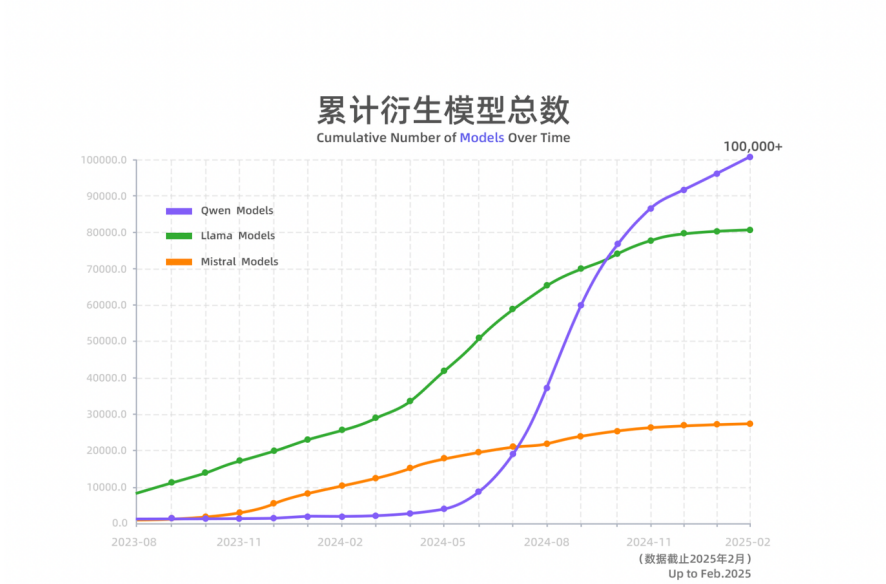
算法工程师视角看Qwen2.5系列

Qwen 开源模型在全球范围内取得了骄人的成绩，下载量**超过 1 亿**，衍生开发模型**超过 10w 个**，稳居**全球第一。**这一成绩的背后，是阿里云团队对技术的不懈追求和对社区的积极贡献。Qwen2.5 的推出，更是将这一辉煌推向了新的高度。



Qwen2.5系列的技术亮点

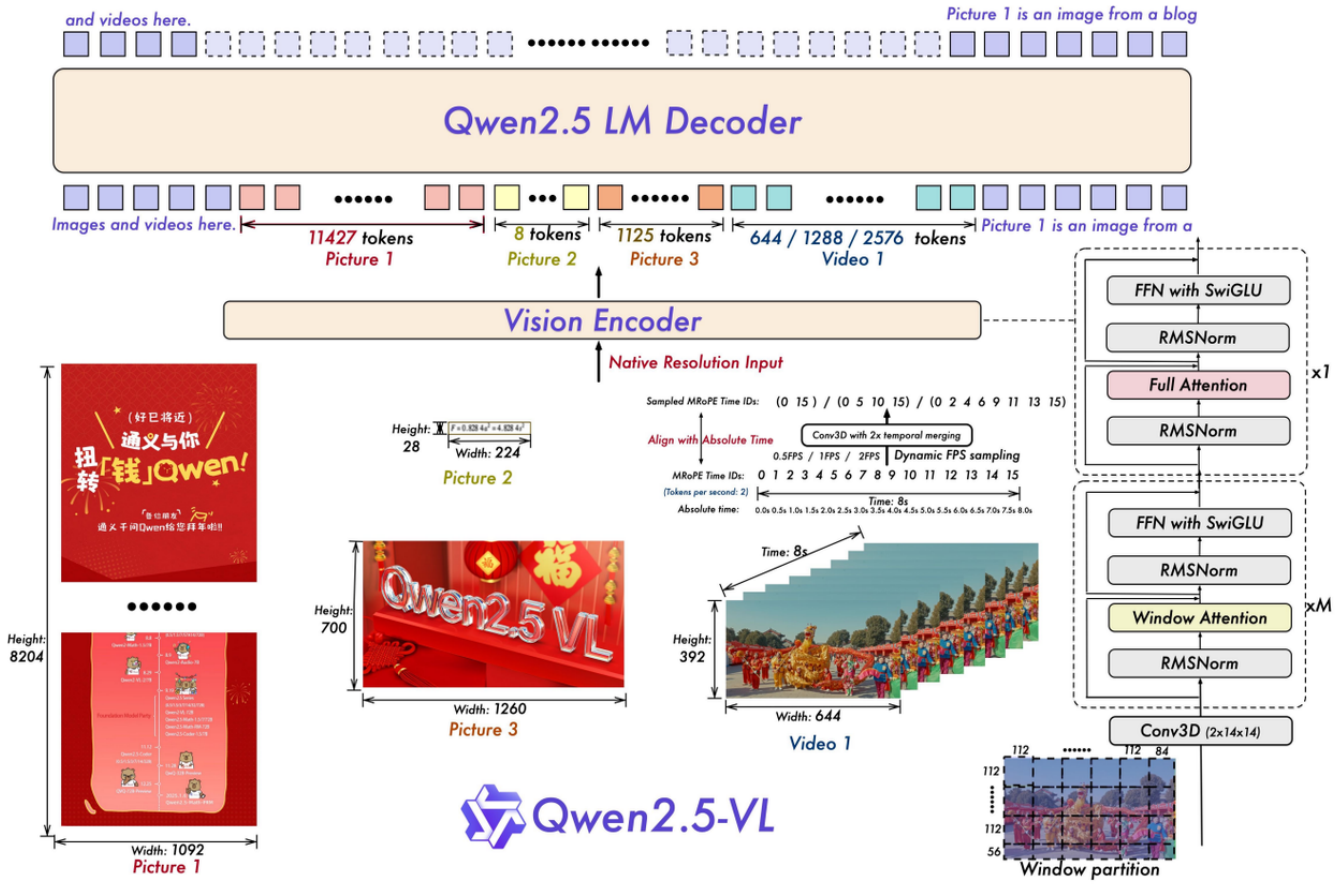
Qwen2.5-VL：视觉理解的冠军

Qwen2.5-VL 在视觉理解领域取得了令人瞩目的成就，斩获了 **“视觉理解开源大模型冠军”** 的殊荣。这一成绩的取得，离不开其在技术上的创新和突破。

Qwen2.5-VL共包含3个尺寸，3B（更易端侧部署）、7B（速度与效果的平衡）以及72B（效果最强）。

架构革新：给AI装上"动态感知器官"

整体架构由三个核心组件构成：视觉编码器（ViT）、语言模型（LLM）和基于 MLP 的视觉 - 语言融合器。兼顾视觉-语言模态的深度融合与计算效率优化：



模型框架图：展示视觉编码器动态处理图像/视频，LLM解码生成结果的流程

动态分辨率视觉编码器（ViT）

- 基于原生分辨率输入，通过**窗口注意力（Window Attention）**策略解决计算复杂度问题：仅4层使用全局注意力，其余层采用最大112×112窗口的局部注意力，实现线性计算复杂度（$O(N)$ vs. 传统ViT的 $O(N^2)$）
- 引入**2D旋转位置编码（2D RoPE）**，保留空间位置信息；视频处理扩展为3D分割（14×14图像块+连续帧分组），减少时序冗余。
- 结构优化：采用SwiGLU激活函数与RMSNorm归一化，提升视觉-语言组件兼容性。

多模态对齐语言模型语言模型（LLM）

基于Qwen2.5 LLM初始化，升级**多模态旋转位置嵌入（MRoPE-Aligned）**：将位置编码分解为时间、高度、宽度三个维度，视频场景下通过时间ID增量与绝对时间对齐，支持变帧率视频的时序推理。

视觉 - 语言融合器

提出**动态特征压缩策略**：将相邻4个图像块特征拼接后，通过两层MLP投影至文本嵌入空间，实现可变长视觉特征序列的高效压缩（序列长度缩减75%），降低LLM计算负载。

实战能力：AI的"十八般武艺"从何而来

1. 文档解析：秒变"全能文书官"

- **结构化HTML引擎**：表格、乐谱、化学式等元素被编码为带坐标的HTML标签。例如水分子（H_2O）会转化为：

```
<formula x1=120 y1=80 x2=200 y2=160>H<sub>2</sub>O</formula>
```

这种"视觉元素字典"使模型能同时理解排版逻辑与语义内容。

- **多语言OCR突破**：通过合成引擎生成法语菜单、阿拉伯路牌等稀缺数据，配合对抗训练，使小语种识别准确率提升58%。

2. 视频理解：打造"时间管理大师"

- **动态FPS训练**：随机抽取1-30 FPS的视频片段进行训练，让AI适应各类播放场景。测试显示，在5 FPS低帧率视频中，事件检测准确率仍保持82%。
- **长视频记忆架构**：采用32K超长上下文窗口，可连续解析768帧（约1小时视频），在LVBench测试中对《星际穿越》5维空间场景的因果关系推理准确率达63%。

3. 代理操作：从"旁观者"到"实操达人"

- **GUI交互预训练**：合成20万组手机/电脑界面截图，标注按钮坐标与操作链。例如"微信发送图片"被分解为：

```
{"步骤":["点击聊天框(x=120,y=300)", "选择图库(x=80,y=600)", "滑动选择图片(x1=200,y1=400,x2=500,y2=800)"]}
```

- **多模态思维链**：在Android Control任务中，模型需先解析屏幕文字（"未连接WiFi"），再定位设置图标，最终执行点击操作。这种"观察-推理-行动"闭环使任务成功率提升至93.7%。

性能对决：挑战GPT-4o的"六边形战士"

1. 杀手锏场景

- **医疗报告解析**：在SEED-Bench-2-Plus测试中，Qwen2.5-VL对CT影像的异常区域定位精度达73.2%，比GPT-4o高9.8%，关键在动态分辨率下可识别3mm级病灶。
- **工业流程图理解**：解析化工厂P&ID图纸时，能自动标注阀门（V-101）、管道（L-203）等元素，设备关联推理准确率91.4%，超越专业解析软件15%。

2. 效率革命

- **72B模型推理优化**：

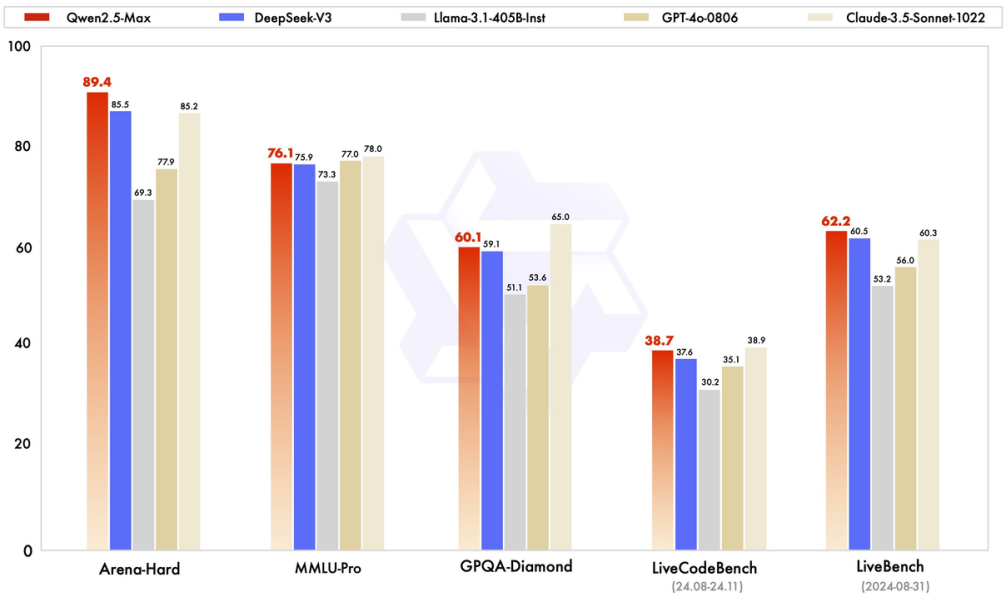
优化项	传统方案	Qwen2.5-VL	提升幅度

图像编码速度	380ms	220ms	42%
视频帧处理内存	24GB	14GB	41%
长文档解析吞吐量	12 docs/min	28 docs/min	133%

Qwen2.5-Max：超大规模模型的探索

Qwen2.5-Max 是 Qwen 团队在超大规模MoE模型领域的一次重要探索。其使用超过 20 万亿 token 的预训练数据及精心设计的后训练方案进行训练，在多个基准测试中展现出了卓越的性能。

1. **基准测试表现**：在 Arena-Hard、LiveBench、LiveCodeBench 和 GPQA-Diamond 等基准测试中，Qwen2.5-Max 的表现超越了 DeepSeek V3 等业界领先的模型。这一成绩的取得，得益于其在模型架构、训练数据和训练方法上的创新。



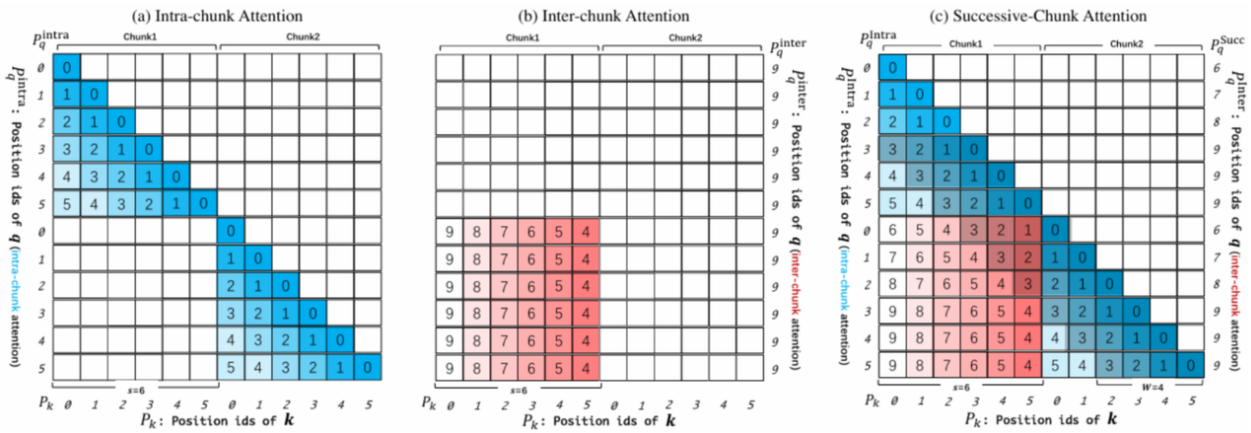
2. **模型对比**：在基座模型的对比中，Qwen2.5-Max 与领先的开源 MoE 模型 DeepSeek V3、最大的开源稠密模型 Llama-3.1-405B 等进行了对比。结果显示，Qwen2.5-Max 在大多数基准测试中都展现出了显著的优势。

	Qwen2.5-Max	Qwen2.5-72B	DeepSeek-V3	LLaMA3.1-405B
MMLU	**87.9**	86.1	87.1	85.2
MMLU-Pro	**69.0**	58.1	64.4	61.6
BBH	**89.3**	86.3	87.5	85.9
C-Eval	**92.2**	90.7	90.1	72.5
CMMLU	**91.9**	89.9	88.8	73.7
HumanEval	**73.2**	64.6	65.2	61.0
MBPP	**80.6**	72.6	75.4	73.0
CRUX-I	**70.1**	60.9	67.3	58.5
CRUX-O	**79.1**	66.6	69.8	59.9
GSM8K	**94.5**	91.5	89.3	89.0
MATH	**68.5**	62.1	61.6	53.8

Qwen2.5-1M：长上下文模型的突破

Qwen2.5-1M 是 Qwen 团队在长上下文模型领域的一次重要突破。其将模型窗口提升到了 1M，为处理长文本任务提供了强大的支持。

1. **技术亮点**：
    - **Dual Chunk Attention (DCA)**：DCA 是一种无需训练即可有效进行窗口长度外推的方法。其通过将长输入拆分成多个 chunk，并计算三种 attention（intra-chunk attention、inter-chunk attention 和 successive-chunk attention），实现了对长文本的有效处理。核心思想是**将长文本分割成多个较小的chunks**，然后分别在这些chunk内和chunk之间应用注意力机制。论文：training-Free Long-Context Scaling of Large Language Models



- **MInference 1.0**：MInference 1.0 是一个理论有损的推理加速框架，加速的是 pre-filling 阶段。其通过动态估计 input-dependent 的稀疏 mask，实现了对长文本的高效推理。
- **chunked prefill**：通过将输入 prompt 切分成小的 chunk，并在推理时同时进行 prefill 和 decode，提升了 GPU 的利用率和吞吐量。

2. **训练与数据**：

- **多阶段预训练**：Qwen2.5-1M 的预训练分成 5 个阶段，渐进式提升训练的窗口长度。在各个阶段中，使用的数据里有 75% 和当前的窗口长度相同，而另外 25% 的数据则是较短的。
- **预训练数据**：其使用了真实世界的数据和合成数据进行预训练。合成数据主要包括 Fill in the Middle、Keyword-Based and Position-Based Retrieval 和 Paragraph Reordering 等类型。
- **SFT 数据和 Qwen-Agent**：Qwen2.5-1M 从预训练语料中选择长文档，并根据这些长文档来生成 QA，进行 SFT。Qwen-Agent 框架通过 RAG 的方式，让较短窗口的模型可以处理长文档。

3. **推理与效果**：
   - **Length Extrapolation**：使用 DCA 和 YaRN 的注意力缩放，将推理窗口提升到 1M。
   - **使用 MInference**：通过 MInference 将推理窗口提升到 1M，并配合 chunked prefill 使用，提升了吞吐量。
   - **sparse attention 配合 DCA**：在结合 MInference 和 DCA 时，通过恢复距离值的连续，解决了 performance drop 的问题。
   - **Sparsity Refinement**：随着长度增加，跟踪 MInference 的 pattern 的 attention score 召回值，如果召回值低于阈值，增加 vertical 或者 slash 的预算，提升了召回率。