

Dynamic Image Tokenization for Efficient VLMs

Anonymous authors

Paper under double-blind review

Abstract

Vision Language Models (VLMs) exhibit impressive cross-modal understanding, but the large quantity of image tokens results in substantial computational overhead. Prior efforts to reduce image tokens have mainly relied on training-free approaches, which do not properly align the LLM with the compressed visual representations at inference time. We observe that, after finetuning, even the straightforward baseline of merging a fixed number of tokens at a time can outperform current training-free approaches. However, fixed pooling cannot adjust to the specific content of an image, leading to degraded performance on complex OCR tasks that require fine-grained understanding. In this paper, we introduce DRIP, a straightforward approach for dynamically tokenizing images that uses a lightweight MLP-based boundary predictor to close this performance gap. Our findings indicate that dynamic image tokenization can restore accuracy on several OCR benchmarks while preserving performance on other coarse-grained tasks. We will release the source code and model checkpoints once the paper is accepted.

1 Introduction

Vision-Language Models (VLMs) (Alayrac et al., 2022; Liu et al., 2023; Bai et al., 2023; Chen et al., 2024), which generate textual outputs conditioned on both image inputs and text prompts, have demonstrated strong performance on a wide range of visual question answering benchmarks, including reasoning (Hendrycks et al., 2021; Ainslie et al., 2023), OCR-based understanding (Singh et al., 2019; Liu et al., 2024c), hallucination evaluation (Li et al., 2023), and open-ended responses (Yu et al., 2024). However, the excessively large number of image tokens incurs substantial computational overhead and undermines both efficiency and scalability (Shang et al., 2026; Yu et al., 2026; Zhang et al., 2025). Numerous previous approaches have been introduced to compress image tokens, both in standalone vision transformers (Dosovitskiy et al., 2021) and in vision-language models (VLMs). For standalone ViT architectures, EViT (Liang et al., 2022) leverages attention from the class token to detect salient tokens and merge the remaining ones, whereas ToME (Bolya et al., 2023) relies on token cosine similarity to aggregate redundant image tokens. For vision-language models such as LLaVA (Liu et al., 2023), LLaVA-PruMerge (Shang et al., 2026) leverages both class-attention and cosine similarity to cluster the image features produced by ViTs.

Nevertheless, it is important to note that all current approaches are **training-free**. While this makes them straightforward to use in a plug-and-play fashion, they do not align LLM training with the compressed visual representations, indicating room for further improvement. We claim that by finetuning the LLM on compressed image tokens, the language model can be conditioned to answer questions even when provided with only limited visual information. In our experiments, we observe that finetuning with the simplest baseline, fixed pooling, which merges tokens by aggregating every fixed number of consecutive tokens in the one-dimensional sequence, can surpass existing training-free approaches such as LLaVA-PruMerge (Shang et al., 2026) in performance. This observation calls into question the prevailing direction of designing training-free approaches that avoid any retraining.

Although finetuning with fixed pooling is effective, this strategy cannot adjust to the specific content of each image. As a result, we observe a substantial drop in performance on

complex OCR tasks that require much more fine-grained understanding. Building on this observation, we argue that dynamic image tokenization is necessary to tackle complex VQA tasks. In this paper, we introduce DRIP, a straightforward approach that dynamically tokenizes images using a lightweight MLP-based boundary predictor, aiming to address this performance gap. Our boundary predictor outputs a boundary mask that classifies tokens as either boundary or non-boundary tokens, where each sequence of non-boundary tokens is grouped together with the subsequent boundary token. During training, we sample from a Relaxed Bernoulli distribution to obtain the boundary mask in the forward pass, while incorporating a binomial boundary loss during backpropagation to regulate the compression ratio. During inference, to fairly compare with other compression methods under an identical compute budget, we choose the ‘top-K’ boundary tokens with the largest boundary probabilities so that the overall compression matches theirs.

Our experiments demonstrate that dynamic image tokenization can restore accuracy on several OCR benchmarks, while still preserving performance on other coarse-grained tasks. Overall, our approach offers the most favorable FLOP–accuracy balance when compared with fixed pooling strategies and other training-free methods.

2 Method

In this section, we describe the architecture of our proposed dynamic image tokenization for VLMs, and detail how the boundaries are predicted by the boundary predictor module.

2.1 Architecture

In this section, we introduce our proposed approach, DRIP, illustrated in Figure 1. Overall, our design largely adheres to the LLaVA architecture (Liu et al., 2023; 2024a), with the main difference being the insertion of a lightweight boundary predictor between the ViT-L (Dosovitskiy et al., 2021) image encoder and the MLP projector. The input image is patchified into a sequence of patches, which are encoded by the ViT-L image encoder. The boundary predictor, which will be described in detail in Section 2.2, generates a boundary mask that divides tokens into boundary tokens and non-boundary tokens. All non-boundary tokens will be combined together and attached to the following boundary token. We use the most basic form of average pooling to combine these image tokens. The compressed features of the input image are then fed into an MLP projector and subsequently provided to the vicuna-7b-v1.5 language model (Zheng et al., 2023). On the language-processing side, the prompt is first tokenized, then passed through an MLP projector, and finally into the language model. During the token alignment phase (Liu et al., 2023), we train only the MLP projectors and the boundary predictor, keeping all other components fixed. In contrast, during the instruction finetuning phase (Liu et al., 2023), we freeze only the image encoder and update all remaining parameters.

2.2 Boundary Prediction

Boundary Predictor The boundary predictor is implemented as a simple two-layers Multi-Layer Perceptron. We report its parameter overhead in Appendix E. To model content dynamics and handle the non-uniform information density within images, the boundary predictor is used to learn a sequence of binary boundaries. Formally, for each hidden-state input $\mathbf{x}$ composed of l image patches, the dynamic boundary sequence $\mathbf{b}$ takes values in $\{0, 1\}^l$, where $b_t = 1$ indicates a predicted boundary between the adjacent patches x_t and x_{t+1} (Nawrot et al., 2023). To enable end-to-end training, we adopt the Gumbel-Sigmoid reparameterization, which makes the dynamic boundary sequence $\mathbf{b}$, originally a discrete Bernoulli random variable, differentiable. Formally, the probability of classifying a given image patch as a boundary, denoted as $\hat{b}_t = p(b_t = 1)$, is obtained by introducing a stochastic random variable u as follows:

$$\hat{b}_t = \text{sigmoid} \left[\log \frac{\hat{b}_t u}{(1 - \hat{b}_t)(1 - u)} \right]^{1/\tau}, u \sim \text{Uniform}(0, 1). \quad (1)$$

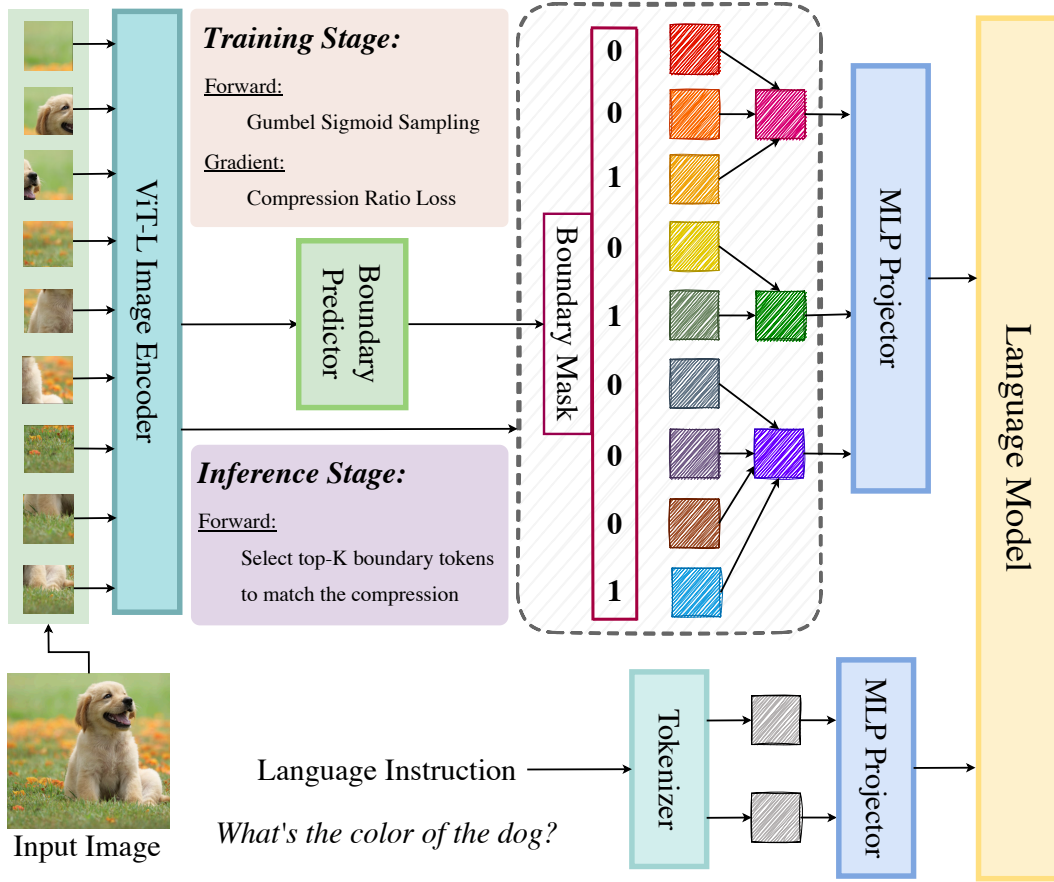


Figure 1: **DRIP architecture.** Our proposed boundary predictor outputs a boundary mask that divides tokens into boundary and non-boundary tokens, where each group of consecutive non-boundary tokens is merged together with the subsequent boundary token.

where τ represents a temperature hyperparameter. During inference, to fairly compare with other compression approaches under an identical compute budget, we choose the ‘top-K’ boundary tokens with the highest boundary probabilities so that the compression rate exactly matches the specified budget. The pseudocode for the training and inference stages is provided in Appendix B.

Boundary Loss The boundary predictor is further characterized by a boundary rate ϕ , or compression ratio, such that $(100 * \phi)\%$ of the patches are designated as boundaries. Regarding the training objective, we incorporate an auxiliary boundary loss, which encourages the boundary predictor to classify approximately $(100 * \phi)\%$ of image patches as boundaries as accurately as possible. Formally, let $\text{preds}_{i,t} \in [0, 1]$ denote the boundary predictions for sequence i of length T . Then, the boundary predictor outputs the following number of boundary patches:

$$k_i = \sum_{t=1}^T \text{preds}_{i,t} \quad (2)$$

Assuming a Binomial distribution $K \sim \text{Binomial}(T, \phi)$ with boundary rate ϕ , the boundary loss is defined as the negative log-likelihood of the number of boundary patches under this distribution:

$$\mathcal{L}_{\text{boundary}} = -\frac{1}{BT} \sum_{i=1}^B \log \left[\binom{T}{k_i} \phi^{k_i} (1 - \phi)^{T-k_i} \right], \quad (3)$$

where B is the batch size. Intuitively, this loss discourages deviations from the expected number of boundaries, $T\phi$, pushing the model to preserve a stable boundary rate (i.e., compression ratio). The pseudocode for the boundary loss is provided in Appendix B, and the corresponding loss curve is presented in Appendix F, where the derivation of the theoretical minimum boundary loss is also included.

Total Loss To jointly optimize the main objective, namely the language modeling cross-entropy loss, together with the auxiliary boundary prediction task, we simply include the auxiliary boundary loss as an additional term in the primary training loss. Formally, the model is trained under the following combined objective:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{primary}} + \mathcal{L}_{\text{boundary}}, \quad (4)$$

where $\mathcal{L}_{\text{primary}}$ denotes the primary training loss, and $\mathcal{L}_{\text{boundary}}$ represents the boundary loss.

3 Experiments

3.1 Experiment Setup

Finetuning Recipe. We largely adopt the finetuning setup and hyperparameter configuration of LLaVA 1.5 (Liu et al., 2024a). In the token alignment phase (Liu et al., 2023), we train only the MLP projectors and the boundary predictor while freezing all other components; in the subsequent instruction finetuning phase (Liu et al., 2023), we freeze the image encoder and update all remaining modules. Our language backbone is vicuna-1.5-7B (Zheng et al., 2023), and we use ViT-L-14 (Radford et al., 2021) as the image encoder with an input resolution of 336×336 . Further hyperparameter settings are provided in Appendix A.

General VQA benchmarks. We evaluate our method on 9 general VQA benchmarks. Among them, 4 are widely used, comprehensive visual question answering datasets in a multiple-choice format: VQAv2 (Agrawal et al., 2016), SQA (also referred to as ScienceQA) (Lu et al., 2022), MME (Fu et al., 2025a), and MM-Bench (Liu et al., 2024b). In addition, we include 2 benchmarks focused on complex reasoning, GQA (Ainslie et al., 2023) and MMMU (Hendrycks et al., 2021). To assess hallucination, we adopt POPE (Li et al., 2023). We also consider benchmarks featuring free-form responses, such as LLaVA-Bench-in-the-wild (Liu et al., 2023) and MM-Vet Yu et al. (2024), where performance is scored by an LLM-based judge. Further details about these benchmarks are provided in Appendix H.

OCR benchmarks. OCR benchmarks are specifically designed to assess how well VLMs handle text perception, recognition, and comprehension within images. In this work, we adopt five such benchmarks: TextVQA (Singh et al., 2019), OCRBench (Liu et al., 2024c), OCRBench v2 (Fu et al., 2025b), DocVQA (Mathew et al., 2021), and ChartQAPro (Masry et al., 2025). Unlike general-purpose VQA benchmarks, OCR benchmarks require fine-grained perception over the specific regions where text appears, which makes visual compression more difficult. Further details on these OCR benchmarks can be found in Appendix H.

FLOP Measurement We adopt TFLOPs as our efficiency metric and use the fvcare (FAIR) library to automatically measure it via a single forward pass of the model in inference-only mode. Our FLOPs calculation is restricted to the prefill stage of inference, since this phase accounts for the majority of vision-related computation, whereas the decode stage primarily involves autoregressive language-model generation with images and prompts already stored in the KV cache.

3.2 Main Results

A Motivating Example. We provide a motivating example in Figure 2 to illustrate why finetuning the LLM with compressed visual features is necessary. Specifically, we study

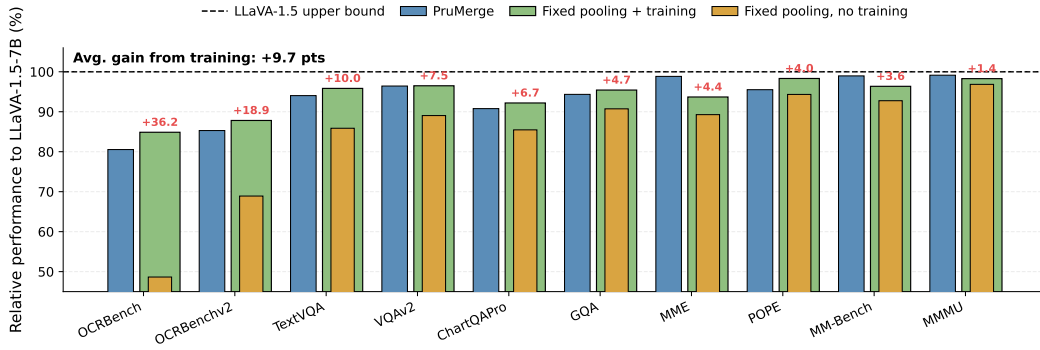


Figure 2: **A motivating example demonstrating that training can reliably and substantially restore performance.** Under 4x compression, we observe a significant performance difference between training-free fixed pooling and trained fixed pooling approaches.

155 a 4x compression setup where the number of image tokens is reduced by 75%. We first
 156 observe that using fixed pooling as a training-free strategy leads to substantial performance
 157 degradation compared to state-of-the-art training-free baselines such as LLaVA-PruMerge.
 158 In contrast, when we finetune the MLLM (i.e., LLaVA) with fixed pooling, the model
 159 largely recovers its performance and can even surpass PruMerge. This result supports our
 160 hypothesis that training the LLM to align with the compressed visual representation can
 161 substantially boost performance.

Model	VQAv2	SQA	MME	MM-Bench	GQA	MMMU	TextVQA	OCRBench	OCRBench2	DocVQA	ChartQAPro	POPE	LLaVA-Wild	MM-Vet	TFL0Ps
Benchmark Type	general				reasoning		OCR				hallucination		free-response		
upper bound (keep 576/576 tokens out of the image encoder)															
LLaVA-1.5-7B	76.75	68.42	1425.34	74.43	58.34	34.9	58.16	18.5	23.8	25.61	12.04	85.18	57.4	29.0	8.89
4x compression (keep 144/576 tokens out of image encoder)															
PruMerge(ICCV'25)	74.01	68.07	1408.83	73.66	55.04	34.6	54.69	14.9	20.3	16.37	10.93	81.36	61.7	27.0	3.18
fixed pooling	74.06	67.82	1335.40	71.72	55.67	34.3	55.74	15.7	20.9	18.78	11.10	83.75	64.1	29.9	3.16
DRIP	74.59	69.31	1341.93	74.05	55.16	34.7	56.08	16.8	20.9	21.27	11.38	82.83	59.5	28.6	3.18
8x compression (keep 72/576 tokens out of the image encoder)															
PruMerge(ICCV'25)	70.67	68.07	1311.21	72.33	52.01	34.3	54.46	14.6	18.4	14.28	10.48	77.53	58.8	25.3	2.22
fixed pooling	72.13	69.11	1280.78	71.65	53.78	34.0	54.05	14.3	19.5	16.00	10.81	82.15	50.7	25.8	2.21
DRIP	72.58	68.47	1310.01	71.90	54.23	32.9	54.75	15.7	20.4	18.50	10.54	82.52	63.6	29.5	2.23
10x compression (keep 58/576 tokens out of the image encoder)															
PruMerge(ICCV'25)	69.15	68.17	1258.86	71.40	50.90	33.6	54.01	13.7	17.9	13.76	10.09	76.40	52.5	26.7	2.02
fixed pooling	71.49	68.02	1359.05	72.17	53.02	34.3	53.40	13.9	19.1	15.32	11.25	82.14	57.1	26.1	2.01
DRIP	71.59	69.61	1313.76	71.35	52.93	33.2	54.83	15.2	20.4	17.42	10.84	82.59	55.8	26.8	2.04

Table 1: **LLaVA-1.5-7B full-parameter finetuning results using the last layer’s image features.** The **best** results are highlighted among **training-free** methods and **training-based** methods.

162 **Overall Performance.** We consider three baselines: the original LLaVA that keeps all
 163 image tokens, LLaVA-PruMerge, and a finetuning approach with fixed pooling. We finetune
 164 LLaVA 1.5 7B using full-parameter tuning (Table 1). To more comprehensively assess its
 165 finetuning behavior, we additionally report LoRA finetuning results (Table 5) in Appendix C.
 166 All baselines and our method are evaluated under three compression settings: 4x, 8x, and
 167 10x. The experiments show that finetuning with fixed pooling outperforms PruMerge on
 168 multiple benchmarks under the same compute budget. Therefore, we find that the simplest
 169 baseline of merging every certain number of tokens, after finetuning, can outperform exist-
 170 ing training-free approaches. We present additional result analysis including benchmark
 171 compressibility and McNemar’s test in Appendix D.

172 **OCR Performance.** We conduct a detailed analysis of results on five selected OCR bench-
 173 marks. Our observations show that dynamic image tokenization consistently outperforms
 174 fixed pooling at all compression ratios, with particularly notable improvements at higher

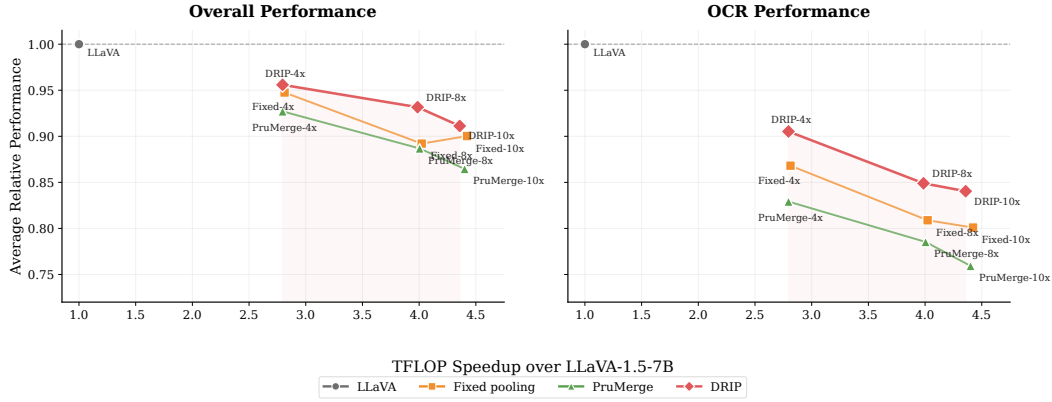


Figure 3: **FLOP-Accuracy tradeoff** (results from full finetuning based on image features from the final layer of the image encoder).

175 compression levels. On additional coarse-grained VQA benchmarks, dynamic tokenization
 176 preserves its performance. Based on these findings, we contend that dynamic image tok-
 177 enization can restore accuracy on multiple OCR benchmarks while still maintaining strong
 178 performance on other coarse-grained tasks. In Section 3.2, we further provide concrete
 179 OCR examples from TextVQA to illustrate the scenarios in which dynamic tokenization is
 180 particularly beneficial.

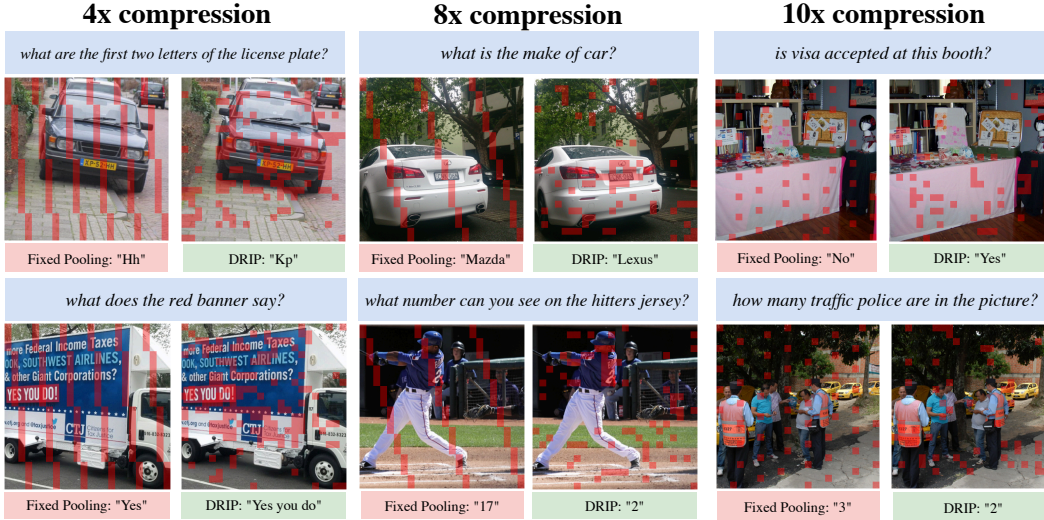


Figure 4: **TextVQA examples at various compression ratios**. With dynamic tokenization, more boundary tokens are assigned to text-dense regions, retaining richer visual details there while achieving stronger compression in the background areas.

181 **FLOP-Accuracy Tradeoff.** To compare the average performance across different ap-
 182 proaches, we further examine the tradeoff between TFLOPs and VQA accuracy, as illustrated
 183 in Figure 3. The x-axis reports the relative TFLOP speedup with respect to the original
 184 LLaVA-1.5-7B, while the y-axis indicates the relative performance to LLaVA-1.5-7B, aggre-
 185 gated over all 14 benchmarks. In general, models that lie closer to the upper-right region
 186 achieve a more favorable FLOP-accuracy tradeoff. We find that dynamic tokenization

(DRIP) attains the best *overall* performance and significantly outperforms fixed pooling and PruMerge baselines on OCR benchmarks.

Boundary Visualization. To assess the effectiveness of dynamic tokenization for images in OCR-focused tasks, we examine a set of TextVQA (Singh et al., 2019) benchmark questions where fixed pooling (i.e., fixed tokenization) produces incorrect answers, but dynamic tokenization (i.e., DRIP) succeeds. In Figure 4, we present two examples for each compression ratio under consideration. We mark *boundary tokens* with red squares for both fixed pooling and dynamic tokenization to illustrate how each method partitions the image. Using the same image and question, we observe that fixed pooling often fails to capture the relevant text regions and consequently answers incorrectly. In contrast, dynamic tokenization typically places more boundary tokens in text-rich areas, retaining crucial visual details while applying stronger compression in background regions. Additional qualitative examples and validation are provided in Appendix G.

Image Feature Layer Analysis. In LLaVA (Liu et al., 2023), features from both the last layer and second to last layer are considered. Following this setup, we additionally report our full-parameter training and evaluation results based on image features extracted from the second to last layer of the vision encoder, as shown in Table 2. We find no notable performance drift compared to using features from the last ViT layer. This suggests that our method is robust to variations in visual features and that our experimental outcomes are largely insensitive to the specific choice of image feature layer.

Model	VQAv2	SQA	MME	MM-Bench	GQA	MMMU	TextVQA	OCRBench	OCRBenchv2	DocVQA	ChartQAPro	POPE	LLaVA-Wild	MM-Vet	TFLOPs
Benchmark Type	general			reasoning		OCR				hallucination		free-response			
upper bound (keep 576/576 tokens out of the image encoder)															
LLaVA-1.5-7B	76.96	68.32	1459.94	73.61	58.79	34.3	57.43	19.3	24.0	25.46	11.74	85.13	60.7	29.2	8.89
4x compression (keep 144/576 tokens out of image encoder)															
PruMerge(ICC'25)	74.54	69.31	1424.85	73.50	55.29	34.9	55.26	15.8	20.6	16.82	11.26	82.37	60.4	27.3	3.18
fixed pooling	74.86	67.87	1413.55	73.95	55.93	34.1	55.62	15.8	22.1	18.75	10.92	84.88	69.0	28.5	3.16
DRIP	75.12	69.26	1362.44	74.14	55.79	34.4	56.16	17.1	22.6	21.08	10.89	84.07	62.8	28.0	3.18
8x compression (keep 72/576 tokens out of the image encoder)															
PruMerge(ICC'25)	71.50	68.96	1357.71	72.84	52.37	34.3	54.65	14.8	19.5	15.01	10.87	78.42	58.7	26.8	2.22
fixed pooling	73.08	68.96	1320.00	72.68	54.14	34.4	53.93	14.3	20.4	16.22	10.99	82.56	59.8	26.3	2.21
DRIP	73.18	69.01	1356.79	73.13	54.05	34.6	55.51	15.8	21.0	18.28	11.18	82.34	62.3	27.2	2.23
10x compression (keep 58/576 tokens out of the image encoder)															
PruMerge(ICC'25)	70.17	69.41	1325.20	72.49	51.43	34.3	54.25	14.6	18.8	14.35	10.77	77.28	50.1	27.2	2.02
fixed pooling	72.48	67.77	1325.38	73.00	54.01	33.0	53.66	13.7	20.3	15.49	10.17	83.06	60.0	26.8	2.01
DRIP	72.42	70.15	1310.24	73.47	53.56	33.6	55.39	14.5	20.9	17.31	10.93	82.81	60.7	28.3	2.04

Table 2: LLaVA-1.5-7B full-parameter finetuning results using the second-to-last layer’s image features. The best results are highlighted among training-free and training-based methods.

Scaling to Qwen2.5-14B-instruct LLM. The original LLaVA 1.5 (Liu et al., 2024a) is restricted to the Vicuna 1.5 family of LLMs, which are instruction-tuned variants of LLaMA 2. Yet, Vicuna 1.5 is a relatively dated LLM series, potentially constraining LLaVA’s visual question answering capabilities and resulting in less-than-optimal performance across benchmarks. To address this, we scale our experiments using Qwen2.5-14B-instruct (Qwen et al., 2025) as the LLM (Table 3), which offers stronger capabilities and a larger model size. We first note that, although this choice substantially increases TFLOPs, the Qwen2.5-14B-instruct-based model yields large and consistent performance improvements over the original LLaVA-1.5-7B when using the same visual encoder. Surprisingly, DRIP does not consistently outperform fixed pooling at 4x compression, where the token budget remains sufficiently large for fixed pooling to preserve most visual information. Nevertheless, under more aggressive compression (both 8x and 10x), dynamic tokenization becomes increasingly important to consistently recover performance, particularly on OCR benchmarks.

Model	VQAv2	SQA	MME	MM-Bench	GQA	MMMU	TextVQA	OCRBench	OCRBenchv2	DocVQA	ChartQAPro	POPE	LLaVA-Wild	MM-Vet	TFL0Ps
Benchmark Type	general			reasoning			OCR			hallucination			free-response		
upper bound (keep 576/576 tokens out of the image encoder)															
LLaVA-Qwen2.5	77.63	76.00	1573.12	79.19	58.92	47.6	61.25	19.3	29.1	29.44	13.45	85.16	70.2	30.7	17.71
4x compression (keep 144/576 tokens out of image encoder)															
PruMerge(ICCV'25)	74.06	75.31	1490.82	77.93	54.60	46.4	56.95	14.9	24.6	17.19	12.56	81.79	66.5	26.0	5.62
fixed pooling	75.06	74.17	1510.43	78.16	55.75	48.3	59.06	17.0	26.2	21.48	12.91	83.96	66.4	27.2	5.60
DRIP	74.24	73.92	1460.11	76.54	54.86	48.6	57.89	15.9	25.0	22.21	12.28	83.75	68.8	26.6	5.63
8x compression (keep 72/576 tokens out of the image encoder)															
PruMerge(ICCV'25)	70.89	75.21	1388.75	77.02	52.59	46.9	56.23	13.9	23.1	15.29	12.29	78.73	65.5	23.9	3.61
fixed pooling	72.71	75.16	1422.09	77.15	54.01	47.6	55.86	14.8	24.9	17.14	12.90	83.59	64.5	24.5	3.58
DRIP	73.64	75.56	1459.39	77.93	54.98	47.9	58.42	15.9	26.8	20.45	12.61	83.43	65.0	27.6	3.61
10x compression (keep 58/576 tokens out of the image encoder)															
PruMerge(ICCV'25)	69.38	75.21	1352.54	76.42	51.53	46.9	56.29	13.8	22.9	13.77	12.19	77.34	64.6	24.5	3.18
fixed pooling	72.04	75.31	1427.13	76.47	54.27	47.6	55.92	13.5	24.5	15.96	13.12	83.95	65.6	23.1	3.16
DRIP	72.48	76.25	1478.91	76.54	53.99	47.9	57.88	15.7	24.8	18.37	12.58	83.46	67.9	24.9	3.22

Table 3: Evaluation results when using Qwen2.5-14B-instruct as the LLM. The best results are highlighted among training-free and training-based methods.

4 Ablation Studies

In this section, we analyze several important ablations related to training the boundary predictor, specifically focusing on sampling temperature, the architecture of the boundary predictor, and the loss ratio.

Sampling Temperature. During training, the relaxed Bernoulli sampling relies on a temperature hyperparameter τ to adjust the sharpness of the sampling process. In our main experiments, we adopt the default temperature of 0.1. In this ablation study, we examine a sharper sampling scheme by reducing the temperature to 0.01, as reported in Table 4. We observe that using this lower, sharper temperature leads to additional, moderate gains on OCR and reasoning benchmarks.

Model	VQAv2	SQA	MME	MM-Bench	GQA	MMMU	TextVQA	OCRBench	OCRBenchv2	DocVQA	ChartQAPro	POPE	LLaVA-Wild	MM-Vet	TFL0Ps
Benchmark Type	general				reasoning			OCR			hallucination	free-response			
4x compression (keep 144/576 tokens out of image encoder)															
DRIP	74.59	69.31	1341.93	74.05	55.16	34.7	56.08	16.8	20.9	21.27	11.38	82.83	59.5	28.6	3.18
Sampling Temperature															
temp = 0.01	74.60	68.67	1295.99	73.41	55.35	35.7	56.32	17.0	21.9	21.34	11.33	83.06	64.9	28.7	3.18
Boundary Predictor Architecture															
H-Net	73.60	69.11	1377.26	72.42	55.70	35.1	54.28	14.9	20.4	18.28	11.19	83.86	56.5	27.8	3.18
Loss Ratio															
loss ratio = 1:10	74.56	68.37	1327.69	73.86	55.31	33.7	56.23	16.8	21.0	21.00	11.31	83.48	60.2	28.3	3.18

Table 4: Key ablations for training the boundary predictor. We study the effects of sampling temperature, the architecture of the boundary predictor, and the loss weighting ratio. All experiments are conducted at 4x compression.

Boundary Predictor Architecture. In our proposed method, the boundary predictor is implemented as a lightweight 2-layer MLP. Other strategies for boundary prediction have also been explored. For instance, H-Net (Hwang et al., 2025) relies on token cosine similarity: the less similar two tokens are, the higher the probability that a boundary is placed between them. As illustrated in Figure 5, our MLP-based boundary predictor produces more confident decisions and yields boundaries that better align with semantic structure compared to H-Net. Furthermore, the results in Table 4 confirm that the H-Net strategy leads to overall inferior performance, particularly on OCR benchmarks.

Loss Ratio. By default, we set the language model loss and boundary loss to have a 1:1 ratio:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{primary}} + \mathcal{L}_{\text{boundary}}. \quad (5)$$

We then investigate whether altering this loss balance yields different outcomes during supervised instruction finetuning. Because the boundary loss is much smaller in magnitude,

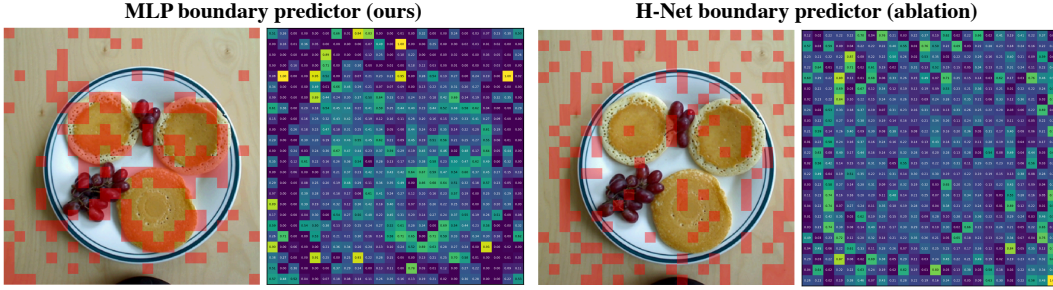


Figure 5: The predicted boundaries and probability heatmaps between our MLP boundary predictor and the ablated H-Net boundary predictor. Our MLP boundary predictor leads to more confident predictions and more semantically meaningful boundaries.

we also evaluate a 1:10 ratio to emphasize it. As shown in Table 4, adjusting the loss ratio does not lead to any substantial performance differences.

5 Related Work

Image Token Compression. The original motivation for compressing image tokens was to lower the computational overhead of vision transformers (Dosovitskiy et al., 2021) on vision-only tasks like ImageNet classification (Deng et al., 2009). For instance, DynamicViT (Rao et al., 2021) is trained to prune less informative tokens, EViT (Liang et al., 2022) identifies salient tokens using class-token attention and merges the rest, and ToME (Bolya et al., 2023) leverages token cosine similarity to aggregate redundant image tokens. With the emergence of open-source VLMs such as LLaVA (Liu et al., 2023), which show strong performance on multimodal benchmarks, image token compression has also been extended to VLMs: LLaVA-PruMerge (Shang et al., 2026) exploits class-token attention to select important image tokens, which serve as centroids around which other tokens are clustered and merged. More recently, SparseVLM (Zhang et al., 2025) and VisionTrim (Yu et al., 2026) have investigated leveraging the text prompt to guide the image compression process. Nevertheless, all of these approaches are *training-free*: they do not explicitly adapt the LLM to the compressed visual representation, suggesting that further gains remain possible.

Dynamic Token Pooling. Dynamic Token Pooling (Nawrot et al., 2023) is proposed for language modeling to make Transformers more efficient and adaptive. Unlike Hourglass (Nawrot et al., 2022), which merges tokens at fixed intervals, it dynamically decides where to merge tokens based on linguistic cues. A boundary predictor segments sequences into variable-length groups, which are pooled, processed at shorter sequence length, and then upsampled back to the original resolution for autoregressive prediction. By adapting computation to each region’s information content, Dynamic Token Pooling allows Transformers to model language more efficiently, achieving lower perplexity and faster training under the same compute budget.

6 Conclusion

In this work, we first question the trend of training-free visual token compression, demonstrating that finetuning VLMs with a straightforward fixed pooling baseline can already be competitive, although fixed pooling remains inadequate for OCR-heavy tasks. To close this OCR performance gap, we introduce dynamic image tokenization, which jointly trains an MLP-based boundary predictor. We finetune LLaVA under multiple configurations and assess performance on 14 VQA benchmarks, where our method achieves the best FLOP-accuracy tradeoff. Our goal is to establish a strong baseline for learning image tokenization during training, guiding future research on multimodal LLM token compression.

References

- Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Dhruv Batra, and Devi Parikh. Vqa: Visual question answering, 2016. URL <https://arxiv.org/abs/1505.00468>.
- Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints, 2023. URL <https://arxiv.org/abs/2305.13245>.
- Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning, 2022. URL <https://arxiv.org/abs/2204.14198>.
- Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond, 2023. URL <https://arxiv.org/abs/2308.12966>.
- Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your ViT but faster. In *International Conference on Learning Representations*, 2023.
- Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, Bin Li, Ping Luo, Tong Lu, Yu Qiao, and Jifeng Dai. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks, 2024. URL <https://arxiv.org/abs/2312.14238>.
- Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.
- Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL <https://arxiv.org/abs/2010.11929>.
- Facebook AI Research (FAIR). fvcore: A lightweight computer vision library. <https://github.com/facebookresearch/fvcore>, 2022–2025. Accessed: 2025-08-30.
- Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, Rongrong Ji, Caifeng Shan, and Ran He. Mme: A comprehensive evaluation benchmark for multimodal large language models, 2025a. URL <https://arxiv.org/abs/2306.13394>.
- Ling Fu, Zhebin Kuang, Jiajun Song, Mingxin Huang, Biao Yang, Yuzhe Li, Linghao Zhu, Qidi Luo, Xinyu Wang, Hao Lu, Zhang Li, Guozhi Tang, Bin Shan, Chunhui Lin, Qi Liu, Binghong Wu, Hao Feng, Hao Liu, Can Huang, Jingqun Tang, Wei Chen, Lianwen Jin, Yuliang Liu, and Xiang Bai. Ocrbench v2: An improved benchmark for evaluating large multimodal models on visual text localization and reasoning, 2025b. URL <https://arxiv.org/abs/2501.00321>.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021. URL <https://arxiv.org/abs/2009.03300>.
- Sukjun Hwang, Brandon Wang, and Albert Gu. Dynamic chunking for end-to-end hierarchical sequence modeling, 2025. URL <https://arxiv.org/abs/2507.07955>.

- Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models, 2023. URL <https://arxiv.org/abs/2305.10355>.
- Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations, 2022. URL <https://arxiv.org/abs/2202.07800>.
- Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023. URL <https://arxiv.org/abs/2304.08485>.
- Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning, 2024a. URL <https://arxiv.org/abs/2310.03744>.
- Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. Mmbench: Is your multi-modal model an all-around player?, 2024b. URL <https://arxiv.org/abs/2307.06281>.
- Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xu-Cheng Yin, Cheng-Lin Liu, Lianwen Jin, and Xiang Bai. Ocrbench: on the hidden mystery of ocr in large multimodal models. *Science China Information Sciences*, 67(12), December 2024c. ISSN 1869-1919. doi: 10.1007/s11432-024-4235-6. URL <http://dx.doi.org/10.1007/s11432-024-4235-6>.
- Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering, 2022. URL <https://arxiv.org/abs/2209.09513>.
- Ahmed Masry, Mohammed Saidul Islam, Mahir Ahmed, Aayush Bajaj, Firoz Kabir, Aaryaman Kartha, Md Tahmid Rahman Laskar, Mizanur Rahman, Shadikur Rahman, Mehrad Shahmohammadi, Megh Thakkar, Md Rizwan Parvez, Enamul Hoque, and Shafiq Joty. Chartqapro: A more diverse and challenging benchmark for chart question answering, 2025. URL <https://arxiv.org/abs/2504.05506>.
- Minesh Mathew, Dimosthenis Karatzas, and C. V. Jawahar. Docvqa: A dataset for vqa on document images, 2021. URL <https://arxiv.org/abs/2007.00398>.
- Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947. doi: 10.1007/BF02295996.
- Piotr Nawrot, Szymon Tworkowski, Michał Tyrolski, Łukasz Kaiser, Yuhuai Wu, Christian Szegedy, and Henryk Michalewski. Hierarchical transformers are more efficient language models, 2022. URL <https://arxiv.org/abs/2110.13711>.
- Piotr Nawrot, Jan Chorowski, Adrian Lancucki, and Edoardo Maria Ponti. Efficient transformers with dynamic token pooling. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6403–6417. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.acl-long.353. URL <http://dx.doi.org/10.18653/v1/2023.acl-long.353>.
- 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, Jiayi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL <https://arxiv.org/abs/2412.15115>.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL <https://arxiv.org/abs/2103.00020>.

- 376 Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dy-
 377 namicvit: Efficient vision transformers with dynamic token sparsification, 2021. URL <https://arxiv.org/abs/2106.02034>.
 378
- 379 Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. Llava-prumerge: Adaptive
 380 token reduction for efficient large multimodal models, 2026. URL <https://arxiv.org/abs/2403.15388>.
 381
- 382 Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi
 383 Parikh, and Marcus Rohrbach. Towards vqa models that can read, 2019. URL <https://arxiv.org/abs/1904.08920>.
 384
- 385 Hanxun Yu, Wentong Li, Xuan Qu, Song Wang, Junbo Chen, and Jianke Zhu. Visiontrim:
 386 Unified vision token compression for training-free mllm acceleration, 2026. URL <https://arxiv.org/abs/2601.22674>.
 387
- 388 Weihao Yu, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao
 389 Wang, and Lijuan Wang. Mm-vet: Evaluating large multimodal models for integrated
 390 capabilities, 2024. URL <https://arxiv.org/abs/2308.02490>.
- 391 Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng,
 392 Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, and Shanghang Zhang.
 393 Sparsevlm: Visual token sparsification for efficient vision-language model inference, 2025.
 394 URL <https://arxiv.org/abs/2410.04417>.
- 395 Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao
 396 Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez,
 397 and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023.

Appendix

A Implementation Details

Hyperparameters. As noted above, our finetuning setup largely mirrors the training recipe of LLaVA 1.5. In the initial token-alignment/pretraining phase, where we train only the MLP projectors (plus the boundary predictor in our method), we adopt DeepSpeed ZeRO-2 with an initial learning rate of $1e-3$ and a cosine schedule. For the Qwen2.5-14B DRIP model under $4\times$ compression, we reduce this learning rate to $2e-4$ to maintain stable training. The global batch size is 256, and we apply gradient accumulation over 32 steps to reduce memory usage. This pretraining phase is run for exactly 1 epoch. In the subsequent supervised instruction finetuning phase, where the LLM itself is updated (using either full-parameter finetuning or LoRA), we switch to DeepSpeed ZeRO-3 with an initial learning rate of $2e-5$ and a cosine scheduler. Here, we use 1 batch per device and accumulate gradients over 128 steps, resulting in a global batch size of 128. This finetuning phase also runs for exactly 1 epoch. In both stages, we set the learning rate warmup ratio to 0.03.

Computing Environments. We run inference (i.e., evaluation) on a single NVIDIA A100-40G GPU. The token pretraining is likewise carried out on one A100-40G GPU. For full-parameter finetuning, we use 4 A100-80G GPUs, whereas LoRA finetuning is conducted on a single A100-80G GPU.

B Boundary Prediction Algorithm Pseudocode

Algorithm 1 Pseudocode for Boundary Prediction – training

```
# B: batch size; L: sequence length D: feature dimension
# temperature: sampling temperature

# features in shape [L x B x D]
boundary_logits = self.boundary_predictor(hidden) # in shape [B, L]
boundary_probs = sigmoid(boundary_logits) # in shape [B, L]

# sample boundaries from a relaxed bernoulli distribution
bernoulli = RelaxedBernoulli(boundary_probs, temperature)
soft_boundaries = bernoulli.sample() # in shape [B, L]

# threshold to assign hard boundaries (i.e, 0 or 1)
hard_boundaries = soft_boundaries > 0.5 # in shape [B, L]
return soft_boundaries, hard_boundaries
```

Algorithm 2 Pseudocode for Boundary Prediction – inference

```
# B: batch size; L: sequence length D: feature dimension
# prior: the compression ratio in float

# features in shape [L x B x D]
boundary_logits = self.boundary_predictor(hidden) # in shape [B, L]
boundary_probs = sigmoid(boundary_logits) # in shape [B, L]

# use boundary probabilities to select Top-K image tokens
# to match the compression ratio
K = integer(L * prior)

# binary mask with exactly K selected tokens per sample
hard_boundaries = TopK(boundary_probs, K)
return soft_boundaries, hard_boundaries
```

In this section, we provide pseudocode for boundary prediction during both training and inference. Algorithm 1 illustrates how we sample from the relaxed Bernoulli distribution defined in Equation 2.2 to obtain the boundaries, while Algorithm 2 describes how we choose the Top-K image tokens as boundaries to achieve the desired compression ratio. We additionally include the pseudocode for computing the auxiliary boundary loss in Algorithm 3.

Algorithm 3 Pseudocode for **Boundary Loss Calculation** during training

```

# B: batch size; L: sequence length
# prior: the compression ratio in float

# preds in shape [B x L]
# Count the number of selected boundary tokens in each sample
target_count = preds.sum(dim=-1)

# Define a Binomial prior over the expected number of selections
binomial = Binomial(total_count=L, probs=prior)

# Compute the boundary loss
return -binomial.log_prob(target_count).mean() / total_count

```

C LoRA finetuning results

We report our LoRA finetuning outcomes in Table 5 and the FLOP–accuracy tradeoff in Figure 6. As before, training with fixed pooling yields better performance than PruMerge, and dynamic tokenization outperforms on OCR benchmarks while preserving its effectiveness on the remaining tasks. The results for image features from second to last layer is also shown in Table 6.

Model	VQAv2	SQA	MME	MM-Bench	GQA	MMMU	TextVQA	OCRBench	OCRBenchv2	DocVQA	ChartQAPro	POPE	LLaVA-Wild	MM-Vet	TFLOPs
Benchmark Type	general			reasoning			OCR			hallucination	free-response				
upper bound (keep 576/576 tokens out of the image encoder)															
LLaVA-1.5-7B	75.94	66.19	1450.84	71.60	58.06	33.3	55.87	18.1	23.0	25.41	11.17	85.10	61.1	32.2	8.89
4x compression (keep 144/576 tokens out of image encoder)															
PruMerge(ICCV'25)	73.11	65.99	1410.31	69.98	53.97	34.1	52.18	14.2	20.3	16.17	10.57	81.82	64.6	28.6	3.18
fixed pooling	73.51	66.78	1355.09	71.42	55.76	34.4	53.18	15.8	21.5	18.45	9.42	84.23	61.4	30.8	3.16
DRIP	74.11	66.78	1346.59	71.90	55.81	33.4	54.47	17.2	22.0	21.77	9.82	82.96	60.4	31.0	3.18
8x compression (keep 72/576 tokens out of the image encoder)															
PruMerge(ICCV'25)	69.80	66.19	1291.15	68.56	51.51	33.2	51.37	14.4	18.5	14.69	10.81	78.05	58.5	28.2	2.22
fixed pooling	71.53	68.32	1277.98	70.00	53.53	32.9	50.95	13.4	20.2	15.85	9.28	82.65	62.4	29.1	2.21
DRIP	72.03	67.53	1280.09	70.60	53.70	32.0	52.74	14.9	21.0	18.48	9.63	82.96	60.3	31.2	2.23
10x compression (keep 58/576 tokens out of the image encoder)															
PruMerge(ICCV'25)	68.27	66.14	1257.86	67.81	50.46	32.9	51.16	13.8	18.2	14.07	10.54	76.81	55.0	26.5	2.02
fixed pooling	70.84	67.87	1249.51	70.71	53.03	32.3	50.99	13.3	19.5	14.92	8.83	82.78	58.5	27.4	2.01
DRIP	71.15	66.93	1290.33	69.66	52.95	32.9	52.93	15.1	21.1	16.90	9.38	83.14	57.8	26.9	2.04

Table 5: LLaVA-1.5-7B LoRA finetuning results using the last layer’s image features. The best results are highlighted among training-free methods and training-based methods.

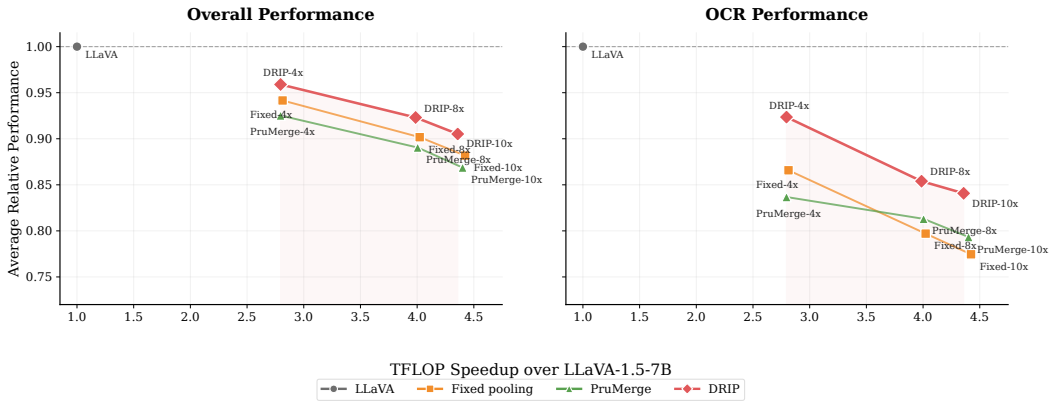


Figure 6: FLOP-Accuracy tradeoff (lora finetuning results using the last layer’s image features).

Model	VQAv2	SQA	MME	MM-Bench	GQA	MMMU	TextVQA	OCRBench	OCRBenchv2	DocVQA	ChartQAPro	POPE	LLaVA-Wild	MM-Vet	TFL0Ps
Benchmark Type	general			reasoning			OCR			hallucination		free-response			
upper bound (keep 576/576 tokens out of the image encoder)															
LLaVA-1.5-7B	76.33	67.48	1437.89	72.58	58.01	32.0	55.96	19.0	23.1	24.45	10.75	85.17	70.5	30.5	8.89
4x compression (keep 144/576 tokens out of image encoder)															
PruMerge(ICCV'25)	73.79	67.18	1410.93	71.37	54.43	31.6	52.77	16.0	20.4	17.16	10.21	82.36	61.4	29.8	3.18
fixed pooling	74.10	67.23	1428.06	72.08	56.08	32.8	54.30	15.8	21.4	18.90	10.38	83.90	70.2	30.2	3.16
DRIP	74.57	67.58	1400.51	72.20	56.08	33.9	54.22	17.3	23.0	21.31	8.97	83.54	67.6	31.7	3.18
8x compression (keep 72/576 tokens out of the image encoder)															
PruMerge(ICCV'25)	70.91	66.83	1361.00	69.77	51.88	32.2	51.87	15.2	19.3	15.36	10.13	78.75	62.2	29.1	2.22
fixed pooling	72.40	68.67	1252.21	71.10	54.08	30.9	51.93	13.8	20.4	16.00	9.37	83.55	66.5	28.8	2.21
DRIP	72.49	67.97	1347.32	71.10	54.01	31.8	53.92	15.9	21.6	18.37	9.30	83.08	62.7	31.0	2.23
10x compression (keep 58/576 tokens out of the image encoder)															
PruMerge(ICCV'25)	69.54	67.33	1333.20	69.50	50.99	32.8	52.00	14.3	18.7	14.95	9.76	77.64	58.9	27.9	2.02
fixed pooling	71.85	67.87	1281.67	71.30	53.78	32.7	51.36	13.7	20.5	15.44	8.86	83.06	66.3	27.9	2.01
DRIP	71.86	69.21	1311.59	70.30	53.41	32.7	52.87	15.9	20.9	16.54	8.89	83.09	66.2	27.2	2.04

Table 6: LLaVA-1.5-7B LoRA finetuning results using the second to last layer’s image features. The best results are highlighted among training-free methods and training-based methods.

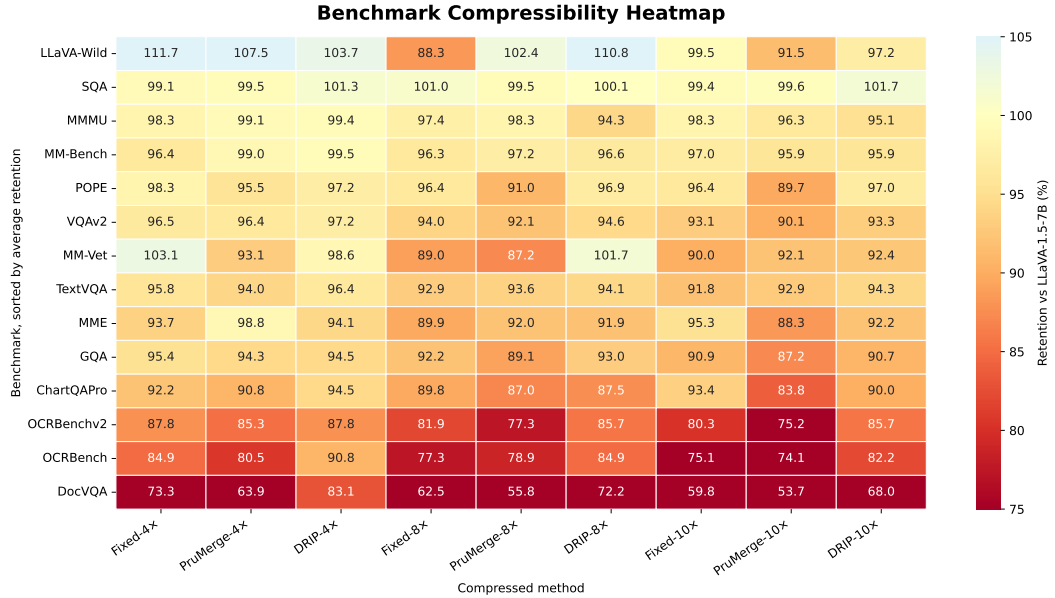


Figure 7: **Benchmark Compressibility Heatmap**. Each cell’s value represents the percentage of performance preserved relative to the original, uncompressed LLaVA. The x-axis lists all compression methods, while the y-axis presents VQA benchmarks ordered by their level of performance retention.

D Additional Result Analysis

Benchmark Compressibility Heatmap. We further study the *compressibility* of each VQA benchmark by examining their relative performance under token compression. Figure 7 presents all 14 VQA benchmarks ranked by performance retention. We observe that benchmarks such as LLaVA-in-the-wild, SQA, MMMU, and MM-Bench are highly compressible, suggesting that they primarily depend on a global understanding of the visual input. In contrast, most OCR benchmarks, which need fine-grained recognition, are highly sensitive to token compression. Nonetheless, our dynamic tokenization approach is able to recover more accuracy on these OCR tasks.

McNemar’s Test on benchmarks when appropriate. We further apply McNemar’s test statistic (McNemar, 1947) to assess whether the observed performance improvements are

statistically significant. McNemar’s test evaluates the null hypothesis that two models achieve the same performance. Specifically, let n_{01} be the number of instances answered correctly by model A but not by model B, and n_{10} be the number of instances answered correctly by model B but not by model A. The McNemar’s test statistic is given by

$$\chi^2 = \frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}}, \quad (6)$$

which, under the null hypothesis that the two models have identical error rates, asymptotically follows a chi-square distribution with one degree of freedom. We examine the combination of fixed pooling and dynamic tokenization and observe that, at a 10x compression rate, dynamic tokenization delivers statistically significant gains on several OCR benchmarks, including TextVQA, OCRBenchv2, and DocVQA. Based on these results, we conclude that dynamic tokenization is strongly preferable when high compression is required.

Model	VQAv2	SQA	MME	MM-Bench	GQA	MMMU	TextVQA	OCRBench	OCRBenchv2	DocVQA	ChartQAPro	POPE	LLaVA-Wild	MM-Vet	TFLOPs
Benchmark Type	general				reasoning		OCR				hallucination		free-response		
4x compression (keep 144/576 tokens out of image encoder)															
fixed pooling	74.06	67.82	1335.40	71.72	55.67	34.3	55.74	15.7	20.9	18.78	11.10	83.75	64.1	29.9	3.16
DRIP	74.59	69.31	1341.93	74.05	55.16	34.7	56.08	16.8	20.9	21.27	11.38	82.83	59.5	28.6	3.18
McNemar's test	-	4.163	4.663	26.156	3.025	0.032	0.0733	2.127	0.237	30.303	0.179	9.592	-	-	-
p-value	-	0.041	0.031	0.000	0.082	0.858	0.786	0.145	0.626	0.000	0.672	0.002	-	-	-
8x compression (keep 72/576 tokens out of the image encoder)															
fixed pooling	72.13	69.11	1280.78	71.65	53.78	34.0	54.05	14.3	19.5	16.00	10.81	82.15	50.7	25.8	2.21
DRIP	72.58	68.47	1310.01	71.90	54.23	32.9	54.75	15.7	20.4	18.50	10.54	82.52	63.6	29.5	2.23
McNemar's test	-	0.64	2.952	0.224	2.297	0.75	1.481	3.25	0.575	32.397	0.455	1.266	-	-	-
p-value	-	0.424	0.086	0.635	0.129	0.386	0.223	0.071	0.448	0.000	0.499	0.261	-	-	-
10x compression (keep 58/576 tokens out of the image encoder)															
fixed pooling	71.49	68.02	1359.05	72.17	53.02	34.3	53.40	13.9	19.1	15.32	11.25	82.14	57.1	26.1	2.01
DRIP	71.59	69.61	1313.76	71.35	52.93	33.2	54.83	15.2	20.4	17.42	10.84	82.59	55.8	26.8	2.04
McNemar's test	-	4.620	0.565	2.849	0.092	0.75	9.689	2.938	16.644	23.971	0.613	1.868	-	-	-
p-value	-	0.031	0.452	0.091	0.761	0.386	0.002	0.086	0.000	0.000	0.434	0.171	-	-	-

Table 7: McNemar’s Statistical test upon the Table 1. Green cells indicate $p_{value} < 0.05$ and yellow cells imply $p_{value} < 0.1$. VQA2 requires external evaluation, and free-response benchmarks like LLaVA-wild and MM-vet use LLM-as-a-judge, not suitable for this test.

E Efficiency Details

Comprehensive Efficiency report.

Other than TFLOPs during prefill stage, we also collect and report memory metrics such as KV cache and allocated memory across different compression budgets, as shown in Table 8. We find that when the number of image tokens is reduced, not only the TFLOP drops, but also the KV cache and allocated memory are less consumed due to the much reduced sequence length. This indicates that image token compression can mitigate both computation and memory consumption.

Method	Tokens	TFLOPs	KV Cache	Memory
LLaVA	576	8.89	321.0 MB	14.95 GB
DRIP-4x	144	3.18	105.5 MB	14.15 GB
DRIP-8x	72	2.23	69.5 MB	14.02 GB
DRIP-10x	58	2.04	62.5 MB	14.00 GB

Table 8: Comprehensive efficiency report.

Boundary Predictor Overhead. From the results in Table 1, we find that adding our boundary predictor leads to only a negligible increase in TFLOPs compared with using fixed pooling. Therefore, in Table 9, we additionally report the number of parameters to assess the overhead introduced by the proposed boundary predictor. We base our analysis on the original LLaVA 1.5 model, which uses Vicuna-1.5-7B as the LLM and openai/clip-vit-large-patch14-336 as the vision encoder. Our lightweight MLP-based boundary predictor adds only 0.06% to the total number of parameters.

Module	#Params	%
Vicuna-1.5-7B	6.61B	95.26%
ViT-L/14	303.51M	4.37%
MLP projector	20.98M	0.30%
Boundary predictor	4.20M	0.06%

Table 9: Boundary Predictor Overhead.

473 F Training Curves

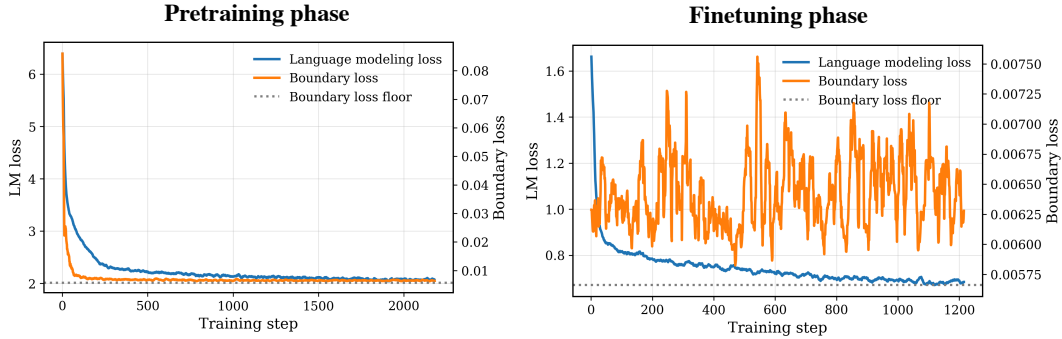


Figure 8: The loss curve for both LM loss and boundary loss in the pretraining phase (left) and finetuning phase (right).

474 We additionally present the training loss curves for both the token alignment/pretrain-
 475 ing phase and the instruction finetuning phase, as shown in Figure 8. To derive the
 476 theoretical lower bound of the boundary loss, we consider a binomial random variable
 477 $K \sim \text{Binomial}(L, \phi)$, where L is the sequence length and ϕ is the compression ratio. Our
 478 boundary loss in Equation 3 can then be written as

$$\mathcal{L}_{\text{boundary}} = -\frac{1}{L} \log P(K = k). \quad (7)$$

479 This loss is minimized when $k = L \cdot \phi$. Hence, the theoretical minimum of the boundary
 480 loss, which is nonzero, is given by

$$\mathcal{L}_{\min} = -\frac{1}{L} \log P(K = L \cdot \phi). \quad (8)$$

481 In Figure 8, we assume a 4x compression setting and also plot this theoretical bound for
 482 the boundary loss. Concretely, at the beginning of pretraining, both the LM loss and the
 483 boundary loss are very large because the boundary predictor is randomly initialized and
 484 does not yet enforce the target compression ratio. Both losses then drop rapidly, with
 485 the boundary loss converging particularly fast. During instruction finetuning, the LM
 486 loss starts relatively high (above 1.6) and decreases gradually over training steps, while
 487 the boundary loss, which is already very small after pretraining, fluctuates around the
 488 theoretical minimum. It does not exactly attain this minimum because boundary positions
 489 are continually sampled during training.

490 G Additional Boundary Validation

491 **Qualitative validation.** To further assess the effectiveness of dynamic tokenization, we
 492 compare our predicted boundaries against the attention maps of the CLS token. The attention
 493 received by the CLS token, commonly used as a simple proxy for token importance (Liang
 494 et al., 2022), indicates which image tokens encode the most salient semantics of an image.
 495 Using the CLS token attention averaged across all heads, Figure 9 shows that our selected
 496 boundaries largely encompass the image tokens that obtain the highest CLS-token attention.

497 **More case studies.** We provide additional boundary visualizations in Figure 10, Figure 11,
 498 and Figure 12 for various compression ratios. These examples span multiple text types,
 499 such as English characters, numerical digits, symbols, and even vertically oriented text.
 500 In general, dynamic image tokenization retains a greater number of image tokens around
 501 textual regions, resulting in improved performance on OCR tasks.

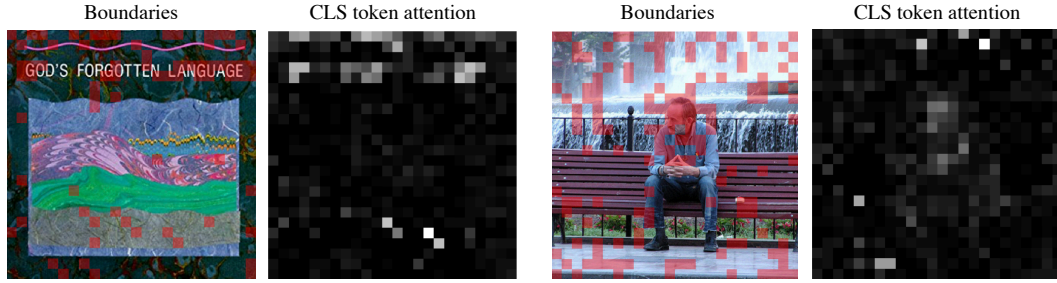


Figure 9: **Qualitative validation against CLS token attention maps.** Our selected boundaries largely encompass the image tokens that obtain the highest CLS-token attention.

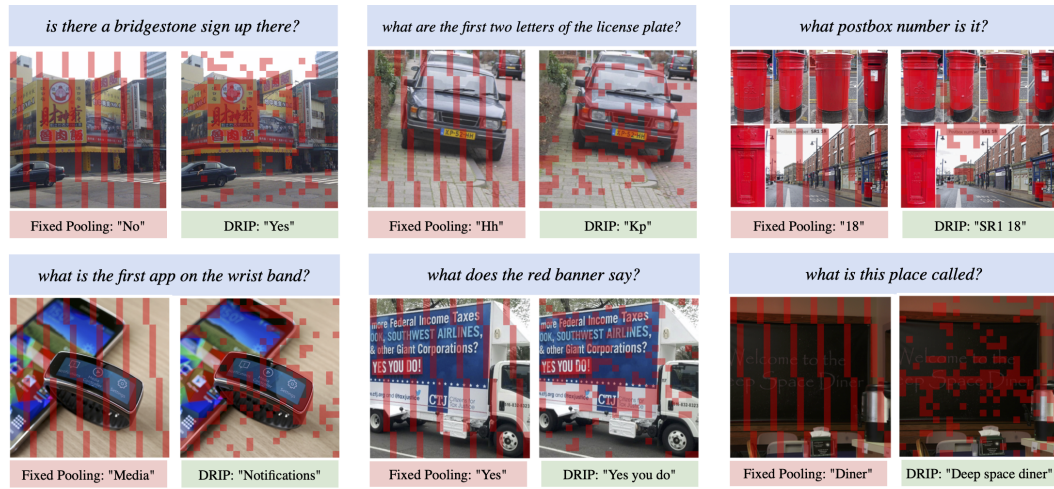


Figure 10: TextVQA examples: 4x compression.

502 H VQA Benchmarks

503 In this section, we introduce each of 14 VQA benchmarks we use in our experiments.

504 **VQAv2.** VQAv2 (Agrawal et al., 2016) is a large scale VQA benchmark that consist more
 505 than 265K open-ended visual questions. All questions are paired with 10 ground truth
 506 answer candidates to accurately evaluate the model’s predictions. Given the fact the ground
 507 truths are not publicly available, all evaluation results will need to be submitted to EvalAI
 508 for scoring.

509 **SQA.** SQA, also known as ScienceQA (Lu et al., 2022), is a VQA benchmark especially
 510 for science domains. Questions are organized hierarchically, covering more than 25 topics.
 511 Note that this benchmark contains multimodal questions and text-only questions: to rigorously
 512 examine visual understanding capabilities of models, we only report the aggregated
 513 accuracy results of multimodal questions.

514 **MME.** The MME benchmark (Fu et al., 2025a) evaluates model performance across a total
 515 of 14 tasks targeting both perceptual and cognitive abilities. We only report the score of the
 516 *perceptual split* since cognitive abilities have little to do with visual compression. MME tries
 517 to prevent data leakage for fair evaluation by manually designed instruction-answer pairs.



Figure 11: TextVQA examples: 8x compression.



Figure 12: TextVQA examples: 10x compression.

518 **MMBench.** MMBench (Liu et al., 2024b), a bilingual (English & Chinese) VQA multiple-
 519 choice benchmark, is rigorously curated with well-designed quality control. It also intro-
 520 duces a rigorous CircularEval strategy to ensure the model is robust to the order of choices
 521 (i.e., A, B, C, D). We report our results with a single round.

522 **GQA.** The GQA (Ainslie et al., 2023) benchmark evaluates visual scene understanding
 523 and reasoning capabilities from both images and scene graphs. It also includes spatial
 524 attributes and object features paired with well-designed questions. However, we follow
 525 LLaVA-1.5 (Liu et al., 2024a) pipeline to exclude object-level questions during our evaluation.

526 **MMMU.** MMMU (Hendrycks et al., 2021) evaluates multimodal models on hard tasks
 527 that require a wide range of knowledge and strong reasoning abilities. The benchmark
 528 consists of more than 10K questions spanning six different disciplines, sourced
 529 from exams and textbooks. We only report the overall aggregated accuracy for the simplicity
 530 of comparison.

531 **TextVQA.** TextVQA (Singh et al., 2019) evaluates a MLLM’s ability to capture text embed-
532 ded within images. It’s the most widely used and popular OCR benchmark, which provides
533 a list of candidates for ground truth answers to mitigate the effect of formatting issues in
534 model’s responses.

535 **OCRBench.** OCRBench (Liu et al., 2024c) is a comprehensive OCR evaluation benchmark.
536 It contains almost 30 datasets spanning different forms of embedded text: multilingual text,
537 handwritten text, non-semantic text, and mathematical expression recognition.

538 **OCRBenchv2.** As an extension of OCRBench, OCRBenchv2 (Fu et al., 2025b) presents a
539 larger-scale bilingual text-centric benchmark and provides 4x more tasks than OCRBench.
540 Concretely, it covers more than 30 diverse scenarios with 10K human-verified question-
541 answering pairs.

542 **DocVQA.** DocVQA (Mathew et al., 2021) specifically offers a testbed for document images,
543 consisting of 50K questions within 12K document images. We use $Accuracy@NL < 0.5$,
544 which quantifies the percentage of prediction examples whose normalized edit distance is
545 less than 0.5, as our evaluation metric reported in this paper.

546 **ChartQAPro.** ChartQAPro (Masry et al., 2025) is a recent benchmark that consists of over
547 1K chart images such as dashboards and infographics. It’s characterized by almost 2K
548 questions spanning multiple-choice, conversational, hypothetical, and unanswerable ones.

549 **POPE.** POPE (Li et al., 2023) evaluates object hallucination in models via curated binary
550 questions (i.e., ‘yes’ or ‘no’) that asks whether something is present in the given image. Three
551 splits of hallucinated objects are provided: random (a randomly selected object), common (a
552 very common object in daily life), and adversarial (a very similar but still different object
553 in the image). Metrics like Accuracy, Recall, Precision, and F1 Score measure hallucination
554 levels across three sampling strategies, and we report the arithmetic average of accuracy
555 across 3 different splits above.

556 **LLaVA-in-the-wild.** LLaVA-Bench (Liu et al., 2023), also known as LLaVA-in-the-wild,
557 collects a small but diverse set of 24 images along with 60 questions. All questions expect
558 open-ended free-response answers, thus LLM-as-a-judge is applied to evaluate model’s
559 performance.

560 **MM-Vet.** MM-Vet (Yu et al., 2024) is a similar free-response benchmark to LLaVA-Bench,
561 but it offers a more structured family of visual questions spanning recognition, OCR,
562 knowledge, language generation, spatial awareness, and math. LLM-as-a-judge is used to
563 evaluate answers from models.