PR #41129 完整报告

vllm-project/vllm

[New Model] Laguna XS.2 implementation

合并时间：2026-04-29 02:23

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/41129

执行摘要

- 一句话：新增 Laguna XS.2 模型支持，包括 MoE、推理解析和工具调用
- 推荐动作：推荐精读 laguna.py 中 MoE 层和混合注意力层的实现，以及 poolside_v1_tool_parser.py 中的增量流式工具调用逻辑。poolside_v1_reasoning_parser.py 展示了如何基于现有解析器定制行为，具有参考价值。PR 合并前应解决 review 中的全局配置和默认值问题。

功能与动机

支持 Laguna XS.2 模型在 vLLM 中推理，该模型具备稀疏 MoE 和混合注意力特性，并通过解析器适配池化端的工具调用和推理标记。

实现拆解

1. 定义模型配置：新增 LagunaConfig（vllm/transformers_utils/configs/laguna.py），继承 PretrainedConfig，包含 MoE、混合注意力等参数，并处理 rope_parameters 兼容性。
2. 实现模型架构：新增 laguna.py（vllm/model_executor/models/laguna.py），包含 LagunaMLP、LagunaMoE（sigmoid 路由、共享专家）、LagunaAttention（支持 softplus gating 和滑动窗口）、LagunaDecoderLayer（分层类型选择）、LagunaModel（流水线并行）等类。
3. 添加推理解析器：PoolsideV1ReasoningParser（vllm/reasoning/poolside_v1_reasoning_parser.py）继承 DeepSeekV3ReasoningParser，但将 </think> 搜索范围限定在当前助手掌内，避免历史对话中误判。
4. 添加工具解析器：PoolsideV1ToolParser（vllm/tool_parsers/poolside_v1_tool_parser.py）支持增量流式工具调用，通过 _streaming_string_value 状态实现字符串参数的逐字符流式输出。
5. 注册与集成：更新 config.py 添加 laguna 配置映射，更新 registry.py 注册模型，更新 __init__.py 注册推理和工具解析器，更新 tests/models/registry.py 添加测试入口。

关键文件：

- vllm/model_executor/models/laguna.py（模块 模型执行器；类别 source；类型 data-contract；符号 LagunaMLP, LagunaMoE, LagunaAttention, LagunaDecoderLayer）：模型核心实现文件，包含所有模型层（LagunaMLP、LagunaMoE、LagunaAttention、LagunaDecoderLayer、LagunaModel），定义了正向

传播和并行支持。

- vllm/tool_parsers/poolside_v1_tool_parser.py（模块 工具解析器；类别 source；类型 dependency-wiring；符号 PoolsideV1ToolParser, _deserialize, _json_escape_string_content, _is_string_type）：工具解析器，实现增量流式工具调用，支持字符串参数逐字符输出，解决流式延迟问题。
- vllm/reasoning/poolside_v1_reasoning_parser.py（模块 推理解析器；类别 source；类型 dependency-wiring；符号 PoolsideV1ReasoningParser, is_reasoning_end）：推理解析器，基于 DeepSeekV3 定制，避免历史对话中的误判。
- vllm/transformers_utils/configs/laguna.py（模块 配置；类别 source；类型 core-logic；符号 LagunaConfig）：模型配置定义，包含所有超参数和 TP/PP 映射。
- vllm/transformers_utils/config.py（模块 配置；类别 source；类型 core-logic）：配置映射，添加 laguna 到 LagunaConfig 的映射，并处理嵌套 rope_parameters 兼容性。
- vllm/reasoning/__init__.py（模块 推理解析器；类别 source；类型 core-logic）：注册 poolside_v1 推理解析器。
- vllm/tool_parsers/__init__.py（模块 工具解析器；类别 source；类型 core-logic）：注册 poolside_v1 工具解析器。
- vllm/model_executor/models/registry.py（模块 模型执行器；类别 source；类型 data-contract）：注册 Laguna 模型。
- vllm/transformers_utils/configs/__init__.py（模块 配置；类别 source；类型 core-logic）：导出 LagunaConfig。
- tests/models/registry.py（模块 测试；类别 test；类型 test-coverage）：更新模型测试注册表，添加 Laguna 模型测试入口。

关键符号：LagunaMLP.forward, LagunaMoE.forward, LagunaAttention.forward, LagunaDecoderLayer.forward, LagunaModel.forward, PoolsideV1ReasoningParser.is_reasoning_end, PoolsideV1ToolParser.extract_tool_calls, PoolsideV1ToolParser.adjust_request

关键源码片段

vllm/tool_parsers/poolside_v1_tool_parser.py

工具解析器，实现增量流式工具调用，支持字符串参数逐字符输出，解决流式延迟问题。

```
# SPDX-License-Identifier: Apache-2.0
# SPDX-FileCopyrightText: Copyright contributors to the vLLM project
"""
GLM-4 Tool Call Parser with incremental string streaming support.

This parser fixes the streaming issue reported in Issue #32829 where long string
parameters (e.g., file content with 4000+ characters of code) are buffered until
complete, causing multi-second delays before the user sees any content.

The fix streams string values incrementally as they arrive, providing a true
streaming experience for long content.
```

```
"""

import ast
import json
from collections.abc import Sequence
from typing import Any

import partial_json_parser.core.complete
import regex as re
from partial_json_parser.core.options import Allow

from vllm.entrypoints.chat_utils import make_tool_call_id
from vllm.entrypoints.openai.chat_completion.protocol import (
    ChatCompletionRequest,
)
from vllm.entrypoints.openai.engine.protocol import (
    DeltaFunctionCall,
    DeltaMessage,
    DeltaToolCall,
    ExtractedToolCallInformation,
    FunctionCall,
    ToolCall,
)
from vllm.entrypoints.openai.responses.protocol import (
    ResponsesRequest,
)
from vllm.logger import init_logger
from vllm.tokenizers import TokenizerLike
from vllm.tool_parsers.abstract_tool_parser import (
    Tool,
    ToolParser,
)

logger = init_logger(__name__)


class PoolsideV1ToolParser(ToolParser):
    """Tool parser for GLM-4 models with incremental string streaming.

    This parser emits tool-call deltas incrementally as arguments arrive.
    For string-type parameters, content is streamed character-by-character
    rather than waiting for the complete </arg_value> tag.
    """

    def __init__(self, tokenizer: TokenizerLike, tools: list[Tool] | None = None):
        super().__init__(tokenizer, tools)
        self.current_tool_name_sent: bool = False
        self.prev_tool_call_arr: list[dict[str, Any]] = []
        self.current_tool_id: int = -1
```

```
        self.streamed_args_for_tool: list[str] = []

        self.tool_call_start_token: str = "<tool_call>"
        self.tool_call_end_token: str = "</tool_call>"
        self.arg_key_start: str = "<arg_key>"
        self.arg_key_end: str = "</arg_key>"
        self.arg_val_start: str = "<arg_value>"
        self.arg_val_end: str = "</arg_value>"

        self.tool_calls_start_token = self.tool_call_start_token

        self.func_call_regex = re.compile(r"<tool_call>.*?</tool_call>", re.DOTALL)
        self.func_detail_regex = re.compile(
            r"<tool_call>([^\n]*)\n(.*)</tool_call>", re.DOTALL)
        self.func_arg_regex = re.compile(
            r"<arg_key>(.*?)</arg_key>\s*<arg_value>(.*?)</arg_value>", re.DOTALL)

        if not self.model_tokenizer:
            raise ValueError(
                "The model tokenizer must be passed to the ToolParser "
                "constructor during construction.")

        self.tool_call_start_token_id = self.vocab.get(self.tool_call_start_token)
        self.tool_call_end_token_id = self.vocab.get(self.tool_call_end_token)
        self._buffer: str = ""
        self._in_tool_call: bool = False
        self._current_tool_name: str | None = None
        self._pending_key: str | None = None
        self._streaming_string_value: bool = False
        self._tool_call_ids: list[str] = []
        self._args_started: list[bool] = []
        self._args_closed: list[bool] = []
        self._seen_keys: list[set[str]] = []
```

vllm/reasoning/poolside_v1_reasoning_parser.py

推理解析器，基于 DeepSeekV3 定制，避免历史对话中的误判。

```
# SPDX-License-Identifier: Apache-2.0
# SPDX-FileCopyrightText: Copyright contributors to the vLLM project
"""
Laguna reasoning parser.

``DeepSeekV3ReasoningParser.is_reasoning_end`` walks the entire
token sequence backwards and returns ``True`` on the first ``</think>`` it
sees. When called on ``prompt_token_ids`` that mistakes any stray
``</think>`` in conversation history, few-shot examples or tool descriptions
for a template-injected "thinking already ended" marker. In the streaming
path (see ``vllm/entrypoints/openai/chat_completion/serving.py``,
``prompt_is_reasoning_end_arr``) that false positive short-circuits the
```

```
reasoning parser for the whole response, so any ``<think>...</think>`` the
model emits itself ends up in the content field instead of the reasoning
field.

As we have more flexible templates, we instead scope
the backward search to the current assistant turn: the
walk terminates as soon as we hit the ``<assistant>`` start-of-message
token. A ``</think>`` in a prior user turn or few-shot example is no longer
visible.
"""

from collections.abc import Sequence

from transformers import PreTrainedTokenizerBase

from vllm.reasoning.deepseek_r1_reasoning_parser import DeepSeekR1ReasoningParser
from vllm.reasoning.deepseek_v3_reasoning_parser import DeepSeekV3ReasoningParser
from vllm.reasoning.identity_reasoning_parser import IdentityReasoningParser


class PoolsideV1ReasoningParser(DeepSeekV3ReasoningParser):
    """Drop-in replacement for ``deepseek_v3`` that tolerates ``</think>``
    tokens appearing anywhere in the prompt other than the generation prefix.
    """

    _start_of_assistant_message = "<assistant>"

    def __init__(self, tokenizer: PreTrainedTokenizerBase, *args, **kwargs):
        super().__init__(tokenizer, *args, **kwargs)

        if self._start_of_assistant_message not in self.vocab:
            raise ValueError(
                f"Tokenizer must contain {self._start_of_assistant_message!r} token")
        self._start_of_assistant_message_token_id = self.vocab[
            self._start_of_assistant_message]

    def is_reasoning_end(self, input_ids: Sequence[int]) -> bool:
        # IdentityReasoningParser always returns True: no reasoning to parse.
        if isinstance(self._parser, IdentityReasoningParser):
            return True

        assert isinstance(self._parser, DeepSeekR1ReasoningParser)
        for tok_id in reversed(input_ids):
            # <think>: reasoning is not yet ended.
            if tok_id == self._parser.start_token_id:
                return False
            # </think>: reasoning has ended.
            if tok_id == self._parser.end_token_id:
                return True
```

```
            # <assistant>: reached the start of the current assistant turn
            # without seeing either marker. Anything further back belongs to
            # the prior conversation and should be ignored.
            if tok_id == self._start_of_assistant_message_token_id:
                return False
        return False


__all__ = ["PoolsideV1ReasoningParser"]
```

评论区精华

Review 中 gemini-code-assist[bot] 指出三个问题：

- 在 LagunaMoE.__init__ 中直接修改全局 eplb_config.num_redundant_experts（行 159），可能影响依赖该配置的其他模型（如 DeepSeek）。
- 缺少对物理专家数能被 EP 大小整除的校验（行 162），不满足时会导致运行时错误。
- 使用 getattr(config, "moe_apply_router_weight_on_input", True) 默认值 True 与 LagunaConfig 默认值 False 不一致。robertgshaw2-redhat 也确认了该问题。 这些评论均未在 PR 中看到修复，需后续跟进。
- 全局配置修改风险 (correctness): 未确认修复，需后续跟进。
- 专家分布整除校验缺失 (correctness): 未确认修复，需后续跟进。
- getattr 默认值不一致 (correctness): 未确认修复，需后续跟进。

风险与影响

- 风险：
  1. 全局配置副作用：LagunaMoE 修改 eplb_config.num_redundant_experts 可能影响其他组件，建议使用局部变量。
  2. 专家分布校验缺失：若物理专家数不能被 EP 大小整除，将导致索引错误，建议添加显式检查。
  3. 默认值不一致：moe_apply_router_weight_on_input 的默认值在代码中与配置定义不一致，可能导致行为差异。
  4. 新模型未经大规模测试，可能在边界情况（如长上下文、特殊注意力设置）下出错。 - 影响：对用户：可以加载 Laguna XS.2 模型并利用其工具调用和推理功能。对系统：新增 4 个源文件，修改 6 个注册文件，不影响现有模型。对团队：需维护新增模型及解析器，并修复 review 中提出的问题。 - 风险标记：全局配置副作用，专家分布校验缺失，默认值不一致

关联脉络

- 暂无明显关联 PR