NEURAL INFORMATION PROCESSING SYSTEMS

# ElasticMM: Efficient Multimodal LLMs Serving with Elastic Multimodal Parallelism

*Zedong Liu, Shenggan Cheng, Guangming Tan, Yang You*, Dingwen Tao**

中科院计算所 INSTITUTE OF COMPUTING TECHNOLOGY ,CAS

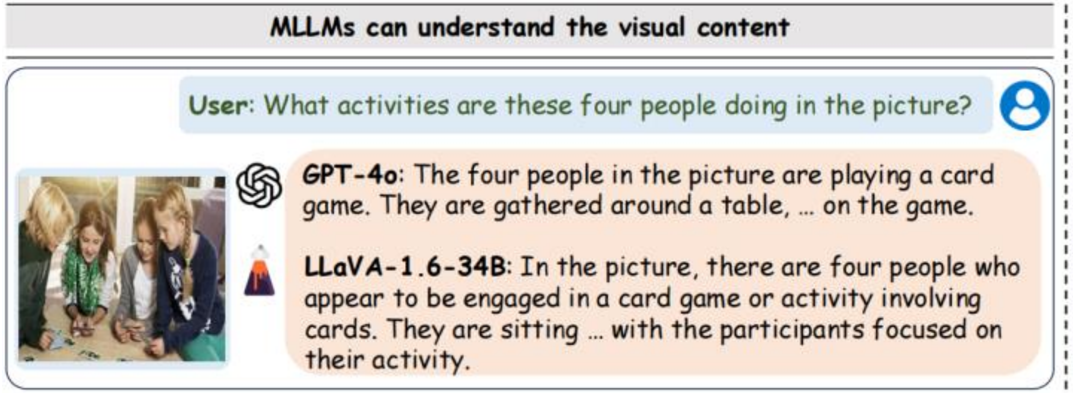
NUS National University of Singapore

中国科学院 CHINESE ACADEMY OF SCIENCES

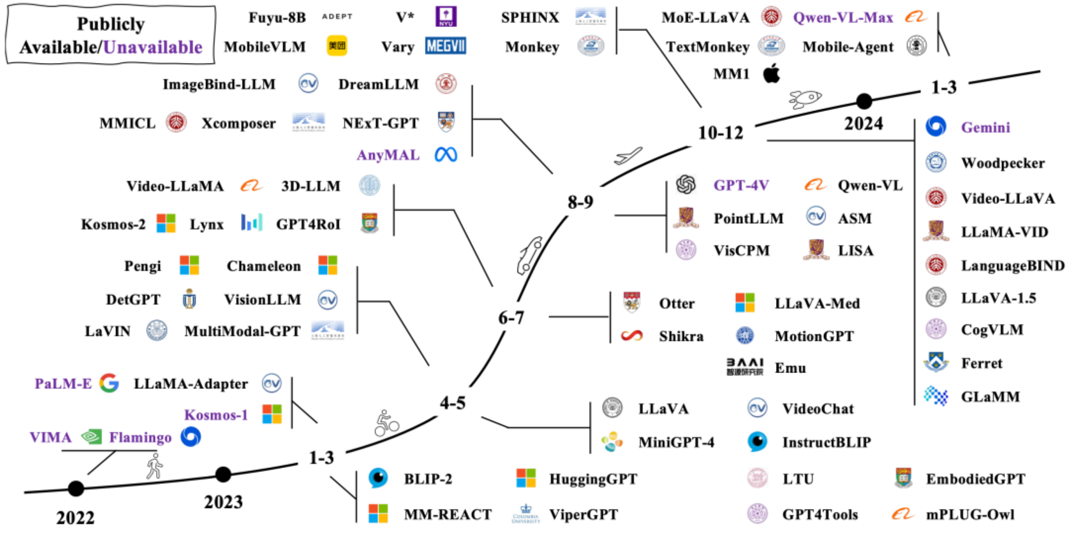
UESTC 1956

# Multimodal Large Language Model

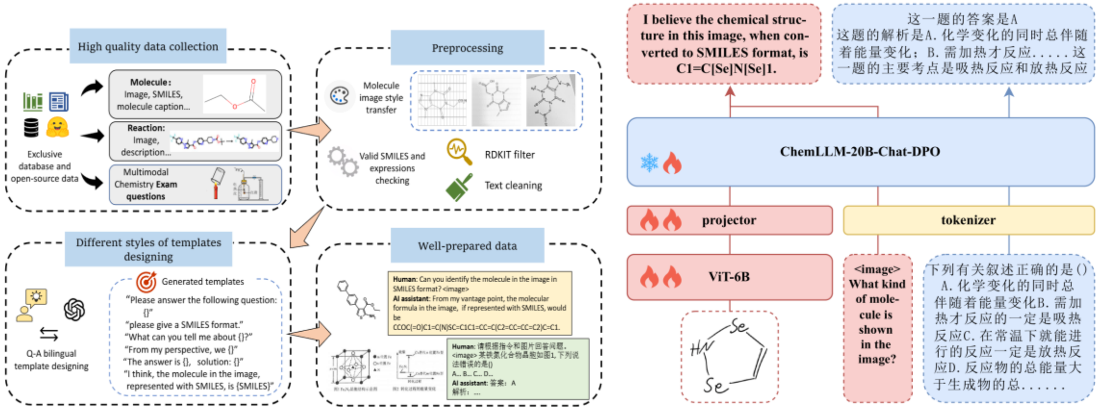
- ➢ MLLMs extend LLM capabilities to images, video by **integrating additional components.**
- ➢ Larger Size and higher architectural complexity lead to significantly **increased resource usage and latency.**




Models like GPT-4o and LLaVA are widely adopted



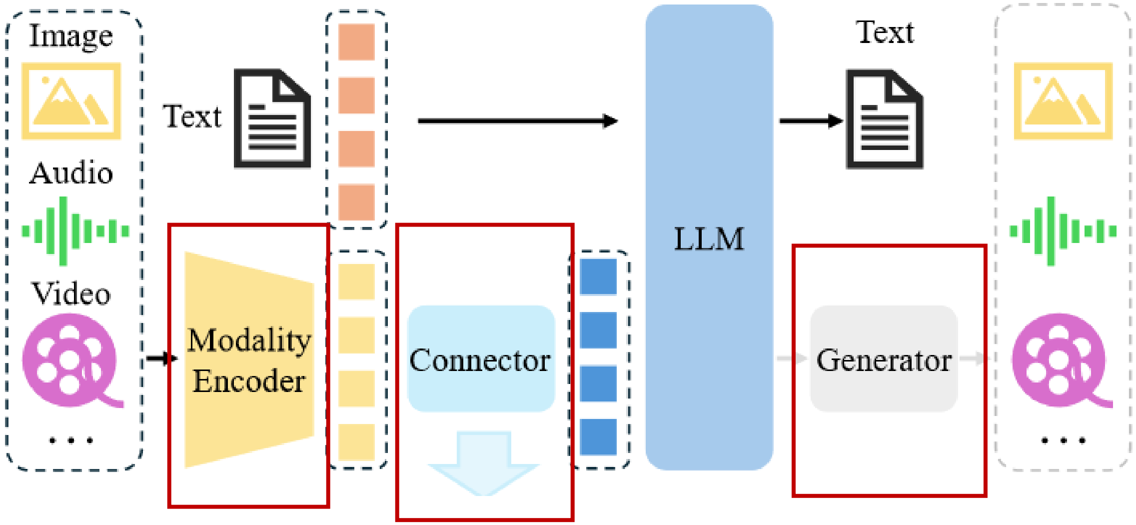

Rapid Progress of Multimodal LLMs




Multimodal is essential for chemistry & materials models

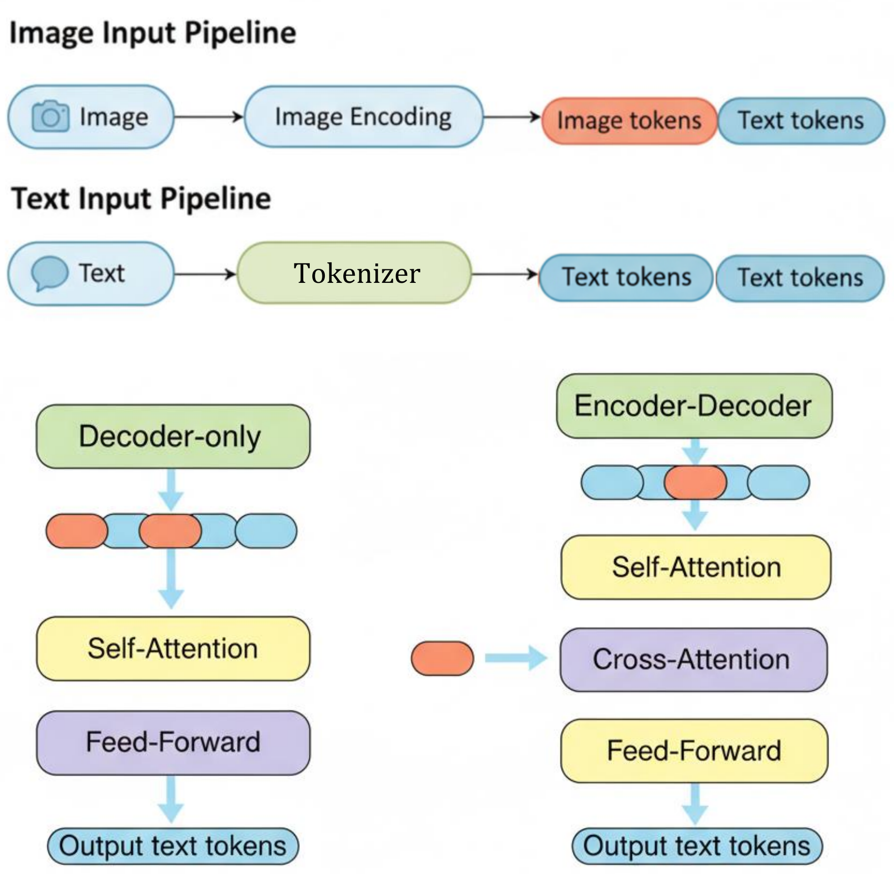
# Background: MLLM Inference Pipeline

- **Image Preprocessing:** resize & tile images
- **Image Encoding:** extract features → vision tokens
- **Text Generation:** LLM produces responses from image+text




**Overview of a MLLM Inference Architecture**

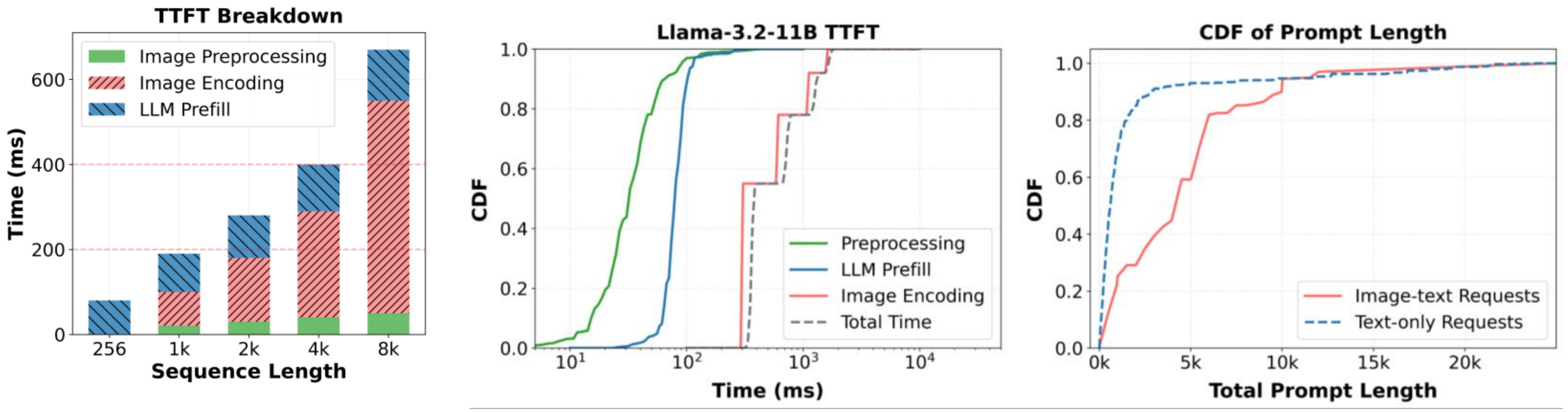
# Background: MLLM Architectures




**Two Class:** Decoder-only vs. Encoder-decoder

- **Decoder-only:** vision + text tokens
  all tokens processed **in every generation step**
- **Encoder-decoder:** Use cross-attention
  vision interacts with text **only through cross-attention layers**

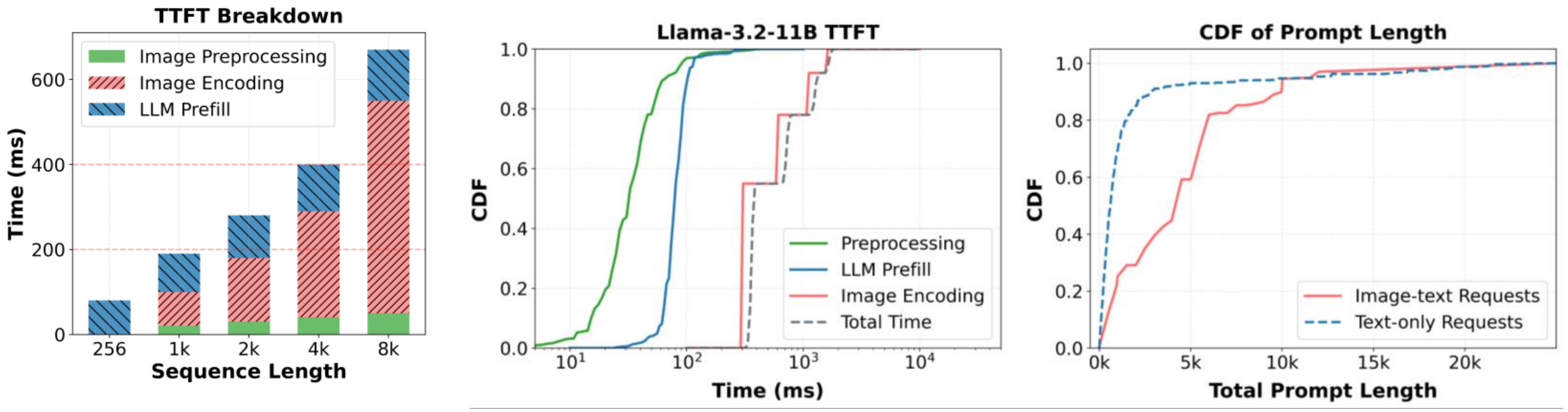
# Background: Additional Overheads




## Additional Overhead：

1. **Structural Complexity:**
   Extra components and stages significantly raise first-token latency; encoding can be **3–8×** the prefill time.

# Background: Additional Overheads




**Additional Overhead：**

2. **Longer Context:**
   Vision tokens greatly expand the input context, often **100–2000×** longer than pure text.
3. **Request Heterogeneity:**
   Mixed multimodal and text-only requests reduce inference efficiency and GPU utilization.

# Limitation on Existing Systems

Systems like vLLM and SGLang show clear limitations in MLLM inference due to **tightly coupled execution**.

- **Service-Level :** mixed execution text-only and multimodal hurts efficiency and increases SLO violations.
- **Architectural-Level:** EncDec models have heterogeneous compute; mixing request types in a batch increases latency for text requests and reduces overall efficiency.

# Motivations

**Key Insight 1 — Modality-Aware Decoupling:** Text-only and multimodal requests should be served independently to meet their distinct requirements.

**Limitation of Static Allocation:** Fixed resource allocation cannot adapt to shifting request patterns or changing parallelism needs.

**Key Insight 2 — Elastic Serving:** Dynamic resource reallocation and stage-specific parallelism are essential for handling fluctuating multimodal workloads at scale.

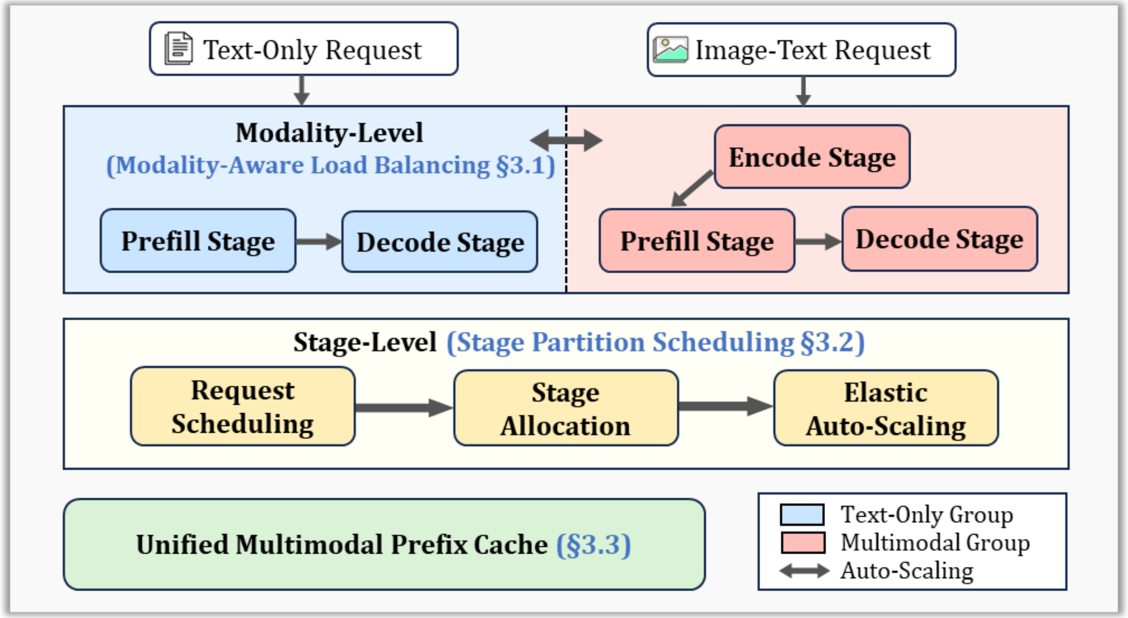
# Overview of ElasticMM

## Elastic Multimodal Parallelism (EMP)

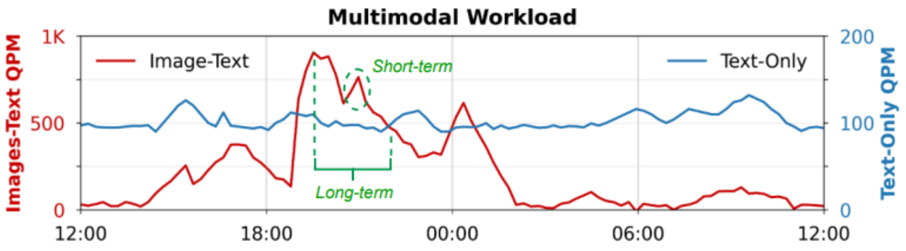
- **Modality-Level Scheduling:**
  Elastic instances grouped by modality
- **Stage-Level Scheduling:**
  Inference split into encoding, prefill, and decode; resources elastically scaled

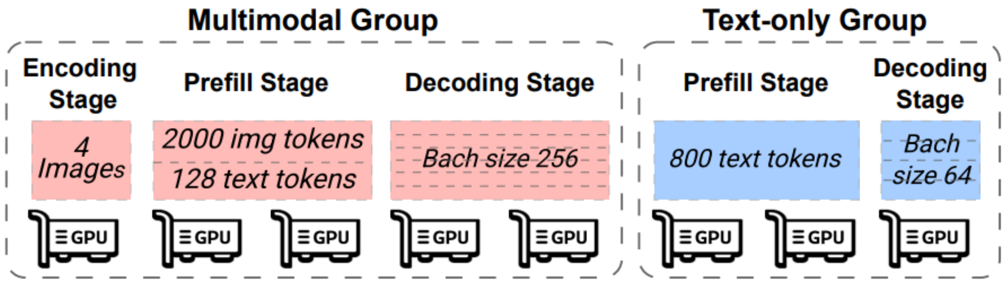
**Key Technique 1 — Modality-Aware Load Balancing**:
Addresses load imbalance **across** modality groups.

**Key Technique 2 — Elastic Partition Scheduling**:
stage-level elastic parallelism **within** each group.



# Modality-Aware Load Balancing (Group-level)



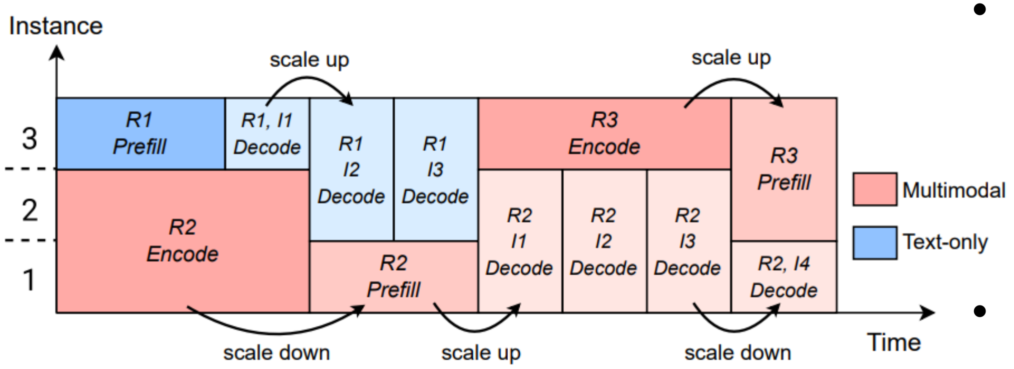




- **Proactive Mechanism:** exhibit smooth and periodic long-term patterns; a greedy algorithm maximizes each modality group's **minimum peak tolerance** (bt).

$$bt(i) = \frac{\#\text{ Instances } i \text{ can use for its peak load}}{\#\text{ Instances } i \text{ can use for its average load}} = \frac{N_i^{\text{peak}}}{N_i^{\text{avg}}}$$

- **Reactive Scaling:** Unpredictable short-term spikes are handled by dynamic expansion and inter-group resource preemption.

# Elastic Partition Scheduling (Stage-level)




**Three-Step Decoupling:**
1) Request scheduling 2) Stage allocation 3)Elastic auto-scaling

- **Stage allocation :** Prefill prioritized due to higher scalability.

$$\text{Gain} = \sum_{r \in R_p} \frac{T(R_p, E_p) - T(R_p, E_p \cup e_{max})}{r.\text{input\_len}} \qquad \text{Cost} = \sum_{r \in B_d} \frac{M(e_{max}) + w \cdot L(B_d, E_d - e_{max})}{r.\text{output\_len}}$$

- **Elastic auto-scaling:** Decode-stage shortages trigger scaling; a cost model decides intra- or inter-group preemption.

$$\text{Gain} = \sum_{r \in B_d} \frac{\text{AvgLat}_d - T(B_d, E_d \cup e_{max})}{r.\text{output\_len}} \qquad \text{Cost} = \sum_{r \in R'_p} \frac{M(e_{max}) + w \cdot L(R'_p, E'_p - e_{max})}{r.\text{input\_len}}$$

# Multimodal Inference Optimization

- **Unified Prefix Cache:** Two-level cache shared across multimodal and text tokens, reducing redundant computation and data transfer during encoding.

- **Non-Blocking Encoding:** Decouples preprocessing and encoding; uses asynchronous pipelining so these stages run in parallel with other inference stages on separate instances.

# Experimental Setup

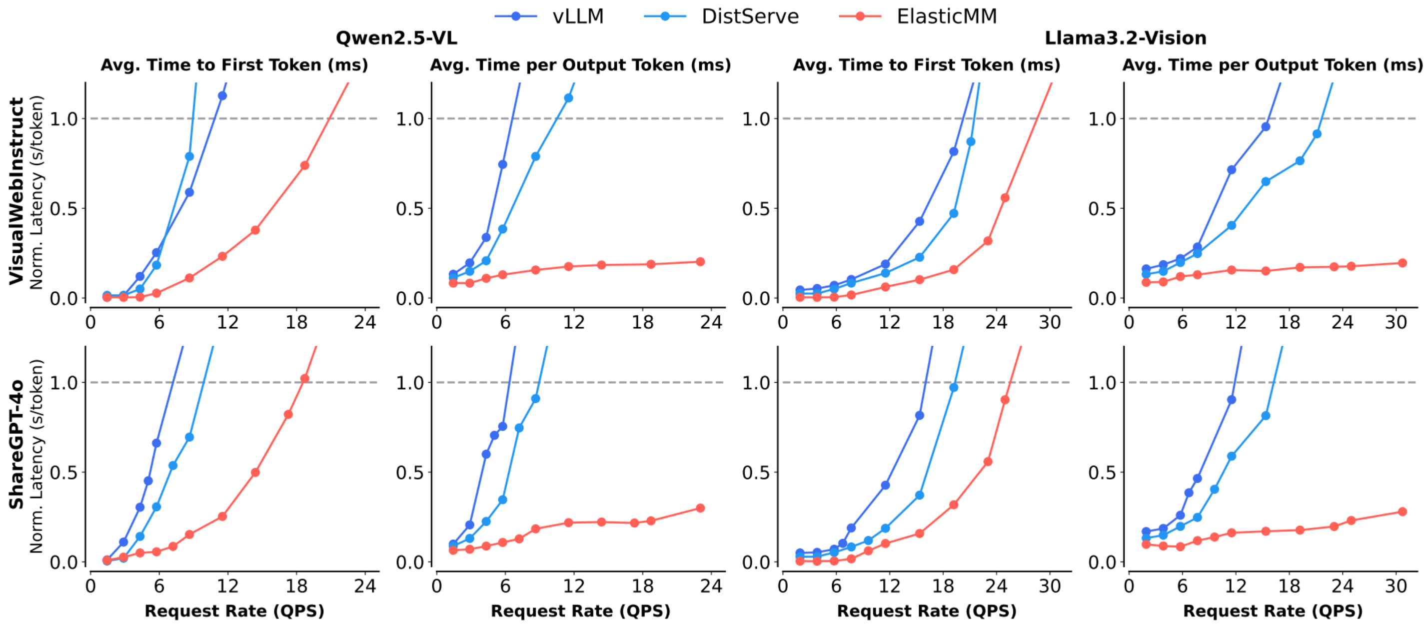
**Test Bed：NVIDIA H800 * 8**

**Baseline：SOTA Systems vLLM，DistServe**

**Model：Llama3.2-Vision 11B and Qwen2.5-VL 8B**

**Dataset：VisualWebInstruct and ShareGPT-4o**

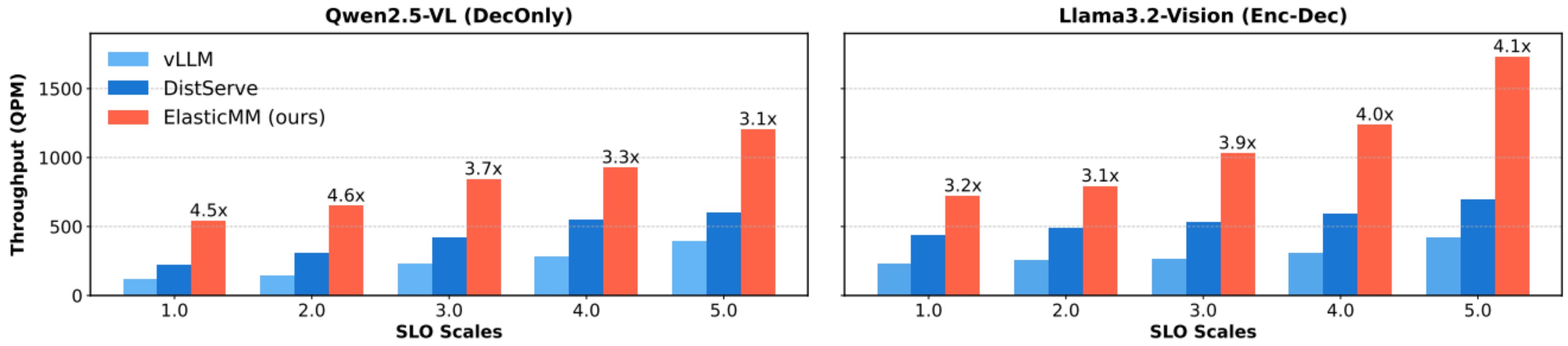
# Evaluation 1： End to End Latency




- ➢ Up to **4.2×** lower TTFT compared with vLLM
- ➢ Up to **2.6×** lower TTFT compared with DistServe

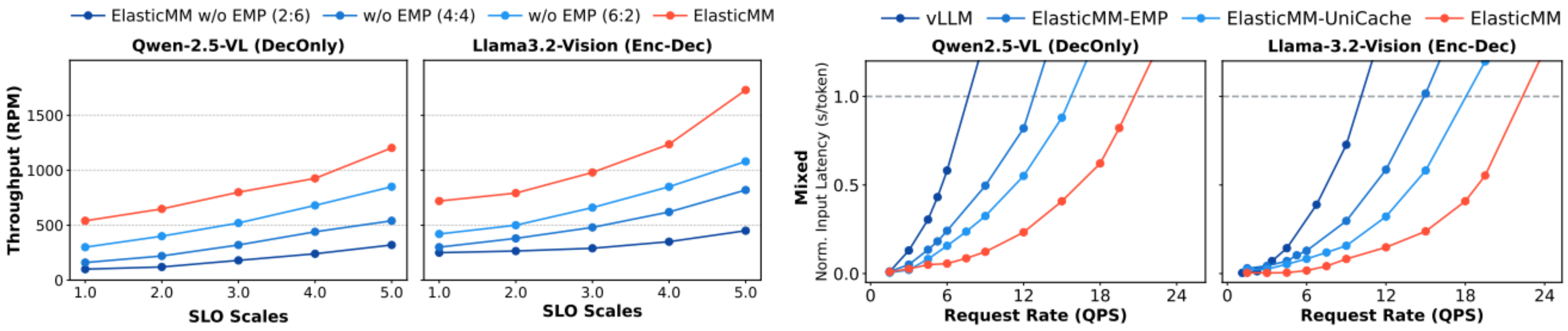
# Evaluation 2： Throughput

**Evaluated under both relaxed and strict SLO settings**




- **ElasticMM improves throughput by up to 4.5× and 2.3× over vLLM and DistServe**

# Evaluation 3： Ablation Study

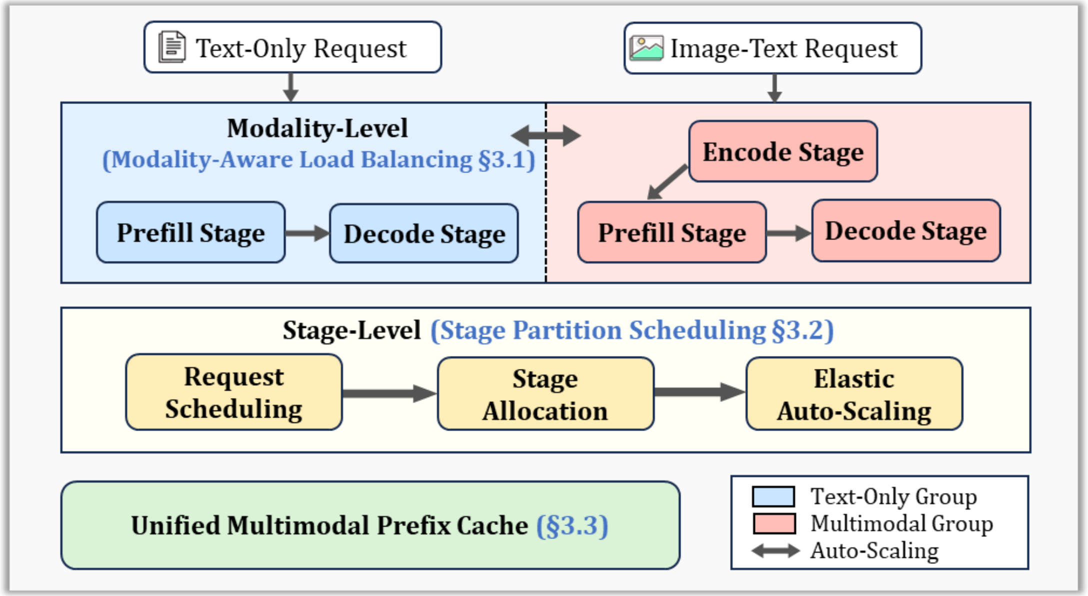



- **Static Allocation vs. EMP:** EMP consistently achieves higher GPU utilization across different static allocation ratios.

- **Prefix Cache & Non-Blocking Encoding:** Both optimizations consistently improve latency on mixed multimodal workloads.

# Conclusion

- ➢ **Principles — Decoupled & Elastic Serving**
- ➢ **Design — ElasticMM with EMP**
- ➢ **Results — Strong Performance Gains**



# Thank you : )

Contact: captain.liu77@gmail.com

Github: https://github.com/CapitalLiu/ElasticMM