PR #22867 完整报告

sgl-project/sglang

[feat] Add language_model_only parameter support for Qwen35

合并时间：2026-08-10 02:47

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/22867

执行摘要

- 一句话：Qwen3.5 纯文本模式支持，跳过视觉编码器加载
- 推荐动作：值得精读。核心看点：一是三层联动开关的设计（配置层判定 + 模型层分支 + processor 短路），把单一 config 字段贯穿到整个加载链路；二是 _require_vision 守卫函数兼顾生产与单测的写法（模块级函数 + getattr 兜底），是对“现有单测以未绑定方法调用”这类隐性约束的教科书式响应；三是 gemini-code-assist 高优先级告警被驳回的论证——属性初始化顺序问题最终由维护者确认父类无条件赋值而关闭，展示了 review 中不必要的告警如何被快速、有理有据地化解。若团队后续要支持其他模型的纯文本变体，本 PR 提供了可直接复用的模式。

功能与动机

PR body 原文：Add language_model_only parameter support to ensure compatibility with the previous Qwen 3.5 architecture and avoid redundant architectures。背景是 Qwen 3.5 的纯文本权重仍使用 Qwen3VL 多模态架构类，若不识别该参数，引擎会为纯文本检查点也构建视觉编码器并加载多模态 processor，造成显存与加载开销冗余，甚至因 config 中缺少 vision_config 等字段而崩溃；同时会让同一个模型被迫维护两套架构入口（多模态与纯文本），架构上冗余。

实现拆解

1. 模型入口改造（python/sglang/srt/models/qwen3_vl.py）：Qwen3VLForConditionalGeneration.__init__读取 config.language_model_only（默认 False），为 True 时 self.visual=None 并旁路 deepstack 状态初始化；is_mrope_enabled 增加 not language_model_only 条件。forward 新增纯文本直通分支：直接调用 self.model(...) 而非 general_mm_embed_routine，同时跳过 mrope 的 (3, seq_len) 形状断言；pad_input_ids、get_image_feature、get_video_feature 三个多模态入口统一加 _require_vision 守卫。
2. 新增模块级守卫函数 _require_vision（qwen3_vl.py 文件末尾）：当 language_model_only=True 且 visual=None 时抛出 RuntimeError，把“纯文本模型收到图像输入”从隐式崩溃变成显式可诊断错误。实现选择模块级函数 + getattr 双保险，是为了兼容现有单测 test_qwen3_vl_feature_materialization 以未绑定方法在 SimpleNamespace 上调用特征入口的写法（提交 f19ed3a 的说明）。

3. 配置判定层（python/sglang/srt/configs/model_config.py）：集中读取 self.is_lm_only = getattr(self.hf_config, "language_model_only", False)，并给 model_is_mrope、is_multimodal、is_image_understandable_model、is_audio_understandable_model 统一加 not self.is_lm_only 门控——这一步同时堵住了早期版本在 model_is_mrope 上的漏判（提交 d10034a 说明），防止纯文本检查点在 forward_batch 构造阶段被误判为 mrope 模型。
4. 子类适配（python/sglang/srt/models/qwen3_5.py）：Qwen3_5ForConditionalGeneration 与 Qwen3_5MoeForConditionalGeneration 的 is_mrope_enabled 加 not self.language_model_only 条件，deepstack_visual_indexes 在 visual 为 None 时回退为空列表，避免初始化崩溃。
5. 处理器加载短路（python/sglang/srt/utils/hf_transformers/processor.py）：get_processor 检测到 config.language_model_only=True 时直接返回 AutoTokenizer，跳过多模态 processor 初始化，防止 TokenizerManager 走入多模态路径。
6. 测试与配套：本次没有新增测试文件；验证依赖现有单元测试的间接覆盖与手工回归（JustinTong0323 在 APPROVE 时注明 checked with no regressions），CI 经历 3 轮触发与重跑后才全绿，期间发现并修复了 _require_vision 的未绑定调用问题。

关键文件：

- python/sglang/srt/models/qwen3_vl.py（模块 模型层；类别 source；类型 data-contract；符号 Qwen3VLForConditionalGeneration, _require_vision）：PR 核心文件：Qwen3VLForConditionalGeneration 是所有 Qwen3 VL 系模型的统一入口，这里实现 visual=None 分支、mrope/deepstack 旁路、forward 纯文本直通与 _require_vision 守卫。
- python/sglang/srt/configs/model_config.py（模块 配置判定；类别 source；类型 data-contract；符号 ModelConfig）：全局配置判定层：集中读取 is_lm_only 并给 model_is_mrope、is_multimodal、is_image_understandable_model、is_audio_understandable_model 统一加门控，影响调度、chunked prefill 与 CUDA graph 等下游开关。
- python/sglang/srt/models/qwen3_5.py（模块 模型层；类别 source；类型 data-contract；符号 Qwen3_5ForConditionalGeneration, Qwen3_5MoeForConditionalGeneration）：Qwen3.5 两个子类（Dense 与 MoE）继承父类 language_model_only 语义，需要同步调整 mrope 开关并让 deepstack_visual_indexes 在无视觉编码器时回退，否则初始化崩溃。
- python/sglang/srt/utils/hf_transformers/processor.py（模块 处理器加载；类别 source；类型 core-logic；符号 get_processor）：多模态 processor 加载的短路逻辑：纯文本检查点直接返回 AutoTokenizer，避免 TokenizerManager 初始化 Qwen3VL 多模态 processor 失败或多余加载。

关键符号：Qwen3VLForConditionalGeneration.init, Qwen3VLForConditionalGeneration.forward, _require_vision, ModelConfig.init, Qwen3_5ForConditionalGeneration.init, Qwen3_5MoeForConditionalGeneration.init, get_processor

关键源码片段

python/sglang/srt/models/qwen3_vl.py

PR 核心文件：Qwen3VLForConditionalGeneration 是所有 Qwen3 VL 系模型的统一入口，这里实现 visual=None 分支、mrope/deepstack 旁路、forward 纯文本直通与 _require_vision 守卫。

```
# 模块级守卫：纯文本模式（language_model_only=True）下若仍收到图像或视频
# 输入，显式抛错而不是在 self.visual 为 None 时隐式崩溃。
# 选择模块级函数 + getattr 兜底，是为了兼容现有单测以未绑定方法
# 在 SimpleNamespace 上调用特征入口的写法。
def _require_vision(model) -> None:
    if (
        getattr(model, "language_model_only", False)
        and getattr(model, "visual", None) is None
    ):
        raise RuntimeError(
            "Checkpoint is marked language_model_only=True and was loaded "
            "without a vision encoder; multimodal inputs are not supported."
        )


class Qwen3VLForConditionalGeneration(nn.Module):
    def __init__(self, config, quant_config=None, prefix="", language_model_cls=Qwen3LLMModel)
    :
        super().__init__()
        # 读取一次 config 标记：True 表示纯文本模型，不构建视觉编码器，
        # 省去视觉权重加载与显存占用；默认 False 保持多模态原行为。
        self.language_model_only = getattr(config, "language_model_only", False)
        if self.language_model_only:
            self.visual = None
        else:
            self.visual = Qwen3VLMoeVisionModel(
                config.vision_config,
                quant_config=None,
                norm_eps=getattr(config, "rms_norm_eps", 1e-6),
                prefix=add_prefix("model.visual", prefix),
                use_data_parallel=self.use_data_parallel,
            )

        # mrope 只对多模态路径有意义，纯文本模式直接关闭，
        # 避免 forward_batch 构造阶段误判为 mrope 模型。
        self.is_mrope_enabled = (
            not self.language_model_only and "mrope_section" in self.config.rope_scaling
        )

        # deepstack 依赖 vision_config 的索引：纯文本模式清空状态，
        # 否则访问缺失字段会抛 AttributeError。
        if not self.language_model_only:
            self.deepstack_visual_indexes = config.vision_config.deepstack_visual_indexes
```

```
            self.num_deepstack_embeddings = len(self.deepstack_visual_indexes)
            self.use_deepstack = {Modality.IMAGE: True, Modality.VIDEO: True}
        else:
            self.deepstack_visual_indexes = []
            self.num_deepstack_embeddings = 0
            self.use_deepstack = {}

    def forward(self, input_ids, positions, forward_batch, get_embedding=False, pp_proxy_
    tensors=None):
        if self.is_mrope_enabled:
            positions = forward_batch.mrope_positions

        if self.language_model_only:
            # 纯文本直通：跳过 general_mm_embed_routine，同时跳过
            # mrope 的 (3, seq_len) 形状断言。
            hidden_states = self.model(
                input_ids=input_ids,
                forward_batch=forward_batch,
                positions=positions,
                pp_proxy_tensors=pp_proxy_tensors,
            )
        else:
            if not (
                forward_batch.forward_mode.is_decode()
                or not forward_batch.contains_image_inputs()
            ):
                if self.is_mrope_enabled:
                    assert positions.ndim == 2 and positions.size(0) == 3, (
                        "multimodal section rotary embedding requires "
                        f"(3, seq_len) positions, but got {positions.size()}"
                    )
            hidden_states = general_mm_embed_routine(
                input_ids=input_ids,
                forward_batch=forward_batch,
                language_model=self.model,
                multimodal_model=self,
                positions=positions,
                use_deepstack=self.use_deepstack,
                pp_proxy_tensors=pp_proxy_tensors,
            )
```

python/sglang/srt/configs/model_config.py

全局配置判定层：集中读取 is_lm_only 并给 model_is_mrope、is_multimodal、is_image_understandable_model、is_audio_understandable_mode统一加门控，影响调度、chunked prefill 与 CUDA graph 等下游开关。

```
        # 集中读取一次 language_model_only，统一记作 is_lm_only；
        # 早期版本散落的 getattr 曾在 model_is_mrope 上漏判，导致纯文本
        # 检查点仍被当作 mrope 模型处理。
```

```
        self.is_lm_only = getattr(self.hf_config, "language_model_only", False)
        self.model_is_mrope = (
            not self.is_lm_only
            and rope_scaling is not None
            and "mrope_section" in rope_scaling
        )

        # 纯文本模式强制关闭多模态、图像、音频能力判定，避免调度器、
        # 多模态 chunked prefill 与 CUDA graph 开关被误触发——这些标志
        # 的下游影响远超 Qwen3.5 本身。
        self.is_multimodal = (
            enable_multimodal
            and not self.is_lm_only
            and (
                is_multimodal_model(self.hf_config.architectures)
                or has_multimodal_subconfig
            )
        )
        self.is_image_understandable_model = (
            enable_multimodal
            and not self.is_lm_only
            and hasattr(self.hf_config, "vision_config")
        )
        self.is_audio_understandable_model = (
            enable_multimodal
            and not self.is_lm_only
            and (
                hasattr(self.hf_config, "audio_config")
                or hasattr(
                    getattr(self.hf_config, "thinker_config", None), "audio_config"
                )
                or getattr(self.hf_config, "sound_config", None) is not None
                or is_audio_model(self.hf_config.architectures)
            )
        )
```

python/sglang/srt/utils/hf_transformers/processor.py

多模态 processor 加载的短路逻辑：纯文本检查点直接返回 AutoTokenizer，避免 TokenizerManager 初始化 Qwen3VL 多模态 processor 失败或多余加载。

```
    # 关键分流：language_model_only=True 的检查点虽属多模态架构族，
    # 实际是纯文本模型。直接返回 AutoTokenizer，跳过多模态 processor
    # 初始化，避免 TokenizerManager 走入多模态路径。
    if getattr(config, "language_model_only", False):
        kwargs.pop("use_fast", None)
        return AutoTokenizer.from_pretrained(
            tokenizer_name,
            *args,
            trust_remote_code=trust_remote_code,
```

```
        revision=revision,
        **kwargs,
    )
```

评论区精华

gemini-code-assist[bot] 在 python/sglang/srt/models/qwen3_5.py 上提出两条 high 优先级告警（Qwen3_5ForConditionalGeneration 与 Qwen3_5MoeForConditionalGeneration 各一条）：The attribute self.language_model_only is used ... but it is not initialized in the init__method of this class. This will result in an AttributeError at runtime。维护者 JustinTong0323 逐一反驳：self.language_model_only is initialized unconditionally in Qwen3VLForConditionalGeneration.__init(qwen3_vl.py:

1241) before any early return; the subclass only reads the attribute after super().init(...). The flagged path is unreachable on the rebased head。两条告警最终均以 resolved 关闭，JustinTong0323 APPROVED 并注明 LGTM, checked with no regressions。另一条有信号的讨论隐藏在提交 f19ed3a 中：现有单测以未绑定方法在 SimpleNamespace 上调用特征入口，导致初版实例方法 _require_vision 抛 AttributeError，最终改为模块级函数并使用 getattr 兜底。

- 子类读取 language_model_only 的初始化顺序告警 (correctness): JustinTong0323 澄清：该属性在父类 Qwen3VLForConditionalGeneration.init（qwen3_vl.py:1241）中无条件初始化，早于任何 early return；子类仅在 super().init() 之后读取，flagged path 不可达。告警以 resolved 关闭，最终 APPROVED（LGTM, checked with no regressions）。
- _require_vision 需兼容未绑定测试调用 (testing): _require_vision 从实例方法改为模块级函数，用 getattr 同时兜底 language_model_only 与 visual：裸测试命名空间可正常通过，真实纯文本检查点仍抛出相同的 RuntimeError。

风险与影响

- 风险：
  1. 核心加载路径变更：Qwen3VLForConditionalGeneration.forward 是所有 Qwen3VL 系模型的统一入口，本 PR 在 forward、pad_input_ids、get_image_feature、get_video_feature 四个入口加了分支或守卫，任何与 config 字段约定不一致都可能影响正常多模态加载；默认 getattr(..., False) 兜底后，多模态原行为保持。
  2. 反向误标风险：若多模态检查点错误携带 language_model_only=True，会静默禁用视觉能力，且只在收到图像输入时由 _require_vision 显式抛错，错误暴露存在延迟。
  3. 全局判定门控：model_config.py 的 is_lm_only 影响 is_multimodal、is_audio_understandable_model、model_is_mrope 等标志，下游调度器、多模态 chunked prefill、CUDA graph 开关均依赖这些标志，影响面不止 Qwen3.5 一族。
  4. 测试缺口：无新增测试覆盖 language_model_only=True 的纯文本加载、不加载 visual 权重和 processor 短路路径，仅靠现有单测间接覆盖。
  5. 类型契约：get_processor 短路返回 AutoTokenizer（而非 processor），调用方若按 processor 协议处理返回值可能出现类型不符，需关注 TokenizerManager 的实际用法。
  - 影响：用户侧：Qwen3.5 纯文本检查点（Dense 与 MoE 变体）可开箱即用，不再加

载视觉编码器，显存占用与权重加载时间下降；多模态用户行为不变（默认 False）。系统侧：模型初始化、配置判定、processor 加载三条路径各新增一条分支，虽然改动面横跨 4 个核心文件，但所有开关默认关闭，向后兼容。团队侧：确立了“多模态架构族的纯文本变体”处理范式——在配置判定层集中读标记、在模型层按标记旁路视觉组件、在 processor 层短路加载，后续其他模型族的同类需求可直接复用该模式。 - 风险标记：核心模型加载路径变更，缺少新增测试覆盖，依赖 HF config 字段约定，多模态判定全局门控

关联脉络

- PR #32858 Fix DCP KV head mapping for GQA models: 与本次 PR 同样修改 python/sglang/srt/models/qwen3_5.py 与 python/sglang/srt/configs/model_config.py，属于 Qwen3.5 模型支持线的另一项重要修复（DCP 下 GQA KV head 映射），两条线共同表明 Qwen3.5 系列近期在持续补齐功能与正确性。