

白皮书 · 2026 年 10 月

大语言模型推理能效评估

Benchmark 1.5

构建“测-评-用-优”闭环体系，推动 AI 可持续发展

编制单位



清华大学
清华四川能源互联网研究院
Sichuan Energy Internet Research Institute
Tsinghua University

本白皮书由清华大学电机工程与应用电子技术系信息能源实验室、清华四川能源互联网研究院信息能源与绿色智算技术研究所、新型电力系统运行与控制全国重点实验室电化学储能高效集成与控制团队联合牵头编制，Linux 基金会研究部门与 SODA 基金会（通过其 Aitra 项目）提供开放协作生态系统支持。在此谨向 Linux 基金会、SODA 基金会及 Aitra 项目致以诚挚感谢。

支持单位



Research



soda foundation



目录

- 目录.....2
- 前言.....4
- 第一章 研究背景与评测目标.....4
 - 1.1 大模型推理能耗的刚性增长.....4
 - 1.2 能效评测缺位与选型困境.....5
 - 1.3 评测目标与 Benchmark 1.5 的定位.....5
- 第二章 能效评价指标体系.....6
 - 2.1 模型筛选.....6
 - 2.1.1 筛选原则.....6
 - 2.1.2 模型版本连续性与分组构成.....7
 - 2.1.3 新增模型的代表性.....8
 - 2.2 评价指标.....9
 - 2.3 能效电费映射.....11
- 第三章 测试方法与实验设计.....12
 - 3.1 测试环境.....12
 - 3.2 测试数据集.....13
 - 3.3 测试参数.....14
 - 3.4 测试流程.....15
- 第四章 测试结果.....17
 - 4.1 能耗对比.....18
 - 4.1.1 不同场景稳定态能耗对比.....18
 - 4.1.2 不同场景全生命周期能耗热力图对比.....20
 - 4.2 生命周期能耗结构.....23
 - 4.3 吞吐率对比.....25
 - 4.4 典型模型分析.....26
 - 4.4.1 Gemma 4 31B 的跨场景部署表现.....26

4.4.2 多模态模型对比.....	27
4.5 不同推理引擎下的部署表现.....	28
4.6 测试结果总结.....	30
第五章 结论与展望.....	31
5.1 核心结论.....	31
5.2 未来展望.....	32
5.3 局限性说明.....	33
附录 A 参测模型制品与版本标识.....	34
附录 B 逐运行配置与阶段能耗.....	35

前言

随着大模型进入生产环境，推理侧的能耗规模持续上升。本白皮书关注的不是训练能耗与推理能耗的总体占比，而是推理部署环节的测量口径问题：同一硬件平台上，不同模型、精度与推理引擎组合的单位输出能耗可相差数倍，若测量边界不统一，这些差异无法用于选型比较。

《大模型能效评估与排名白皮书》（以下简称 Benchmark 1.0）只比较稳定运行阶段的能效，无法回答按需启动、弹性扩缩容和短任务场景下的成本问题。《大语言模型推理能效评估白皮书》（Benchmark 1.5）将测量边界由运行与稳定阶段扩展至包含启动、正式运行和资源释放的完整服务生命周期，并将参测模型由 8 个扩展至 13 个，补充了大规模原生多模态、低激活参数 MoE、混合线性注意力与轻量 Dense 企业模型等路线。

本白皮书延续“统一测试点、可复现、可追溯”的原则，所有结论均限定于本次列出的模型制品、推理引擎版本、并行配置与工作负载，不代表各模型经独立调优后的能效上限。

第一章 研究背景与评测目标

1.1 大模型推理能耗的刚性增长

以 Transformer 为基础的大语言模型与多模态模型持续扩张，参数规模与调用量的增长共同推高了算力需求。与集中投入的训练阶段相比，推理阶段的能耗随服务时间延续，其总量取决于请求量、请求长度、批量大小与服务利用率。当模型进入生产环境，推理能耗在数据中心电力负荷中的比重随之上升。

词元是推理过程的基本计量单元，其消耗规模直接映射推理侧的能源需求。当词元调用量快速增长，而单次词元处理的能耗因模型架构、精度格式与系统适配差异可达数倍甚至数十倍时，缺少以词元为最小粒度的统一能效口径，产业界便难以在模型能力、调用成本与能耗约束之间建立可比较的权衡依据。

与此同时，算力基础设施的能源密度持续攀升，单机柜功率显著提高，液冷等技术从可选方案变为必选项，数据中心的选址逻辑也在从靠近用户转向靠近能源。能耗成本与供应稳定性已成为算力经济性的核心约束之一。

1.2 能效评测缺位与选型困境

当前大模型生态在能力评测方面已有较成熟的基准，而在能效维度上仍缺乏统一、透明、可复现的评估口径，由此带来三类具体困难。

第一，部署选型缺乏量化依据。用户在私有化部署时通常只能依据模型在特定任务上的准确率或生成质量，对完成同等任务所需的能耗成本缺乏可比数据。同一任务在不同模型、不同精度制品与不同推理引擎下可能呈现数倍差异，缺少实测数据会使选型偏离总拥有成本最优解。

第二，优化缺少共同靶点。量化、稀疏化、算子融合与内存优化等工作的效果，需要跨模型、跨配置的一致口径才能横向比较。缺少公开可比的能效数据，不同技术路径的收益难以判断，能效优化也缺少外部的竞争牵引。

第三，评估口径本身容易失真。能耗数值高度依赖测量边界：是否包含模型加载与服务初始化，是否包含资源释放，是否扣除空闲基线，都会显著改变结论。若不明确边界，稳定运行阶段的效率数据会被直接当作完整服务成本使用，从而高估短任务场景下的实际能效水平。

1.3 评测目标与 Benchmark 1.5 的定位

本白皮书聚焦建立统一、透明、可对比的大模型推理能效评估口径，具体目标包括以下四项。

(1) 明确指标定义与测量边界。给出稳定态单位输出能耗与全生命周期单位输出能耗的定义、适用场景与相互关系，并说明各阶段的起止时刻、采样方式和统计范围。

(2) 覆盖完整服务生命周期。在保留稳定态能耗、吞吐与时延指标的基础上，将模型加载、服务启动与服务释放产生的固定成本纳入测量，使按需启动与弹性扩缩容场景具备可比较的能效数据。

(3) 扩展模型与配置覆盖。在统一测试点下扩大模型范围，覆盖不同参数规模、架构类型、精度格式与输入模态，为跨模型比较提供同口径样本。

(4) 界定结论适用范围。明确所有结论仅对应本次列出的模型制品、推理引擎版本、并行配置与工作负载，并逐项披露复现条件与已知局限，避免将统一基线下的结果外推为模型的能效上限。

本白皮书的结构如下：第二章说明模型筛选原则与评价指标体系；第三章说明测试环境、数据集、参数与流程；第四章给出三类场景下的测试结果与分析；第五章总结核心结论、未来展望与局限性。

第二章 能效评价指标体系

2.1 模型筛选

2.1.1 筛选原则

Benchmark 1.5 面向大模型本地推理部署场景，选取具有代表性的模型开展性能与能效评测。参测范围覆盖不同参数规模、架构类型、精度格式、模型来源和输入模态，并要求模型制品可获取、版本可固定、部署过程可复现。

具体筛选原则如下：

（1）**可部署性**。模型权重必须能够在本地获取并固定版本，同时能够通过 vLLM 或 SGLang 建立完整的推理服务。仅提供商业 API、无法获取本地权重或无法固定模型版本的模型不纳入测试。

（2）**覆盖范围**。参测模型覆盖不同参数规模、架构、精度格式、来源地区和输入模态。架构包括 Dense、MoE 和混合线性注意力，精度包括 BF16、FP8 类和 MX 低精度格式，来源地区包括美国、欧洲、中国和日本，输入能力包括纯文本和图文多模态。

（3）**分组比较**。按总参数量将模型分为轻量级（不高于 50B）、中等规模（高于 50B 且低于 200B）和大规模（不低于 200B）。每个规模组尽量保留架构或精度不同的模型，避免比较对象过于单一。

（4）**任务适用性**。文本聊天、代码补全和图文问答根据模型实际能力执行。不支持的任务标记为“不适用”，不计为测试失败。

表 2-1 参与评测的模型规格详情

模型名称	总参数量	架构类型	模型能力			
			文本生成		代码补全	图文问答
			MMLU-Pro	Arena Text Rating (Style Control)	LiveCodeBench v6	MMMU / MMMU-Pro
DeepSeek-V4-Flash-0731	284B	MoE	—	—	—	不支持
Qwen3.5-397B-A17B-FP8	397B	MoE	87.8	1442	83.6	85.0 (MMMU) 79.0 (MMMU-Pro)
GLM-5.3-Flash	320B	MoE	—	1474	—	—
MiniMax-M3-MXFP8	≈428B	MoE	—	1440	—	78.1 (MMMU-Pro)

模型名称	总参数量	架构类型	模型能力			
			文本生成		代码补全	图文问答
			MMLU-Pro	Arena Text Rating (Style Control)	LiveCodeBench v6	MMMU / MMMU-Pro
MiMo-V2.5	310B	MoE	—	1434	—	77.9 (MMMU-Pro)
Hy3-FP8	295B	MoE	—	1457	—	不支持
Llama-4-Maverick-17B-128E-Instruct-FP8	400B	MoE	80.5	—	—	73.4 (MMMU) 59.6 (MMMU-Pro)
gpt-oss-120b	117B	MoE	—	1352	—	不支持
Gemma-4-31B-it	30.7B	Dense	85.2	—	80.0	76.9 (MMMU-Pro)
Mistral-Medium-3.5-128B	128B	Dense	—	—	—	—
Granite-4.2-30B-FP8	30B	Dense	77.6	1341	75.77	不支持
Ling-3.0-Flash-FP8	124B	MoE	—	—	—	不支持
Sarashina2.2-3B-Instruct-v0.1	3B	Dense	—	—	—	不支持

注：

- “—”表示未公布或缺少同口径数据；“不支持”表示模型不具备图像输入能力。
- 模型能力数据核验日期为 2026 年 9 月 28 日。MMLU-Pro、LiveCodeBench v6 和 MMMU/MMMU-Pro 取自各模型官方模型卡或技术报告；Arena Text Rating 取自 LMArena Text Overall 榜单的 Style Control 口径，榜单发布日期为 2026 年 9 月 25 日。
- 部分量化制品的能力数据采用对应基础模型的官方公开成绩，不代表本次量化制品的独立能力复测。MMMU 与 MMMU-Pro 为不同评测口径，已在表中分别标注。

2.1.2 模型版本连续性与分组构成

为保持与 Benchmark 1.0 的连续性，同时覆盖新的技术路线，Benchmark 1.5 将参测模型划分为延续类、更新类和新增类。延续类保留原有模型路线；更新类在原有路线基础上更新模型版本或部署制品；新增类用于补充 Benchmark 1.0 尚未覆盖的代表性模型类别

Benchmark 1.0 参测 8 个模型，Benchmark 1.5 参测 13 个模型，其中延续类 5 个、更新类 3 个、新增类 5 个。

表 2-2 Benchmark 1.0 与 Benchmark 1.5 模型演进关系

分类	Benchmark 1.0 模型	Benchmark 1.5 模型	主要变化	精度格式	权重来源
延续类	Qwen3.5-397B-A17B-FP8	Qwen3.5-397B-A17B-FP8	—	FP8	Qwen
延续类	Llama-4-Maverick-17B-128E-Instruct-FP8	Llama-4-Maverick-17B-128E-Instruct-FP8	—	FP8	RedHatAI
延续类	gpt-oss-120b	gpt-oss-120b	—	MXFP4	openai
延续类	Mistral-Medium-3.5-128B	Mistral-Medium-3.5-128B	—	FP8	mistralai
延续类	Gemma-4-31B-it	Gemma-4-31B-it	—	BF16	google
更新类	DeepSeek-V4-Flash	DeepSeek-V4-Flash-0731	更新至 0731 版本，并固定本轮使用的模型 Revision 和制品	FP8 + FP4	deepseek-ai
更新类	MiniMax-M2.7-REAP-172B-A10B-BF16	MiniMax-M3-MXFP8	由 M2.7 迭代至 M3，改用 MXFP8 制品并增加图文输入能力	MXFP8	MiniMaxAI
更新类	MiMo-V2-Flash	MiMo-V2.5	由 V2-Flash 迭代至 V2.5，扩展为原生多模态模型	FP8	XiaomiMiMo
新增类	—	GLM-5.3-Flash	补充中国大规模原生多模态模型和混合注意力路线	FP8	zai-org
新增类	—	Hy3-FP8	补充大规模 MoE 模型样本	FP8	tencent
新增类	—	Granite-4.2-30B-FP8	补充 30B 级 Dense 企业模型和 FP8 制品路线	FP8	ibm-granite
新增类	—	Ling-3.0-Flash-FP8	补充低激活参数 MoE 和混合线性注意力路线	FP8	inclusionAI
新增类	—	Sarashina2.2-3B-Instruct-v0.1	补充日本及日语 3B Dense 官方模型路线	BF16	sbintuitions

注：

1. 精度格式与权重来源均对应 Benchmark 1.5 本轮参测制品。权重来源列展示 Hugging Face 仓库的发布组织；完整权重仓库 ID、Revision 及提交日期见附录 A。
2. 精度格式表示固定 Revision 对应制品的主要权重格式，不代表所有模块或计算均使用相同精度。DeepSeek-V4-Flash-0731 采用 FP8 与 FP4 混合格式。

2.1.3 新增模型的代表性

新增模型用于填补 Benchmark 1.0 在架构、规模、精度、区域和语言覆盖上的空白，其技术定位如表 2-3 所示。

表 2-3 新增模型的技术特点与覆盖价值

新增模型	模型采用的主要技术	纳入本次评测的原因
GLM-5.3-Flash	采用大规模 MoE 结构，结合稀疏注意力和线性注意力，并支持图文输入	用于观察中国原生多模态模型及混合注意力方案的部署能效
Hy3-FP8	总参数约 295B、每 Token 激活约 21B，采用 MoE 结构、FP8 制品，模型本身支持专家并行部署	用于补充大规模 MoE 模型在轻量级等高规模组之外的实测样本。本次测试统一采用 TP8，未启用专家并行，因此不能作为专家并行条件下的实测样本
Granite-4.2-30B-FP8	采用 30B Dense Transformer 和 FP8 制品，面向企业级部署	用于在轻量级分组内与 Sarashina 和 Gemma 形成规模与精度格式对照，并补充轻量 Dense 企业模型样本
Ling-3.0-Flash-FP8	总参数约 124B、每 Token 激活约 5.1B，采用混合线性注意力 MoE	用于观察低激活参数和线性注意力方案在中等规模模型中的表现
Sarashina2.2-3B-Instruct-v0.1	采用 3B Dense Decoder-only 架构的官方日语指令模型，测试制品为 BF16	用于补充日本及日语模型样本，并观察小规模官方 Dense 模型在统一八卡基线下的部署表现

从本次全部 13 个参测模型看，轻量级模型包括 Gemma、Granite 和 Sarashina，中等规模模型包括 gpt-oss、Mistral 和 Ling，大规模模型包括 DeepSeek、Qwen、GLM、MiniMax、MiMo、Hy3 和 Llama。参测模型同时覆盖 Dense、MoE 和混合线性注意力架构，以及 BF16、FP8 类和 MX 低精度格式；其中，7 个模型支持图文多模态输入，6 个模型支持纯文本输入。

参测模型来源覆盖美国、中国、欧洲和日本。其中，美国和中国均有多个模型参测，欧洲和日本分别由 Mistral 和 Sarashina 代表。

2.2 评价指标

Benchmark 1.5 保留 Benchmark 1.0 的五项核心指标，并新增三项生命周期指标。沿用指标定义如下：

- 输出吞吐率 (tok/s)：**正式请求阶段生成的输出 Token 总数除以该阶段时长，单位为 Token/秒。
- 稳定态单位输出能耗 (J/token)：**在稳态窗口内，GPU 能耗除以同一窗口内的输出 Token 数，单位为焦耳/Token。分子与分母取自同一窗口。
- GPU 平均功率 (W)：**统计窗口内 GPU 能耗除以该窗口时长。本白皮书以多卡合计功率报告。
- 首 Token 响应时间 (TTFT)：**自请求发出至首个输出 Token 返回的时间，单位为秒。本节按 P50 报告。

单 Token 生成时间（TPOT）：首个 Token 之后的生成时长除以该阶段输出 Token 数，单位为秒/Token。

新增的生命周期指标如下：

启动能耗：从推理服务启动至正式请求开始期间的 GPU 能耗，用于反映模型加载、服务初始化和预热带带来的固定成本。

资源释放能耗：从正式请求结束至推理进程退出、显存释放且功率恢复期间的 GPU 能耗。

全生命周期单位输出能耗（J/token）：启动、正式请求和资源释放阶段 GPU 能耗之和除以输出 Token 数，用于评价一次完整服务过程的单位输出成本。

稳定态 J/token 与全生命周期 J/token 对应不同测量窗口，前者适用于持续运行效率分析，后者适用于冷启动、短任务和弹性服务分析。两者作为“单位输出能耗”项下的两个并列指标列示，共用 J/token 单位但测量边界不同，不可混用，也不可直接相减。

全生命周期单位输出能耗的分母口径为：仅计入正式请求阶段成功完成的请求所对应的接口 completion Token 数，不含预热请求、失败请求。不同模型与推理引擎对思考内容的返回和计数方式存在差异，部分运行的思考内容混合在普通输出中，部分运行保存独立 reasoning 输出字段，当前计数不能统一解释为仅可见输出 Token 数。由于各项测试对可见输出与推理 Token 的记录方式不同，本文采用接口返回的 completion Token 数计算单位输出能耗，不再统一拆分两类 Token。各项测试的 thinking 设置及 Token 统计口径见附录 B 表 B-1。各阶段起止时刻的定义见第 3.4 节。

对于一次启动后连续处理多个请求的服务，可按下式估算生命周期内的 GPU 总能耗，并据此计算启动与释放固定开销的摊销：

$$E(N) = E_{start} + E_{release} + N \cdot e_{ss} \quad (1)$$

式中， N 为同一实例生命周期内的请求数； E_{start} 和 $E_{release}$ 分别为启动与释放阶段的 GPU 能耗，单位为 J； e_{ss} 为稳定运行阶段的平均每请求 GPU 能耗，单位为 J/request。启动与释放固定开销摊销至每个请求的能耗为 $\frac{(E_{start} + E_{release})}{N}$ 。

若根据稳定态 J/token 计算 e_{ss} ，需乘以相同负载条件下的平均每请求 completion Token 数。使用附录 B 表 B-2 中的能耗数据时，需将 kJ 换算为 J，即乘以 1000。

该估算适用于请求负载和运行配置与测量条件一致的情况，不包含额外请求间隔产生的空闲能耗。实际全生命周期能耗仍以实测结果为准。

2.3 能效电费映射

为便于将技术指标转化为部署决策依据，本节给出由能耗指标推算电力成本的换算方式，并同时说明其适用边界。

本白皮书第 2.2 节与第四章报告的全部能耗数值均为 GPU 能耗。由 GPU 能耗推算数据中心电力成本，需依次补入两级未测量的损耗：一是同一服务器内 CPU、内存、存储、网络等部件及供电转换的损耗，二是数据中心制冷、配电与照明等设施损耗。因此，不能由 GPU 的 J/token 直接推算数据中心总电费。

单位词元电力成本的换算关系如下：

$$C_{elec} = \frac{e_{GPU}}{3.6 \times 10^6} \times \eta_{sys} \times PUE \times P_{elec} \quad (2)$$

式中： e_{GPU} 为每词元 GPU 能耗，单位 J/token，即本白皮书正文与图表报告的口径；除以 3.6×10^6 是将其换算为 kWh/token，该中间量的换算结果记作 E_{GPU} （单位 kWh/token），以避免与 J/token 口径混用同一符号。 η_{sys} 为同一统计窗口内服务器整机电量与 GPU 电量之比，等价于该窗口的平均整机功率与平均 GPU 功率之比； PUE 为数据中心电能利用效率； P_{elec} 为电价（元/kWh）。

η_{sys} 必须与 e_{GPU} 取自同一统计窗口。稳定运行阶段的 η_{sys} 不能直接用于启动或资源释放阶段：后两个阶段 GPU 可能处于加载或空闲状态，整机与 GPU 的功率比与稳定态不同。因此，将稳定态 η_{sys} 用于全生命周期换算会引入额外误差，全生命周期口径若需电费换算，应另行给出对应阶段的 η_{sys} 取值及来源。

整机计量边界定义为：服务器电源输入端的电量为界，即包含 CPU、内存、存储、网络与供电转换损耗。数据中心侧的制冷、配电与照明等设施损耗不重复计入，统一由 PUE 表达。 η_{sys} 与 PUE 均未在本次测试中测量，取值依赖具体服务器与机房条件，属于外部假设而非实测结果。在给出实测数据前，本节不提供电费结论。

作为换算示例，取文本聊天场景的稳定态中位数 $e_{GPU} = 0.747$ J/token，并假设 $\eta_{sys} = 1.30$ 、 $PUE = 1.40$ 、 $P_{elec} = 1.0$ 元/kWh：

$$C_{elec} = \frac{0.747}{3.6 \times 10^6} \times 1.30 \times 1.40 \times 1.0 \approx 3.8 \times 10^{-7} \text{ 元/token} \quad (3)$$

即每百万词元的电力成本约为 0.38 元。该数值同时依赖上述三项外部假设，且仅覆盖稳定运行阶段，不含模型加载与服务释放的固定成本；若按全生命周期口径计算，短任务场景下的单位成本会明显高于该值。因此该示例仅用于说明换算路径，不作为对外报价或成本承诺的依据。

需要说明的是，若 η_{sys} 与 PUE 改取其他取值，结果会按比例变化；相应取值的来源与适用范围应在正式发布时一并披露。

第三章 测试方法与实验设计

3.1 测试环境

全部测试在同一硬件平台和共同测试基线（TP8、序列调度上限 128）上运行，以消除平台差异对能效和性能的影响。参测模型的软件配置存在差异，涉及推理引擎版本、镜像和量化制品，个别模型使用特定开发版推理引擎。这些模型专属的软件配置差异在测试中予以保留，并随正式结果一并记录。硬件方面，测试平台为本地服务器，配置 8 张 GPU。软件方面，我们构建了一套统一且可固化的基准测试工具链。

表 3-1 汇总本节测试点的硬件与软件环境。为使结果可复核，下表列出复现同一测试点所需的全部环境要素；标“待补”的条目为当前底稿尚未披露、需由测试记录补充的项。

表 3-1 测试平台与可复现环境清单

项目	配置
GPU 型号与形态	NVIDIA H100 80GB HBM3
GPU 数量	8
单卡显存	80 GB
CPU 配置	双路 CPU，96 个物理核心、192 个逻辑核心
系统内存容量	约 1 TiB
GPU 驱动版本	560.35.03
CUDA 版本	容器 CUDA 运行时 12.9.1 或 12.9.2，按模型镜像固定
容器镜像与标签	按模型固定，镜像 Digest 随正式结果记录
模型 Revision	见附录 A，完整权重仓库 ID、Revision 及提交日期已列示
基准框架	ml-energy/benchmark v3.0
能耗测量库	Zeus 0.13.1（NVML）
推理引擎	vLLM 与 SGLang，具体版本按模型配置固定并随正式结果记录
稳态判定阈值	连续 100 请求 CV < 0.03

基准框架选用 ml-energy/benchmark v3.0，该框架已被 NeurIPS 2025 Datasets & Benchmarks Spotlight 收录，负责驱动测试、收集能耗与性能指标。

推理引擎以 vLLM 为主，并通过 Docker 容器锁定版本，避免 CUDA 或 PyTorch 版本漂移；在实际测试中同时使用了 vLLM 0.29.x 为主线版本，并针对个别模型的兼容性问题启用了 SGLang 作为备选。

首轮测试中，MiMo-V2.5 使用 SGLang，其余模型主要使用 vLLM；后续选取 6 个模型开展 SGLang 测试，用以对比展示推理引擎的影响。

本白皮书所报告的能耗数值口径为 GPU 能耗。能耗测量采用 Zeus 0.13.1 库，在硬件与权限支持的前提下，其监测接口可提供 GPU、CPU 及内存的能耗读数；CPU 与 DRAM 读数是否可用取决于具体硬件与运行环境，并非必然存在。本次统计仅使用其中 GPU 相关读数，CPU 与内存读数未纳入本白皮书任何指标的计算。此外测量过程还设计了两个并行的观察窗口：一个覆盖自首个正式请求发出至最后一个正式请求完成的区间，用于评估正式请求阶段能耗；另一个则通过统计方法自动判定系统进入稳态后的区间，仅使用这一区间的能耗来计算最关键的每词元消耗焦耳量指标，从而排除冷启动、突发排队等不稳定因素的干扰。其中稳态判定采用滑动窗口法：候选窗口按请求完成顺序形成，每次向后滑动一个请求，窗口长度为 100 个请求；以窗口内逐请求 J/token 的算术平均值作为该窗口的耗能水平，并以该窗口内各请求 J/token 的标准差除以平均值得到变异系数（Coefficient of Variation，CV）。当连续无重叠窗口全部满足 $CV < 0.03$ 时，判定系统进入稳态，稳态窗口取首个满足条件的窗口起点至正式请求结束。若正式请求结束时仍未有窗口满足该阈值，则该次运行记为稳态未成立，其稳定态 J/token 不参与跨模型比较，并在结果中标注。并发请求的能耗按各请求的生成时间区间归属，重叠部分的 GPU 能耗按请求在该区间内的输出 Token 数比例分摊。

同时，我们还在旁路部署了 Prometheus 监控系统以便在运行时采集数据，用于后期的稳态验证。

3.2 测试数据集

本次测试使用三类数据集，分别对应文本聊天、代码补全和多模态图像问答三大典型生产场景。三类数据集分别取自 LMArena 的真实人机对话、Sourcegraph 的真实代码库上下文与 VisionArena 的用户上传图文，均为公开的真实交互或代码语料，而非合成模板，以保障测试结果对实际部署的参考价值。测试数据集的关键参数如表 3-2 所示。

表 3-2 测试数据集关键参数

场景	数据集来源	请求数	最大输出词元	平均输入图片数	平均输入文本词元
文本聊天	lmarena-ai/arena-human-preference-100k	256	1,024	—	约 580
代码补全	sourcegraph/context-aware-fim-code-completions	1,024	2,048	—	约 2k
多模态图文问答	lmarena-ai/VisionArena-Chat	1,024	4,096	1	—

注：

1. 三类场景的主要测试点均为 TP=8、序列调度上限 128。统一测试点用于比较相同硬件配置下的部署结果，不代表各模型的最佳配置。
2. 通用文本聊天的数据基于 LMArena 平台收集的用户与各主流模型之间的真实对话，涵盖中英双语问答、编程、写作、推理等多种任务类型。我们以固定随机种子抽取 256 条对话，保留原始的提示词长度分布，并限制最大输出长度设为 1024 词元，以模拟日常对话的典型回复长度。
3. 代码补全的数据采用 Sourcegraph 的真实代码库上下文补全代码数据集。平均输入长度约 2k 词元，测试共 1024 条请求，并限制输出上限为 2048 词元。
4. 多模态图像问答的数据取自 VisionArena 的用户上传图片与文本提问，每条请求包含一张图像与问题文本，共 1024 条请求。此外我们将最大输出长度放宽至 4096 词元，以测量视觉编码器联合推理及长生成场景的能耗特征。

这三类数据集在输入长度、输出长度和模态方面各有侧重，综合覆盖了大模型在实际应用中的主要工作模式。

3.3 测试参数

Benchmark 1.5 以统一测试点开展模型横向比较，张量并行度设为 8，序列调度上限设为 128。与 Benchmark 1.0 的 TP×BS 多配置扫描相比，本次测试重点扩展模型覆盖范围和完整生命周期测量。工作负载及请求参数见表 3-2，具体运行参数见表 3-3。

表 3-3 统一测试点与运行参数

参数	设定值	说明
张量并行度 (tensor-parallel-size)	8	统一使用 8 张 GPU
序列调度上限 (max-num-seqs)	128	服务端单轮可调度的序列数上限，不等于实际批量
客户端并发上限	不设置	请求进入服务端队列，由推理引擎调度
预热请求数	8	最大输出 16 词元，并发数为 1
随机种子	48105	固定数据抽样

驱动端不设置并发上限，请求由推理引擎按序列调度上限分批处理。SGLang 采用与 vLLM 功能对应的配置。统一测试点用于比较相同资源条件下的实际部署结果，不代表各模型的最佳配置。

模型因架构、量化方式和推理引擎产生的差异，以实际配置为准。

测试采用的 thinking 设置与 Token 统计口径见附录 B 表 B-1，作为单位输出能耗结果的解释依据。推理引擎版本见表 B-3。

3.4 测试流程

每项测试均按照统一流程执行：

测试准备：检查模型 ID、Revision、精度、模型快照、推理引擎、镜像、并行配置和请求集，确认 GPU、存储、遗留进程及容器状态正常。

启动生命周期监测：测试前采集 20 秒空载功率，用于确认 GPU 初始状态。启动阶段采用驱动层 GPU 板卡功率遥测，以约 1 秒周期采集每卡瞬时功率。测量范围自首条有效采样至首个正式请求发出，包含服务启动、模型加载、服务就绪和预热。阶段边界处的功率采用线性插值，各卡能耗按复合梯形公式积分后求和。

启动服务并核验运行状态：按照冻结配置启动推理服务并持续检查运行状态。确认服务就绪后，再次核对模型身份、Revision、推理引擎、上下文长度及并行配置，确保实际运行环境与测试计划一致。

预热运行：测试采用推理服务冷启动，模型权重预先保存为固定本地快照，每次测试新建推理实例，使用独立的引擎运行目录和编译缓存目录，操作系统文件页缓存保持原有状态。启动后执行 8 个非正式请求，触发首次请求处理、JIT 及 Kernel 编译和 Tokenizer 处理。预热请求不计入正式性能指标；权重加载、编译和预热过程中的 GPU 能耗均计入启动阶段，网络下载不计入启动能耗。

正式请求：驱动端以不限制请求速率和客户端并发数的方式提交全部请求；推理服务按照 max-num-seqs=128 定义的最大并发序列数进行调度，驱动端在途请求数可以高于该值。测试过程中逐条记录请求是否成功、输入和输出 Token 数、首 Token 延迟（TTFT）、单 Token 生成时间（TPOT）、相邻 Token 间隔（ITL）及端到端延迟，并统计请求吞吐和输出 Token 吞吐。

稳定态测量：正式请求阶段的 GPU 能耗由 Zeus 基于 NVML 进行测量。Prometheus 同步采集推理引擎/metrics 接口提供的队列、KV Cache 和调度状态，仅用于运行状态分析，不参与稳态边界判定和 GPU 能耗计算。

服务关闭与资源释放：最后一个正式请求完成后，停止推理服务，并继续采集板卡功率和显存占用。资源释放阶段以最后一个正式请求完成为起点，以进程退出、显存释放且 GPU 功率恢复至空载基线允许范围为终点。释放能耗沿用启动阶段的边界处理与积分方法；未在规定时间内完成资源恢复的运行不纳入正式结果。

结果校验与归档：全部请求完成且能耗窗口关闭后，框架生成标准结果文件，并保存运行配置、生命周期数据、GPU 遥测、请求记录和运行日志，用于结果校验、汇总和追溯。

Benchmark 1.5 将测试边界由 Benchmark 1.0 的运行与稳定阶段扩展至完整服务生命周期：

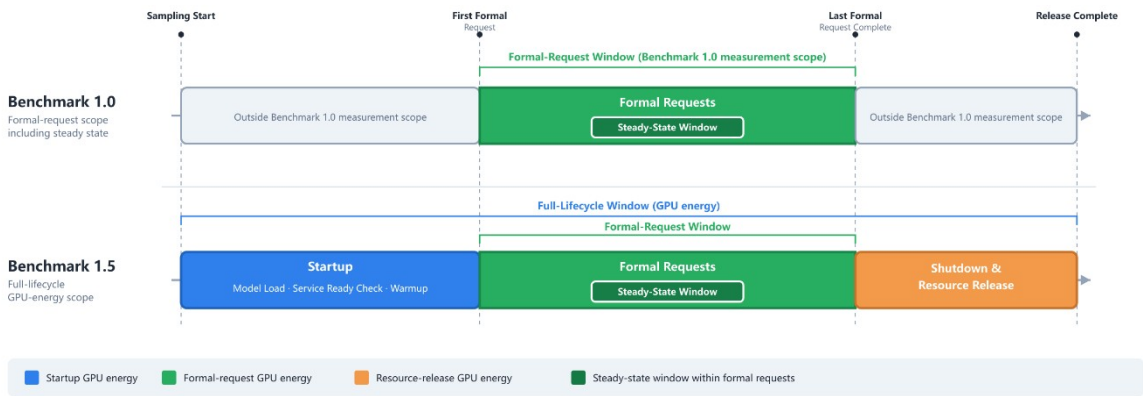


图 3-1 Benchmark 1.0 与 Benchmark 1.5 测量生命周期对比

全生命周期 GPU 能耗由启动、正式请求和资源释放三个阶段组成。正式请求阶段和稳定态能耗由 Zeus/NVML 测量；启动和资源释放阶段采用约 1 秒一次的 GPU 功率采样，按相邻采样点的时间间隔做梯形积分。各阶段的测量边界如下：启动阶段自 GPU 连续采样开始至正式请求开始，覆盖模型加载、服务就绪和预热；正式请求阶段自首个正式请求发出至最后一个正式请求完成；资源释放阶段自正式请求结束至推理进程退出、显存释放且 GPU 功率恢复至空载基线允许范围。全生命周期能耗采用各阶段的实际测量值，不扣除空载基线功率；测试前后的空载基线仅用于确认 GPU 空闲状态和测试结束后的资源恢复状态。空载基线允许范围定义为：各卡功率回落至测试前基线值的 $\pm 5\%$ 以内并稳定维持不少于 30 秒，即判定资源释放完成；若超过设定的最长等待时间仍未回落至该范围，则按超时记录并标注该次运行的释放阶段区间未闭合。Prometheus 仅用于解释推理引擎运行状态，不作为能耗结算来源。

尽管总体流程保持一致，各模型因架构、量化制品和推理引擎不同，存在少量特殊处理，具体见表 3-4。

表 3-4 各模型在测试中的特殊处理

模型	推理引擎	主要适配
GLM-5.3-Flash	特定开发版 vLLM；SGLang	特殊推理引擎版本和镜像身份
MiniMax-M3-MXFP8	vLLM	使用 MXFP8 模型制品
MiMo-V2.5	SGLang	首轮正式测试采用 SGLang 路径
gpt-oss-120b	vLLM；SGLang	使用 MXFP4 模型制品
Gemma-4-31B-it	vLLM	使用 BF16 制品
Sarashina2.2-3B-Instruct-v0.1	vLLM；SGLang	使用官方 3B Dense BF16 模型制品

每次测试完成后，均需核验正式请求的完成情况、实际运行配置、生命周期测量边界和 GPU 遥测数据，并确认推理进程已退出、显存已释放且 GPU 功率已恢复。全部检查通过后，测试结果方可

纳入正式结果；未通过校验的测试保留原始记录，重新测试时另行生成独立记录，不覆盖已有数据。

为便于复核稳态判定，每次正式运行均记录模型名称、测试任务、推理引擎和稳态判定结果。稳态成立时，保存稳态窗口的起止时间及对应的起止请求序号，同时记录窗口内的 GPU 能耗 (J) 和输出 Token 数。稳定态单位输出能耗 (J/token) 的计算结果一并归档。

正式请求阶段的输出 Token 总数一并保存，并注明其中用于计算全生命周期单位输出能耗的 Token 数。资源释放阶段 GPU 功率是否恢复至空载基线允许范围的判定结果同步归档。

稳定态单位输出能耗仅对稳态成立的运行计算并用于跨模型比较。稳态未成立的运行在结果表中标注并说明原因，不纳入该指标的比较。

第四章 测试结果

本章分析 Benchmark 1.5 的 48 条正式测试结果。首轮横向测试覆盖 13 个模型，共形成 33 条结果，以 vLLM 为主要推理引擎，MiMo-V2.5 采用 SGLang。随后，选取其中 6 个模型开展 SGLang 补充测试，形成 15 条结果。本章基于首轮结果，比较各模型在文本聊天、代码补全和图文问答场景下的能耗与吞吐率，分析生命周期能耗构成和典型模型表现，并结合补充测试结果，比较 vLLM 与 SGLang 的部署表现。

本章按模型总参数量划分 $\leq 50\text{B}$ 、 $>50\text{B}$ 且 $<200\text{B}$ 和 $\geq 200\text{B}$ 三个规模组，在相同任务场景和评价指标下进行组内比较。参数规模用于分组，不表示计算量相同。各模型结果基于对应部署配置下的单次测试；组内有多个适用模型时按指标排序，只有一个时列示其数值。

4.1 能耗对比

4.1.1 不同场景稳定态能耗对比

我们在统一 TP8、序列调度上限 128 测试点下，横向比较各模型在文本聊天、代码补全和图文问答场景中的稳定态单位输出能耗。该指标数值越小，表示生成单位输出所需的 GPU 能耗越低。结果见图 4-1（a）至图 4-1（c）。

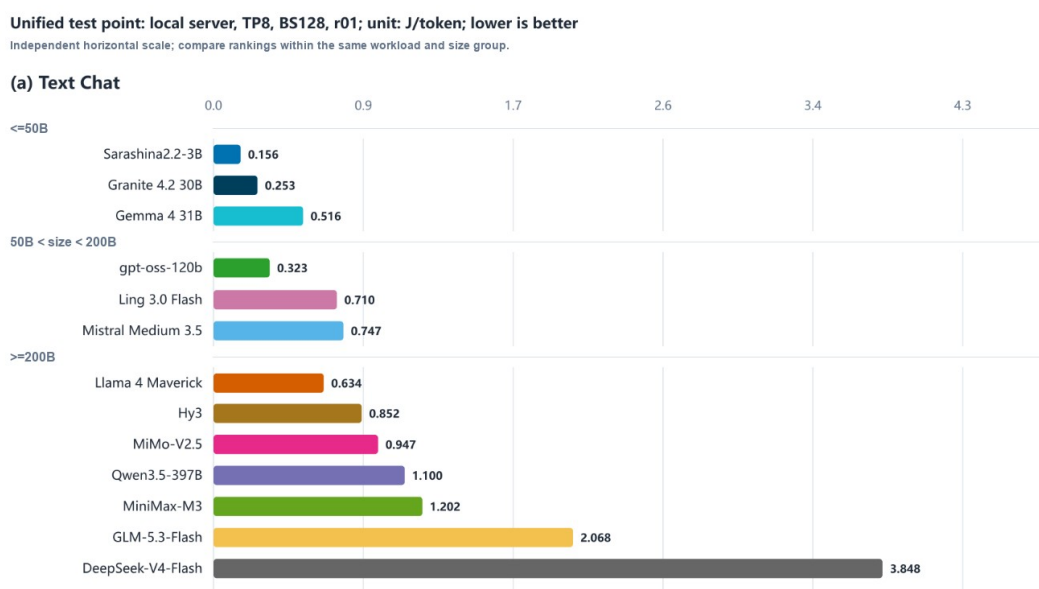


图 4-1（a） 文本聊天场景稳定态单位输出能耗

文本聊天场景共纳入 13 个模型，稳定态单位输出能耗的总体中位数为 0.747 J/token。在轻量级、中等规模和大规模三个组内，Sarashina 2.2、gpt-oss-120b 和 Llama 4 Maverick 的稳定态单位输出能耗分别低于各自组内其他参测模型，对应数值为 0.156、0.323 和 0.634 J/token。

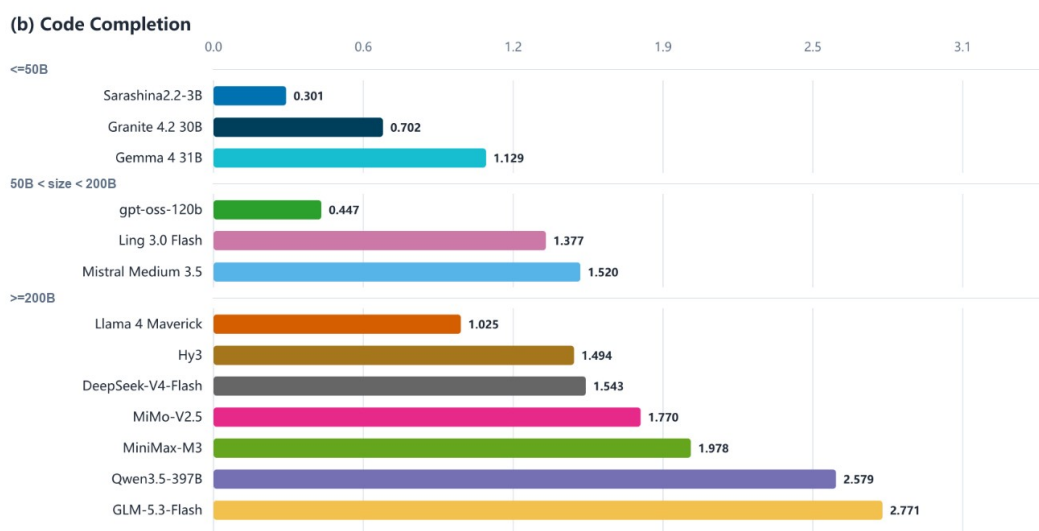


图 4-1 (b) 代码补全场景稳定态单位输出能耗

代码补全场景共纳入 13 个模型，稳定态单位输出能耗的总体中位数为 1.494 J/token。在轻量级、中等规模和大规模三个组内，Sarashina 2.2、gpt-oss-120b 和 Llama 4 Maverick 的稳定态单位输出能耗分别低于各自组内其他参测模型，对应数值为 0.301、0.447 和 1.025 J/token。

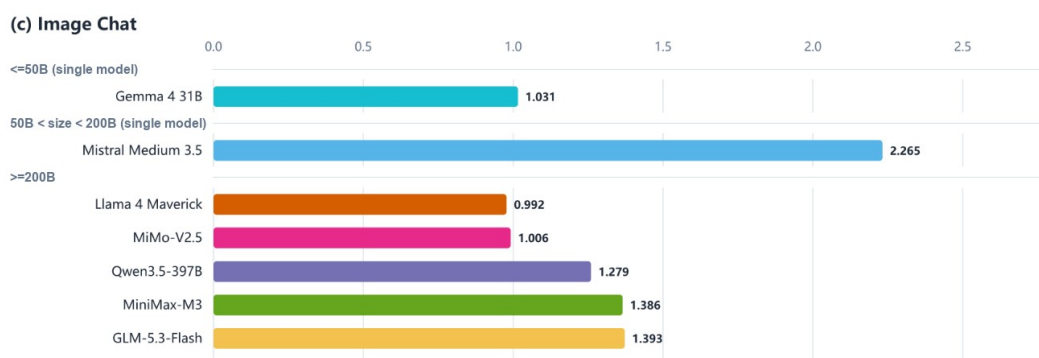


图 4-1 (c) 图文问答场景稳定态单位输出能耗

图文问答场景共纳入 7 个模型，稳定态单位输出能耗的总体中位数为 1.279 J/token。大规模组中，Llama 4 Maverick、MiMo-V2.5 和 Qwen3.5 的稳定态单位输出能耗分别为 0.992、1.006 和 1.279 J/token，均低于该组其余参测模型。轻量级组与中等规模组各有一个参测模型，分别为 Gemma 4 31B 和 Mistral Medium 3.5，其稳定态单位输出能耗分别为 1.031 和 2.265 J/token。

跨场景比较

为保持参测范围一致，跨场景比较选取同时完成三类场景测试的 7 个模型。文本聊天、图文问答和代码补全的稳定态单位输出能耗中位数分别为 0.947、1.279 和 1.770 J/token，图文问答和代码补全分别比文本聊天高约 35%和 87%。其中，5 个模型各自在文本聊天、图文问答和代码补全场景下的稳定态单位输出能耗依次增加。三类场景的负载有所不同：文本聊天平均输入约 580 Token，代码补全平均输入约 2k Token，图文问答涉及图像编码。

在 7 个共同参测模型中，Llama 4 Maverick 的跨场景能耗差异相对较小：其三类场景的稳定态单位输出能耗最大值与最小值之比为 1.62，低于其余 6 个模型。MiMo-V2.5 的图文问答能耗接近文本聊天，仅高约 6.3%，而代码补全比文本聊天高约 87%，其跨场景差异主要体现在代码补全任务上。

GLM-5.3-Flash 和 Mistral Medium 3.5 的能耗变化顺序与前述 5 个模型不同。GLM 的图文问答能耗低于文本聊天；Mistral 的图文问答能耗则高于代码补全。因此，三类场景的总体能耗顺序并不适用于每个模型，具体部署仍需查看目标模型在相应任务下的结果。

面向多任务部署，应在相同任务场景和测试配置下比较模型，并结合各任务的稳定态单位输出能耗、吞吐率、时延和全生命周期单位输出能耗进行选型。

4.1.2 不同场景全生命周期能耗热力图对比

图 4-2 至图 4-4 分别给出了文本聊天、代码补全和图文问答场景下的能耗热力图。Benchmark 1.5 首轮测试统一采用 TP8、序列调度上限 128 的测试点，图中并列展示各模型的稳定态与全生命周期单位输出能耗。各图按模型规模分组，组内按全生命周期单位输出能耗由低到高排列，两列数值分别用于比较对应规模组内的稳定态与全生命周期单位输出能耗。

为便于比较，本节使用“量级比”和 Spearman 等级相关系数两个统计量。量级比为同一场景全生命周期 J/token 中位数与稳定态 J/token 中位数之比；数值越大，说明计入启动和资源释放后，两种口径的整体差距越大。Spearman 等级相关系数用于比较两种口径下的模型排序，取值范围为 -1 至 1；数值越接近 1，排序越一致。例如，0.74 表示两种排序总体较为一致，但部分模型的名次仍会变化。

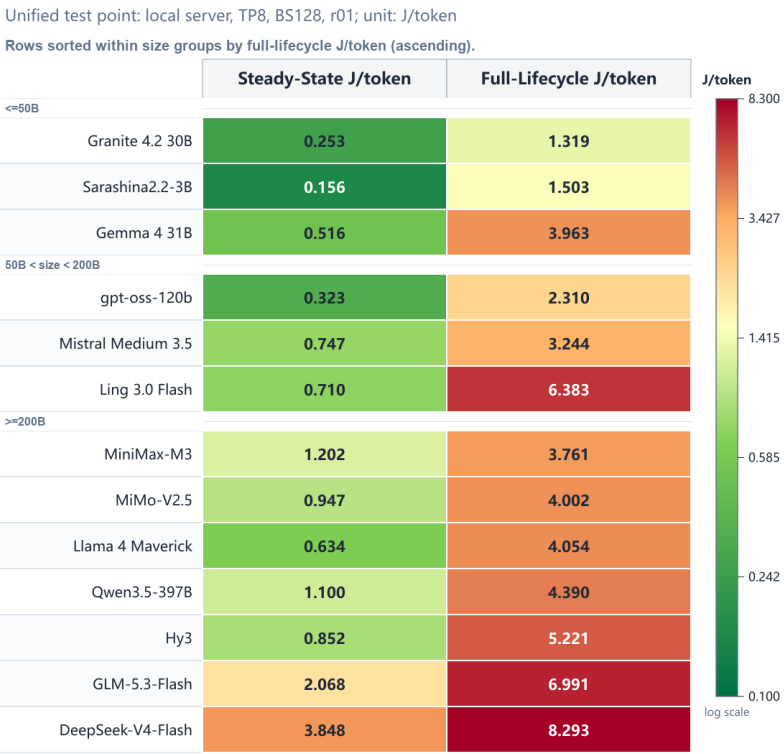


图 4-2 文本聊天场景能耗热力图

文本聊天场景的稳定态与全生命周期单位输出能耗中位数分别为 0.747 和 4.002 J/token，量级比为 5.36。两种口径下的模型排序的 Spearman 等级相关系数约为 0.74，整体较为接近，但部分模型之间的能耗高低关系发生变化。例如，Sarashina 的稳定态单位输出能耗低于 Granite，全生命周期单位输出能耗则高于 Granite；Ling 的稳定态单位输出能耗低于 MiniMax，全生命周期单位输出能耗则高于 MiniMax。

本次文本聊天测试的正式运行时间较短，启动和资源释放开销在全生命周期单位输出能耗中占比较高。稳定态指标衡量服务稳定运行时生成单位输出所需的 GPU 能耗；全生命周期指标则将启动、正式运行和资源释放阶段的 GPU 总能耗分摊到正式请求阶段的输出 Token，衡量一次完整服务的单位输出能耗。

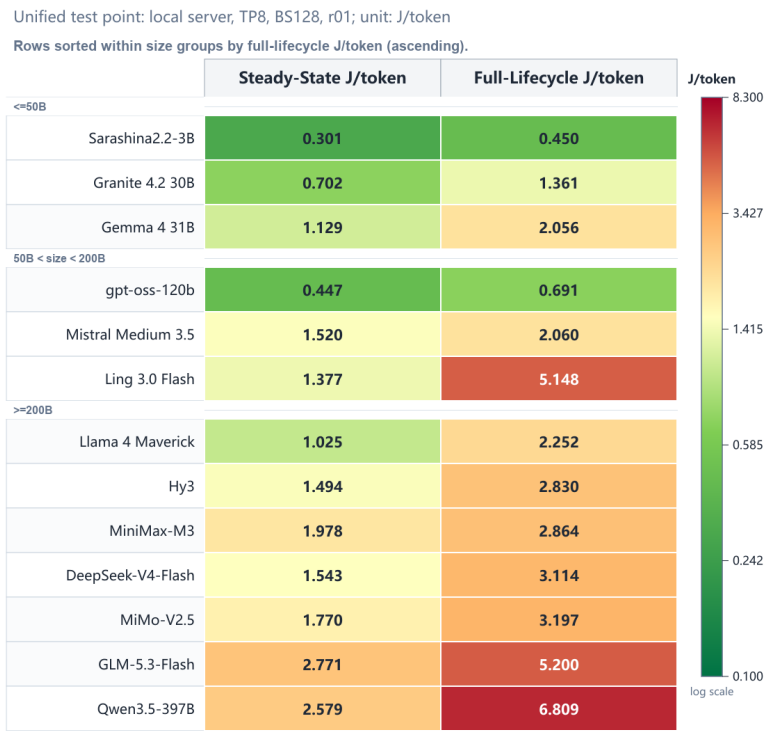


图 4-3 代码补全场景能耗热力图

代码补全场景的稳定态与全生命周期单位输出能耗中位数分别为 1.494 和 2.830 J/token，量级比为 1.89。两种口径下的模型排序的 Spearman 等级相关系数约为 0.86，表明排序总体较为一致。与文本聊天相比，代码补全的稳定态排序与全生命周期排序更为接近。

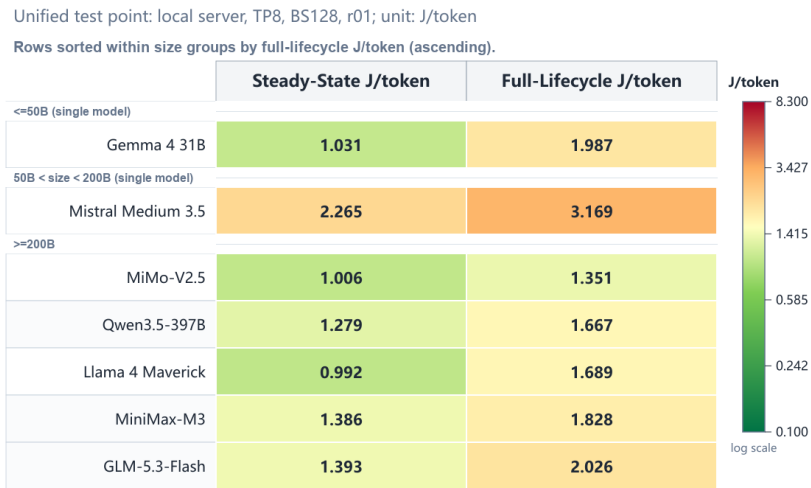


图 4-4 图文问答场景能耗热力图

图文问答场景的稳定态与全生命周期单位输出能耗中位数分别为 1.279 和 1.828 J/token，量级比为 1.43，在三个场景中最小。两种口径下的模型排序的 Spearman 等级相关系数约为 0.75。稳定态口径下，Llama 4 Maverick 的单位输出能耗低于 MiMo-V2.5 和 Qwen3.5；全生命周期口径下，MiMo-

V2.5 和 Qwen3.5 的单位输出能耗均低于 Llama 4 Maverick。两种口径的整体数值差距较小，部分模型之间的能耗高低关系仍发生变化。

跨场景比较

能耗数值方面，文本聊天、代码补全和图文问答的全生命周期单位输出能耗中位数，分别为相应稳定态中位数的 5.36、1.89 和 1.43 倍。因此，文本聊天中两种口径的数值差距较大，代码补全次之，图文问答相对较小。

总体排序相关性方面，基于各场景的全部参测模型，比较稳定态与全生命周期单位输出能耗排序的一致性。代码补全场景的 Spearman 等级相关系数约为 0.86，高于文本聊天的 0.74 和图文问答的 0.75，表明代码补全场景中，按稳定态与全生命周期单位输出能耗得到的模型排序更为接近。

Benchmark 1.5 在稳定态测量的基础上，增加覆盖启动、正式运行和资源释放阶段的全生命周期测量，为不同服务方式提供相应的能耗评估依据。长期常驻、连续处理请求的服务，可重点参考稳定态 J/token，评估稳定运行期间的单位输出能耗；涉及冷启动、短时运行或弹性扩缩容的服务，可结合全生命周期 J/token，评估包含启动和资源释放开销的完整服务能耗。

4.2 生命周期能耗结构

图 4-5 展示 13 个模型在文本聊天场景下的启动、正式运行和资源释放能耗占比。各模型均采用横向测试的主结果，其中 MiMo-V2.5 采用 SGLang 测试结果，其余 12 个模型采用 vLLM 测试结果。各项测试的启动、正式运行、资源释放及全生命周期绝对 GPU 能耗见附录 B 表 B-2，单位为 kJ。

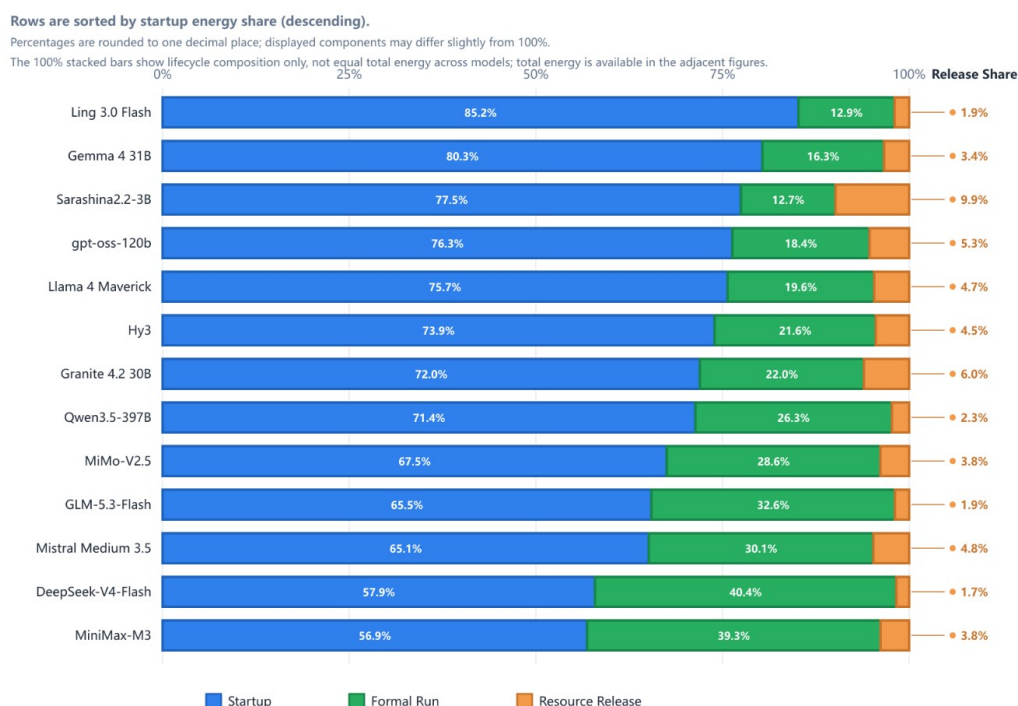


图 4-5 文本聊天场景典型模型的生命周期能耗结构

在本次文本聊天测试中，13 个模型的启动能耗均占全生命周期 GPU 能耗的 50% 以上，其中启动能耗占比最小的 MiniMax 为 56.87%。启动、正式运行和资源释放阶段的能耗占比中位数分别为 71.97%、21.98% 和 3.85%。在当前请求规模下，启动阶段是各模型完整服务能耗的主要组成部分。

各模型的阶段能耗构成存在差异。Ling、Gemma 和 Sarashina 的启动能耗占比分别为 85.18%、80.34% 和 77.46%，正式运行占比分别为 12.88%、16.30% 和 12.68%；DeepSeek 和 MiniMax 的正式运行占比分别为 40.37% 和 39.30%，启动占比分别为 57.92% 和 56.87%。这些占比反映本次模型制品、推理引擎和测试负载下的实际部署结果。部署优化中，减少重复启动、提高实例复用率，有助于降低累计启动能耗。对于 DeepSeek 和 MiniMax 等正式运行能耗占比较大的方案，还应关注请求处理阶段的能耗，并结合稳定态单位输出能耗和吞吐率评估优化效果。

资源释放阶段的能耗占比整体较小，但模型间存在差异。Sarashina 的资源释放能耗占比为 9.86%，在本次 13 个模型中占比最高，部分模型则低于 2%。在频繁创建和销毁实例的服务中，关闭服务、进程退出和显存释放等环节的能耗会随实例启停次数累积，因此资源释放过程也应纳入部署评估。

Sarashina 的启动能耗占比为 77.46%，高于 DeepSeek 的 57.92%，但其全生命周期单位输出能耗为 1.503 J/token，低于 DeepSeek 的 8.293 J/token。阶段能耗占比的分母是各模型自身的 GPU 总能耗，反映的是该模型内部的能耗分布。评估部署方案时，全生命周期 J/token 用于比较完整服务的单位输出能耗，各阶段的能耗占比用于分析主要能耗来源，为确定优化重点提供参考。

4.3 吞吐率对比

图 4-6（a）至图 4-6（c）分别给出了不同模型在文本聊天、代码补全和图文问答场景下的输出吞吐率。各图按规模组分块，组内按吞吐率由高到低排列。

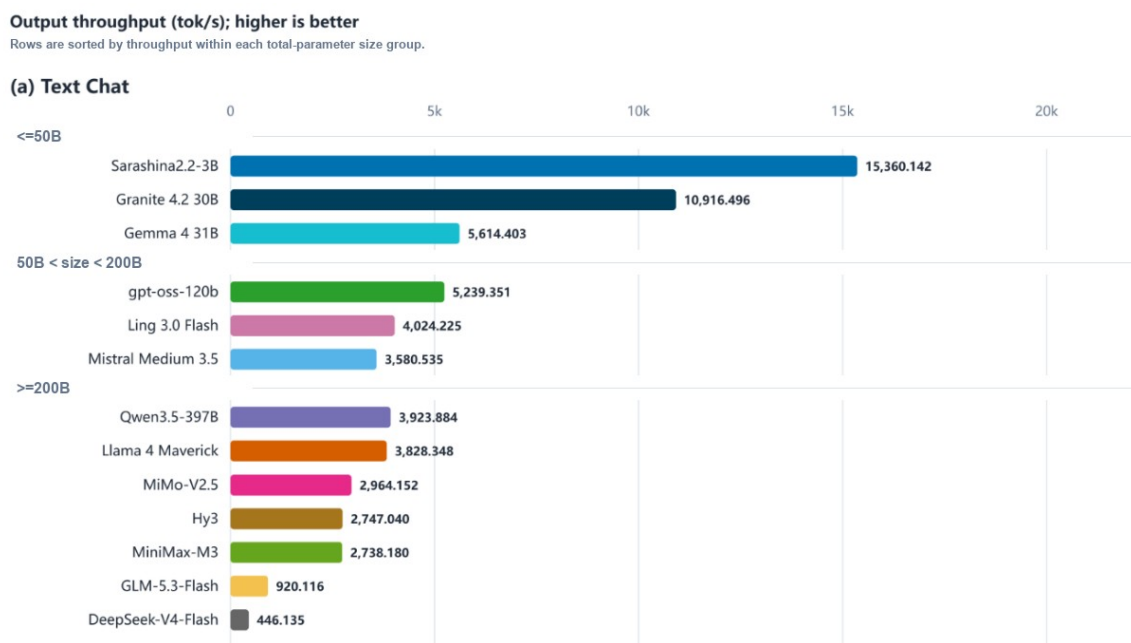


图 4-6（a） 文本聊天场景输出吞吐率

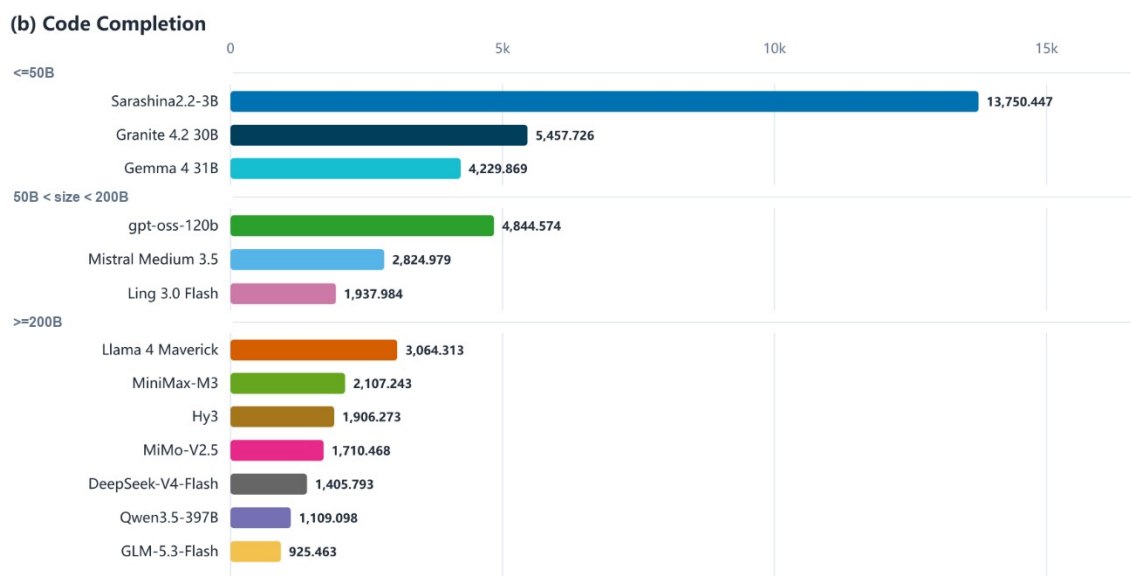


图 4-6（b） 代码补全场景输出吞吐率

文本聊天与代码补全的吞吐特征

文本聊天与代码补全的模型吞吐率排序较为一致，13 个模型的 Spearman 等级相关系数约为 0.79。在本次测试配置下，Sarashina 2.2 的文本聊天和代码补全输出吞吐率分别为 15,360 和 13,750 tok/s，均为相应场景轻量级组中的最高值。

Qwen3.5 与 MiniMax-M3 的吞吐表现随任务变化。文本聊天中，Qwen3.5 的输出吞吐率为 3,924 tok/s，高于 MiniMax-M3 的 2,738 tok/s；代码补全中，MiniMax-M3 的输出吞吐率为 2,107 tok/s，高于 Qwen3.5 的 1,109 tok/s。

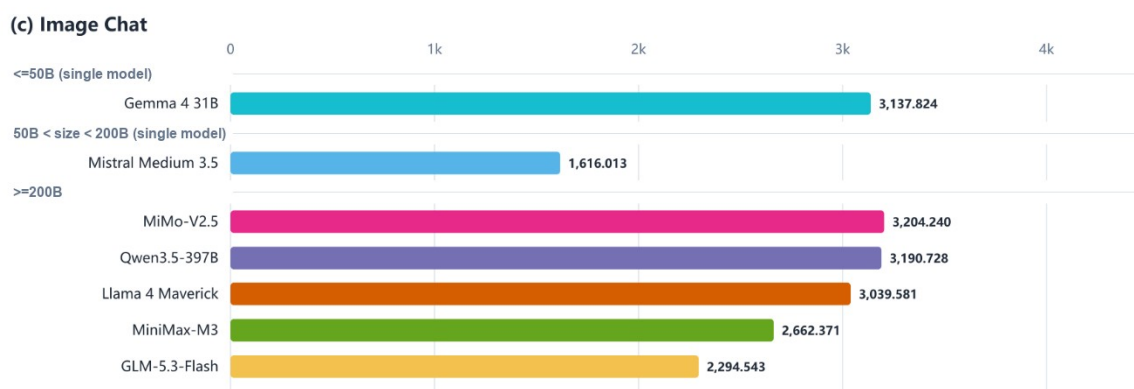


图 4-6 (c) 图文问答场景输出吞吐率

在同时完成三类任务测试的 7 个模型中，文本聊天与图文问答吞吐率的 Spearman 等级相关系数约为 0.43，代码补全与图文问答约为 -0.04。部分模型之间的吞吐率高低关系随场景变化。例如，文本聊天中，Mistral Medium 3.5 的输出吞吐率为 3,581 tok/s，高于 GLM-5.3-Flash 的 920 tok/s；图文问答中，GLM-5.3-Flash 的输出吞吐率为 2,295 tok/s，高于 Mistral Medium 3.5 的 1,616 tok/s。

图文问答中，MiMo-V2.5、Qwen3.5、Gemma 4 31B 和 Llama 4 Maverick 的输出吞吐率较为接近，介于 3,040 至 3,204 tok/s。其中，MiMo-V2.5 的输出吞吐率比 Llama 4 Maverick 高约 5.4%。

部署选型时，依据目标任务下的实测吞吐率评估模型的输出能力，并结合稳定态单位输出能耗、全生命周期单位输出能耗、首 Token 延迟（TTFT）及目标任务的质量要求进行综合评估。本节吞吐率结果对应本次模型制品、推理引擎和统一 TP8 测试配置。

4.4 典型模型分析

4.4.1 Gemma 4 31B 的跨场景部署表现

为分析同一模型在不同工作负载下的部署差异，本节选取完成三类场景测试的 Gemma 4 31B 作为典型案例。该模型在自身三类场景中，稳定态单位输出能耗与全生命周期单位输出能耗呈现不同的高低关系，适合用于说明固定启动成本对部署能效的影响。

Gemma 4 31B 采用 BF16 制品、vLLM 0.29.0、TP8、序列调度上限 128 和 32K 上下文。图 4-7 比较该模型在三类场景下的吞吐率、稳定态能耗和全生命周期能耗。

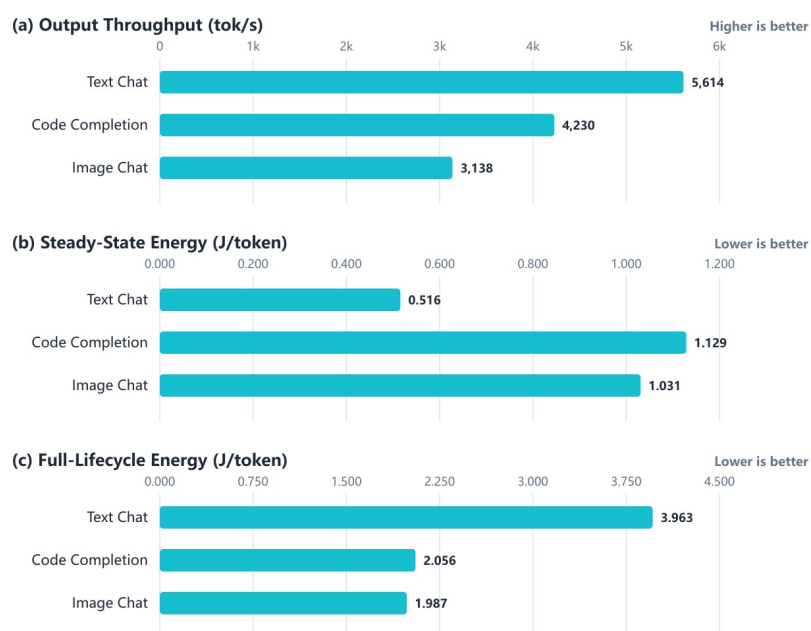


图 4-7 Gemma 4 31B 在三类场景下的吞吐率、稳定态及全生命周期单位输出能耗

在 Gemma 自身的三类场景中，文本聊天的吞吐率最高，为 5,614 tok/s，稳定态单位输出能耗最低，为 0.516 J/token，但全生命周期单位输出能耗最高，为 3.963 J/token。代码补全和图文问答的吞吐率分别为 4,230 和 3,138 tok/s，稳定态单位输出能耗分别为 1.129 和 1.031 J/token，全生命周期单位输出能耗分别为 2.056 和 1.987 J/token。上述高低关系仅指 Gemma 三类场景之间的比较。

这种差异主要来自固定启动成本的分摊。三个场景的启动时长接近，均为 362~369 秒；文本聊天的正式运行时长仅 24.0 秒，明显短于代码补全的 113.8 秒和图文问答的 169.2 秒，输出 Token 数也更少。因此，固定启动能耗分摊到每个输出 Token 后，文本聊天的全生命周期单位输出能耗反而较高。常驻服务的持续运行成本可主要依据稳定态结果评估；频繁启停或短时运行的服务还需结合全生命周期结果。

4.4.2 多模态模型对比

7 个图文模型的吞吐中位数为 3,040 tok/s，稳定态和全生命周期能耗中位数分别为 1.279 和 1.828 J/token。吞吐前四名集中在 3,040 至 3,204 tok/s 之间，但能耗差异仍然明显，说明当前图文场景已出现“性能接近、成本分化”的特征。对于这类结果，吞吐率更适合作为准入条件，而能耗指标更适合进一步区分部署方案。

同时考虑吞吐率和稳定态单位输出能耗时，Llama 4 Maverick 和 MiMo-V2.5 的表现较为突出。Llama 以 0.992 J/token 取得大规模组内最低稳定态能耗；MiMo 的稳定态能耗为 1.006 J/token，仅高 1.4%，但吞吐率达到 3,204 tok/s，比 Llama 高 5.4%。在本次结果中，Llama 的优势体现在更低的持续运行能耗，MiMo 的优势体现在能耗接近时的更高输出吞吐率。

进一步比较吞吐率和全生命周期单位输出能耗，MiMo 在本次测试的大规模组中同时取得最高吞吐率和最低全生命周期单位输出能耗。其全生命周期能耗为 1.351 J/token，比 Llama 的 1.689 J/token 低约 20.0%。Qwen 的吞吐率为 3,191 tok/s，与 MiMo 仅相差约 0.4%，但稳定态和全生命周期能耗分别高约 27.1%和 23.4%。

跨场景观察则显示，Llama 的优势更偏向一致性。其文本聊天、代码补全和图文问答稳定态能耗分布在 0.634 至 1.025 J/token 之间，最大值约为最小值的 1.62 倍；MiMo 对应范围为 0.947 至 1.770 J/token，最大值约为最小值的 1.87 倍，并在代码补全中出现更明显上升。因此，若业务只面向当前图文问答负载且重视弹性服务成本，MiMo 的当前部署方案更具优势；若需要同一模型覆盖文本、代码和图文任务，Llama 较稳定的跨场景表现具有额外价值。上述判断仅对应本次模型制品、推理引擎、TP8、序列调度上限 128 和工作负载，不涉及回答质量，也不能将差异单独归因于模型架构。

4.5 不同推理引擎下的部署表现

vLLM。采用 PagedAttention，将注意力键值缓存（KV Cache）划分为按需分配的内存块，减少显存碎片和过度预留；通过连续批处理，在请求执行过程中动态调整批次。其前缀缓存利用缓存块的哈希匹配，复用相同输入前缀的计算结果。

SGLang。同样支持连续批处理和前缀缓存。其 RadixAttention 通过基数树组织请求前缀及对应的 KV Cache，识别并复用共享前缀的缓存；重叠调度机制使 CPU 准备后续批次与 GPU 执行当前批次并行进行，减少调度等待。

两套引擎在缓存组织、请求调度、显存配置、算子适配和输出解析等方面存在实现差异。本节以各模型实际使用的推理引擎版本和运行配置为基础，比较两套部署方案在吞吐率、时延和能耗方面的综合表现。

为兼顾服务能力、用户响应、稳定运行效率和全生命周期单位输出能耗，本节选取输出吞吐率、TTFT P50、稳定态 J/token 和全生命周期 J/token 四个指标。本次共形成 15 条 SGLang 正式结果，其中 14 条可与同模型、同任务的 vLLM 结果配对，另 1 条为 DeepSeek 关闭推理模式的补充测试，不纳入主对比。14 组配对中，图 4-8 仅纳入两种引擎输出 Token 数相对差异不超过 20%的 11 组配对。

在 DeepSeek 代码补全、GLM-5.3-Flash 图文问答和 Granite 4.2 代码补全测试中，vLLM 与 SGLang 的输出 Token 数相对差异均超过 20%。相应的引擎对比结果未纳入图 4-8 及本节的部署表现分析。其中，GLM-5.3-Flash 图文问答还存在输出解析路径差异。

此处输出 Token 数的相对差异定义为两者输出 Token 总数之差的绝对值与较大值之比，计算方法见式 (4)。

$$\delta_N = \frac{|N_{\text{vLLM}} - N_{\text{SGLang}}|}{\max[N_{\text{vLLM}}, N_{\text{SGLang}}]}, \delta_N \leq 0.20 \quad (4)$$

其中， N_{vLLM} 和 N_{SGLang} 分别表示同一模型、同一任务下，两套部署方案正式请求阶段的输出 Token 总数。

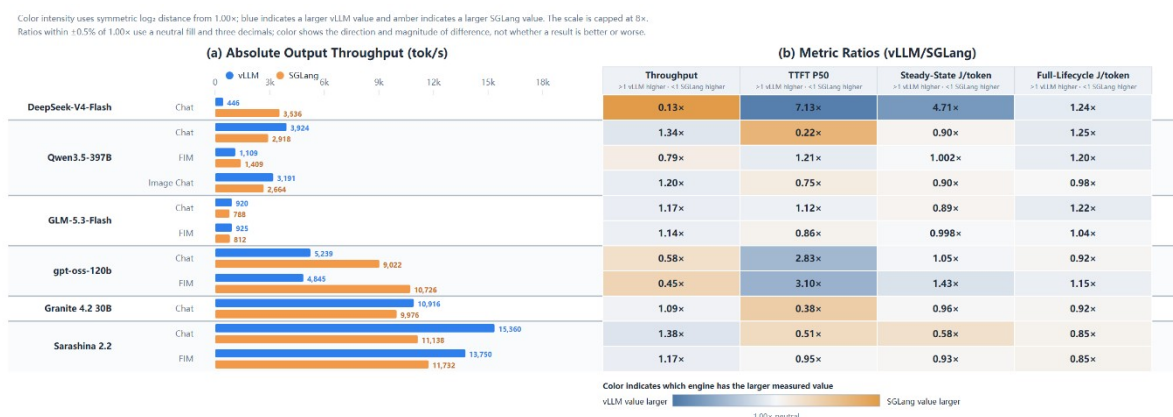


图 4-8 vLLM 与 SGLang 关键指标变化

注：图中所有倍数均以 vLLM 实测值除以 SGLang 对应实测值计算。倍数大于 1 表示 vLLM 的实测值更高，小于 1 表示 SGLang 的实测值更高；具体优劣需结合指标方向判断。接近 1.00 倍表示两套部署方案的该项实测结果接近。

DeepSeek-V4-Flash-0731 文本聊天是 11 组配对中吞吐率相对差异最大的一组。vLLM 和 SGLang 的吞吐率分别为 446 和 3,536 tok/s，SGLang 约为 vLLM 的 7.9 倍，首 Token 延迟中位数约低 86%。SGLang 的单位输出能耗优势在完整服务中有所缩小：稳定态单位输出能耗较 vLLM 低约 79%，按启动、正式运行和资源释放全过程计算，全生命周期单位输出能耗低约 19%。

其余配对中，gpt-oss-120b 代码补全的 SGLang 吞吐率为 10,726 tok/s，高于 vLLM 的 4,845 tok/s，同时 TTFT P50 和两项单位输出能耗均更低。Qwen3.5 图文问答、Granite 4.2 文本聊天及 Sarashina 2.2 的两类文本任务，则均由 vLLM 取得更高吞吐率、更低 TTFT P50 和更低单位输出能耗。同一模型在不同任务中的差异幅度也有所不同：Sarashina 文本聊天中，vLLM 的稳定态单位输出能耗较 SGLang 低约 42%，代码补全中则低约 7%。

Qwen3.5 和 GLM-5.3-Flash 代码补全的稳定态单位输出能耗比值分别为 1.002 和 0.998，均基本持平；全生命周期单位输出能耗均以 SGLang 较低。吞吐率和 TTFT P50 方面，Qwen3.5 的 SGLang 方案表现较好，GLM-5.3-Flash 的 vLLM 方案表现较好。

文本聊天中，Qwen3.5 采用 vLLM 时吞吐率更高、TTFT P50 和稳定态单位输出能耗更低，全生命周期单位输出能耗则比 SGLang 高 25%。gpt-oss-120b 在前三项指标上以 SGLang 较好，但 vLLM 的全生命周期单位输出能耗约低 8%。GLM-5.3-Flash 的 vLLM 方案吞吐率更高、稳定态单位输出能耗更低，SGLang 方案的 TTFT P50 和全生命周期单位输出能耗更低。

本次配对结果表明，vLLM 与 SGLang 的相对表现随模型和任务变化；部署方案的评价应结合吞吐率、首 Token 延迟中位数，以及稳定态和全生命周期单位输出能耗。

4.6 测试结果总结

综合上述结果，可以得到以下结论：

- 不同任务场景呈现不同的单位输出能耗水平。** 在同时完成三类场景测试的 7 个模型中，文本聊天、图文问答和代码补全的稳定态单位输出能耗中位数分别为 0.947、1.279 和 1.770 J/token，依次升高，但并非所有模型均保持相同的跨场景排序。因此，文本聊天、代码补全和图文问答需要分别比较，不能将一个场景中的结果直接外推至其他任务。
- 稳定态能耗与全生命周期能耗反映不同阶段的能效表现。** 文本聊天、代码补全和图文问答的全生命周期与稳定态能耗中位数量级比分别约为 5.36、1.89 和 1.43，且三类场景中均出现不同程度的模型排序变化。稳定态 J/token 适用于评价服务运行后的持续效率，全生命周期 J/token 适用于评价包含启动、正式运行和资源释放的一次完整服务过程。
- 文本聊天的全生命周期能耗中启动阶段占比较高。** 13 个模型的启动、正式运行和资源释放占比中位数分别为 71.97%、21.98% 和 3.85%，启动占比最低的模型仍达到 56.87%。不同模型的阶段结构存在差异，生命周期占比用于识别能耗主要集中在哪个阶段，全生命周期 J/token 用于比较完整服务过程的单位输出能耗，两类信息不能相互替代。
- 吞吐率应在同一任务场景内比较，并结合能耗和时延综合判断。** 文本聊天与代码补全的吞吐排序具有较强相关性，而图文问答会明显改变模型的吞吐性能排序。图文问答吞吐前四名集中在 3,040 至 3,204 tok/s 之间，此时稳定态能耗、全生命周期能耗和时延比小幅吞吐差异更具区分意义。
- 模型选择应结合任务场景和评价指标。** 文本聊天和代码补全场景中，轻量级组的 Sarashina 2.2 兼具较高吞吐率和较低稳定态单位输出能耗，但其文本聊天全生命周期单位输出能耗高于 Granite 4.2；中等规模组的 gpt-oss-120b 在吞吐率、稳定态及全生命周期单位输出能耗三项指标上均优于组内其他参测模型。大规模组中，Llama 4 Maverick 在三个场景的稳定态单位输出能耗均低于组内其他参测模型，而 MiMo-V2.5 在图文问答场景中兼具较高吞吐率和较低全生命周期单位输出能耗。
- vLLM 与 SGLang 的实测表现因模型、任务和评价指标而异。** 11 组输出 Token 差异不超过 20% 的 vLLM/SGLang 配对中，既有 SGLang 四项指标同向改善的组合，也有 vLLM 四项指标同时领先的组合，还有吞吐、TTFT、稳定态能耗和全生命周期能耗方向不一致的组合。推理引擎选择需要综合比较输出吞吐率、TTFT、稳定态单位输出能耗和全生命周期单位输出能耗。

第五章 结论与展望

5.1 核心结论

本次测试对 13 个大模型在文本聊天、代码补全和图文问答三类场景下进行了能耗与性能评测。Benchmark 1.5 在延续稳定态评价口径的基础上，将测量范围扩展至服务启动、正式运行和资源释放阶段，参测模型由 8 个扩展至 13 个，其中延续 5 个、更新 3 个、新增 5 个。测试结果表明：

- 1. 全生命周期测量能够更完整地反映模型服务成本。** 稳定态单位输出能耗主要反映持续运行阶段的生成效率，全生命周期单位输出能耗则进一步包含模型加载、服务启动和资源释放成本。文本聊天场景的运行时间较短，图 4-5 显示多数模型的启动能耗占比较高。本节以文本聊天场景说明生命周期能耗结构；代码补全和图文问答的阶段能耗占比尚未逐项统计，不能仅凭全生命周期与稳定态 J/token 之比判断其主要能耗阶段。因此，仅使用稳定态 J/token 难以完整评价冷启动、短任务和弹性服务。
- 2. 不同场景下的高效模型存在明显差异。** 文本聊天和代码补全场景中，轻量级组的 Sarashina 2.2、Granite 4.2 和中等规模组的 gpt-oss-120b 分别形成较好的能耗与吞吐组合；图文问答场景的大规模组中，Llama 4 Maverick 的稳定态单位输出能耗最低，MiMo-V2.5 的吞吐最高、全生命周期单位输出能耗最低。模型选型应结合具体工作负载，不宜使用单一场景结果代替整体表现。
- 3. 模型规模不是决定能效的唯一因素。** Granite 4.2 和 Sarashina 2.2 在统一八卡基线下表现突出，而 Gemma 4 31B 未延续 Benchmark 1.0 中的能效优势。模型制品、量化方式、推理引擎、上下文长度和并行配置均会影响最终结果，小模型也可能因张量并行切分过细而增加通信和同步开销。
- 4. 统一测试配置不代表各模型的最优配置。** Benchmark 1.5 采用 TP8、序列调度上限 128 建立共同测试点，有利于比较相同资源条件下的部署结果，但不能代表各模型的最优配置。Gemma 在 Benchmark 1.0 与 1.5 两轮之间的结果存在差异，但两轮之间模型版本、推理引擎、精度制品、工作负载和并行度可能同时变化，因此不能将跨轮差异直接视为并行度不匹配的证据。并行度与模型规模需要匹配仅为一种可能解释，需通过控制变量实验验证。
- 5. 推理引擎和模型制品是能效评价的重要组成部分。** 首轮横向测试中，MiMo-V2.5 采用 SGLang，其余模型主要采用 vLLM；参测模型覆盖 BF16、FP8 类、MXFP4 和 MXFP8 等精度格式。测试结果反映的是模型、精度、推理引擎和配置组成的实际部署方案，不能将全部差异归因于模型架构。
- 6. 模型范围扩展提高了评测的代表性。** Benchmark 1.5 将参测模型由 8 个增加至 13 个，其中延续类 5 个、更新类 3 个、新增类 5 个，新增路线为 GLM-5.3-Flash、Hy3-FP8、Granite-4.2-30B-FP8、Ling-3.0-Flash-FP8 和 Sarashina2.2-3B-Instruct-v0.1；完成图文问答测试的模型由 3 个增加至 7 个，进一步扩展了不同来源、参数规模和输入模态的模型覆盖范围。

总体而言，Benchmark 1.5 实现了从稳定运行能效比较向完整服务生命周期评价的扩展。模型部署应遵循“场景决定需求、配置影响效率、推理引擎释放能力、生命周期反映成本”的原则，在能耗、吞吐、时延和服务方式之间进行综合权衡。

5.2 未来展望

面向大模型技术和智能计算产业的长期发展，评测体系与产业实践将在以下方向持续深化：

- 1. 全栈评测体系拓展。** 评测范围可从当前模型推理层向上延伸至 Agent、检索增强、工具调用、记忆管理和多模型协同等上层软件栈，形成从底层计算、推理服务到智能应用的端到端评测链路。该方向有助于识别能耗在复杂智能系统中的真实分布，避免仅优化单一模型环节而忽略上层编排和长链路交互成本。
- 2. 模型架构范围拓展。** 当前主流评测对象仍以 Transformer 及其 MoE 变体为主。未来可进一步覆盖状态空间模型、线性注意力、混合递归架构及其他非 Transformer 创新路线，研究不同计算范式在长上下文、稀疏激活和持续推理场景中的能效特征，为新型模型架构的发展提供统一的实证基础。
- 3. 系统能耗与算法能力融合。** 将系统侧的 J/token、吞吐和时延指标与算法侧的准确率、任务完成率、推理质量及安全性指标关联，在能力约束下构建“有效智能产出/单位能耗”等融合评价逻辑。参考学界成熟研究经验，可进一步研究质量—性能—能耗三维前沿，避免以降低输出质量为代价获得表面能效优势。
- 4. 异构算力与能源系统协同。** 研究范围可由单一服务器拓展至不同 GPU、NPU、CPU 及国产加速器，并进一步关联数据中心 PUE、可再生能源利用、碳排放因子和电力负荷特征。该方向能够将模型能效研究与绿色数据中心、算力调度和新型电力系统协同连接起来，为“算力—电力—碳排”联合优化提供数据基础。
- 5. 标准体系与产业生态建设。** 推动能效指标、测试边界、数据披露和结果解释形成可复现、可互认的行业规范，并逐步服务于云服务定价、企业模型选型、绿色算力认证和政策评估。公开透明的评测体系可促进模型、软件与硬件之间形成持续反馈，推动产业竞争从单纯追求能力规模转向能力、效率与可持续性的协同发展。

5.3 局限性说明

本白皮书存在以下局限，恳请读者与业界同仁批评指正：

- 1. 统计重复次数有限。** 每套配置均完成一次正式测试，尚未形成三次重复后的均值、标准差和统计显著性结论。

2. **统一测试点不代表各模型的最佳配置。** 本次主要采用 TP8、序列调度上限 128。不同模型规模、架构和并行方式可能具有不同的效率最优点，因此结果主要用于共同基线下的部署比较。
3. **部署方案存在异质性。** 各模型使用的精度、模型版本、推理引擎版本和专属配置不同。本次形成 15 条 SGLang 正式结果，其中 14 条可与同模型、同任务的 vLLM 结果配对，另 1 条为补充测试。14 组配对中，11 组纳入图 4-8；DeepSeek FIM、Granite FIM 和 GLM Image Chat 因输出 Token 数差异超过 20%，不用于比较两种推理引擎的相对表现。
4. **能耗边界限于 GPU。** 本白皮书全部能耗数值均为 GPU 能耗口径。当前结果不包含 CPU、内存、存储、网络、制冷和整机供电损耗；启动和资源释放能耗来自 GPU 功率采样积分，不能等同于数据中心墙插能耗。
5. **Token 口径会影响跨模型比较。** 不同 Tokenizer、输出长度和 Reasoning 模式会影响 J/token 及吞吐结果。J/token 及输出吞吐率均按接口返回的 completion Token 数计算，测试采用的 thinking 设置及 Token 统计口径见附录 B 表 B-1。跨模型判断还需结合 J/request、requests/s 和实际任务完成情况。
6. **工作负载和能力评价仍有限。** 当前覆盖文本聊天、代码补全和图文问答三类任务，尚未覆盖 Agent 长程任务、检索增强、视频、多模型协同及专业领域应用；同时，本次测试不直接评价回答准确率和任务质量。

附录 A 参测模型制品与版本标识

本附录列表 2-2 中 13 个参测模型对应的完整权重仓库 ID、固定 Revision 及提交日期。权重仓库 ID 链接指向该 Revision 对应的仓库页面。

参测模型	权重仓库 ID	Revision	提交日期
Qwen3.5-397B-A17B-FP8	Qwen/Qwen3.5-397B-A17B-FP8	9f1f3de9a3a48cfd340fd73ca98c02625b7afb3b	2026-02-26
Llama-4-Maverick-17B-128E-Instruct-FP8	RedHatAI/Llama-4-Maverick-17B-128E-Instruct-FP8	ad751fe44a9a92326ce966d980e325fbc2c832c3	2025-09-22
gpt-oss-120b	openai/gpt-oss-120b	b5c939de8f754692c1647ca79fbf85e8c1e70f8a	2025-08-26
Mistral-Medium-3.5-128B	mistralai/Mistral-Medium-3.5-128B	ed0c85631cf20929c29bc36d7cb37519ed0cdd28	2026-05-04
Gemma-4-31B-it	google/gemma-4-31B-it	439edf5652646a0d1bd8b46bfdc1d3645761a445	2026-04-10
DeepSeek-V4-Flash-0731	deepseek-ai/DeepSeek-V4-Flash-0731	7872f01b1d1fe23eabc4c98b48bfffce5a386062	2026-08-01
MiniMax-M3-MXFP8	MiniMaxAI/MiniMax-M3-MXFP8	c5454eb03678d8710e54a4e0fc681b9f3b4a3dba	2026-07-11
MiMo-V2.5	XiaomiMiMo/MiMo-V2.5	63651580ca774f8504f676040460aed3e1244ac1	2026-07-09
GLM-5.3-Flash	zai-org/GLM-5.3-Flash	eb9eb208eb0d988989d07a6a12d0fdeb5f52574a	2026-09-07
Hy3-FP8	tencent/Hy3-FP8	ecc1d8e194e093f33177f2f0ef7ce8f397b2d68b	2026-07-20
Granite-4.2-30B-FP8	ibm-granite/granite-4.2-30b-fp8	bee6d7463fd7064455f59ac480a3fcdc ae5cb801	2026-09-02
Ling-3.0-Flash-FP8	inclusionAI/Ling-3.0-flash-fp8	88d48c5a46e1b2f609144e83d95b8ac47528b12e	2026-09-04
Sarashina2.2-3B-Instruct-v0.1	sbintuitions/sarashina2.2-3b-instruct-v0.1	4f3626fb1b64b3e97c908e67f27b2d627ba2a999	2026-08-25

注：提交日期为所列 Revision 在权重仓库中的提交日期，按 UTC 日期列示，不代表模型或权重的首次发布日期。FP8、FP4 及 MX 低精度制品中，部分模块可能保留 BF16 或其他精度，具体配置以对应 Revision 的仓库文件为准。

附录 B 逐运行配置与阶段能耗

本附录提供各项测试的配置和阶段能耗，便于查阅正文结果对应的测试条件。48 项测试分为首轮模型测试、SGLang 引擎补充测试，以及 DeepSeek 关闭 thinking 的补充测试。表 B-1 与表 B-2 通过测试编号对应，能耗均为 8 卡 GPU 合计。首轮测试中，MiMo-V2.5 采用 SGLang，其余模型采用 vLLM。

共同缓存条件：测试采用推理服务冷启动，权重为预先准备的固定本地快照，运行与编译缓存目录按测试独立设置，操作系统文件页缓存保持原有状态。启动能耗包含权重加载、编译和预热，不含网络下载。

表 B-1 测试配置与 thinking 设置方式

首轮模型测试（33 项）

对应第四章的模型横向比较。

编号	模型	测试场景	推理引擎	权重精度	thinking 设置方式
0003	Sarashina 2.2	文本聊天	vLLM	BF16	使用默认设置
0004	Sarashina 2.2	代码补全	vLLM	BF16	使用默认设置
0012	Granite 4.2	文本聊天	vLLM	FP8	使用默认设置
0013	Granite 4.2	代码补全	vLLM	FP8	使用默认设置
0035	Gemma 4 31B	文本聊天	vLLM	BF16	使用默认设置
0036	Gemma 4 31B	代码补全	vLLM	BF16	使用默认设置
0034	Gemma 4 31B	图文问答	vLLM	BF16	使用默认设置
0024	gpt-oss-120b	文本聊天	vLLM	MXFP4	使用默认设置
0025	gpt-oss-120b	代码补全	vLLM	MXFP4	使用默认设置
0014	Ling 3.0 Flash	文本聊天	vLLM	FP8 E4M3	使用默认设置
0015	Ling 3.0 Flash	代码补全	vLLM	FP8 E4M3	使用默认设置
0020	Mistral Medium 3.5	文本聊天	vLLM	FP8	使用默认设置
0021	Mistral Medium 3.5	代码补全	vLLM	FP8	使用默认设置
0019	Mistral Medium 3.5	图文问答	vLLM	FP8	使用默认设置
0008	DeepSeek-V4-Flash	文本聊天	vLLM	FP8 + FP4	默认开启
0009	DeepSeek-V4-Flash	代码补全	vLLM	FP8 + FP4	使用默认设置
0040	Hy3	文本聊天	vLLM	FP8	使用默认设置
0041	Hy3	代码补全	vLLM	FP8	使用默认设置
0043	MiMo-V2.5	文本聊天	SGLang	FP8 E4M3	使用默认设置

编号	模型	测试场景	推理引擎	权重精度	thinking 设置方式
0044	MiMo-V2.5	代码补全	SGLang	FP8 E4M3	使用默认设置
0042	MiMo-V2.5	图文问答	SGLang	FP8 E4M3	使用默认设置
0049	GLM-5.3-Flash	文本聊天	vLLM	FP8 E4M3	使用默认设置
0050	GLM-5.3-Flash	代码补全	vLLM	FP8 E4M3	使用默认设置
0048	GLM-5.3-Flash	图文问答	vLLM	FP8 E4M3	使用默认设置
0032	Qwen3.5-397B	文本聊天	vLLM	FP8	使用默认设置
0033	Qwen3.5-397B	代码补全	vLLM	FP8	使用默认设置
0031	Qwen3.5-397B	图文问答	vLLM	FP8	使用默认设置
0038	Llama 4 Maverick	文本聊天	vLLM	FP8	使用默认设置
0039	Llama 4 Maverick	代码补全	vLLM	FP8	使用默认设置
0037	Llama 4 Maverick	图文问答	vLLM	FP8	使用默认设置
0017	MiniMax-M3	文本聊天	vLLM	MXFP8	使用默认设置
0018	MiniMax-M3	代码补全	vLLM	MXFP8	使用默认设置
0016	MiniMax-M3	图文问答	vLLM	MXFP8	使用默认设置

SGLang 引擎补充测试（14 项）

用于与相同模型、相同任务下的 vLLM 测试结果对照。

编号	模型	测试场景	推理引擎	权重精度	thinking 设置方式
0001	Sarashina 2.2	文本聊天	SGLang	BF16	使用默认设置
0002	Sarashina 2.2	代码补全	SGLang	BF16	使用默认设置
0010	Granite 4.2	文本聊天	SGLang	FP8	使用默认设置
0011	Granite 4.2	代码补全	SGLang	FP8	使用默认设置
0022	gpt-oss-120b	文本聊天	SGLang	MXFP4	使用默认设置
0023	gpt-oss-120b	代码补全	SGLang	MXFP4	使用默认设置
0005	DeepSeek-V4-Flash	文本聊天	SGLang	FP8 + FP4	指定开启
0007	DeepSeek-V4-Flash	代码补全	SGLang	FP8 + FP4	使用默认设置
0046	GLM-5.3-Flash	文本聊天	SGLang	FP8 E4M3	使用默认设置
0047	GLM-5.3-Flash	代码补全	SGLang	FP8 E4M3	使用默认设置
0045	GLM-5.3-Flash	图文问答	SGLang	FP8 E4M3	使用默认设置
0029	Qwen3.5-397B	文本聊天	SGLang	FP8	指定开启
0030	Qwen3.5-397B	代码补全	SGLang	FP8	使用默认设置
0028	Qwen3.5-397B	图文问答	SGLang	FP8	使用默认设置

DeepSeek 关闭 thinking 测试（1 项）

单独列示关闭 thinking 后的文本聊天测试结果。

编号	模型	测试场景	推理引擎	权重精度	thinking 设置方式
0006	DeepSeek-V4-Flash	文本聊天	SGLang	FP8 + FP4	指定关闭

注：“使用默认设置”表示测试参数不另行指定 thinking 开关，具体行为由模型及推理引擎决定；“默认开启”依据配置记录标注。J/token 按接口返回的 completion Token 数计算。

表 B-2 各测试阶段 GPU 能耗（kJ）

首轮模型测试（33 项）

编号	模型	测试场景	启动能耗	正式运行 能耗	释放能耗	总能耗
0003	Sarashina 2.2	文本聊天	150.002	24.561	19.096	193.660
0004	Sarashina 2.2	代码补全	147.889	379.138	25.757	552.784
0012	Granite 4.2	文本聊天	209.915	64.119	17.645	291.679
0013	Granite 4.2	代码补全	209.201	273.269	23.723	506.193
0035	Gemma 4 31B	文本聊天	428.682	87.000	17.919	533.601
0036	Gemma 4 31B	代码补全	432.114	532.620	24.641	989.376
0034	Gemma 4 31B	图文问答	442.615	584.812	27.187	1054.614
0024	gpt-oss-120b	文本聊天	305.774	73.658	21.220	400.652
0025	gpt-oss-120b	代码补全	320.174	657.298	28.030	1005.501
0014	Ling 3.0 Flash	文本聊天	1054.382	159.398	24.070	1237.850
0015	Ling 3.0 Flash	代码补全	1072.905	415.851	24.784	1513.540
0020	Mistral Medium 3.5	文本聊天	279.065	128.877	20.508	428.450
0021	Mistral Medium 3.5	代码补全	273.513	995.478	26.070	1295.061
0019	Mistral Medium 3.5	图文问答	277.942	754.756	27.128	1059.826
0008	DeepSeek-V4-Flash	文本聊天	885.103	616.880	26.198	1528.180
0009	DeepSeek-V4-Flash	代码补全	1054.118	1074.524	27.118	2155.760
0040	Hy3	文本聊天	400.277	116.888	24.170	541.336
0041	Hy3	代码补全	388.812	586.474	27.249	1002.534
0043	MiMo-V2.5	文本聊天	441.068	186.918	25.141	653.128
0044	MiMo-V2.5	代码补全	447.075	614.935	28.809	1090.820
0042	MiMo-V2.5	图文问答	449.530	1994.613	38.601	2482.743
0049	GLM-5.3-Flash	文本聊天	923.078	460.091	26.702	1409.872

编号	模型	测试场景	启动能耗	正式运行 能耗	释放能耗	总能耗
0050	GLM-5.3-Flash	代码补全	959.978	1028.126	27.025	2015.128
0048	GLM-5.3-Flash	图文问答	962.563	2529.878	32.350	3524.792
0032	Qwen3.5-397B	文本聊天	785.070	289.457	25.389	1099.917
0033	Qwen3.5-397B	代码补全	783.167	494.229	27.509	1304.905
0031	Qwen3.5-397B	图文问答	795.462	3069.200	34.089	3898.750
0038	Llama 4 Maverick	文本聊天	290.231	75.334	17.965	383.530
0039	Llama 4 Maverick	代码补全	274.728	293.973	24.271	592.973
0037	Llama 4 Maverick	图文问答	267.711	524.834	27.706	820.251
0017	MiniMax-M3	文本聊天	353.892	244.546	23.868	622.307
0018	MiniMax-M3	代码补全	361.205	990.942	26.564	1378.711
0016	MiniMax-M3	图文问答	353.723	1759.398	29.337	2142.458

SGLang 引擎补充测试 (14 项)

编号	模型	测试场景	启动能耗	正式运行 能耗	释放能耗	总能耗
0001	Sarashina 2.2	文本聊天	178.384	29.908	17.166	225.458
0002	Sarashina 2.2	代码补全	180.677	367.744	28.592	577.013
0010	Granite 4.2	文本聊天	221.829	73.890	23.548	319.268
0011	Granite 4.2	代码补全	224.904	261.292	27.320	513.515
0022	gpt-oss-120b	文本聊天	354.201	62.286	19.260	435.746
0023	gpt-oss-120b	代码补全	350.967	471.525	31.828	854.320
0005	DeepSeek-V4-Flash	文本聊天	1193.996	191.471	24.625	1410.092
0007	DeepSeek-V4-Flash	代码补全	1182.147	712.763	29.714	1924.624
0046	GLM-5.3-Flash	文本聊天	655.851	470.434	25.304	1151.588
0047	GLM-5.3-Flash	代码补全	764.482	959.256	25.778	1749.516
0045	GLM-5.3-Flash	图文问答	609.840	1698.709	29.026	2337.576
0029	Qwen3.5-397B	文本聊天	516.960	335.126	25.799	877.884
0030	Qwen3.5-397B	代码补全	509.992	476.035	28.686	1014.713
0028	Qwen3.5-397B	图文问答	513.598	3385.570	36.832	3936.001

DeepSeek 关闭 thinking 测试（1 项）

编号	模型	测试场景	启动能耗	正式运行 能耗	释放能耗	总能耗
0006	DeepSeek-V4-Flash	文本聊天	1173.428	163.273	24.568	1361.269

注：总能耗为启动、正式运行和资源释放三个阶段的 GPU 能耗之和。各阶段保留三位小数，分项之和与总计可能存在末位舍入差异。请求规模对应的固定开销摊销见式（1）。

表 B-3 推理引擎版本

推理引擎	版本	适用测试
vLLM	0.29.0	首轮测试，除 Ling 3.0 Flash、Mistral Medium 3.5 及 GLM-5.3-Flash 外的 vLLM 测试
vLLM	0.29.0-cu129	Ling 3.0 Flash 首轮测试
vLLM	0.20.1	Mistral Medium 3.5 首轮测试
vLLM	0.28.1rc1.dev580+g385dce36b.cu129	GLM-5.3-Flash 首轮测试
SGLang	0.5.19-cu129	MiMo-V2.5 首轮测试；除 GLM-5.3-Flash 外的 SGLang 补充测试；DeepSeek 关闭 thinking 测试
SGLang	nightly-dev-cu12-20260827-20621aa1	GLM-5.3-Flash 的 SGLang 补充测试

感谢阅读

大语言模型推理能效评估白皮书 (Benchmark 1.5)

构建"测-评-用-优"闭环体系，推动 AI 可持续发展