rule-based methods, or even the LLM's own self-segmentation capabilities, remains an open question. Investigating this is crucial for understanding the method's applicability in diverse, potentially resource-constrained production environments.

Validation at Larger Scales and Pre-training. Although our ICL experiments include very large models, our supervised fine-tuning was conducted on 7B-level LLMs due to resource constraints. A full investigation of how sentence-aware fine-tuning interacts with scaling laws at a larger scale is a necessary next step. Furthermore, while our SFT results suggest strong potential, the ultimate impact of incorporating sentence-level objectives during the pre-training phase has yet to be empirically verified.

Deeper Interpretability. Our analysis, based on attention scores and performance on reasoning-centric tasks, provides initial evidence for the mechanism behind our method's success. However, a more profound understanding is needed. Employing more advanced interpretability techniques, such as causal mediation analysis or probing for specific linguistic features in neuron activations, could more definitively trace how explicit structural signals modulate the model's internal computations and lead to improved reasoning.

Acknowledgments

We sincerely thank all the reviewers for their feedback on the paper. This study is funded by Shenzhen Science and Technology Program (No. JCYJ20240813094612017) and Guangdong Province ZJRC Program (No. 2024QN11X145).

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, et al. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, et al. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, et al. 2022. Palm: Scaling language modeling with pathways. *Preprint*, arXiv:2204.02311.

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

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, et al. 2025a. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, et al. 2025b. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arXiv:1810.04805.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, Baobao Chang, Xu Sun, Lei Li, and Zhifang Sui. 2024. A survey on in-context learning. *Preprint*, arXiv:2301.00234.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *Preprint*, arXiv:1903.00161.

Markus Frohmann, Igor Sterner, Ivan Vulić, Benjamin Minixhofer, and Markus Schedl. 2024. Segment any text: A universal approach for robust, efficient and adaptable sentence segmentation. *Preprint*, arXiv:2406.16678.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. 2024. Think before you speak: Training language models with pause tokens. *Preprint*, arXiv:2310.02226.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. *Preprint*, arXiv:2009.03300.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, et al. 2022. Training compute-optimal large language models. *Preprint*, arXiv:2203.15556.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. *Advances in Neural Information Processing Systems*, 28.

Anton Bulle Labate and Fabio Gagliardi Cozman. 2024. Infusing prompts with syntax and semantics. *Preprint*, arXiv:2412.06107.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, et al. 2025. Tulu 3: Pushing frontiers in open language model post-training. *Preprint*, arXiv:2411.15124.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, et al. 2022. Training language models to follow instructions with human feedback. *Preprint*, arXiv:2203.02155.

Zhenting Qi, Mingyuan Ma, Jiahang Xu, Li Lyna Zhang, Fan Yang, and Mao Yang. 2024. Mutual reasoning makes smaller llms stronger problem-solvers. *Preprint*, arXiv:2408.06195.

Wenjie Qiu, Yi-Chen Li, Xuqin Zhang, Tianyi Zhang, Yihang Zhang, Zongzhang Zhang, and Yang Yu. 2025. Sentence-level reward model can generalize better for aligning llm from human preference. *Preprint*, arXiv:2503.04793.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Matthew Renze and Erhan Guven. 2024. The benefits of a concise chain of thought on problem-solving in large language models. In *2024 2nd International Conference on Foundation and Large Language Models (FLLM)*, page 476–483. IEEE.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. 2025. Kimi k1.5: Scaling reinforcement learning with llms. *Preprint*, arXiv:2501.12599.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, et al. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, et al. 2024. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Preprint*, arXiv:2305.10601.

Wen Yin, Cencen Liu, Yi Xu, Ahmad Raza Wahla, Huang Yiting, and Dezhang Zheng. 2024. SynPrompt: Syntax-aware enhanced prompt engineering for aspect-based sentiment analysis. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 15469–15479, Torino, Italia. ELRA and ICCL.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. 2024. Rest-mcts*: Llm self-training via process reward guided tree search. *Preprint*, arXiv:2406.03816.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. 2025. Group sequence policy optimization. *Preprint*, arXiv:2507.18071.

A Evaluation Settings

Detailed evaluation settings of n-shot and CoT in ICL and SFT experiments are as follows:

- MMLU: 4-shot CoT for ICL, 0-shot CoT for SFT
- MMLU-Pro: 5-shot CoT for SFT
- GSM8k: 8-shot CoT for 7B-level LLMs and 4-shot CoT for large LLMs for ICL, 8-shot CoT for SFT
- MATH: 4-shot CoT for both ICL and SFT
- DROP: 3-shot for both ICL and SFT (DROP requires no CoT)
- GPQA: 0-shot CoT for SFT
- HumanEval: 0-shot for SFT (completion task cannot apply CoT)

B Combination between SFT with and without ICL

In experiments, we assumed that the model obtained from sentence-segmented SFT would be used with ICL during inference on downstream tasks, which means the input of SFT model is well-segmented. This section explore whether a well-segmented input is strictly required by the SFT model.

	GSM8k	DROP
no-seg	71.42	50.90
seg	74.91	54.26

Table 4: Comparison between SFT models with segmented input (seg) and raw input (no-seg).

Using the fine-tuned Llama3-8B model in Table 2, we evaluated its performance on GSM8k and DROP under two conditions: with sentence-segmented input, and with raw, unsegmented input. As shown in Table 4, the performance with segmented input is significantly better than without segmentation. This indicates that the SFT model has internalized the delimiter-augmented reasoning format; removing the delimiters leads to a distribution mismatch between training and evaluation, resulting in the performance degradation.

C Details about Sentence Segmentation Model

All sentence segmentation is performed using `wtpsplist`, with default segmentation parameter [4]. Some details about the model SAT-12L-SM's usage are listed below:

- stride: 256
- block_size: 512
- pad_last_batch: False
- weighting: uniform
- model size: ∼300M

D SFT training details

The SFT training parameters are listed below:

```
trainer:
    use_flash_attn: true
    max_seq_length: 2048
    train_batch_size: 128
    learning_rate: 5.0e-06
    lr_scheduler_type: linear
    warmup_ratio: 0.03
    weight_decay: 0.0
    num_train_epochs: 1
deepspeed:
    zero_stage: 2
    gradient_clipping: 1.0
    offload: none
```

[4] https://github.com/segment-any-text/wtpsplit/blob/main/wtpsplit/__init__.py

E An example of Segmented Input

This is an example of segmented prompt and response from GSM8k to demonstrate how sentence-level inference works in our approaches. Delimiter here is "<seg>"

Example of delimited text

Mark has a garden with flowers. <seg> He planted plants of three different colors in it. <seg> Ten of them are yellow, and there are 80% more of those in purple. <seg> There are only 25% as many green flowers as there are yellow and purple flowers. <seg> How many flowers does Mark have in his garden? <seg>

There are 80/100 * 10 = < <80/100 * 10=8> >8 more purple flowers than yellow flowers. <seg> So in Mark's garden, there are 10 + 8 = < <10+8=18> >18 purple flowers. <seg> Purple and yellow flowers sum up to 10 + 18 = < <10+18=28> >28 flowers. <seg> That means in Mark's garden there are 25/100 * 28 = < <25/100 * 28=7> >7 green flowers. <seg> So in total Mark has 28 + 7 = < <28+7=35> >35 plants in his garden. <seg> #### 35 <seg>

F Attention Map

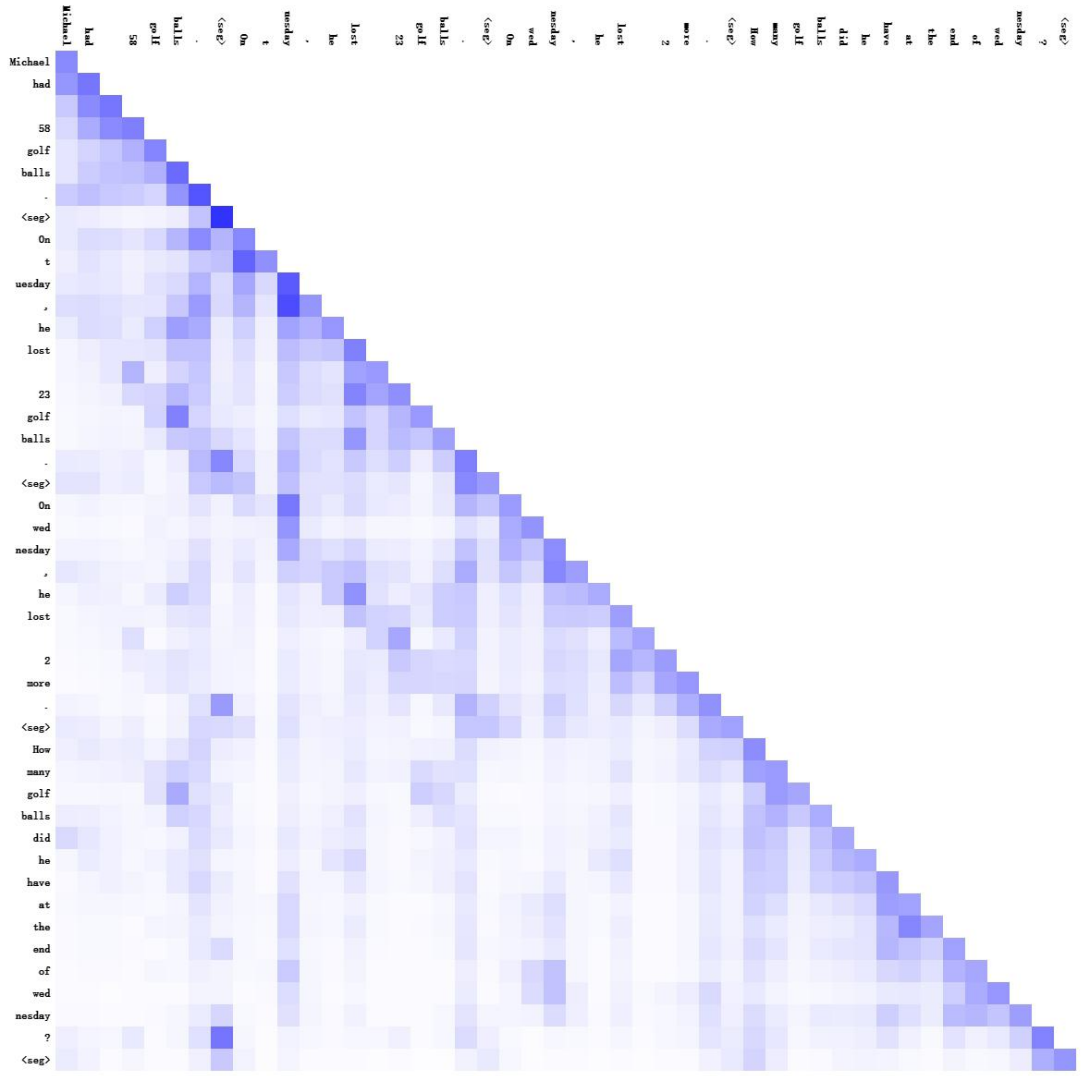

Figure 7: Attention map of Llama3-8b-seg

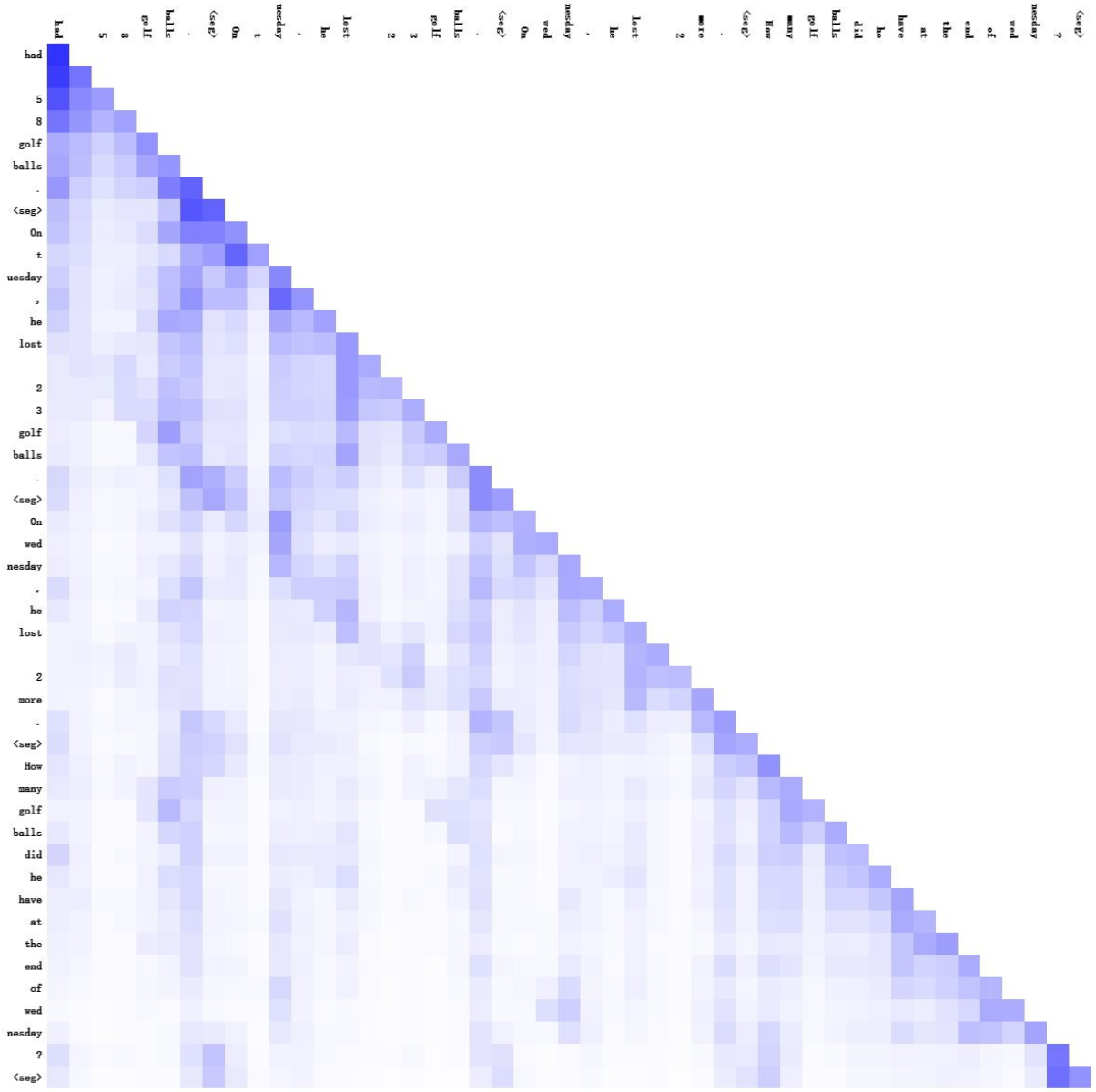

Figure 8: Attention map of Qwen2-7b-Instruct. The segmentation token we used is "**####**". We replaced it to "**<seg>**" only when visualization.