PR #30615 完整报告

sgl-project/sglang

[Bench] Add fixed-prompt mode and per-request spec accept length metrics

合并时间：2026-07-09 17:06

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/30615

执行摘要

- 一句话：新增固定 prompt 模式与 per-request spec_accept_length 指标
- 推荐动作：该 PR 值得关注，特别是对从事 LLM 推理和投机解码评估的工程师。其中 _encode_fixed_prompt 的设计体现了对不同模型编码路径的抽象，FIXME 注释也暴露了客户 - 服务器编码逻辑重复的问题，未来可能需统一。

功能与动机

引用 PR body: 'Bench/eval tooling for speculative decoding: --fixed-prompt-file + --apply-chat-template in one_batch_server for a controlled constant accept length across batch sizes, and per-request spec_accept_length collection in bench_serving and run_eval.' 该变更旨在解决此前投机解码评估缺少可控 prompt 和 per-request 指标的问题。

实现拆解

1. 在 python/sglang/benchmark/one_batch_server.py 中新增 fixed_prompt_file 和 apply_chat_template 配置项，并实现 _encode_fixed_prompt 函数。该函数支持三种编码路径：纯 tokenizer encode、DeepSeek-V4 自定义编码（通过 _is_deepseek_v4_model 检测）、以及标准 HF chat template 编码。通过 @lru_cache 加速模型类型判断。
2. 在 python/sglang/benchmark/serving.py 的 RequestFuncOutput 数据类中新增 spec_accept_length 字段，并在非流式和流式响应的解析分支中从 meta_info 提取该值（非流式从 response_json['choices'][0]['meta_info']，流式从每个 chunk 的 meta_info）。
3. 在 python/sglang/test/simple_eval_common.py 的 ChatCompletionSampler 中新增 record_meta_info 参数，当其启用时，在 API 请求中加入 return_meta_info: true，并将返回的 meta_info 收集到 _meta_infos 列表中。
4. 在 python/sglang/test/run_eval.py 中新增 print_accept_length_summary 函数，统计所有 sampler 收集的 spec_accept_length 并输出 n/mean/min/max；在单次和多次 eval 流程中收集 sampler 并最终调用该函数。

关键文件：

- python/sglang/benchmark/one_batch_server.py（模块 基准测试工具；类别 source；类型 dependency-wiring；符号 _is_deepseek_v4_model, _encode_fixed_prompt）：核心变更文件，新增固定 prompt 模式和 DeepSeek-V4 编码支持

- python/sglang/benchmark/serving.py（模块 基准测试工具；类别 source；类型 core-logic；符号 spec_accept_length）：在 RequestFuncOutput 和响应解析中增加 spec_accept_length 收集
- python/sglang/test/run_eval.py（模块 评估脚本；类别 test；类型 test-coverage；符号 print_accept_length_summary）：新增 print_accept_length_summary 函数并在 eval 流程中收集 sampler
- python/sglang/test/simple_eval_common.py（模块 评估基类；类别 test；类型 test-coverage；符号 record_meta_info, _meta_infos）：ChatCompletionSampler 新增 record_meta_info 参数和 _meta_infos 收集

关键符号：_is_deepseek_v4_model, _encode_fixed_prompt, print_accept_length_summary

关键源码片段

python/sglang/benchmark/one_batch_server.py

核心变更文件，新增固定 prompt 模式和 DeepSeek-V4 编码支持

```
@lru_cache(maxsize=None)
def _is_deepseek_v4_model(name_or_path: str) -> bool:
    # 尝试通过 HuggingFace config 判断模型是否为 DeepSeek-V4
    from transformers import AutoConfig
    from sglang.srt.configs.model_config import is_deepseek_v4

    try:
        hf_config = AutoConfig.from_pretrained(name_or_path, trust_remote_code=True)
    except Exception as e:
        # 离线环境无法加载时，默认返回 False 并警告
        print(
            f"Warning: could not load config for {name_or_path!r} ({e}); "
            "assuming a non-DeepSeek-V4 model for --apply-chat-template."
        )
        return False
    return is_deepseek_v4(hf_config)


def _encode_fixed_prompt(
    tok_inner, prompt_text: str, apply_chat_template: bool
) -> List[int]:
    # 如果不应用聊天模板，直接 encode
    if not apply_chat_template:
        return tok_inner.encode(prompt_text)

    messages = [{"role": "user", "content": prompt_text}]
    # DeepSeek-V4 使用自定义编码，不走 HF chat template
    if _is_deepseek_v4_model(getattr(tok_inner, "name_or_path", "") or ""):
        from sglang.srt.entrypoints.openai import encoding_dsv4
        real_input = encoding_dsv4.encode_messages(messages, thinking_mode="chat")
        return tok_inner.encode(real_input)
```

```
    # 普通模型必须有 chat template
    if getattr(tok_inner, "chat_template", None) is None:
        raise ValueError(
            "--apply-chat-template requires a tokenizer with a chat template, "
            f"but {getattr(tok_inner, 'name_or_path', tok_inner)!r} has none."
        )
    return tok_inner.apply_chat_template(
        messages, add_generation_prompt=True, tokenize=True
    )
```

python/sglang/test/simple_eval_common.py

ChatCompletionSampler 新增 record_meta_info 参数和 _meta_infos 收集

```
  def __call__(self, message_list: MessageList) -> str:
    # 前置逻辑：系统消息前缀
    if self.system_message:
        message_list = [
            self._pack_message("system", self.system_message)
        ] + message_list
    extra_body = self.extra_body
    # 如果需要记录 meta_info，在 extra_body 中添加 return_meta_info
    if self.record_meta_info:
        extra_body = {**(self.extra_body or {}), "return_meta_info": True}
    trial = 0
    while trial < 6:
        try:
            response = self.client.chat.completions.create(
                model=self.model,
                messages=message_list,
                temperature=self.temperature,
                top_p=self.top_p,
                max_tokens=self.max_tokens,
                reasoning_effort=self.reasoning_effort,
                extra_body=extra_body, # 使用组装后的 extra_body
            )
            # 收集 meta_info（包含 spec_accept_length）
            if self.record_meta_info:
                meta_info = getattr(response.choices[0], "meta_info", None)
                if meta_info:
                    self._meta_infos.append(meta_info)
            # 记录 completion tokens
            if response.usage and response.usage.completion_tokens is not None:
                self._completion_tokens.append(response.usage.completion_tokens)
            return response.choices[0].message.content or ""
        except openai.BadRequestError as e:
            print("Bad Request Error", e)
            return ""
        except Exception as e:
            # 指数退避重试
```

```
            exception_backoff = 2 ** trial
            print(f"Rate limit exception, wait {exception_backoff}s", e)
            time.sleep(exception_backoff)
            trial += 1
    return ""
```

评论区精华

Review 中三个核心讨论：

- 离线环境 fast-path 检查（high priority）：_is_deepseek_v4_model 直接调用 AutoConfig.from_pretrained，在无网络环境会失败并返回 False，导致绕过 DeepSeek-V4 自定义编码。建议增加基于名称的字符串快速检查。作者后续通过复用 is_deepseek_v4 函数修复。
- iter_time 计算保护（medium priority）：当 last_ttft 为 0（即 TTFT 测量失败）时，decode_wall 会等于总延迟，导致 iter_time 指标严重膨胀。需增加 last_ttft > 0 守卫。已在 commit "guard iter_time on ttft" 中修复。
- CompletionSampler 缺少 meta_info（medium priority）：目前 CompletionSampler 未传递 record_meta_info，导致 completion 模式（如 GSM8K few-shot）无法收集 spec_accept_length。建议在调用 completions.create 时传入 extra_body={'return_meta_info': True}。该问题尚未确认是否修复。
- 离线环境 fast-path 检查 (correctness): 作者后续通过复用 is_deepseek_v4 模型检测函数修复。
- iter_time 计算保护 (correctness): 已通过 commit 'guard iter_time on ttft' 修复。
- CompletionSampler meta_info 支持 (testing): 待确认是否修复；patch 中已为 ChatCompletionSampler 添加 record_meta_info 支持，但 CompletionSampler 可能仍未覆盖。

风险与影响

- 风险：
  1. _is_deepseek_v4_model 仍可能因网络问题回退，但已有 fast-path 降低风险。
  2. iter_time 指标可能因不完整的 last_ttft 而失真，已添加守卫。
  3. 若服务器未启用投机解码，spec_accept_length 可能缺失，代码中已做了容错（or 0.0）。
  4. 代码中包含 FIXME 注释指出编码分派逻辑与 server 侧重复，存在维护一致性风险。
- 影响：
  - 用户：使用 one_batch_server 和 bench_serving 的工程师可以获得更精确的投机解码 benchmark 数据；使用 run_eval 的开发者可以在 eval 结果中直接看到投机接受长度统计。
  - 系统：无性能影响，仅在 benchmark 和 eval 过程中增加少量额外 HTTP 字段和日志输出。
  - 团队：提供了更丰富的投机解码量化工具，有助于后续优化。

- 风险标记：离线环境兼容风险，指标计算可能失真，编码分派逻辑重复

关联脉络

- PR #27862 Support speculative decoding on CPU: 共享投机解码功能上下文，该 PR 为 CPU 后端引入投机解码，本 PR 为其提供评估工具。
- PR #30461 [DSV4] Fix draft SWA transfer for disaggregated MTP: 涉及 DeepSeek-V4 模型和投机解码的 bug 修复，与本 PR 的 DeepSeek-V4 编码路径相关。