Efficient Multi-modal Models via Stage-wise Visual Context Compression

Jieneng Chen*, **Luoxin Ye***, **Ju He, Zhao-Yang Wang, Daniel Khashabi**†, **Alan Yuille**†
Johns Hopkins University

Abstract

While significant advancements have been made in compressed representations for text embeddings in large language models (LLMs), the compression of *visual* tokens in large multi-modal models (LMMs) has remained a largely overlooked area. In this work, we present the first study on the analysis of redundancy concerning visual tokens and efficient training within these models. Our initial experiments show that eliminating up to 70% of visual tokens at the testing stage by simply average pooling only leads to a minimal 3% reduction in visual question answering accuracy on the GQA benchmark, indicating significant redundancy in visual context. Addressing this, we introduce *Visual Context Compressor*, which reduces the number of visual tokens during training to enhance training efficiency without sacrificing performance. To further avoid the information loss brought by the compression on visual tokens while maintaining training efficiency, we develop *Stage-wise MLLM Training*, which incorporates stage-wise visual context compression to progressively compress the visual tokens from heavily to lightly, and finally no compression at the end of training, yielding no loss of information when testing. Extensive experiments demonstrate that our approach not only improves the performance of MLLMs but also significantly reduces training costs.

1 Introduction

The advent of LLMs [28, 27, 37] has marked a new era in the field of artificial intelligence and natural language processing. LLMs can play a role as a universal interface for a general-purpose assistant, where various task instructions can be explicitly represented in language and guide the end-to-end trained neural assistant to solve a task of interest. For example, the recent success of ChatGPT [27] and GPT-4 [28] have demonstrated the power of aligned LLMs in following human instructions, and have stimulated tremendous interest in developing open-source LLMs (*e.g.*, LLaMA [36] and Gemma [35]). As the horizon of LLM applications broadens and the availability of open-source LLMs increases, the integration of multi-modality into these models presents a new frontier in expanding their capabilities. Multi-modal LLMs [34, 22, 44, 1] (MLLMs), which can process and understand not just text but also visual information, stand at the cutting edge of this evolution.

While MLLMs have made significant strides, a crucial aspect that remains relatively unexplored is the efficient representation and processing of visual information within these models. Substantial efforts [43, 29] have been dedicated to optimizing the efficient representation of text tokens through various compression techniques, such as those proposed in works like Token Dropping [14], VCC [43] and Nugget [29], aimed at enhancing inference efficiency by attentively selecting important tokens. However, the efficient learning of *visual* tokens in MLLM has not garnered comparable attention. Naturally, this raises questions about the potential redundancy present in visual tokens and its implications for the overall computational efficiency of MLLMs.

*Contributed equally.
†Advised equally.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

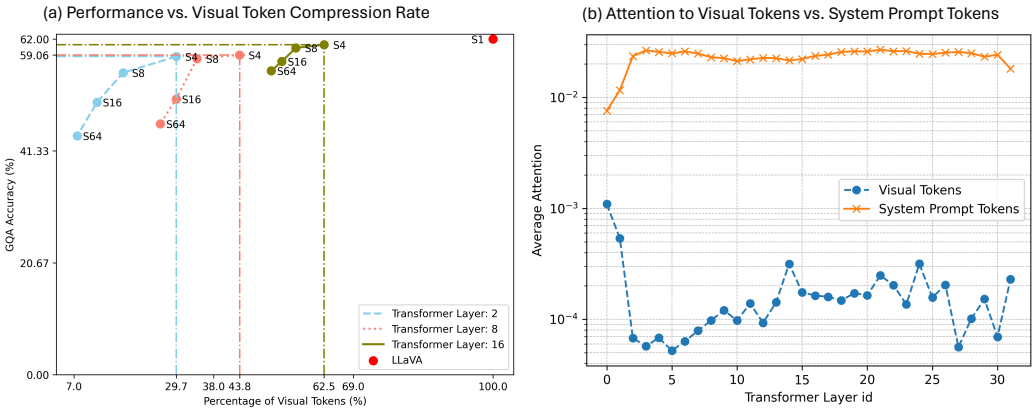

Figure 1: **Visual tokens are redundant in MLLMs. Left:** The accuracy of the LLaVA-1.5-7B [22] model on the GQA [16] benchmarks varies with different percentages of retained visual tokens. The x-axis represents the percentage of original visual tokens preserved after applying 1D average pooling with varying stride sizes S applied in i-th Transformer layer. **Right:** Visual tokens receive less attention from the `[ANS]` token as we go deeper into its layers of LLaVA-1.5-7B model. These findings collectively suggest a significant redundancy within the visual tokens of the MLLMs.

We start our work by addressing the question: *Are visual tokens redundant in multi-modal LLMs?* To explore this, we first experiment with simply reducing the number of visual tokens in a pre-trained LLaVA-1.5-7B [22] at the inference stage via average pooling (§3.2). As shown in Fig.1 (left), our initial results demonstrate that eliminating up to 70% of visual tokens by pooling them with a stride of 4 starting from Transformer layer 2 incurs only a minimal performance loss on the GQA benchmark, specifically a 3% accuracy reduction. Additionally, we compute and present the average attention values from the `[ANS]` token to visual tokens and system prompt tokens across different Transformer layers in the pre-trained LLaVA-1.5-7B [22]. As revealed in Fig. 1 (right; blue trends), the visual tokens are generally less attended to, measured based on average attention from the `[ANS]` token, as the layers get deeper. These two early explorations indicate significant redundancy in visual tokens.

Addressing this, in this work we develop an effective *Visual Context Compressor* that can be integrated into the training of MLLMs. Surprisingly, a simple average pooler nested in LLMs stands out as the most effective compressor, outperforming the attention-based [14, 43] and parametric [18] counterparts. We attribute this to two reasons: (1) The simple pooling operation makes training stable, whereas prior attention-based approaches [14, 43] are specifically designed for accelerating inference rather than training. (2) Visual tokens in the deeper Transformer layers are less attended to (see Fig. 1 (right)) and particularly redundant, making a simple compressor placed in a deeper Transformer layer effective enough. At a lower training cost, the LLaVA-1.5-7B [22] trained with the proposed *Visual Context Compressor* is competitive with the non-compressed baseline across various multi-modal benchmarks (*e.g.*, GQA [16] and MM-Vet [42]). This dual achievement highlights *Visual Context Compressor*'s role as a pivotal advancement in enhancing the efficiency and performance of MLLMs across various multi-modal question-answering benchmarks.

To further mitigate the information loss caused by compressing visual tokens, especially under a large compression ratio (CR), we have devised a multi-stage training scheme, dubbed *Stage-wise MLLM Training*, which progressively employs *Visual Context Compressor* at multiple training stages with different compression ratios (§3.3). Specifically, *Stage-wise MLLM Training* progresses through several stages, beginning with a high level of visual token compression and gradually reducing the compression ratio until the final stages, where full visual tokens are utilized. This multi-stage approach allows for adaptive compression levels that ensure training efficiency without losing information at testing, thus maintaining the overall effectiveness of the model.

Extensive experimental evaluations of *Stage-wise MLLM Training* have been conducted on nine widely-adopted multi-modal LLMs benchmarks: GQA [16], MM-Vet [42], ScienceQA (SQA) [25], MME [10], TextVQA [33], POPE [19], MMBench [24], MMBench-CN [24] and VQA-v2 [11], showing promising results. We observe that *Stage-wise MLLM Training* not only enhances the performance of MLLMs, but also achieves a substantial reduction in training costs. These experiments

validate the effectiveness of our method, demonstrating its capability to optimize resource utilization while maintaining or even improving model performance.

In summary, our paper makes the following contributions:

- We present two initial studies to verify the redundancy of visual tokens in MLLMs.
- We propose *Visual Context Compressor*, a simple yet effective compression technique that utilizes an average pooler, seamlessly integrating into the training of MLLMs.
- We propose *Stage-wise MLLM Training* as an efficient training scheme by leveraging *Visual Context Compressor* at multiple training stages with a progressively decreasing compression rate, thereby enhancing the performance of MLLMs across various benchmarks. Additionally, our approach showcases efficiency gains by reducing training costs by 16%.

2 Related Works

Multi-modal LLMs. The evolution of large language models [28, 27, 7] into their multi-modal counterparts [34, 22] represents a significant leap in their ability to follow instructions and generalize across tasks. This transition has been marked by seminal works such as Flamingo [1], BLIP-2 [18] and LLaVA [22], which have extended LLM capabilities to encompass visual tasks, demonstrating impressive zero-shot generalization and in-context learning abilities. Progress in multi-modal LLMs has primarily been driven by advancements in visual instruction tuning [22, 44], leveraging vision-language datasets and refining visual instruction-following data. Additionally, efforts have been made to enhance the grounding capabilities of multi-modal LLMs through the use of specialized datasets aimed at improving task-specific performance. Despite these advancements, the exploration of visual compression within multi-modal LLMs remains relatively underdeveloped. The design and optimization of compression strategies are crucial for maximizing the effectiveness and efficiency of multi-modal LLMs, suggesting a potential area for future research and development.

Visual Redundancy. In computer vision, reducing redundancy is crucial for creating efficient yet effective models without losing accuracy [4]. Redundancy in images often arises from the inherent characteristics of natural scenes, including repetitive patterns, textures, and areas of uniform color. These features, while contributing to the richness and detail of visual perception, can lead to inefficiencies in both storage and processing when not adequately addressed. Image compression algorithms [39] can reduce file size by eliminating or efficiently encoding redundant data. These methods take advantage of human visual perception's tolerances to subtly reduce data without significantly impacting image quality. Advanced machine learning models, particularly CNNs and autoencoders [3], offer sophisticated approaches to minimizing redundancy. Transformers [38], as a fundamental architecture for LLMs [7, 28], apply self-attention mechanisms to dynamically bind the most informative parts of tokents. Vision Transformers [5, 6, 9, 12] trained with CLIP objective [5, 30] encode an image to a sequence of visual features for multi-modal LLMs [22]. Nevertheless, visual tokens receive less attention in LLMs due to attention shrinkage [40], resulting a waste of computation. In this work, we focus on reducing the redundancy of visual tokens in MLLMs.

Efficient LLMs. Efficient inference and training for LLMs are important. Compressing input sequences for efficiency reasons in Transformers is not a new idea for NLP. Much work is being done to accelerate the inference of LMs. For example, Pyramid Transformer variants [8] and [15] are proposed in Encoder-Decoder LMs that progressively compress the sequence as the layers grow deeper via pooling or core-set selection. Nawrot et al. [26] propose adaptively compressing the sequence based on the predicted semantic boundaries within the sequence. Rae et al. [31] propose compressing the fine-grained past activations to coarser memories. VCC [43] compress the sequence into a much smaller representation at each layer by prioritizing important tokens. Besides efficient inference, accelerating training for LLMs attracts attention as well. A staged training setup [32] is proposed which begins with a small model and incrementally increases the amount of compute used for training by applying a growth operator to increase the model depth and width. However, efficient training for LLMs in multi-modal scenarios is rarely explored.

3 Method

We first introduce an overview of multi-modal LLMs in § 3.1. Then, we define the problem of visual redundancy and introduce *Visual Context Compressor* in § 3.2. Finally, we present our proposed *Stage-wise MLLM Training* in § 3.3.

3.1 Preliminaries: A Multi-modal LLM

We start by reviewing the design of the LLaVA family [22, 21]. For processing an input image $\mathbf{X}_v$, we utilize the pre-trained CLIP visual encoder ViT-L/14, as detailed by [30], to extract the visual feature $\mathbf{Z}_v = g(\mathbf{X}_v)$, where $g(.)$ indicates the visual encoder. To bridge the gap between visual and linguistic modalities, the LLaVA [22, 21] framework as an MLLM implements a straightforward linear/MLP transformation. This involves a trainable projection matrix $\mathbf{W}$, which maps the visual features $\mathbf{Z}_v$ into the linguistic embedding space, producing language embedding tokens $\mathbf{H}_v = \mathbf{W}\mathbf{Z}_v$. These tokens are designed to match the dimensionality of the word embeddings within the LLM.

For each image $\mathbf{X}_v$, one can generate multi-turn conversation data $(\mathbf{X}_q^1, \mathbf{X}_a^1, \cdots, \mathbf{X}_q^T, \mathbf{X}_a^T)$ with T as the number of turns. One can organize them as a sequence, by treating all answers as the assistant's response and the instruction $\mathbf{X}_{\texttt{instruct}}^t$ at the t-th turn as:

$$\mathbf{X}_{\texttt{instruct}}^t = \begin{cases} \text{Random Choose}[\mathbf{X}_q^1, \mathbf{X}_v] \text{ or } [\mathbf{X}_v, \mathbf{X}_q^1], & t = 1 \\ \mathbf{X}_q^t, & t > 1 \end{cases} \tag{1}$$

This approach establishes a standardized format for the multi-modal instruction-following sequence. It allows for the instruction-based tuning of the LLM to be applied to the prediction tokens, utilizing the model's native auto-regressive training objective. Specifically, for a sequence with length L, the likelihood of the target responses $\mathbf{X}_a$ is calculated as:

$$p(\mathbf{X}_a|\mathbf{X}_v, \mathbf{X}_{\texttt{instruct}}) = \prod_{i=1}^{L} p_\theta(x_i|\mathbf{X}_v, \mathbf{X}_{\texttt{instruct},<i}, \mathbf{X}_{a,<i}), \tag{2}$$

3.2 *Visual Context Compressor*

Problem Formulation: The redundancy observed in images often arises from inherent traits of natural scenes, including repetitive patterns, textures, and regions with uniform color. While these traits enrich visual perception by offering detail and depth, they can also present challenges in terms of storage and processing efficiency. Considering the inherent limitations of Transformers in handling long sequences [23, 2, 41], it is critical to minimize any length redundancies to obtain a more effective accuracy/efficiency trade-off.

The objective of this study is to decrease the length of visual tokens $\mathbf{X}_v$ (*i.e.,* its hidden states $\mathbf{H}_v$ if inside LLMs), while simultaneously maximizing the probability of the target response $p(\mathbf{X}_a|\mathbf{X}_v, \mathbf{X}_{\texttt{instruct}})$ as described in Equation (2).

Visual Context Compressor: A key design change that we introduce is a compressor layer that compresses the dimensions of the visual inputs by reducing the effective number of visual tokens. As depicted in Fig. 2, the compressor is simply an average pooler in our setting. It is applied to the visual tokens in k-th Transformer layer of an LLM. Formally, given the hidden visual tokens at k-th Transformer layer $\mathbf{H}_k \in \mathbb{R}^{B\times C\times L}$, the compressor is expected to fulfill the following projection: $f : \mathbb{R}^{B\times C\times L} \mapsto \mathbb{R}^{B\times C\times L_{out}}$, which results in compressed visual tokens $\tilde{\mathbf{H}}_k \in \mathbb{R}^{B\times C\times L_{out}}$, where $L_{out} = \frac{L}{S}$ with s as the compression stride. In §4, we explore multiple variants of compressor f to reduce the token length, including random token dropping [13] with dropping ratio $1 - \frac{1}{S}$, K-Means [17] with number of centroids set to $N_C = \frac{L}{S}$, attention-based token-centric compression [43], attention-based token dropping [14], and average pooling with stride s. To our surprise, we find that the simple average pooler is the most effective compressor for vision tokens within MLLMs, due to its stability during training detailed in § 4.3. Thus, we choose average pooler as the compressor.

Note that the proposed *Visual Context Compressor* can be directly applied to any off-the-shelf MLLMs to assess the visual redundancy, as conducted in §4.2. One can also train an MLLM with

Visual Context Compressor to reduce the number of visual tokens while maintaining competitive multi-modal performance.

Compression Ratio (CR)[3]. For an LLM with N Transformer decoder layers, the compression ratio for visual tokens can be calculated as:

$$\text{CR} = \frac{N \cdot L}{(N-K) \cdot L_{out} + K \cdot L} \quad , \tag{3}$$

where K is the K-th Transformer layer of a multi-modal LLM; L is the the length of visual tokens input into Visual Context Compressor; L_{out} is the compressed length of visual tokens generated by Visual Context Compressor, as illustrated in Fig. 2.

Our architecture modifications thus far mostly impacts the inference efficiency of MLLM, however, its impact on performance-compression trade-off remains unclear. We will study this question in the context of **training** MLLMs with a goal of enhancing efficiency without compromising performance. We then move on to further utilize *Visual Context Compressor* to design an efficient training scheme to incorporates Visual Context Compressor at various stages of the training process.

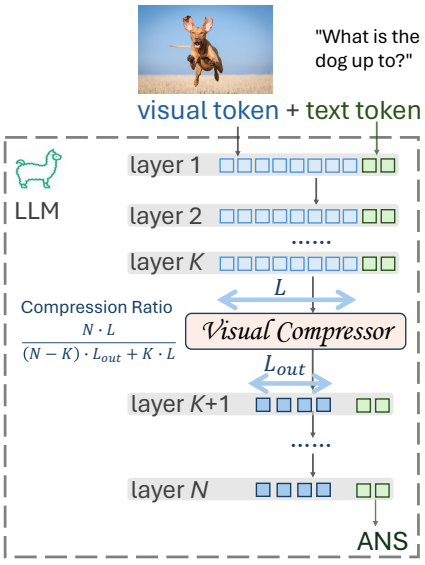

Figure 2: Example of Visual Context Compressor in a multi-modal LLM.

3.3 *Stage-wise MLLM Training*

Training with *Visual Context Compressor* not only facilitates efficient inference but also enhances training efficiency. However, devising an effective training scheme poses challenges when ensuring fair comparisons with the original LLaVA [21], primarily due to differences in the number of tokens involved in inference. This discrepancy may lead to information loss, particularly when operating under a scenario with a high compression ratio. To tackle this issue, we have developed a stage-wise visual token compression approach. Generally, assuming there are N_s total stages, stage i involves $\frac{1}{N_s}$ of the total training epochs with a compression ratio of r_i, and the final stage proceeds without any compression. Essentially, as training progresses, i increases while r_i decreases.

In this work, as depicted in Fig. 3, we primarily explore a three-stage training pipeline that progressively reduces the compression ratio, as detailed below:

Training Stage I: Heavy Compression. The MLLM training at the first one-third of the total training iterations commences with a heavy compression ratio (> 500%), where *Visual Context Compressor* is applied in an early layer of the LLM with a large pooling stride. This setup enables a very fast training speed.

Training Stage II: Light Compression. The MLLM continues training with another one-third of the total training epochs. At this stage, *Visual Context Compressor* is applied at only the deeper layers of the LLM with a smaller pooling stride compared to Training Stage I.

Training Stage III: No Compression. The MLLM continues training with the final one-third of the total training epochs, following the standard MLLM training protocol without compression. Disabling compression in the final stage ensures that the number of tokens remains consistent with the original MLLM during inference, avoiding the loss of information caused by the reduction of visual tokens.

Given the above meta framework, we can instantiate a family of training schemes, as demonstrated in Tab. 1. The single-stage (non-compression) scheme is equivalent to the MLLM baseline. For multi-stage training, the compression stage can either go deeper or wider. "deeper" implies an increase in K (Transformer layer), while "wider" means a decrease in the stride of the pooler.

Note that all training schemes will be standardized to complete just one epoch. Thus, in the three-stage training, each stage will receive one third of an epoch, while in the four-stage training, each stage will receive one fourth of an epoch. Effects of non-uniform stage splitting are presented in the Appendix.

[3]Definition of compression ratio from Wikipedia

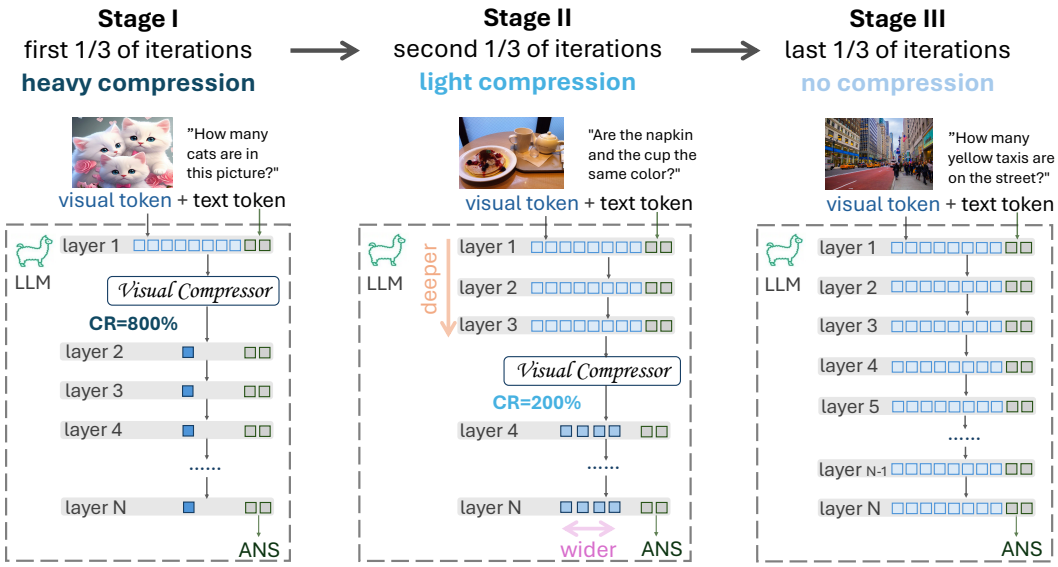

Figure 3: Meta framework of ***Stage-wise MLLM Training***. We observe that visual tokens are redundant and can be compressed. By marrying the observation with efficient training, we propose *Stage-wise MLLM Training* consisting with three training stages: Stage I with heavy visual compression; Stage II with light visual compression in deeper layer with wider token window; Stage III with standard MLLM training (as there is also no compression in standard inference). This can accelerate the training by 16+% while maintaining performance.

#Stages	Scheme	Stage	Layer	Stride	CR	#Epoch
Single	no compression	$S1$	/	/	100%	1
Two	compression	$S1$	2	8	557%	0.5
		$S2$	/	/	100%	0.5
Three	compr. deeper	$S1$	2	8	557%	0.33
		$S2$	16	8	178%	0.33
		$S3$	/	/	100%	0.33
Three	compr. wider	$S1$	2	8	557%	0.33
		$S2$	2	2	188%	0.33
		$S3$	/	/	100%	0.33

#Stages	Scheme	Stage	Layer	Stride	CR	#Epoch
Four	wider then deeper	$S1$	2	8	557%	0.25
		$S2$	2	2	188%	0.25
		$S3$	16	2	133%	0.25
		$S4$	/	/	100%	0.25
Four	deeper then wider	$S1$	2	8	557%	0.25
		$S2$	16	8	178%	0.25
		$S3$	16	2	133%	0.25
		$S4$	/	/	100%	0.25

Table 1: **Instantiations of *Stage-wise MLLM Training* schemes**. deeper indicates that the compressor's position in the LLM shifts from the shallow layer (*e.g.*, 2) to a deeper layer (*e.g.*, 16). wider indicates that the compressor's stride decreases while the number of visual tokens increases.

4 Experiments

In this section, we first detail the experimental setup in § 4.1, elaborate the assessment of the visual redundancy in multi-modal LLMs in § 4.2, validate the proposed *Visual Context Compressor* in § 4.3 and analyze the effectiveness of the proposed *Stage-wise MLLM Training* in § 4.4.

4.1 Experimental Setup

We adopt the Vicuna-v1.5-7B [7] as the language model, leveraging the LLaMA2 codebase [36]. We leverage the pre-trained CLIP ViT-L/14 [30, 9] with an input resolution of 336×336, resulting in 576 visual tokens. We employ the LLaVA framework [21] to connect the frozen CLIP vision encoder and the Vicuna LLMs. Along with the projector, we train the entire LLM instead of parameter-efficient finetuning. We follow LLaVA-1.5 [21] to perform data preparation and training schedule for pretraining and instruction tuning. We conduct all the experiments with the machine of $8\times$ Nvidia RTX 6000 Ada. Due to multiple invalid image links in the dataset of instruction tuning stage, the scores of LLaVA-1.5 reported in our analysis are reproduced by ourselves to ensure a fair comparison under the same experimental environment.

Compressor	#Tokens	CR	GQA	MM-Vet	ScienceQA	MME	VQA^T	POPE	MMB	MMB^{CN}	VQA^{v2}	LLaVA^W	VisWiz	SEED^I	MMMU	AVG
*Train **without** compression; Testing with compression*																
Random Dropping	3312	556%	50.6	21.4	69.3	1142.2	46.5	55.8	39.7	33.3	59.3	47.6	47.2	52.2	34.3	47.3
K-Means	3312	556%	54.4	25.9	**69.7**	1155.2	49.0	**78.6**	55.3	46.1	**69.3**	**57.6**	48.9	56.1	32.9	54.0
Token Dropping [14]	3312	556%	52.1	**30.6**	69.4	**1298.0**	**53.4**	65.6	60.1	**53.0**	68.6	54.8	**50.0**	56.3	**34.9**	**54.9**
VCC [43]	3582	514%	**54.7**	26.9	69.2	1246.1	49.2	72.3	**60.8**	52.0	68.1	55.6	47.8	**57.0**	34.8	54.7
Average Pooling	3312	556%	53.7	25.6	69.4	1150.1	47.7	70.1	56.4	46.5	67.0	55.6	50.0	55.7	34.3	53.0
*Train **with** compression; Testing with compression*																
Random Dropping	3312	556%	53.4	25.0	69.4	1186.4	49.4	64.9	52.0	41.1	59.7	51.5	47.9	52.6	34.6	50.8
K-Means	3312	556%	57.5	25.9	55.6	1279.2	51.4	79.4	62.6	54.6	75.7	59	46.1	59.2	34.1	57.9
Token Dropping [14]	3312	556%	55.9	27.9	70.4	1327.1	49.7	79.8	62.9	55.9	69.5	61.7	**49.6**	56.8	**35.1**	57.0
VCC [43]	3582	514%	57.7	29.3	70.7	1398.1	53.0	83.6	65.0	55.8	74.1	58.0	48.2	60.1	35.0	58.5
Average Pooling	3312	556%	**60.0**	**30.7**	**70.8**	**1450.2**	**55.1**	**85.5**	**65.0**	**59.5**	**75.9**	**66.9**	46.4	**62.6**	33.8	**60.4**

Table 2: **Comparison among different visual compressors**. Higher values are preferred. All methods except VCC are set to the compression ratio of 556% to approximate VCC's 514% [43] for a fair comparison. The best scores are marked as gray and the second best are underlined. Attention-based compressors (*i.e.*, Token Dropping and VCC) excel during the inference phase, yet their application to the training phase proves challenging. Average pooling shows a more stable performance during the training phase.

It is worth mentioning that assessing visual token redundancy only necessitates the inference of existing off-the-shelf models, whereas the other experiments involve the training of multi-modal LLMs, specifically projectors and LLMs.

Benchmarks and Metrics: We adopt nine benchmarks specifically designed for multi-modal LLM evaluation, including GQA [16], MM-Vet [42], ScienceQA (SQA)[25], MME[10], TextVQA [33], POPE [19], MMBench [24], MMBench-CN [24], and VQA-v2 [11]. GQA and VQA-v2 evaluate the model's visual perception capabilities on open-ended short answers. MME-Perception evaluates model's visual perception with yes/no questions. ScienceQA with multiple choice are used to evaluate the zero-shot generalization on scientific question answering. TextVQA contains text-rich visual question answering. MMBench and the CN version evaluate a model's answer robustness with all-round shuffling on multiple choice answers. MM-Vet evaluates a model's capabilities in engaging in visual conversations.
We report the official metrics calculated using the standard implementations provided for each benchmark for a fair comparison. Latency is reported as the time taken during inference until the first answer token is produced. When reporting average performance in Table 6, the score of MME is divided by 2000, as its range is from 800 to 2000. TFLOPs are profiled via DeepSpeed. For total number of tokens, $\#\text{Tokens} = \sum_i^N \#\text{Token}^i$. The training time is reported for one epoch of training during the LLaVA instruction-tuning stage. The Compression Ratio (CR) is defined as in Equation 3.

4.2 Assessing Visual Context Redundancy

To assess the redundancy of visual tokens, we perform average pooling within an off-the-shelf LLaVA-1.5-7B checkpoint at the testing stage, using different pooling stride sizes S across various Transformer layers K. As shown in Fig. 1, the model still exhibits strong performance even when retaining only 62.5% of the visual tokens ($S = 4, K = 16$) in the MM-Vet benchmark, without the need for additional training. When adopting the same setting ($S = 4, K = 16$), a similar trend can be observed in the GQA benchmark as well, where the compressed model only has 1% performance drop than the uncompressed counterpart. Surprisingly, in the GQA benchmark, eliminating up to 70% of visual tokens ($S = 4, K = 16$) results in a mere 3% decrease in performance. This proof-of-concept shows a certain level of redundancy in the visual tokens within MLLMs.

4.3 Effectiveness of *Visual Context Compressor*

In this section, we first study the choice of visual compressors by comparing various compression methods. We then examine the effectiveness of the proposed *Visual Context Compressor*.

Choice of Visual Compressors. The design choices include (1) random token dropping, (2) K-Means clustering, (3) average pooling, (4) Token Dropping [14], (5) VCC [14], (6) parametric pre-trained Q-Former [18]. We have the following three observations. Firstly, Tab. 2 shows that the attention-based methods, including Token Dropping and VCC win 3/4 best and second best scores, showcasing the high performance when compressing visual tokens in inference. However, they are ineffective when applied to training because the in-training attention scores are unstable. Secondly, and surprisingly, the average pooling obtains the highest scores across four benchmarks when it is used to train MLLMs

with a high CR. Thirdly, Tab. 3 shows that both Q-Former and average pooling can obtain reasonably good performance when trained with extremely high CRs, and the average pooling performs better with less training cost. The reason could be that the Q-Former resamples tokens outside the LLM, potentially causing the LLM to overlook crucial information relevant to the response. In contrast, our approach employs average pooling subsequent to Transformer layer K, allowing the initial K layers of the LLM to effectively retain important information from uncompressed tokens. Given these three insights, we select average pooling as our favored approach for visual compression.

Method	#Param	#Tokens	CR	Train Time	GQA	MMVet	SQA	MME	VQA^T	POPE	MMB	MMB^{CN}	VQA^{v2}	LLaVA^w	VisWiz	SEED	MMMU	Avg.
Q-Former [18]	105M	1024	1800%	10.4h	55.7	26.4	69.3	1217	49.2	83.0	57.7	50.7	71.4	64.6	52.6	55.1	34.0	56.2
Pool S64	0	855	2156%	9.2h	55.9	26.3	71.0	1321	51.6	82.5	63.3	55.9	74.5	63.1	47.8	57.3	35.7	57.8
Pool S16	0	1692	1086%	10.1h	58.0	30.4	71.0	1397	56.4	85.9	64.6	57.9	75.9	66.1	46.5	61.4	34.1	59.5

Table 3: **Parametric *vs*. nonparametric visual compressor.** We follow miniGPT-4 [44] that uses Q-Former pre-trained from BLIP-2 [18] as the parametric compressor (All other aspects are maintained as in LLaVA to ensure a fair comparison). Pool S64/16: pooling with stride 64/16 on LLM layer 1. Under comparable CRs, our nonparametric compressor outshines the parametric Q-Former counterpart in terms of both performance and training efficiency.

Performance Across Compression Ratios. Herein, we train the multi-modal LLM with our *Visual Context Compressor* in various settings. As demonstrated in Tab. 4, the proposed method offers certain improvements and trade-offs compared to the state-of-the-art method, LLaVA-1.5-7B. We have the following two observations. Firstly, in the heavy compression level, the performance of MLLM is inversely proportional to the compression ratio (linearly scaling to the number of visual tokens). Secondly, the performance of MLLMs at the light compression level does not correlate directly with the number of visual tokens, making this observation somewhat unexpected. We attribute this to the MLLMs at this level of compression being relatively insensitive to changes in the compression ratio. This indicates that MLLMs trained at a light compression level will not hurt the model performance at all. For instance, the setting of stride 16 in light compression level attains a 188% CR and also outperforms the baseline LLaVA-v1.5-7B across all four metrics. The above observations pave the way for developing a more systematic training scheme.

Method	Compr. level	Stride	#Tokens	CR	Latency	TFLOPs	Train-time	GQA	MM-Vet	ScienceQA	MME	Avg-13
Visual Context Compressor	Heavy layer 2	64	1422	1296%	36.2ms	1.38	9.9	56.0	26.7	71.4	1299.7	57.2
		16	2232	826%	37.6ms	1.71	10.8	58.3	29.2	71.4	1377.3	59.3
		8	3312	557%	37.9ms	2.14	12.0	60.0	30.7	70.8	1450.2	60.4
		4	5472	337%	39.1ms	3.02	12.2	61.1	32.5	71.5	1452.7	61.4
	Light layer 16	64	9360	197%	49.8ms	4.59	12.5	62.3	31.0	71.3	1522.3	62.3
		16	9792	188%	50.2ms	4.77	12.6	63.2	32.9	71.3	1439.8	62.5
		8	10368	178%	51.3ms	5.00	12.8	62.2	31.3	70.2	1505.4	60.8
		4	11520	160%	52.2ms	5.47	13.2	62.4	32.2	70.2	1471.3	61.6
Baseline [21]	-	-	18432	100%	68.5ms	8.26	15.3h	62.0	31.1	70.1	1453.0	61.5

Table 4: **Training MLLMs with *Visual Context Compressor* in various compression levels**. Blue cell: the best number in the heavy compression range, where the performance is inversely proportional to the compression ratio. Gray cell: the best number in the light compression range, where the performance is not sensitive to compression. Performance remains high for models at the light compression level.

4.4 Effectiveness of *Stage-wise MLLM Training*

In this section, we validate the family of *Stage-wise MLLM Training* schemes proposed in § 3.3. We conduct a thorough evaluation of the multi-modal capability across 9 benchmarks in accordance with LLaVA. Tab. 6 demonstrates that our proposed *Stage-wise MLLM Training* not only consistently lowers training costs by 16% (15.3 hours *vs*. 12.8 hours) but also surpasses the non-compression baseline. The four-stage training scheme (deeper, and then wider) achieves the best performance in five out of the nine benchmarks and obtains 65.19% average performance, surpassing LLaVA-v1.5-7B by +0.6% average performance. This indicates the necessity of designing an optimal training scheme.

We extend our training scheme to VideoLLaVA [20] and have similar findings as above. Note that VideoLLaVA doesn't support DeepSpeed ZeRO-3 like LLaVA did.

#Stages	Scheme	#Tokens	CR	Test TFLOPs	Train Time	GQA	MMVet	SQA	MME	VQA^T	POPE	MMB	MMB^{CN}	VQA^{v2}	LLaVA^w	VisWiz	SEED	MMMU	Avg.
Single	no compression	18432	-	8.26	15.3h	62.0	31.1	70.1	1453	58.2	85.9	64.3	58.3	78.5	64.56				
Two	compression	10062	183%	5.20	12.8h	61.9$_{.23}$	31.7$_{1.5}$	70.9$_{.34}$	**1480**$_{23}$	58.3$_{.46}$	86.5$_{.33}$	64.8$_{.23}$	59.0$_{1.1}$	78.5$_{.20}$	67.3$_{.91}$	47.2$_{1.8}$	64.9$_{.17}$	34.9$_{.11}$	61.5$_{.40}$
Three	compr. deeper	10597	174%	5.13	12.8h	62.1$_{.01}$	30.5$_{.40}$	70.5$_{.23}$	1477$_{13}$	58.4$_{.07}$	86.6$_{.14}$	65.6$_{.26}$	59.9$_{.27}$	78.5$_{.22}$	67.5$_{1.4}$	49.2$_{.56}$	65.9$_{.17}$	35.0$_{.19}$	61.8$_{.10}$
Three	compr. wider	10407	177%	3.93	12.8h	61.1$_{1.6}$	**31.8**$_{.61}$	71.0$_{.28}$	1434$_{12}$	58.5$_{.04}$	86.6$_{.06}$	64.8$_{.23}$	59.1$_{.83}$	78.7$_{.02}$	64.3$_{4.8}$	49.8$_{1.1}$	65.3$_{.04}$	34.3$_{.75}$	61.3$_{.28}$
Four	wider then deeper	11088	166%	5.39	12.9h	62.1$_{.09}$	31.6$_{.58}$	**71.4**$_{.36}$	1444$_{15}$	58.7$_{.24}$	**86.8**$_{.21}$	65.3$_{.30}$	59.3$_{.26}$	78.8$_{.05}$	67.7$_{3.1}$	**50.1**$_{.21}$	65.6$_{.15}$	33.8$_{.78}$	61.8$_{.35}$
Four	deeper then wider	10863	170%	5.45	12.8h	62.1$_{.07}$	31.5$_{.20}$	70.5$_{.16}$	1472$_{16}$	**58.7**$_{.08}$	86.3$_{.33}$	**65.6**$_{.52}$	**59.9**$_{.61}$	**78.8**$_{.03}$	**68.2**$_{2.1}$	48.3$_{1.3}$	**66.1**$_{.20}$	**35.1**$_{.02}$	**61.9**$_{.47}$

Table 5: ***Stage-wise MLLM Training* for LLaVA [21]**. See the definition of each training scheme in Tab. 1. #Tokens, CR and TFLOPs are computed average across stages. The derived five training schemes achieve competitive results while reducing 16% training time. We reported the average results across three runs, with the standard deviation written at the bottom right of the average result.

#Stages	Scheme	#Tokens†	CR†	TFLOPs†	Train-time	MSVD-QA		MSRVTT-QA		ActivityNet-QA		Average	
						Score	Acc	Score	Acc	Score	Acc	Score	Acc
Single	no compression	147456	-	29.68	40.7h	3.69	69.1	3.48	56.8	3.28	47.5	3.48	57.8
Two	compression	80496	183%	17.73	37.1h	3.71	69.0	3.50	56.9	3.29	47.9	3.50	57.9
Three	compr. deeper	84776	174%	17.29	37.1h	3.73	69.3	**3.51**	**57.2**	3.28	47.4	3.51	58.0
Three	compr. wider	83256	177%	16.86	37.0h	3.72	69.0	**3.51**	**57.2**	**3.29**	47.7	**3.51**	58.0
Four	wider then deeper	88704	166%	18.32	37.2h	3.72	69.1	**3.51**	**57.2**	3.27	**48.0**	3.50	58.1
Four	deeper then wider	86904	170%	18.64	37.1h	**3.74**	**69.8**	3.49	56.9	3.27	47.8	3.50	**58.2**

Table 6: ***Stage-wise MLLM Training* for VideoLLaVA[20]**. See the definition of each training scheme in Tab. 1. †: average across stages. To implement our multi-stage training, we apply the same compression processing to the 8 frames representing the video respectively. The derived five training schemes achieve competitive results while reducing 9% training time.

Furthermore, we conduct an ablation study on the number of iterations in different stages (uniform *vs*. non-uniform stage splitting), which is detailed in the Appendix. We also explore how the saved 16% training budget can be utilized to enhance performance, as discussed in the Appendix.

5 Conclusion

In this work, we conduct two initial studies to investigate and verify the redundancy of visual tokens in multi-modal LLMs. To address this, we propose *Visual Context Compressor*, a straightforward yet effective compression technique that employs a simple average pooler, seamlessly integrating into the training of MLLMs. This approach enhances training efficiency without compromising performance. To further mitigate the information loss brought by the token compression, we introduce *Stage-wise MLLM Training*, a multi-stage training scheme that utilizes *Visual Context Compressor* with a progressively decreasing compression rate. Experimental results on various visual question answering benchmarks verify the effectiveness of *Stage-wise MLLM Training* in boosting performance while also demonstrating efficiency gains by reducing training costs by 16%. To the best of our knowledge, we are the first to accelerate the training of multi-modal LLM from the compression perspective. We hope that the proposed *Visual Context Compressor* and *Stage-wise MLLM Training* will inspire more in-depth analysis of visual redundancy existing in current MLLMs and call for future designs of efficient training for MLLMs.

References

[1] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

[2] C. Anil, Y. Wu, A. Andreassen, A. Lewkowycz, V. Misra, V. Ramasesh, A. Slone, G. Gur-Ari, E. Dyer, and B. Neyshabur. Exploring length generalization in large language models. *arXiv preprint arXiv:2207.04901*, 2022.

[3] P. Baldi. Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 37–49. JMLR Workshop and Conference Proceedings, 2012.

[4] H. Barlow. Redundancy reduction revisited. *Network: computation in neural systems*, 12(3):241, 2001.

[5] J. Chen, Q. Yu, X. Shen, A. Yuille, and L.-C. Chen. Vitamin: Designing scalable vision models in the vision-language era. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[6] J.-N. Chen, S. Sun, J. He, P. H. Torr, A. Yuille, and S. Bai. Transmix: Attend to mix for vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12135–12144, 2022.

[7] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6, 2023.

[8] Z. Dai, G. Lai, Y. Yang, and Q. Le. Funnel-transformer: Filtering out sequential redundancy for efficient language processing. *Advances in neural information processing systems*, 33:4271–4282, 2020.

[9] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[10] C. Fu, P. Chen, Y. Shen, Y. Qin, M. Zhang, X. Lin, Z. Qiu, W. Lin, J. Yang, X. Zheng, et al. Mme: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*, 2023.

[11] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *CVPR*, 2017.

[12] J. He, J.-N. Chen, S. Liu, A. Kortylewski, C. Yang, Y. Bai, and C. Wang. Transfg: A transformer architecture for fine-grained recognition. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 852–860, 2022.

[13] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[14] L. Hou, R. Y. Pang, T. Zhou, Y. Wu, X. Song, X. Song, and D. Zhou. Token dropping for efficient bert pretraining. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 2022.

[15] X. Huang, A. Khetan, R. Bidart, and Z. Karnin. Pyramid-bert: Reducing complexity via successive core-set based token selection. *arXiv preprint arXiv:2203.14380*, 2022.

[16] D. A. Hudson and C. D. Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709, 2019.

[17] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE transactions on pattern analysis and machine intelligence*, 24(7):881–892, 2002.

[18] J. Li, D. Li, S. Savarese, and S. Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[19] Y. Li, Y. Du, K. Zhou, J. Wang, W. X. Zhao, and J.-R. Wen. Evaluating object hallucination in large vision-language models. *arXiv preprint arXiv:2305.10355*, 2023.

[20] B. Lin, B. Zhu, Y. Ye, M. Ning, P. Jin, and L. Yuan. Video-llava: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*, 2023.

[21] H. Liu, C. Li, Y. Li, and Y. J. Lee. Improved baselines with visual instruction tuning, 2023.

[22] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.

[23] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*, 2023.

[24] Y. Liu, H. Duan, Y. Zhang, B. Li, S. Zhang, W. Zhao, Y. Yuan, J. Wang, C. He, Z. Liu, et al. Mmbench: Is your multi-modal model an all-around player? *arXiv preprint arXiv:2307.06281*, 2023.

[25] P. Lu, S. Mishra, T. Xia, L. Qiu, K.-W. Chang, S.-C. Zhu, O. Tafjord, P. Clark, and A. Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 2022.

[26] P. Nawrot, J. Chorowski, A. Łańcucki, and E. M. Ponti. Efficient transformers with dynamic token pooling. *arXiv preprint arXiv:2211.09761*, 2022.

[27] OpenAI. ChatGPT. `https://openai.com/blog/chatgpt/`, 2023.

[28] OpenAI. Gpt-4 technical report, 2023.

[29] G. Qin and B. Van Durme. Nugget: Neural agglomerative embeddings of text. In *International Conference on Machine Learning*, pages 28337–28350. PMLR, 2023.

[30] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[31] J. W. Rae, A. Potapenko, S. M. Jayakumar, and T. P. Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*, 2019.

[32] S. Shen, P. Walsh, K. Keutzer, J. Dodge, M. Peters, and I. Beltagy. Staged training for transformer language models. In *International Conference on Machine Learning*, pages 19893–19908. PMLR, 2022.

[33] A. Singh, V. Natarajan, M. Shah, Y. Jiang, X. Chen, D. Batra, D. Parikh, and M. Rohrbach. Towards vqa models that can read. In *CVPR*, 2019.

[34] G. Team, R. Anil, S. Borgeaud, Y. Wu, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[35] G. Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S. Kale, J. Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

[36] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[37] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[38] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[39] G. K. Wallace. The jpeg still picture compression standard. *IEEE transactions on consumer electronics*, 38(1):xviii–xxxiv, 1992.

[40] G. Xiao, Y. Tian, B. Chen, S. Han, and M. Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

[41] X. Ye, A. Wang, J. Choi, Y. Lu, S. Sharma, L. Shen, V. Tiyyala, N. Andrews, and D. Khashabi. AnaloBench: benchmarking the identification of abstract and long-context analogies. *arXiv preprint arXiv:2402.12370*, 2024.

[42] W. Yu, Z. Yang, L. Li, J. Wang, K. Lin, Z. Liu, X. Wang, and L. Wang. Mm-vet: Evaluating large multimodal models for integrated capabilities. *arXiv preprint arXiv:2308.02490*, 2023.

[43] Z. Zeng, C. Hawkins, M. Hong, A. Zhang, N. Pappas, V. Singh, and S. Zheng. Vcc: Scaling transformers to 128k tokens or more by prioritizing important tokens. *Advances in Neural Information Processing Systems*, 36, 2024.

[44] D. Zhu, J. Chen, X. Shen, X. Li, and M. Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

Appendix

In the appendix, we provide additional information as listed below:

- § A provides the additional experimental results.
- § B provides the dataset information and licenses.

A Additional Experimental Results

A.1 Non-uniform Stage Splitting

By default, the training time is evenly divided across each stage. To explore how the compression stage affects total training time, we modify the relative proportion of different stages. This variation is tested in the two-stage setup referenced in Tab. 1, adjusting from the standard 50% in Stage 1 and 50% in Stage 2 to different distributions. Tab. 7 below displays the results of these experiments.

Stage 1	Stage 2	#Tokens	CR	GQA	MMVet	SQA	MME	VQA^T	POPE	MMB	MMB^{CN}
0%	100%	18432	-	62.0	31.1	70.1	1453.0	58.2	85.9	64.3	58.3
25%	75%	11088	166%	62.1	31.7	70.6	1474.5	58.8	86.4	65.1	59.6
50%	50%	10863	170%	62.2	30.0	70.3	1443.5	57.5	85.8	64.8	59.7
75%	25%	10597	174%	61.6	32.2	70.8	1471.5	57.5	86.6	65.2	58.9
90%	10%	10407	177%	61.2	31.0	70.5	1447.5	56.3	86.4	64.4	56.9
100%	0%	10062	183%	55.9	29.5	64.1	1257.8	49.1	86.6	47.4	29.2

Table 7: **Effects of non-uniform stage splitting at the two-stage set-up**. Performance decreases as the proportion of Stage 2 decreases, albeit at the expense of lower compression ratios.

We observe that as the Stage 2 increases from 0% to 100%, there is a gradual decrease in the model's performance across various metrics (such as GQA, MMVet, SQA, MME, VQA, POPE, MMB, and MMB^{CN}). Although there is a decline in performance, it is relatively minor when the compression stage makes up to 50% of the training duration. However, when the proportion of the compression stage is reduced below 50%, the decline in performance becomes more significant. In conclusion, keeping the compression stage between 0-50% of the training time minimizes performance loss while still achieving significant compression ratios.

B Datasets Information and Licenses

GQA: The GQA: [16] dataset, consists of 22M questions about various day-to-day images.

License: N/A

Dataset website: `https://cs.stanford.edu/people/dorarad/gqa/download.html`

MM-Vet: The MM-Vet [42] dataset, defining 6 core VL capabilities and examines the 16 integrations of interest derived from the capability combination.

License: Apache License. `https://github.com/yuweihao/MM-Vet/blob/main/LICENSE`

Dataset website: `https://github.com/yuweihao/MM-Vet/tree/main`

SQA: The SQA: [25] dataset, consisting of 21k multimodal multiple choice questions with a diverse set of science topics and annotations of their answers with corresponding lectures and explanations.

License: CC BY-NC-SA (Attribution-NonCommercial-ShareAlike) `https://creativecommons.org/licenses/by-nc-sa/4.0/`

Dataset website: https://scienceqa.github.io/#download

POPE: The POPE [19] dataset can evaluate the object hallucination in a more stable and flexible way.

License: MIT License. https://github.com/RUCAIBox/POPE?tab=MIT-1-ov-file#readme

Dataset website: https://github.com/RUCAIBox/POPE

MMBench: The MMBench [24] dataset is a collection of benchmarks to evaluate the multi-modal understanding capability of large vision language models.

License: Apache License. https://github.com/open-compass/MMBench?tab=Apache-2.0-1-ov-file#readme

Dataset website: https://github.com/open-compass/MMBench

MMBench-CN: The MMBench-CN [24] dataset is a collection of benchmarks to evaluate the multi-modal understanding capability of large vision language models.

License: Apache License. https://github.com/open-compass/MMBench?tab=Apache-2.0-1-ov-file#readme

Dataset website: https://github.com/open-compass/MMBench

MME: The MME [10] dataset containing 30 images with 60 instruction-answer pairs for coarse-grained recognition task; 917 images for fine-grained recognition task; 20 images with 40 instruction-answer pairs for OCR task.

Dataset website: https://github.com/QwenLM/Qwen-VL/blob/master/eval_mm/mme/EVAL_MME.md

License: Tongyi Qianwen LICENSE AGREEMENT. https://github.com/QwenLM/Qwen-VL/tree/master?tab=License-1-ov-file#readme

TextVQA: The TextVQA [33] dataset containing 30 images with 60 instruction-answer pairs for coarse-grained recognition task; 917 images for fine-grained recognition task; 20 images with 40 instruction-answer pairs for OCR task.

Dataset website: https://github.com/facebookresearch/mmf.git

License: BSD LICENSE. https://github.com/facebookresearch/mmf/blob/main/LICENSE

VQA-v2: The VQA-v2 [11] dataset, containing 265,016 images, dataset containing open-ended questions about images. These questions require an understanding of vision, language and commonsense knowledge to answer.

License: Commons Attribution 4.0 International License. https://visualqa.org/terms.html

Dataset website: https://visualqa.org/