# GMSA: Enhancing Context Compression via Group Merging and Layer Semantic Alignment

Anonymous ACL submission

## Abstract

Large Language Models (LLMs) have achieved remarkable performance across a wide range of Natural Language Processing (NLP) tasks. However, in long-context scenarios, they face two challenges: high computational cost and information redundancy. To address these challenges, we propose **GMSA**, an encoder–decoder context compression framework that generates a compact sequence of soft tokens for downstream tasks. GMSA introduces **Group Merging** to achieve more uniform aggregation, mitigating semantic dominance during autoencoder pretraining, and **Layer Semantic Alignment (LSA)** to bridge the semantic gap between high-level abstract semantics and low-level input semantics. We first pretrain GMSA as an autoencoder and then fine-tune it for downstream tasks. Experiments demonstrate that GMSA improves context reconstruction compared to existing soft prompt compression paradigm and outperforms baselines on multiple long-context question answering and summarization benchmarks across two backbone models, while maintaining low end-to-end latency.

## 1 Introduction

Thanks to powerful reasoning and generalization capabilities, Large Language Models (LLMs) have achieved remarkable performance across various Natural Language Processing (NLP) tasks (Qwen et al., 2025; Team et al., 2025; Liu et al., 2025; Zeng et al., 2025). However, directly applying LLMs to long-context scenarios presents two challenges: (1) Computational inefficiency. When processing long prompts, the quadratic complexity of the Transformer's attention mechanism (Vaswani et al., 2017) results in computational inefficiency. (2) Redundant information. Much redundant information in long-context scenarios can degrade model performance (Tang et al., 2025).

Prompt compression methods address these two challenges by significantly reducing input length and redundant information. It can be categorized into hard prompt compression (Li et al., 2023; Jiang et al., 2023; Pan et al., 2024; Jiang et al., 2024; Tang et al., 2025; Zhou et al., 2025; Cao et al., 2025; Chen et al., 2025; Zhao et al., 2025) and soft prompt compression (Mu et al., 2023; Ge et al., 2024; Li et al., 2025; Liao et al., 2025; Dai et al., 2025; Rau et al., 2025; Zhang et al., 2025; Deng et al., 2025). Hard prompt compression methods achieve compression by deleting certain tokens from the original context or generating a summary. However, such explicit compression inevitably compromises semantic integrity. In contrast, by leveraging the inherent redundancy in semantic vectors (Ethayarajh, 2019; Aghajanyan et al., 2021), soft prompt compression learns a set of soft tokens that is much shorter than the original context while preserving more complete semantic information.

However, existing soft prompt compression methods have two limitations: (1) Semantic dominance in autoencoder pretraining process. LLM tends to aggregate information on a few anchor tokens (Zhang et al., 2023; Xiao et al., 2023; Huang et al., 2024; Qiu et al., 2025). As shown in Figure 1, in existing soft prompt compression paradigm, the appending randomly initialized tokens ("<*CT1*>" and "<*CT2*>") learn summary vectors layer by layer. The semantics of anchor token ("*The*") is emphasized layer by layer, resulting in the semantics of the summary vectors being dominated by it while the semantics of other tokens ("*context*", "*is*", and "*lengthy*") are diluted. This limits the retention of *complete semantics* in autoencoder pretraining, which is a common approach in soft prompt compression (Ge et al., 2024; Cheng et al., 2024; Li et al., 2025; Liao et al., 2025; Dai et al., 2025; Rau et al., 2025). (2) Ignoring the large semantic gap between different layers of the LLMs (Liu et al.,

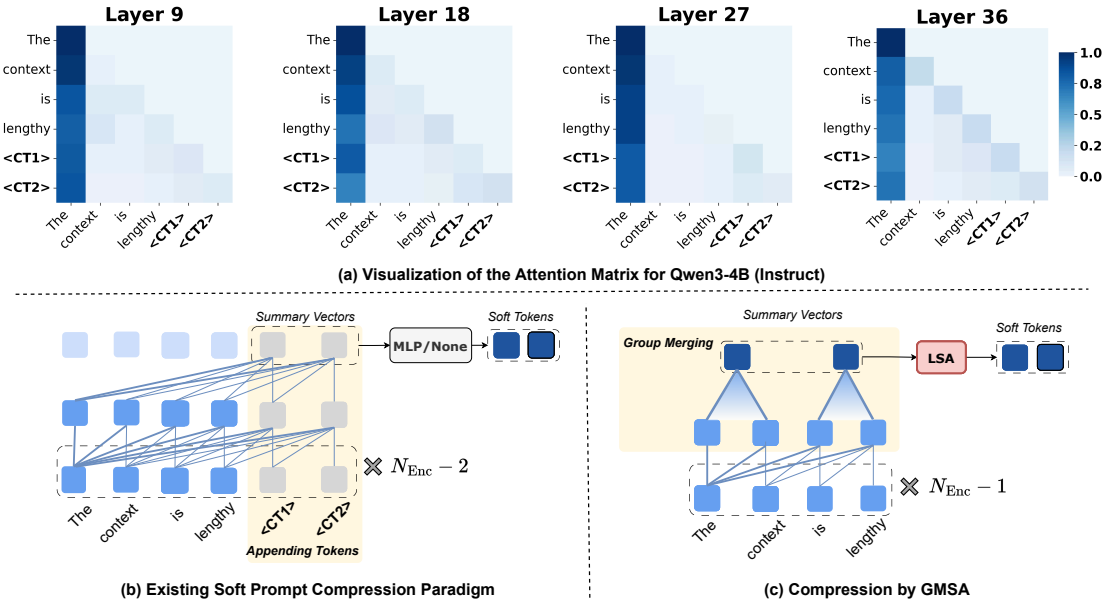



Figure 1: Existing Soft Prompt Compression Paradigm vs. Compression by GMSA. (a) visualizes the attention matrix when processing "*The context is lengthy <CT1> <CT2>*", where *<CT1>* and *<CT2>* are randomly initialized tokens. (b) represents existing soft prompt compression methods. It first learns summary vectors layer by layer, and then either directly use the summary vectors as soft tokens (marked as "None" in (b)) or obtain soft tokens via a MultiLayer Perceptron (MLP), where $N_{\text{Enc}}$ denotes the number of encoder layers. (c) denotes the compression paradigm of GMSA, which first learns summary vectors through group merging and completes semantic alignment via the Layer Semantic Alignment (LSA) module.

2024b; Jin et al., 2025), which cannot be directly bridged by the MLP layer (as demonstrated by the *w/o* LSA results in Tab. 2 and Fig. 6). The summary vectors, which represent high-level abstract semantics, are directly treated as ordinary tokens (i.e., as low-level semantics) and directly fed into the decoder during training and testing, resulting in a large semantic gap. Therefore, two research questions naturally arise: (1) *How can we mitigate semantic dominance in soft prompt compression pretraining process?* (2) *How can we bridge the large semantic gap between different layers?*

To this end, we propose GMSA, a context compression framework based on the encoder-decoder architecture that addresses these limitations via **Group Merging** and **Layer Semantic Alignment (LSA)**. Specifically, group merging partitions the input tokens into equal-sized groups and compresses each group via average pooling. *This uniform aggregation mitigates the dominance of anchor tokens, therefore helping to preserve complete semantics during pretraining.* Furthermore, we introduce a Layer Semantic Alignment (LSA) module to bridge the semantic gap between high-level abstract summary vectors and low-level input semantics. LSA is implemented as a small stack of Transformer blocks, initialized with the weights of the lower decoder layers (Figure 2), thereby inheriting the representation space of low-level semantics. *By feeding the summary vectors through LSA, we project them from a high-level semantic space into the lower-level space, which alleviates cross-layer semantic gap.* We first pretrain GMSA as an autoencoder to encourage the generated soft tokens to retain complete semantics, and then fine-tune it on downstream tasks.

Our contributions are threefold: (1) We identify and analyze two limitations in existing soft prompt compression paradigm: (i) semantic dominance in soft prompt compression pretraining process, and (ii) a semantic gap arising from the direct use of high-level summary vectors as decoder inputs. (2) We propose GMSA, a context compression framework that introduces (i) Group Merging to mitigate semantic dominance in soft compression pretraining process, and (ii) a Layer Semantic Alignment (LSA) module bridges semantic gap. (3) We conduct extensive experiments on context reconstruction, diverse benchmarks, demonstrating that GMSA achieves high semantic fidelity and superior downstream performance compared to baselines, while incurring low end-to-end latency.

## 2 Problem Formulation

Given a retrieval-augmented prompt $X = (X^{\text{ins}}, X^{d_1}, ..., X^{d_k}, ..., X^{d_K}, X^{\text{q}})$, where $X^{ins}$, $\{X^{d_k}\}_{k=1}^{K}$, and $X^{\text{q}}$ represent the instruction, context, and input question respectively. The prompt has a total token length $L$. The key aspect of the context compression system lies in generating a compressed prompt $\widetilde{X}$ with length $\widetilde{L}$, where the compression rate is defined as $\tau = \frac{\widetilde{L}}{L}$. Let $y$ denote the ground truth answer given the original input $X$, and $\widetilde{y}$ denote the answer generated by the large language model (LLM) when input with the compressed prompt $\widetilde{x}$. We aim for the distributions of $y$ and $\widetilde{y}$ to be similar under high compression rates $\tau$. This can be formulated as:

$$\min_{\widetilde{\boldsymbol{x}},\tau} \mathrm{KL}\left(P\left(\widetilde{y} \mid \widetilde{X}\right), P\left(y \mid X\right)\right). \quad (1)$$

## 3 Related Work

**Hard Prompt Compression.** Hard prompt compression refers to the removal of some less important tokens from the original prompt or the generation of summaries to achieve compression. The compressed prompt is explicit text. It can mainly be divided into the following four categories: (1) Perplexity-based methods. Selective-Context (Li et al., 2023) removes certain lexical units based on perplexity, while methods such as LLMLingua (Jiang et al., 2023), LongLLMLingua (Jiang et al., 2024), and Perception Compressor (Tang et al., 2025) adopt a coarse-to-fine framework to gradually eliminate less important parts. (2) Bidirectional semantic-based methods. Considering the unidirectional nature of perplexity, some approaches employ bidirectional semantic information for compression, such as LLMLingua-2 (Pan et al., 2024), MOOSComp (Zhou et al., 2025), and EFPC (Cao et al., 2025). (3) Methods based on intrinsic attention mechanisms. Compression is achieved through the intrinsic attention mechanisms of LLMs, such as PIS (Chen et al., 2025) and AttnComp (Zhao et al., 2025). (4) Summary generation. This involves generating linguistic summaries that contain useful information for long text content, such as CompACT (Yoon et al., 2024) and RECOMP (Xu et al., 2024). *Although these methods improve the computational efficiency of inference through prompt compression, they compromise the semantic integrity of the original prompt.*

**Soft Prompt Compression.** Soft prompt compression has become a research hotspot in the field of Natural Language Processing (NLP). The goal of soft prompt compression is to learn a set of soft tokens (with a sequence length much shorter than the original text) to achieve compression, where the compressed soft prompts cannot be explicitly converted into text. Among existing methods, xRAG (Cheng et al., 2024) focuses on processing short texts and extreme compression. More mainstream methods, such as GIST (Mu et al., 2023), AutoCompressor (Chevalier et al., 2023), 500xCompressor (Li et al., 2024), ICAE (Ge et al., 2024) and VoCo-LLaMA (Ye et al., 2024), learn soft tokens in an autoregressive manner by appending randomly initialized additional tokens. This leads to the semantics of anchor tokens in the input sequence being increasingly emphasized layer by layer, while the semantics of other tokens are diluted and cannot be fully preserved in the summary vectors. Moreover, these methods only use Multilayer Perceptrons (MLPs) for coarse-grained semantic alignment when semantic alignment is required, ignoring the significant differences in representations across different layers of large models. *Our proposed method mitigates semantic dominance during autoencoder pretraining via group merging, which uniformly aggregates within each equal-sized group. Moreover, it bridges the semantic gap across LLM layers through a Layer Semantic Alignment (LSA) module that inherits the representation space of low-level semantics.*

## 4 GMSA

In this section, we elaborate on the architecture of our proposed context compression framework, GMSA, which includes two key components: group merging and layer semantic alignment (LSA). GMSA undergoes a two-stage training process: autoencoder pretraining and fine-tuning (see Figure 2). First, GMSA ensures that the generated soft tokens contain the complete semantic representation of the original text through the autoencoder pretraining process. Then, it applies the knowledge contained in the soft tokens to downstream tasks via fine-tuning.

### 4.1 Group Merging

**Extraction of Semantic Features.** We extract the semantic features of the original text through a language model (e.g., Qwen3-4B) as the encoder. The encoder is trained using LoRA.

$$H = \texttt{Encoder}(X), \quad (2)$$

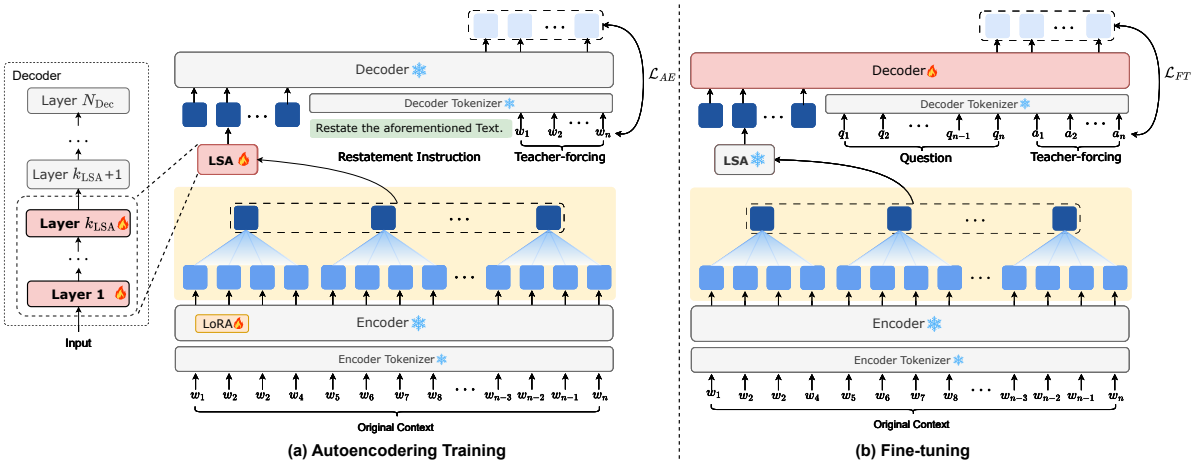



Figure 2: **The Training process of GMSA.** GMSA consists of an encoder and a decoder and is trained in two stages: (a) autoencoding pretraining, followed by (b) task-specific fine-tuning. During autoencoding training, only the encoder and the Layer Semantic Alignment (LSA) module are trained to reconstruct the original context, enabling GMSA to produce semantically complete compressed representations. During fine-tuning, only the decoder is trained, and GMSA is optimized end-to-end on question answering format to enhance its ability to extract knowledge from the compressed representations and improve downstream performance. Notably, a single LSA layer is sufficient to preserve semantics effectively (see Appendix A), so $k_{\text{LSA}} \ll N_{\text{Dec}}$.

where $X$ is the original text and $H$ is the obtained last hidden state.

**Merging.** We divide the obtained $H$ into several groups according to the size of the compression limit, as the group length $L_G$ (e.g., when the compression rate is 4, the group length is also 4). To this end, original text representations are organized as follows:

$$\begin{aligned} H &= \left[H_1, \ldots, H_{\mathbf{G}_{N_g}}\right] \\ &= \left[H_{1:L_G}, \ldots, H_{N_d - L_G + 1:N_d}\right]. \end{aligned} \tag{3}$$

We take the average of each dimension of each group token to obtain the initial compressed representation.

$$\begin{aligned} \widetilde{H} &= \left[\bar{H}_{\mathbf{G}_1}, \ldots, \bar{H}_{\mathbf{G}_N}\right] \\ &= \left[\frac{1}{L_G}\sum H_{\mathbf{G}_1}, \ldots, \frac{1}{L_G}\sum H_{\mathbf{G}_N}\right], \end{aligned} \tag{4}$$

where $\widetilde{H}$ is the obtained initial compressed representation.

## 4.2 Layer Semantic Alignment

The layer semantic alignment (LSA) module is used to complete the alignment from the soft tokens generated by the encoder (high-level semantics) to the primary semantics of the decoder. Given the significant differences in semantic representation between different layers of large language models (LLMs), the LSA is trained via full fine-tuning.

$$\widetilde{m} = \mathcal{F}_{k_{\text{LSA}}}(\widetilde{H}), \tag{5}$$

where $H$ is the final compressed representation, $\mathcal{F}_{k_{\text{LSA}}}$ denotes Transformer blocks initialized with the weights from the first $k$ layers of the decoder, and $\widetilde{m}$ denotes the generated soft tokens. Just one layer of LSA is sufficient to achieve excellent semantic preservation (for space limitations, please refer to Appendix A), so in this work, we can just set $k_{\text{LSA}} = 1$.

## 4.3 Autoencoder Pretraining

The Autoencoder Pretraining process, which aims to encode the complete information of the original text into memory embeddings, is achieved through autoencoder-based training. We hope to minimize the loss of the reconstructed text, which can be expressed as:

$$\mathcal{L}_{AE} = -\sum_{i=1} \log p_\phi\left(x_i \mid \widetilde{m}, X^{\text{ins}}, x_{<i}\right), \tag{6}$$

where $p_\phi(\cdot)$ is the decoder probability distribution obtained after the softmax function, and $x_i$ is the $i$-th token in the original text.

## 4.4 Fine-tuning

After completing autoencoder pretraining, we need to teach the decoder how to utilize the soft tokens.

We achieve this by performing full fine-tuning of the decoder, which can be expressed as:

$$\mathcal{L}_{\mathrm{FT}} = -\sum_{i=1}^{n} \log p_\phi \left( a_i \mid \widetilde{m}, q_1, ..., q_n, a_{<i} \right), \tag{7}$$

where $p_\phi(\cdot)$ is the decoder probability distribution obtained after the softmax function, and $a_i$ denotes the $i$-th token in the predicted answer.

## 5 Experiments

In this section, we attempt to answer the following research questions (RQs): (1) How effective is GMSA in context reconstruction (RQ1)? (2) How does GMSA utilize knowledge compared with other baselines (RQ2)? (3) How effective are the individual components of GMSA (RQ3)?

### 5.1 Settings

**Training.** GMSA involves a two-stage training process: autoencoder pretraining and fine-tuning. We use datasets: PwC (Ge et al., 2024), NaturalQuestions (Liu et al., 2024a), 2WikiMQA (Ho et al., 2020), HotpotQA (Yang et al., 2018), NarrativeQA (Kociský et al., 2018), MultiNews (Fabbri et al., 2019) (see more details on Appendix C). Among them, we use PwC to evaluate the performance of context reconstruction, while the other datasets are employed to measure downstream knowledge application. During training, we randomly sample compression rates (i.e., 4x compression and 8x compression) for each training sample. We set the batch size to 32 and trained all models for one epoch. The learning rate is set to $1 \times 10^{-4}$ during the autoencoder pretraining stage and to $1 \times 10^{-5}$ during the fine-tuning stage. We train GMSA under two distinct experimental configurations: (1) We perform autoencoder pretraining on the PwC dataset to get **GMSA-AE**. (2) We develop the general-purpose **GMSA** for all remaining tasks. This is achieved through total training process: autoencoder pretraining followed by fine-tuning on a balanced hybrid dataset. This dataset is constructed by sampling 20,000 instances from each of the following sources: NaturalQuestions, 2WikiMQA, HotpotQA, NarrativeQA, and MultiNews.

**Implementation.** GMSA is implemented based on LLaMA-3.2-3B (Instruct) and Qwen3-4B (Instruct). Due to the GPU memory constraints of our computational resources, the maximum input length is set to 12K tokens for autoencoding training and 32K tokens for fine-tuning. To ensure fair comparison, all baseline results are obtained from our re-implementations based on official open-source code. All experiments are conducted on 8 NVIDIA H20 GPUs.

**Evaluation Metrics.** For the context reconstruction task on the PwC dataset, we use BLEU (Papineni et al., 2002), Prefix Exact Match, BERT Score (Zhang* et al., 2020), and ROUGE (Lin, 2004) for evaluation. For the QA tasks on Natural Questions, TriviaQA, and 2WikiMQA, we use Exact Match (EM) (Lewis et al., 2020) and F1 (Yang et al., 2018) score for evaluation. For Summary task on MultiNews, we use F1 score too.

**Baselines.** For the task of context reconstruction, we train a **InContext AutoEncoder** (i.e., ICAE-AE) as a baseline *only* using autoencoder pretraining (Ge et al., 2024) and the same training hyperparameters as GMSA. In terms of downstream knowledge application, we conduct comprehensive comparisons with various methods, including: Hard prompt compression (e.g., LongLLMLingua (Jiang et al., 2024), LLMLingua-2-large (Pan et al., 2024), Provence, EXIT), and soft prompt compression (e.g., ICAE (Ge et al., 2024), Activation Beacon (Zhang et al., 2024)). Additionally, we compare against the original input prompt and zero-shot prompting to establish upper and lower bounds on original model performance.

### 5.2 Main Result

We analyze the performance of GMSA along two core dimensions: context reconstruction capability and downstream task effectiveness under compression.

**RQ1: Performance on context reconstruction Task.** As shown in Figure 3, GMSA-AE consistently outperforms ICAE-AE across all evaluation metrics on the context reconstruction task using the PwC dataset. Specifically, under both 4x and 8x compression ratios, GMSA-AE achieves significantly higher scores in token-matching metrics such as BLEU, Prefix Exact Match, and ROUGE-1/2/L. For instance, at sequence length 512, GMSA-AE's BLEU score exceeds that of ICAE-AE by approximately 20–30%, demonstrating its superior ability to precisely recover individual tokens. Furthermore, GMSA-AE also maintains a consistent 5% advantage in BERT Score F1, which measures

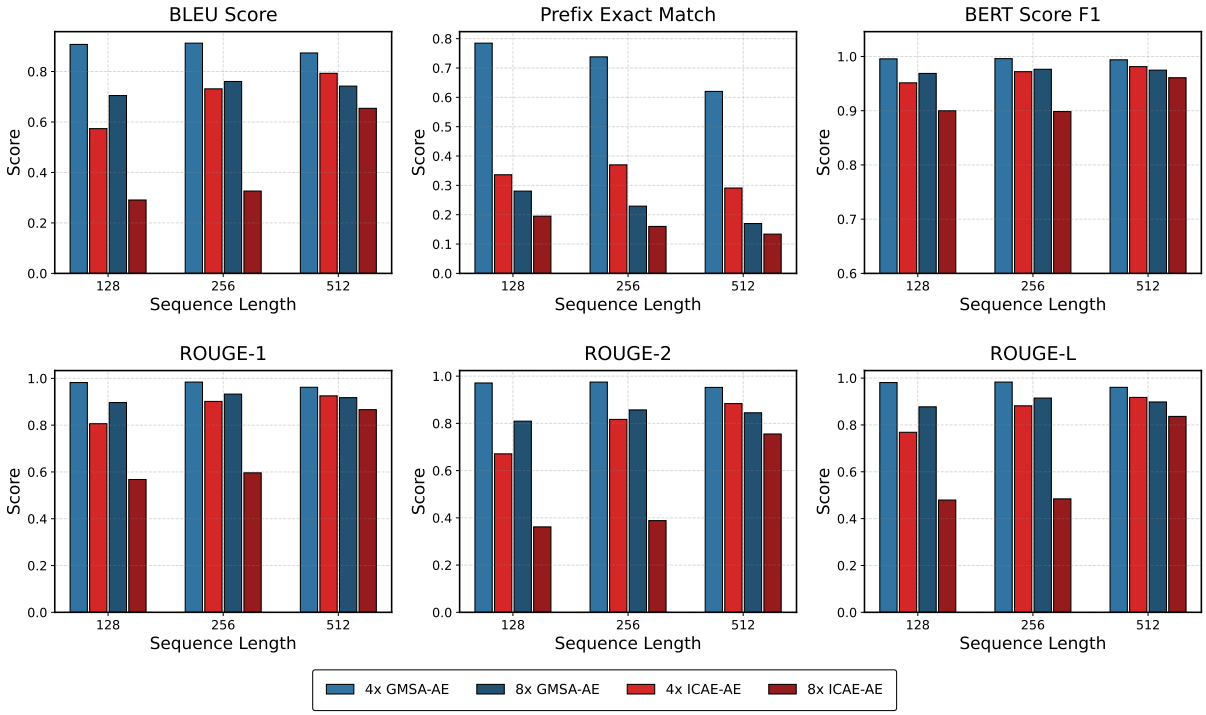



Figure 3: GMSA-AE vs. ICAE-AE on the context reconstruction task (PwC dataset). Sequence Length represents different context reconstruction lengths (i.e., 128, 256, 512).

semantic similarity, indicating that it better preserves overall contextual semantics during compression. This confirms GMSA's effectiveness in encoding and reconstructing semantic information even under various compression.

**RQ2: Effectiveness on Downstream QA and Summarization Tasks under Compression.** Table 1 presents results on multiple long-context QA and summarization benchmarks. GMSA superior performance under both 4x and 8x compression constraints across diverse models (LLaMA-3.1-8B-Instruct and Qwen3-4B-Instruct). Notably, despite employing a *query-independent* compression mechanism, GMSA consistently surpasses query-dependent methods like LongLLMLingua, Provence and EXIT. This highlights GMSA's ability to extract knowledge without relying on query-specific signals. Moreover, GMSA demonstrates more robust and stable gains across varying compression constraint and model scales, further validating the generalizability and effectiveness of GMSA.

### 5.3 Efficiency Analysis

In this section, we discuss the efficiency of our proposed method. By using soft tokens instead of the long original context to enhance the inference process, our method reduces the inference cost of the original context during the generation process by a factor of $r$. The overall floating-point operations (FLOPs) are calculated through two processes: compression and generation.

The compression process can be expressed as:

$$\text{FLOPs}^{comp} = F^{\text{Encoder}}(L) + F^{\text{LSA}}\left(\left\lceil \frac{L}{r} \right\rceil\right).$$

were $L$ denotes the original context length; $L_q$ refers to the question length, and $F^*(\cdot)$ represents the FLOPs complexity measure for module $*$. The symbol $*$ indicates the architectural components, where $* \in \{\text{Decoder}, \text{Encoder}, \text{LSA}\}$. For the generation process, assuming the answer length is $L_a$, the generation process requires $L_a$ forward passes. The FLOPs for the $i$-th forward pass are given by:

$$\text{FLOPs}_i^{forward} = F^{\text{Decoder}}\left(\left\lceil \frac{L}{r} \right\rceil, L_q, i\right).$$

Combining the costs of all components, the total FLOPs complexity is:

$$\text{FLOPs} = \sum_{i=1}^{L_a} \text{FLOPs}_i^{\text{forward}} + \text{FLOPs}^{comp}.$$

GMSA achieves the lowest end-to-end inference latency against, which is approximately 5x faster

Table 1: Main results on benchmarks. We **bold** the optimal results. Closed-book indicates using only the input question as the input, while Original Prompt indicates using original context as the input.

| Methods | NaturalQA | | 2WikiMQA | | HotpotQA | | NarrativeQA | | MultiNews | AVG | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | EM | F1 | EM | F1 | EM | F1 | EM | F1 | F1 | EM | F1 |
| **LLaMA-3.2-3B-Instruct** | | | | | | | | | | | |
| Closed-book | 7.38 | 14.01 | 2.25 | 15.37 | 4.91 | 14.68 | 0.00 | 8.34 | - | 3.64 | 13.10 |
| Original Prompt | 37.55 | 48.00 | 16.29 | 32.34 | 34.94 | 51.35 | 12.78 | 29.27 | 25.25 | 25.39 | 37.24 |
| *4x Compression Constraint* | | | | | | | | | | | |
| ICAE | 25.12 | 27.36 | 25.20 | 28.77 | 17.18 | 24.90 | 2.07 | 11.18 | 18.89 | 17.39 | 22.22 |
| LLMLingua-2-large | 25.76 | 36.55 | 15.09 | 20.00 | 26.79 | 38.22 | 9.02 | 17.90 | 28.40 | 19.17 | 28.21 |
| Activation Beacon | 34.05 | 46.53 | 33.31 | 40.92 | 37.81 | 51.30 | 10.35 | 19.09 | 27.98 | 28.88 | 37.16 |
| Provence | 32.35 | 42.28 | 30.12 | 37.46 | 38.58 | 51.24 | **10.41** | 18.64 | 28.79 | 27.87 | 35.68 |
| EXIT | 44.40 | 54.88 | 27.26 | 33.86 | 43.59 | 57.04 | 8.27 | 18.94 | 33.12 | 30.88 | 39.57 |
| LongLLMLingua | 44.97 | 55.01 | 21.38 | 27.11 | 31.08 | 44.08 | 3.76 | 13.55 | 25.99 | 25.30 | 33.15 |
| **GMSA** | **56.87** | **55.49** | **46.75** | **53.54** | **44.26** | **57.85** | 9.40 | **19.19** | **35.36** | **39.32** | **44.29** |
| *8x Compression Constraint* | | | | | | | | | | | |
| ICAE | 25.46 | 27.83 | 25.61 | 29.39 | 17.60 | 25.38 | 1.97 | 10.42 | 18.96 | 17.66 | 22.40 |
| LLMLingua-2-large | 17.36 | 27.14 | 10.19 | 14.03 | 18.13 | 27.01 | 6.39 | 12.99 | 25.69 | 13.02 | 21.37 |
| Activation Beacon | 29.57 | 42.59 | 33.44 | 40.61 | 34.74 | 47.52 | 6.19 | 16.68 | 25.48 | 25.99 | 34.58 |
| Provence | 31.00 | 40.99 | 27.63 | 34.42 | 36.54 | 48.33 | 6.97 | 16.53 | 24.54 | 25.54 | 32.96 |
| EXIT | 44.67 | 51.21 | 19.78 | 25.10 | 37.67 | 52.46 | 4.98 | 14.00 | 30.62 | 26.78 | 34.68 |
| LongLLMLingua | 35.51 | 47.14 | 17.05 | 21.74 | 25.77 | 37.91 | 2.35 | 10.83 | 22.65 | 20.17 | 28.05 |
| **GMSA** | **53.18** | **52.59** | **45.16** | **52.42** | **39.70** | **53.61** | **7.33** | **17.34** | **33.96** | **36.34** | **41.98** |
| **Qwen3-4B-Instruct** | | | | | | | | | | | |
| Closed-book | 10.49 | 17.44 | 13.50 | 23.36 | 12.33 | 20.03 | 0.61 | 10.17 | - | 9.23 | 17.75 |
| Original Prompt | 32.79 | 44.02 | 33.10 | 43.48 | 44.30 | 60.31 | 8.92 | 20.41 | 31.42 | 29.78 | 39.93 |
| *4x Compression Constraint* | | | | | | | | | | | |
| ICAE | 18.64 | 20.64 | 25.24 | 29.18 | 17.77 | 25.77 | 3.03 | 11.13 | 23.13 | 16.17 | 21.97 |
| LLMLingua-2-large | 23.22 | 35.25 | 25.58 | 31.15 | 28.20 | 40.96 | 7.47 | 18.37 | 29.36 | 21.12 | 31.02 |
| Provence | 31.18 | 43.89 | 39.51 | 48.61 | 42.10 | 56.15 | 11.35 | **24.45** | 28.82 | 31.04 | 40.38 |
| EXIT | 40.00 | 52.28 | 36.97 | 45.62 | 45.16 | 57.47 | 4.99 | 15.61 | 32.03 | 31.78 | 40.60 |
| LongLLMLingua | 40.11 | 53.09 | 26.92 | 31.93 | 30.28 | 43.36 | 3.29 | 11.88 | 24.27 | 25.15 | 32.91 |
| **GMSA** | **60.38** | **58.09** | **55.75** | **63.07** | **52.28** | **67.93** | **12.97** | 24.12 | **36.26** | **45.35** | **49.89** |
| *8x Compression Constraint* | | | | | | | | | | | |
| ICAE | 18.61 | 21.41 | 24.68 | 29.69 | 17.61 | 25.69 | 2.67 | 11.82 | 23.38 | 15.89 | 22.40 |
| LLMLingua-2-large | 14.73 | 26.58 | 21.54 | 25.89 | 19.15 | 28.02 | 5.76 | 15.57 | 26.71 | 15.30 | 24.55 |
| Provence | 31.12 | 42.52 | 37.57 | 45.05 | 37.32 | 48.46 | 9.95 | 21.05 | 24.81 | 28.99 | 36.38 |
| EXIT | 41.45 | 53.38 | 29.63 | 35.01 | 35.25 | 47.83 | 2.16 | 11.55 | 28.26 | 27.12 | 35.21 |
| LongLLMLingua | 31.98 | 45.16 | 23.07 | 27.62 | 24.00 | 35.76 | 1.48 | 9.87 | 19.84 | 20.13 | 27.65 |
| **GMSA** | **50.02** | **51.43** | **50.30** | **57.83** | **43.93** | **59.48** | **10.24** | **21.18** | **33.94** | **38.62** | **44.77** |

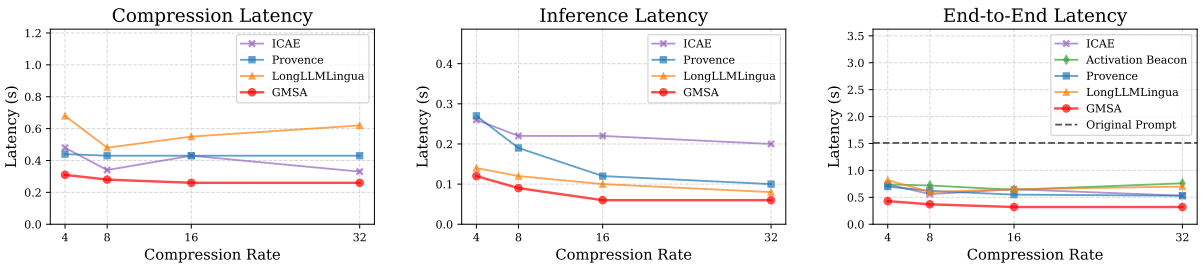
than original prompt input under 32x compression rate on NaturalQuestions (see Figure 4 and Appendix D).

Table 2: Ablation study on NaturalQuestions, 2WikiMQA under 4x compression constraint using Qwen3-4B as backbone.

| Methods | NaturalQA | | 2WikiMQA | |
|---|---|---|---|---|
| | EM | F1 | EM | F1 |
| **Default** | **60.38** | **58.09** | **55.75** | **63.07** |
| *w/o* AE Training | 48.81 | 51.65 | 37.51 | 42.01 |
| *w/o* Group Merging | 21.88 | 24.55 | 28.33 | 32.02 |
| *w/o* LSA | 20.15 | 22.30 | 27.74 | 31.21 |
| *w/* Rand. init. LSA | 19.47 | 21.99 | 28.75 | 32.36 |

### 5.4 Ablation Study

For RQ3, to investigate the impact of each component in GMSA, we conduct the following four ablation experiments (see Table 2): (1) Ours *w/o* AE Training refers to directly perform fine-tuning on GMSA without AutoEncoding (AE) training; (2) Ours *w/o* Group Merging indicates that we replace group merging with appending learnable tokens when generating summary vectors; (3) Ours *w/o* LSA means we do not use the Layer Semantic Alignment (LSA) module, but instead directly pass the summary vectors through an MultiLayer Perceptron (MLP) to obtain the summary vectors; (4) Ours *w/* Rand. init. LSA denotes that the LSA




(a) Compression Latency. (b) Inference Latency. (c) End-to-End Latency.

Figure 4: Latency evaluation across methods. For context compression methods, end-to-end inference latency can be decomposed into compression latency and inference latency. Activation Beacon does not explicitly decouple these two phases and therefore reports only the end-to-end latency. Detailed numerical results are provided in Appendix D.

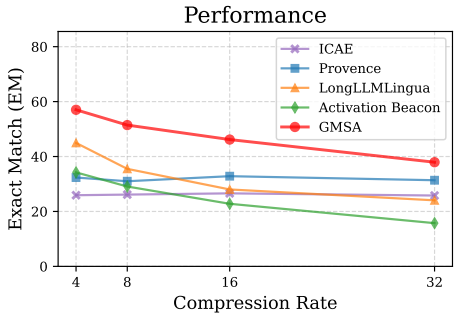



Figure 5: Performance comparison and stress test of GMSA and baselines across various compression rate on the NaturalQuestions dataset, using LLaMA-3.2-3B as the backbone.

module is randomly initialized instead of inheriting the pre-trained weights from the decoder.

In summary, removing any single component leads to a significant performance drop, fully demonstrating the necessity and effectiveness of each component. Removing autoencoder pretraining makes it difficult for GMSA to generate summary vectors that capture complete semantics; replacing Group Merging with learnable appended tokens increases the model's learning difficulty; discarding the Layer Semantic Alignment (LSA) module results in misalignment between the high-level semantic information represented by the summary vectors and the low-level semantic space of the decoder's input; and randomly initializing LSA fundamentally prevents it from inheriting the decoder's initial input semantic space, thereby introducing a substantial semantic gap that is difficult to bridge even through training. Further evidence is provided in Appendix B: under all ablation settings, loss convergence is significantly worse than that of the Default setting, both during the autoencoder pretraining phase and the fine-tuning phase.

### 5.5 Stress Test

As shown in Figure 5, we conduct stress tests on GMSA and its baseline methods across multiple compression rates (2x, 4x, 8x, 16x, and 32x) on the NaturalQuestions dataset. Experimental results show that although certain methods (e.g., LongLLMLingua and Activation Beacon) suffer significant performance degradation under high compression, GMSA consistently maintains the best performance across all compression ratios, with a smaller performance drop compared to most baselines. Notably, while methods such as Providence and ICAE exhibit relatively stable performance across different compression levels, their overall accuracy remains substantially lower than that of GMSA.

## 6 Conclusion

This paper introduces GMSA, a context compression framework based on an encoder-decoder structure. It effectively and efficiently learns summary vectors and bridges the significant gap between the semantics representation of different layers via group merging, and a Layer Semantic Alignment (LSA) module. GMSA first undergoes autoencoder pretraining to ensure that the generated soft tokens contain complete semantics, and then adapts to downstream tasks through fine-tuning. Experiments demonstrate that GMSA has excellent context reconstruction capabilities. It outperforms existing baselines by a large margin in downstream tasks, paving the way for the efficient application of LLMs.

## Limitations

Although GMSA demonstrates strong performance across diverse long-context benchmarks and compression ratios, it has a limitation. Like most soft prompt compression methods, GMSA currently requires a two-stage training pipeline, consisting of autoencoder pretraining followed by task-specific fine-tuning, which incurs modest overhead compared to training-free compression baselines. However, this design is difficult to avoid in task-agnostic compression frameworks, and GMSA's superior downstream performance and inference efficiency adequately compensate for this overhead.

## References


Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. 2021. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7319–7328, Online. Association for Computational Linguistics.

Yun-Hao Cao, Yangsong Wang, Shuzheng Hao, Zhenxing Li, Chengjun Zhan, Sichao Liu, and Yi-Qi Hu. 2025. Efpc: Towards efficient and flexible prompt compression. *Preprint*, arXiv:2503.07956.

Lizhe Chen, Binjia Zhou, Yuyao Ge, Jiayi Chen, and Shiguang Ni. 2025. Pis: Linking importance sampling and attention mechanisms for efficient prompt compression. *arXiv preprint arXiv:2504.16574*.

Xin Cheng, Xun Wang, Xingxing Zhang, Tao Ge, Si-Qing Chen, Furu Wei, Huishuai Zhang, and Dongyan Zhao. 2024. xRAG: Extreme context compression for retrieval-augmented generation with one token. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. 2023. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3829–3846, Singapore. Association for Computational Linguistics.

Yuhong Dai, Jianxun Lian, Yitian Huang, Wei Zhang, Mingyang Zhou, Mingqi Wu, Xing Xie, and Hao Liao. 2025. Pretraining context compressor for large language models with embedding-based memory. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 28715–28732, Vienna, Austria. Association for Computational Linguistics.

Chenlong Deng, Zhisong Zhang, Kelong Mao, Shuaiyi Li, Tianqing Fang, Hongming Zhang, Haitao Mi, Dong Yu, and Zhicheng Dou. 2025. Unigist: Towards general and hardware-aligned sequence-level long context compression. *CoRR*, abs/2509.15763.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Alexander R. Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R. Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *ACL (1)*, pages 1074–1084. Association for Computational Linguistics.

Tao Ge, Hu Jing, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. 2024. In-context autoencoder for context compression in a large language model. In *The Twelfth International Conference on Learning Representations*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Qidong Huang, Xiaoyi Dong, Pan Zhang, Bin Wang, Conghui He, Jiaqi Wang, Dahua Lin, Weiming Zhang, and Nenghai Yu. 2024. Opera: Alleviating hallucination in multi-modal large language models via over-trust penalty and retrospection-allocation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13418–13427.

Huiqiang Jiang, Qianhui Wu, , Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024. LongLLMLingua: Accelerating and enhancing LLMs in long context scenarios via prompt compression. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1658–1677, Bangkok, Thailand. Association for Computational Linguistics.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023. LLMLingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13358–13376, Singapore. Association for Computational Linguistics.

Mingyu Jin, Qinkai Yu, Jingyuan Huang, Qingcheng Zeng, Zhenting Wang, Wenyue Hua, Haiyan Zhao, Kai Mei, Yanda Meng, Kaize Ding, Fan Yang, Mengnan Du, and Yongfeng Zhang. 2025. Exploring concept depth: How large language models acquire

knowledge and concept at different layers? In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 558–573, Abu Dhabi, UAE. Association for Computational Linguistics.

Tomás Kociský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. *Trans. Assoc. Comput. Linguistics*, 6:317–328.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. 2023. Compressing context to enhance inference efficiency of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6342–6353, Singapore. Association for Computational Linguistics.

Zongqian Li, Yixuan Su, and Nigel Collier. 2024. 500xcompressor: Generalized prompt compression for large language models. *Preprint*, arXiv:2408.03094.

Zongqian Li, Yixuan Su, and Nigel Collier. 2025. 500xcompressor: Generalized prompt compression for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 25081–25091.

Huanxuan Liao, Wen Hu, Yao Xu, Shizhu He, Jun Zhao, and Kang Liu. 2025. Beyond hard and soft: Hybrid context compression for balancing local and global information retention. *Preprint*, arXiv:2505.15774.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, and 1 others. 2025. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024a. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Zhu Liu, Cunliang Kong, Ying Liu, and Maosong Sun. 2024b. Fantastic semantics and where to find them: Investigating which layers of generative LLMs reflect lexical semantics. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14551–14558, Bangkok, Thailand. Association for Computational Linguistics.

Jesse Mu, Xiang Lisa Li, and Noah Goodman. 2023. Learning to compress prompts with gist tokens. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Ruhle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. 2024. LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 963–981, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. *CoRR*, abs/2505.06708.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

David Rau, Shuai Wang, Hervé Déjean, Stéphane Clinchant, and Jaap Kamps. 2025. Context embeddings for efficient answer generation in retrieval-augmented generation. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining*, WSDM '25, page 493–502, New York, NY, USA. Association for Computing Machinery.

Jiwei Tang, Jin Xu, Tingwei Lu, Zhicheng Zhang, YimingZhao YimingZhao, LinHai LinHai, and Hai-Tao Zheng. 2025. Perception compressor: A training-free prompt compression framework in long context scenarios. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 4093–4108, Albuquerque, New Mexico. Association for Computational Linguistics.

Kimi Team, 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*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient streaming language models with attention sinks. *arXiv*.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2024. Recomp: Improving retrieval-augmented lms with context compression and selective augmentation. In *The Twelfth International Conference on Learning Representations*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Xubing Ye, Yukang Gan, Xiaoke Huang, Yixiao Ge, and Yansong Tang. 2024. Voco-llama: Towards vision compression with large language models. *arXiv preprint arXiv:2406.12275*.

Chanwoong Yoon, Taewhoo Lee, Hyeon Hwang, Minbyul Jeong, and Jaewoo Kang. 2024. CompAct: Compressing retrieved documents actively for question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21424–21439, Miami, Florida, USA. Association for Computational Linguistics.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, and 1 others. 2025. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*.

Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. 2024. Long context compression with activation beacon. *arXiv preprint arXiv:2401.03462*.

Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. 2025. Long context compression with activation beacon. In *ICLR*. OpenReview.net.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang "Atlas" Wang, and Beidi Chen. 2023. H2o: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, volume 36, pages 34661–34710. Curran Associates, Inc.

Yunlong Zhao, Haoran Wu, and Bo Xu. 2025. Leveraging attention to effectively compress prompts for long-context llms. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(24):26048–26056.

Fengwei Zhou, Jiafei Song, Wenjin Jason Li, Gengjian Xue, Zhikang Zhao, Yichao Lu, and Bailin Na. 2025. Mooscomp: Improving lightweight long-context compressor via mitigating over-smoothing and incorporating outlier scores. *arXiv preprint arXiv:2504.16786*.

## A Impact of different number of LSA layers

We conduct experiments to investigate the impact of Layer Semantic Alignment (LSA) module with varying numbers of layers on the retention of complete semantics, and the results are shown in Figure 7. We can draw the following conclusions: (1) Only one layer of LSA is sufficient to achieve good retention of complete semantics (with a BERT Score F1 close to 1, and it already performs the best among different numbers of LSA layers); (2) When the number of LSA layers becomes too high, e.g., using five layers of LSA, it may actually lead to a decrease in the GMSA's ability to retain semantics. This is likely because as the LSA module becomes deeper, it contains more high-layer semantics and fewer low-layer semantics, thereby increasing the difficulty of semantic alignment.

Table 3: Latency evaluation on NaturalQA using LLaMA-3.2-3B as backbone. Each compression method's total latency can be divided into compression latency and inference latency.

| Methods | Compression Constraint | | | |
|---|---|---|---|---|
| | 4x | 8x | 16x | 32x |
| **Compression Latency (s)** | | | | |
| ICAE | 0.48 | 0.34 | 0.43 | 0.33 |
| Provence | 0.44 | 0.43 | 0.43 | 0.43 |
| LongLLMLingua | 0.68 | 0.48 | 0.55 | 0.62 |
| **GMSA** | **0.31** | **0.28** | **0.26** | **0.26** |
| **Inference Latency (s)** | | | | |
| ICAE | 0.26 | 0.22 | 0.22 | 0.20 |
| Provence | 0.27 | 0.19 | 0.12 | 0.10 |
| LongLLMLingua | 0.14 | 0.12 | 0.10 | 0.08 |
| **GMSA** | **0.12** | **0.09** | **0.06** | **0.06** |
| **End-to-End Latency (s)** | | | | |
| ICAE | 0.74 | 0.56 | 0.65 | 0.53 |
| Activation Beacon | 0.74 | 0.72 | 0.64 | 0.76 |
| Provence | 2.54 | 2.43 | 0.55 | 0.53 |
| LongLLMLingua | 0.82 | 0.60 | 0.65 | 0.70 |
| **GMSA** | **0.43** | **0.37** | **0.32** | **0.32** |
| Original Prompt | 1.51 | | | |

## B Training Loss Comparison

Figure 6a shows the loss curves during the autoencoder pretraining phase under four settings: Default, *w/o* Group Merging, *w/* Rand Init. LSA, and *w/o* LSA. The Default setting converges substantially faster and achieves a lower final loss, whereas the other three settings exhibit similar loss trajectories with significantly higher losses. This indicates that, compared to the Default configuration, each of these ablated variants incurs considerable semantic degradation.

During the fine-tuning phase (Figure 6b), the training loss of the three configurations (i.e., *w/o* Group Merging, *w/* Rand init. LSA, and *w/o* LSA) are significantly higher than that of the *Default* setting, with similar loss trajectories. This suggests that these configurations are less effective at extracting knowledge from the compressed representation compared to the default setup.

## C Datasets Details

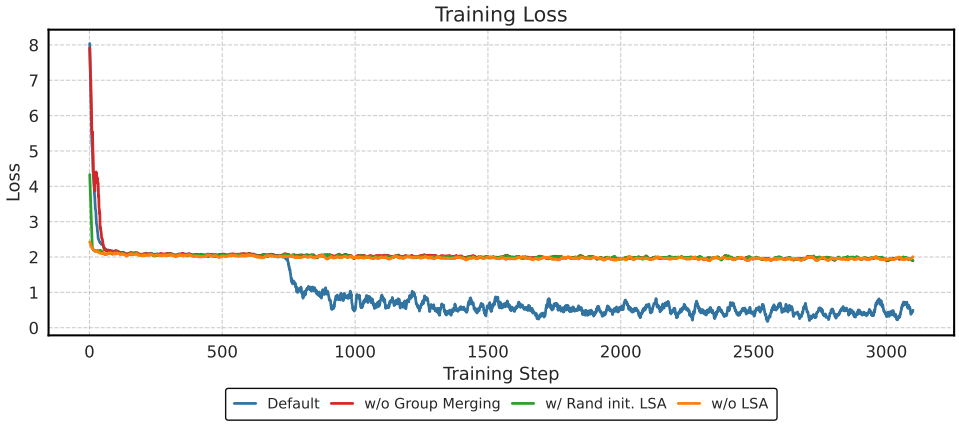
**PwC dataset.** In the PwC dataset (Ge et al., 2024), each sample is a triplet (context, prompt, answer), where the context is sampled from the Pile and the prompt and answer are generated by GPT-4. The training set contains 241,564 samples, the test set contains 18,146 samples.

**NaturalQuestions.** NaturalQuestions (Liu et al., 2024a), in which each question corresponds to 20 relevant documents, 19 of which are distractors and only one contains the ground truth answer. The training set contains 75,322 samples, the test set contains 2,655 samples.

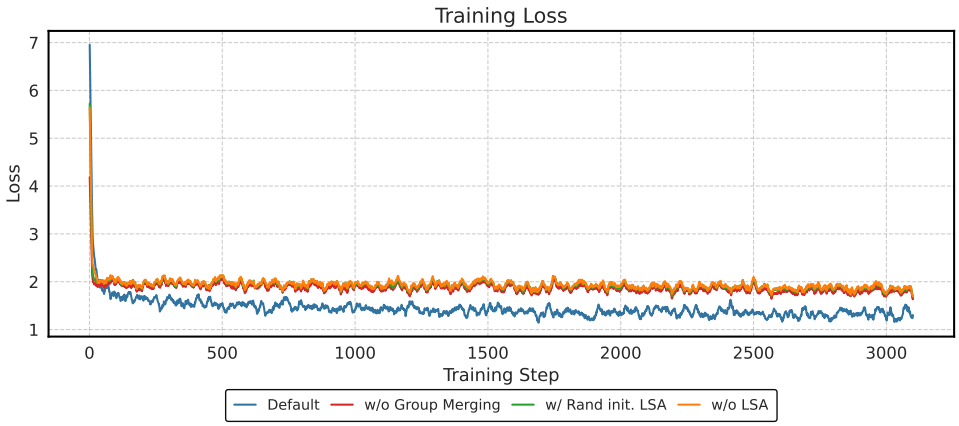
**HotpotQA.** HotpotQA (Yang et al., 2018) is a two-hop reasoning dataset, where the answers are scattered across two documents. Specifically, each question corresponds to 10 relevant documents, two of which are the ground truth documents. The training set contains 89,609 samples, the test set contains 7,345 samples.

**2WikiMQA.** Compared with HotpotQA, 2WikiMQA (Ho et al., 2020) includes more complex reasoning paths, and the combination of structured and unstructured data, usually involving two or more hops and having higher difficulty. The training set contains 167,454 samples, the test set contains 12,576 samples.

**NarrativeQA.** NarrativeQA (Kociský et al., 2018) is a reading comprehension dataset designed to evaluate deeper narrative understanding, where models must answer questions about stories by reading entire books or movie scripts rather than relying on shallow lexical matching. The dataset includes document-level metadata (e.g., story URLs and approximate word counts), Wikipedia summaries, and question-answer pairs. We use the test




(a) Loss comparison during the autoencoder pretraining stage.




(b) Loss comparison during the fine-tuning stage.

Figure 6: Loss comparison of two training stage. (a) and (b) show loss comparisons during autoencoder pretraining and fine-tuning stages, respectively. **Note: *w/o* Group Merging, *w/* Rand init. LSA, and *w/o* LSA show nearly identical loss curves in both stages.**

set to evaluate the model's performance, filtering out test samples longer than 32K.

**MultiNews.** MultiNews (Fabbri et al., 2019) is a multi-document summarization dataset, where each sample consists of multiple news articles paired with a human-written summary. The dataset provides 44,972 training examples, 5,622 validation examples, and 5,622 test examples. We use the test set to evaluate model performance.

## D Numerical Latency Evaluation Results

Table 3 presents the numerical results corresponding to Figure 4. We can see that GMSA consistently achieves the lowest compression latency, inference latency, and end-to-end latency, and is significantly faster than the original prompt, e.g., up to 8× faster under an 8x compression constraint.

## E Ethic Statement

This paper introduces GMSA, a context compression framework based on the encoder-decoder architecture. It effectively and efficiently learns summary vectors and bridges the significant gap between different layers via group merging, and a LSA module. The data and models used in our research are released under open-source licenses and sourced from open platforms. Although our work may have various societal impacts, it does not introduce any additional ethical concerns compared to existing text compression methods. Therefore, we believe it is unnecessary to specifically highlight any particular ethical issues here.

## F Language Model Usage Statement

A large language model was used during manuscript preparation solely for editorial support. Its contributions were limited to language polishing

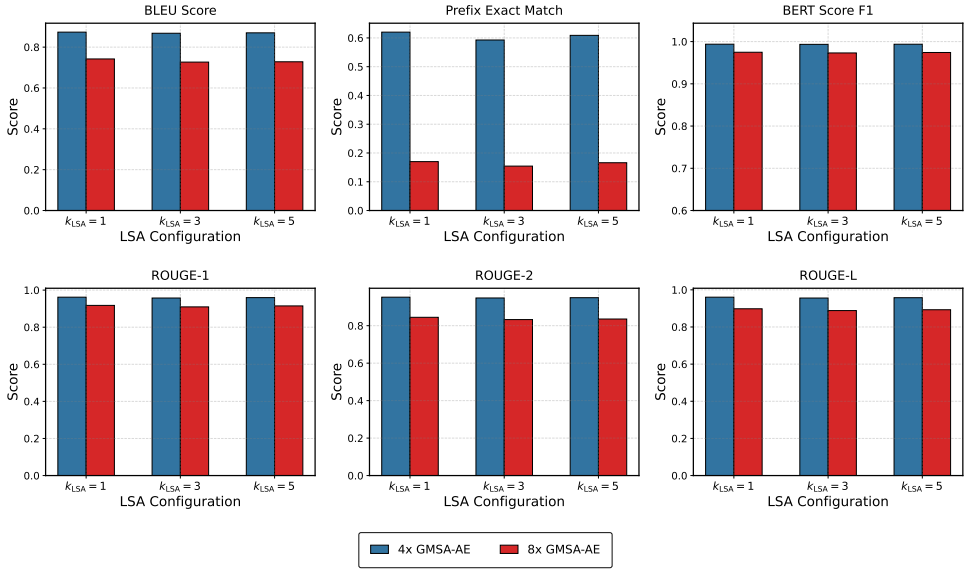



Figure 7: The impact of different number of layers of LSA on semantic retention in GMSA-AE. Sequence Length is set to 512

and clarity enhancement (e.g., rephrasing, tightening explanations, and improving the presentation of formulas/derivations), and it did not generate the underlying research content. All central concepts, methodological decisions, experimental procedures, and empirical findings are the authors' original work.