MMTok

Multimodal Coverage Maximization for Efficient Inference of VLMs

ICLR 2026

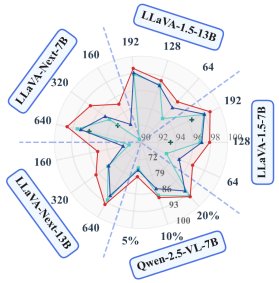
Sixun Dong[1 †], Juhua Hu[2], Mian Zhang[3 †], Ming Yin[4 †], Qi Qian[5]

[1]Arizona State University [2]University of Washington [3]University of Texas at Dallas

[4]Duke University [5]Zoom Communications

† Work done during internship at Zoom Communications ✉ Corresponding author

- ★ A Training-free Framework to Speed Up 1.87 x
- ★ Maintain 95%+ performance of original VLMs
- ★ 87.7% F1 with 4 tokens on POPE

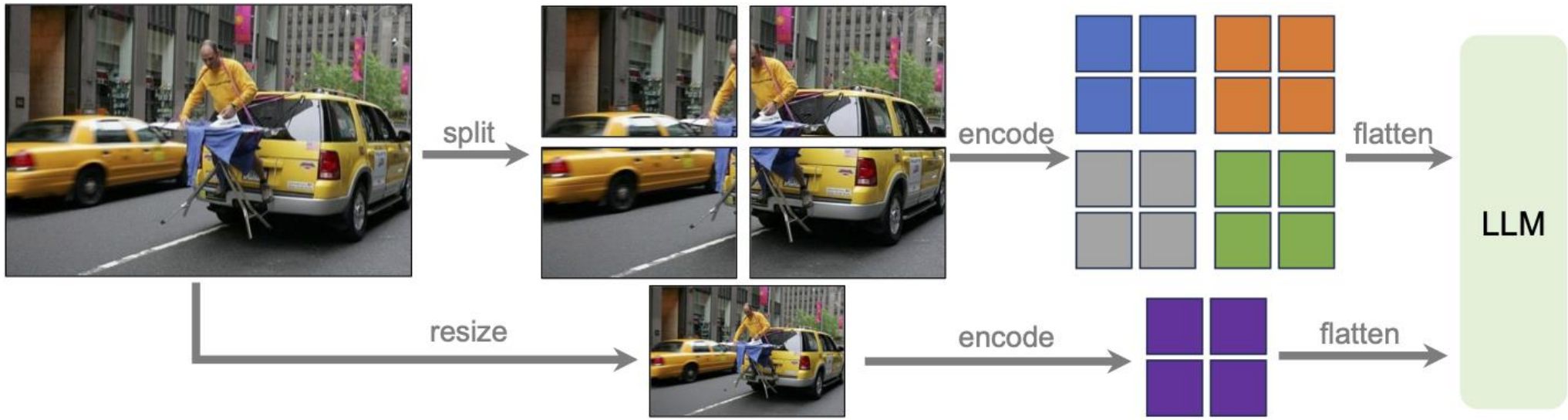


(1) Motivation

- Efficient VLMs Inference

Task: selecting the most **informative 5%** Vision Tokens => Speed up x3

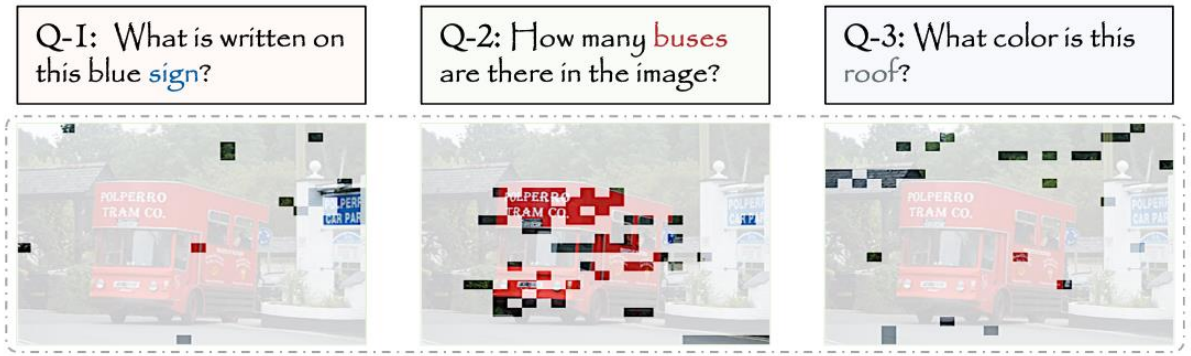
Query: What is in this image? (**7** Tokens) Vision tokens: **2880!**



Llava-Next

(2) Previous Work

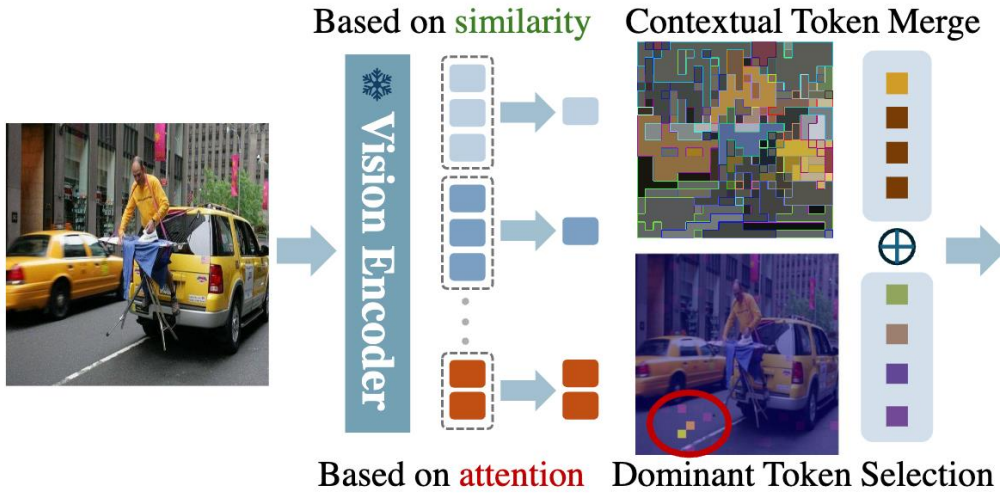
- Based on Language



(b) **Ours.** Text-guided Visual Sparsification

SparseVLM (ICML 2025)

- Base on Visual



VisionZIP (CVPR 2025)

(3) Proposed Method: MMTok

- Preliminary: Max-K-Coverage
  - Why Top-K Fails: The Redundancy Problem

Top-K Ranking

Max-K-Coverage

(3) Proposed Method: MMTok

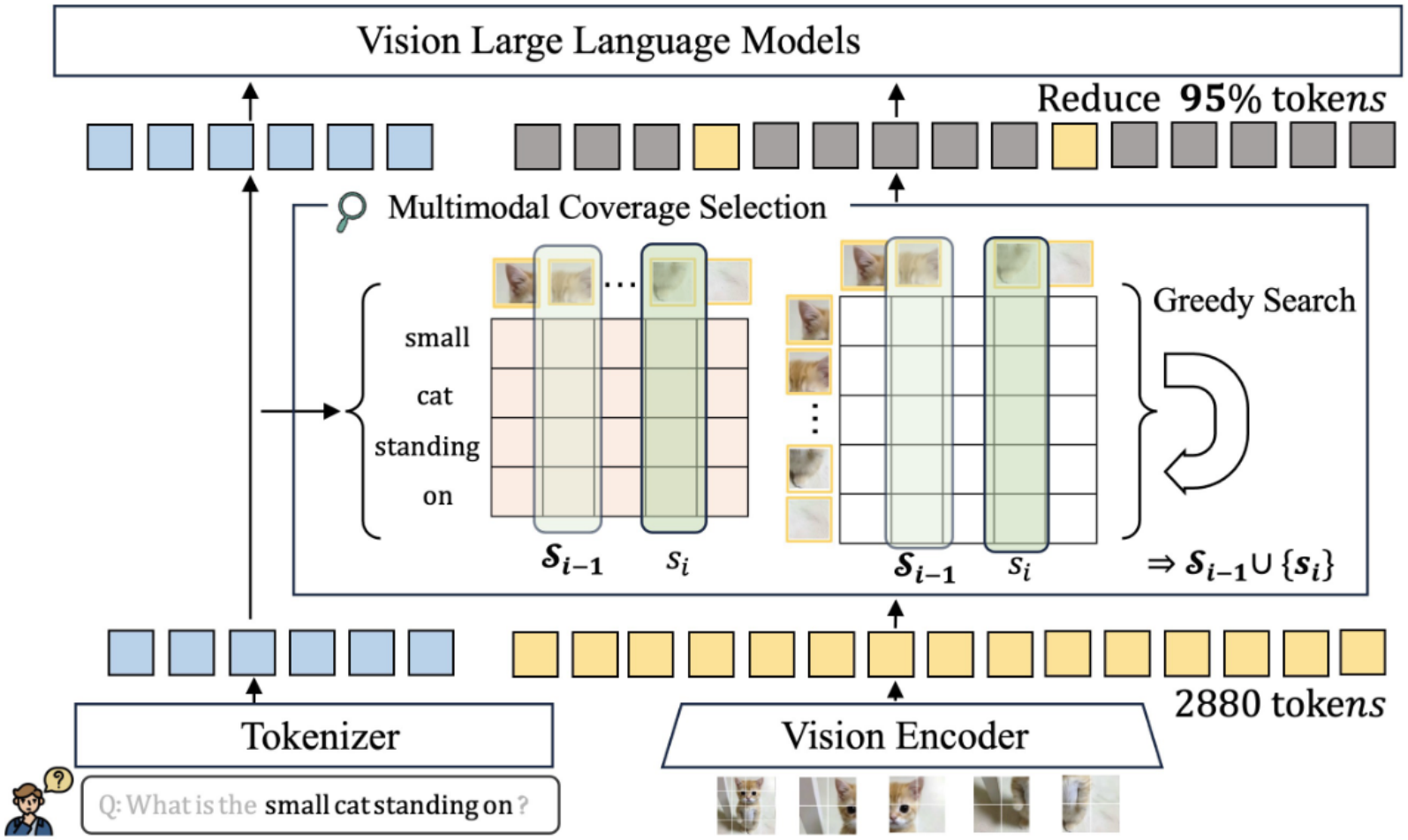
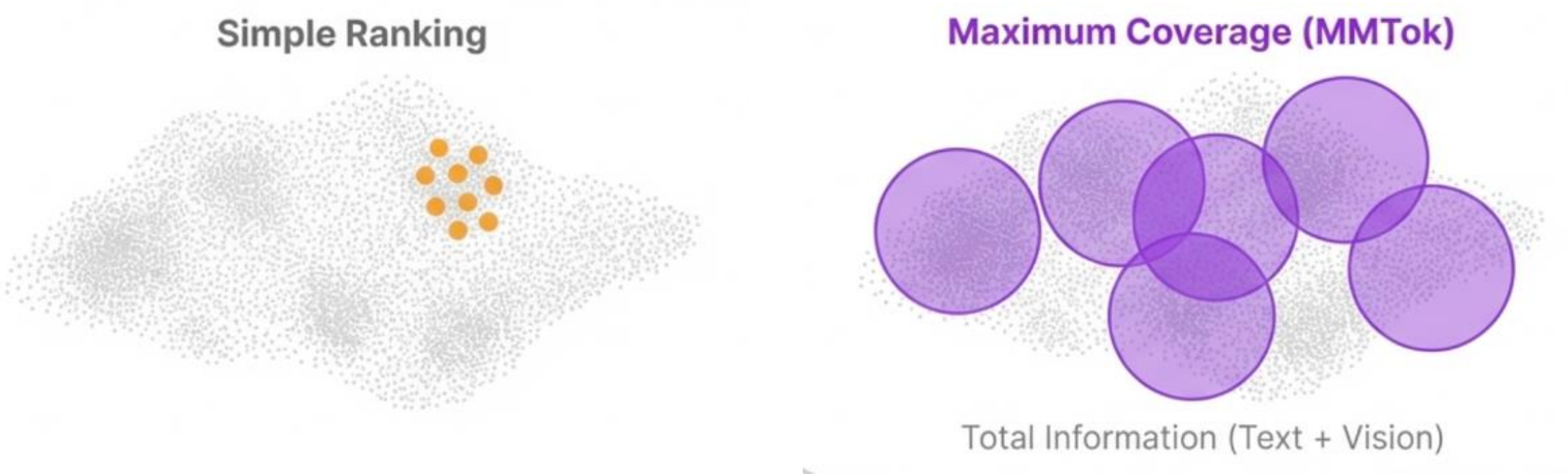


Figure 2: **Overview of MMTok framework.** Our method optimizes two maximum coverage problems simultaneously to leverage text-vision and vision-vision similarity for vision token selections.

(3) Proposed Method: MMTok



(3) Proposed Method: MMTok

Q: Is there a **traffic light** in the image?

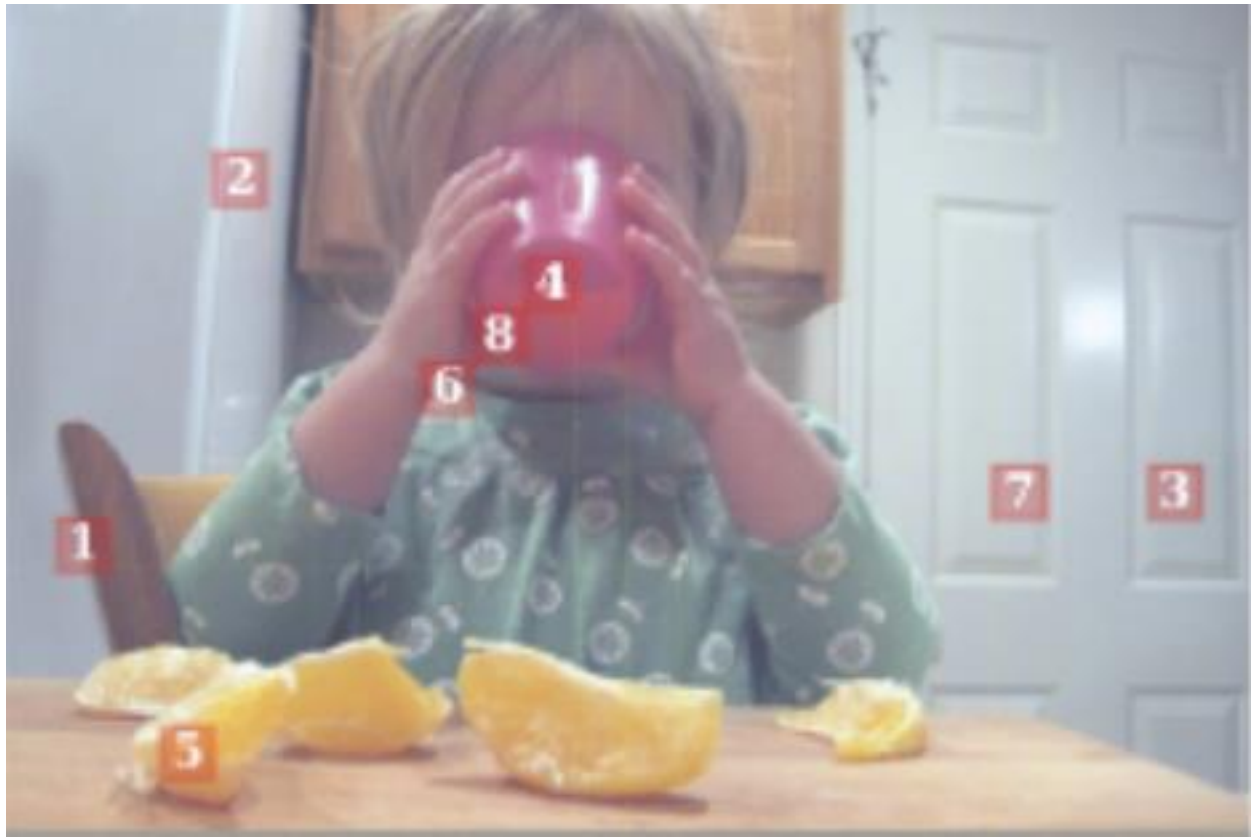
(3) Proposed Method: MMTok

- NP-Hard Selection

$$\mathcal{S}^* = \arg\max_{\mathcal{S}} f(\mathcal{S}; M)$$

- Greedy Selection
  - Marginal Gain Function

$$f(\mathcal{S}; M) = \frac{1}{m} \sum_{i=1}^{m} \max M_{i,\mathcal{S}}$$



❖Submodular Function Maximization

$(\mathbf{1} - \mathbf{1}/\boldsymbol{e})$ Approximation Guarantee

(3) Proposed Method: MMTok

$$\mathcal{S}^* = \arg\max_{\mathcal{S}} f(\mathcal{S}; M)$$

$$f(\mathcal{S}; M) = \frac{1}{m} \sum_{i=1}^{m} \max M_{i,\mathcal{S}}$$

Algorithm 1 A Greedy Algorithm to Cover Text Input with Vision Tokens

Input: Similarity Matrix M^{tv}, k
Initialize $\mathcal{S} = \emptyset$
for $i = 1, \cdots, k$ **do**
  for $s \in \mathcal{N} \setminus \mathcal{S}$ **do** Marginal Gain
    Compute $g(s) = f(\mathcal{S} \cup s; M^{tv})$
  end for
  Obtain $s_i = \arg\max_s g(s)$
  $\mathcal{S} = \mathcal{S} \cup s_i$
end for
return $\mathcal{S}$

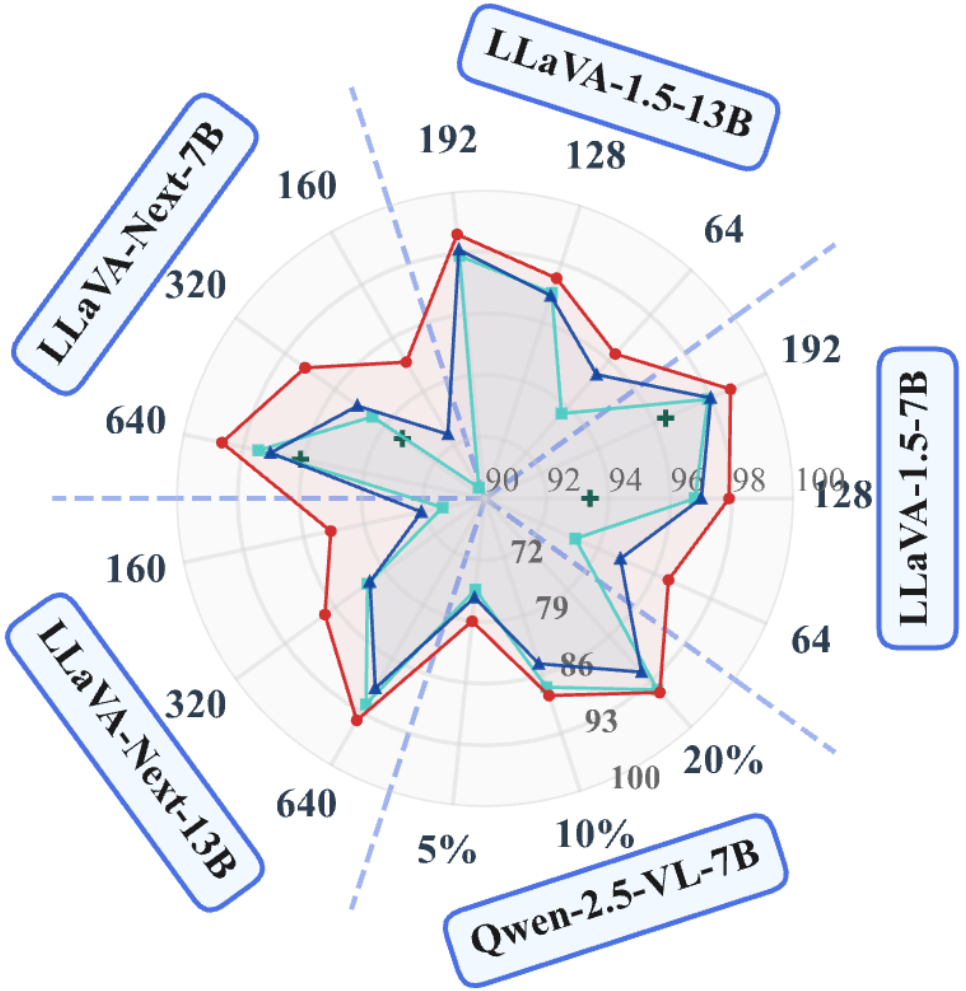
Algorithm 2 MMToK: A Greedy Algorithm for Multimodal Coverage

Input: Similarity Matrices $M^{tv'}$, $M^{vv'}$, k
Initialize $\mathcal{S} = \emptyset$
for $i = 1, \cdots, k$ **do**
  for $s \in \mathcal{N} \setminus \mathcal{S}$ **do** Marginal Gain
    Compute $g(s) = f(\mathcal{S} \cup s; M^{tv'}, M^{vv'})$
  end for
  Obtain $s_i = \arg\max_s g(s)$
  $\mathcal{S} = \mathcal{S} \cup s_i$
end for
return $\mathcal{S}$

The proposed Alg. 1 contains only simple operations (e.g., addition, matrix multiplication, etc.) and is efficient for implementation.

(4) Experiments

✓ Superior performance on **9** datasets with **5** different VLM architectures.



- Nine Benchmarks
  - ✓ GQA
  - ✓ MMBench
  - ✓ MME
  - ✓ POPE
  - ✓ SQA
  - ✓ VQAv2
  - ✓ TextVQA
  - ✓ MMMU
  - ✓ SEED

(4) Experiments

- ✓ MMTok is better than all baselines, even better than the finetuned method, VisionZip🔥

Method	GQA	MMB	MME	POPE	SQA	VQA^{V2}	VQA^{Text}	MMMU	SEED	Avg.
Total 576 Tokens										
LLaVA-1.5-7B	61.90	64.70	1862.00	85.90	69.50	78.50	58.20	36.30	58.60	100%
Retain 192 Tokens ↓ 67%										
FastV	52.70	61.20	1612.00	64.80	67.30	67.10	52.50	34.30	57.10	89.6%
SparseVLM	57.60	62.50	1721.00	83.60	69.10	75.60	56.10	33.80	55.80	95.5%
VisionZip	59.30	63.00	1782.60	85.30	68.90	76.80	57.30	36.60	56.40	97.9%
DivPrune	59.97	62.54	1762.23	87.00	68.66	76.87	56.97	35.44	58.71	98.0%
VisionZip 🔥	60.10	63.40	1834.00	84.90	68.20	77.40	57.80	36.20	57.10	98.4%
MMTok	60.07	63.40	1773.86	86.42	68.76	77.11	57.68	36.33	59.21	**98.7%**
Retain 128 Tokens ↓ 78%										
FastV	49.60	56.10	1490.00	59.60	60.20	61.80	50.60	34.90	55.90	84.4%
SparseVLM	56.00	60.00	1696.00	80.50	67.10	73.80	54.90	33.80	53.40	92.9%
VisionZip	57.60	62.00	1761.70	83.20	68.90	75.60	56.80	37.90	54.90	96.8%
DivPrune	59.25	62.03	1718.22	86.72	68.66	75.96	56.06	35.56	56.98	96.9%
VisionZip 🔥	58.90	62.60	1823.00	83.70	68.30	76.60	57.00	37.30	55.80	97.7%
MMTok	59.29	62.29	1779.14	86.25	68.82	76.35	57.03	35.67	58.59	**97.8%**
Retain 64 Tokens ↓ 89%										
FastV	46.10	48.00	1256.00	48.00	51.10	55.00	47.80	34.00	51.90	75.6%
SparseVLM	52.70	56.20	1505.00	75.10	62.20	68.20	51.80	32.70	51.10	86.9%
VisionZip	55.10	60.10	1690.00	77.00	69.00	72.40	55.50	36.20	52.20	93.2%
DivPrune	57.78	59.28	1674.40	85.56	68.07	74.11	54.69	35.56	55.13	94.8%
VisionZip 🔥	57.00	61.50	1756.00	80.90	68.80	74.20	56.00	35.60	53.40	95.0%
MMTok	58.29	61.17	1715.33	85.77	69.16	75.20	56.01	36.11	57.15	**96.6%**

Table 1: **Performance Comparison on LLaVA-1.5-7B.** More details in Appendix Table 16.

(4) Experiments

✓ MMTok can be applied to Qwen-2.5-VL-7B, which contains the vision token merge layer.

Method	GQA Acc. ↑	MMB Acc. ↑	MME P+C ↑	POPE F1 ↑	VQA^{Text} Acc. ↑	SQA Acc. ↑	OCRBench Acc. ↑	Avg.† ↑
Dynamic Resolution (MinPix = 256 × 28 × 28, MaxPix = 2048 × 28 × 28), Upper Bound **(100%)**								
Avg. Tokens $\bar{T}$	358.5	276.9	867.6	359.6	976.5	323.0	652.8	
Qwen-2.5-VL-7B	60.48	83.25	2327	86.16	77.72	87.46	83.80	100%
Fixed Resolution (MinPix = MaxPix = 2048 × 28 × 28), Upper Bound **(100%)**								
Qwen-2.5-VL-7B	58.59	83.59	2339	86.09	76.64	86.91	76.60	99.3%
Retain 20% $\bar{T}$	71.7	55.4	173.5	71.9	195.3	64.6	130.6	**↓ 80%**
VisionZip	56.80	80.33	2174	83.38	70.43	84.23	59.50	94.2%
DivPrune	56.70	76.98	2163	80.59	65.86	80.91	48.10	91.5%
MMTok	58.09	79.30	2217	82.38	70.49	81.61	59.60	**94.6%**
Retain 10% $\bar{T}$	35.9	27.7	86.8	36.0	97.7	32.3	65.3	**↓ 90%**
VisionZip	52.47	75.60	2003	78.90	63.78	82.30	36.90	87.5%
DivPrune	53.43	72.85	1957	74.99	59.59	79.57	37.30	84.7%
MMTok	55.09	74.74	2051	78.75	63.90	80.47	43.60	**88.5%**
Retain 5% $\bar{T}$	17.9	13.8	43.4	18.0	48.8	16.2	32.6	**↓ 95%**
VisionZip	46.28	67.53	1677	66.38	54.49	79.57	19.70	75.4%
DivPrune	49.01	65.89	1739	68.45	52.02	77.05	24.90	76.3%
MMTok	50.66	65.89	1796	71.35	55.95	77.19	30.70	**79.0%**
0 Token **↓ 100%**								
Qwen-2.5-VL-7B	31.84	20.10	935	0.00*	38.93	71.10	1.80	33.8%

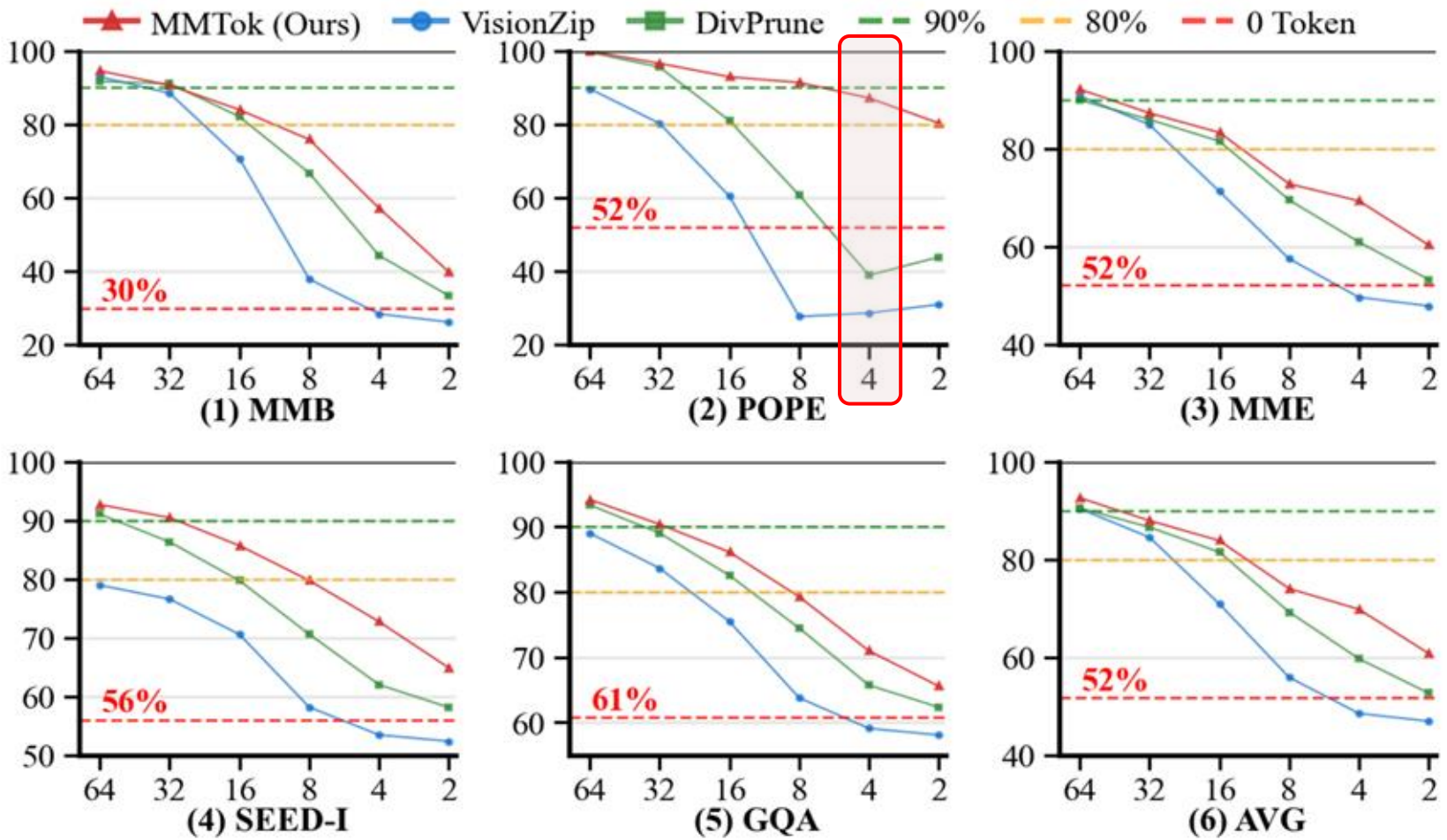
Table 3: **Comparison on Qwen-2.5-VL-7B.** Avg.† are computed over 5 datasets. *When no visual tokens are provided, Qwen-2.5-VL outputs `"No"` for all questions, leading to 0% F1. More detailed results are in Appendix Table 20.

(4) Experiments - Less Token

- ✓ MMTok works well with an extremely aggressive compression ratio (99%+).

Reduce 99.3% ↓

87.7% F1 with **4 tokens** on POPE



(4) Experiments - Ablation Study

- **Multimodal** Coverage Achieve Best Performance

Multimodal Coverage	GQA Acc. ↑	MMB Acc. ↑	MME P+C ↑	POPE F1 ↑	SQA Acc. ↑	VQA^{Text} Acc. ↑	MMMU Acc. ↑	SEED Acc. ↑	Avg. ↑
Total 576 Tokens **(100%)**									
LLaVA-1.5-7B	61.9	64.7	1862	85.9	69.5	58.2	36.3	58.6	100%
Retain 64 Tokens ↓ 88.9%									
T-V (M^{tv})	56.82	59.62	1632.47	83.56	67.87	51.97	35.33	56.36	93.7%
V-V (M^{vv})	<u>58.14</u>	59.88	1662.34	83.43	<u>68.07</u>	53.93	35.33	<u>56.90</u>	94.7%
Softmax T-V ($M^{tv'}$)	56.66	58.85	1674.11	83.69	67.82	52.01	35.33	56.37	93.8%
Softmax V-V ($M^{vv'}$)	57.97	<u>60.31</u>	<u>1684.33</u>	<u>84.31</u>	<u>68.07</u>	<u>55.90</u>	<u>35.89</u>	56.88	<u>95.7%</u>
MMTok ($M^{tv'} + M^{vv'}$)	**58.29**	**61.17**	**1715.33**	**85.77**	**68.86**	**56.01**	**36.11**	**57.15**	**96.6%**

Table 6: **Ablation on multimodal coverage in MMTok**. The best performance with token selection is highlighted in bold and the second-best is underlined.

(4) Experiments - Inference Efficiency

- **Multimodal** Coverage Achieve Best Performance

Model	Token Avg.	Inference Time(s)	GPU. util.	Memory (+15.87GB)
1 × A6000 GPU Performance on MME				
Qwen2.5-VL-7B	867.6	675	77.0%	3.05
VisionZip	86.8	508	66.3%	0.41
DivPrune	86.8	423	55.1%	0.71
MMTok	86.8	419	60.0%	0.71

Table 12: **Comparison of Inference Efficiency on Qwen2.5-VL-7B**. The initial memory usage for loading the model is 15.87GB.

#Input	#Select	Time(ms)	#Input	#Select	Time(ms)	#Input	#Select	Time(ms)
2880	160	6.417	1728	96	3.862	576	32	1.267
2880	80	3.733	1728	48	2.247	576	16	0.774

Table 13: Running time (ms) of MMTok with different numbers of input and selected vision tokens on LLaVA-NeXT-7B. The reported result is averaged over 100 runs on a A6000 GPU.

(4) Experiments - Inference Efficiency

- Linear Time Complexity: **0.7**ms ~ **6.4**ms

#Input	#Select	Time(ms)	#Input	#Select	Time(ms)	#Input	#Select	Time(ms)
2880	160	6.417	1728	96	3.862	576	32	1.267
2880	80	3.733	1728	48	2.247	576	16	0.774

Table 13: Running time (ms) of MMTok with different numbers of input and selected vision tokens on LLaVA-NeXT-7B. The reported result is averaged over 100 runs on a A6000 GPU.

(4) Experiments - Inference Efficiency

- Speed Up 1.6 x for Qwen-2.5 –VL on H100

Model	Token Avg.	Inference Time(s)	GPU. util.	Memory (+15.87GB)
1 × A6000 GPU Performance on MME				
Qwen2.5-VL-7B	867.6	675	77.0%	3.05
VisionZip	86.8	508	66.3%	0.41
DivPrune	86.8	423	55.1%	0.71
MMTok	86.8	419	60.0%	0.71

Table 12: **Comparison of Inference Efficiency on Qwen2.5-VL-7B**. The initial memory usage for loading the model is 15.87GB.

(4) Experiments

- MMTok performs well in multi-turn conversation

Q1: Is there a handbag in the image?

32 / 576 (5.5%)

MMTok: Yes

Q2: Is there a person in the image?

MMTok: No

Q3: Is there a potted plant in the image?

MMTok: Yes

Q4: Is there a vase in the image?

MMTok: No

Q5: Is there a dining table in the image?

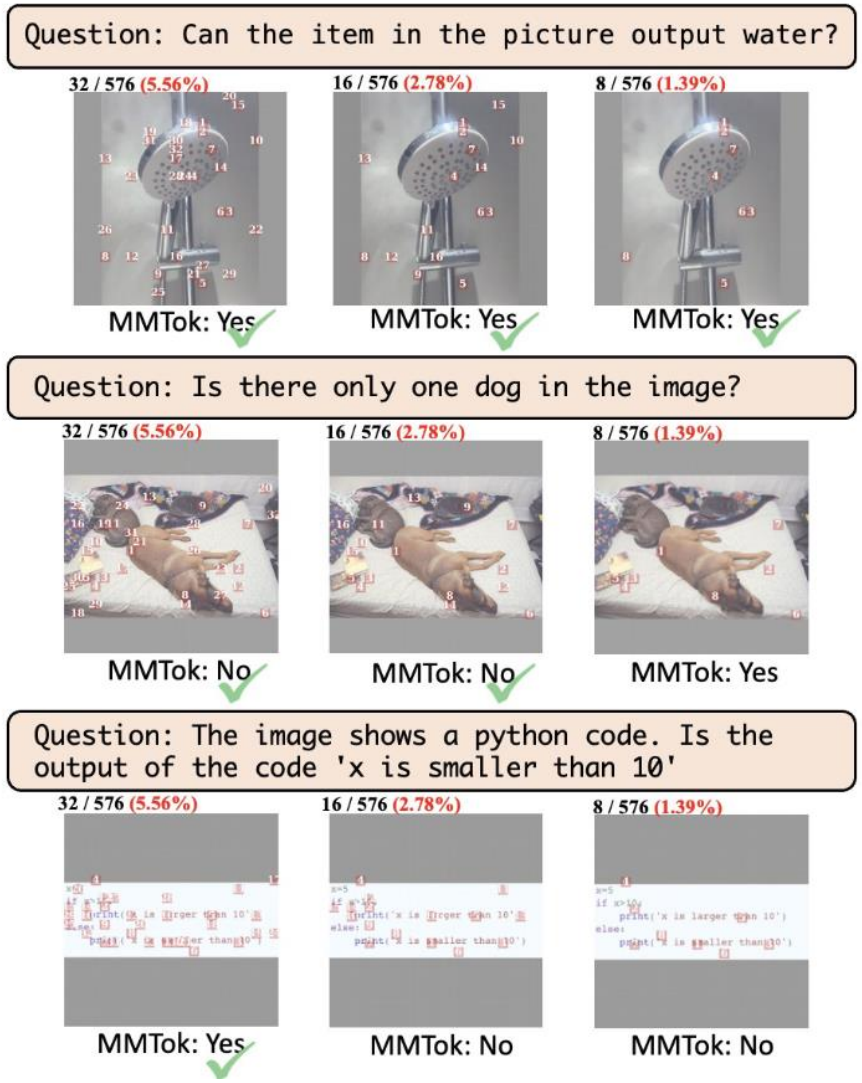
MMTok: Yes

Q6: Is there a cup in the image?

MMTok: No

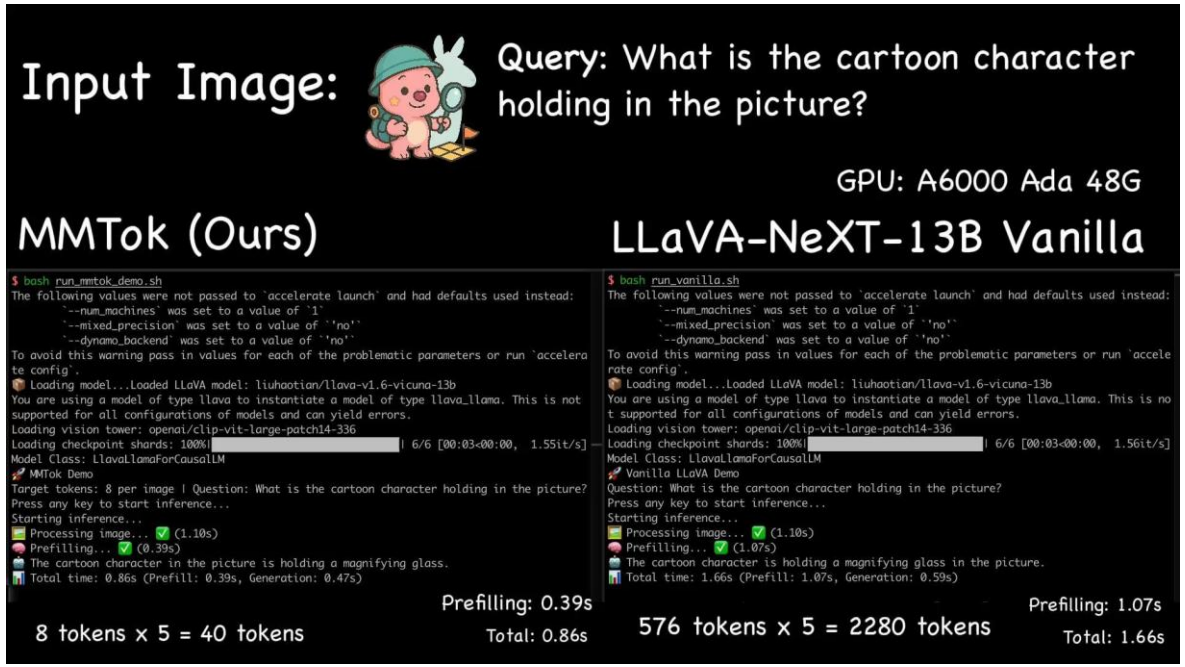
(4) Experiments

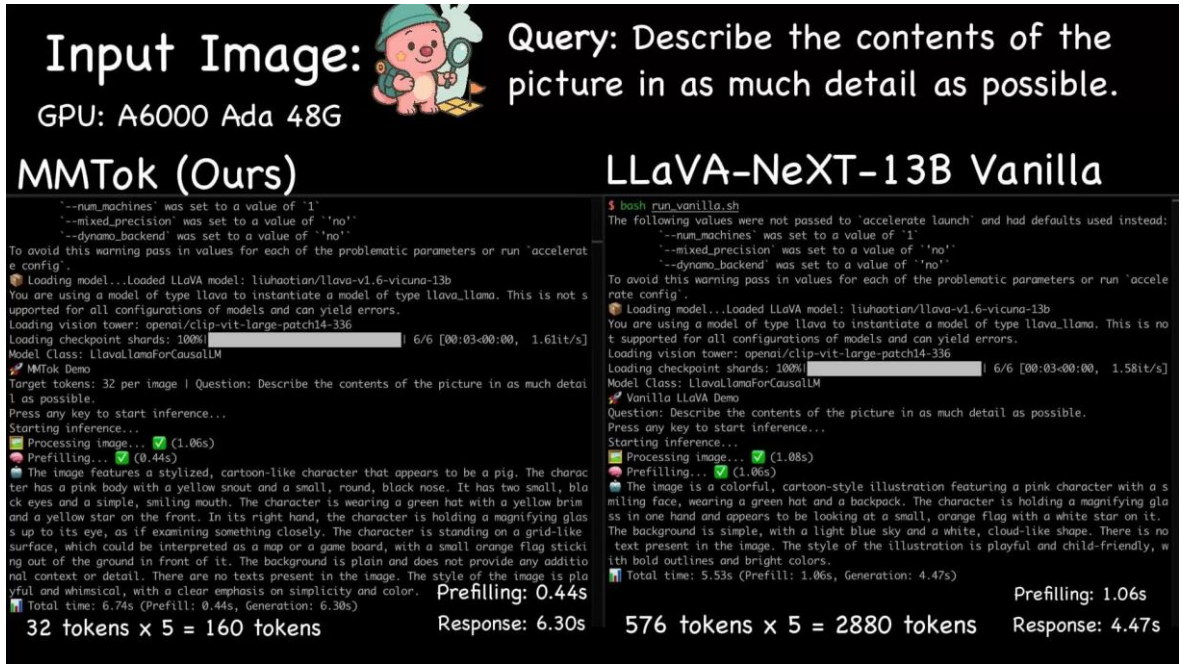
- Future Direction: Hardness-aware Vision Token Selection



(5) Application: Plug-and-Play in VLMs, then Speed up 3x

- ★ A Training-free Framework to Speed Up Inference (3X) on A6000
- ★ Maintain 95%+ performance of original VLMs





MMTok

Multimodal Coverage Maximization for Efficient Inference of VLMs

ICLR 2026

Sixun Dong[1 †], Juhua Hu[2], Mian Zhang[3 †], Ming Yin[4 †], Qi Qian[5]

Paper: https://arxiv.org/abs/2508.18264

Code: https://github.com/Ironieser/MMTok

Project Homepage: https://cv.ironieser.cc/projects/mmtok.html

Code

Homepage