实习项目

工作一：
针对 Berkeley Function-Calling Leaderboard 的性能优化
合成数据构造

工作二：
多轮对话 fun call benchmark构建
评测集的设计和构造

针对 BFCL 的性能优化

为什么需要优化这个 benchmark ?

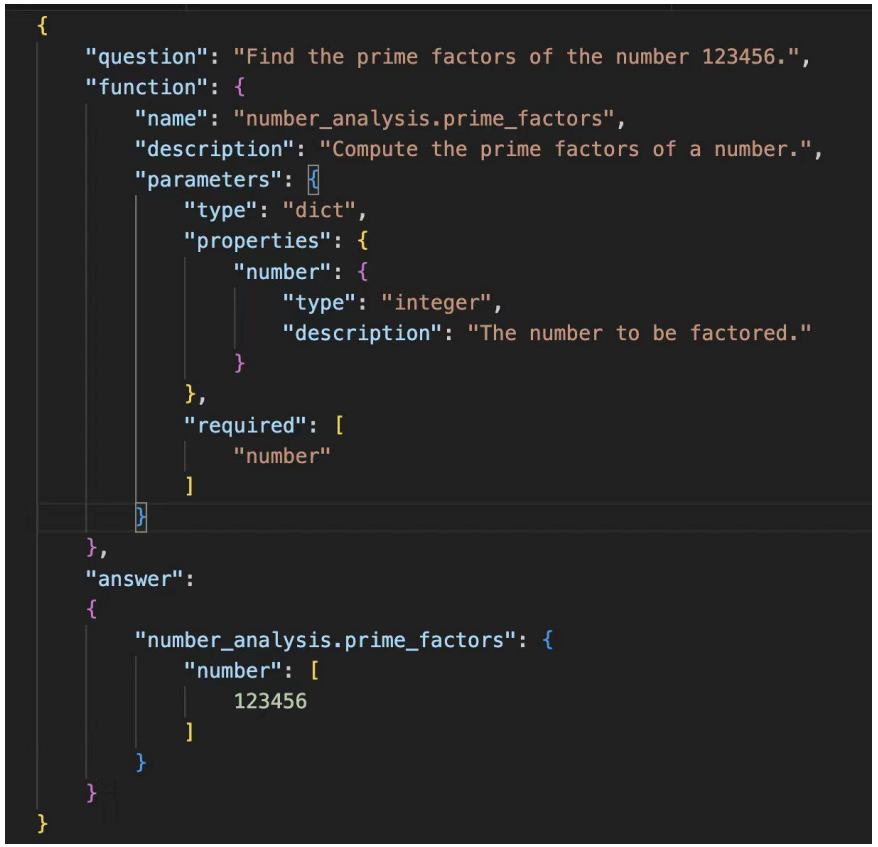
Berkeley Function-Calling Leaderboard

1. 是当前大模型工具学习领域比较有**影响力**的榜单，
GLM-4，LLAMA3.1 等公开了在该榜单上的性能

2. Benchmark 能检测的维度全面：

- 多语言： python, java, javascript, restAPI
- 复杂的用法：**simple / multiple / parallel / parallel_multiple**
- 相关性检测：**是否调用模型的检测**
- 可执行函数

难点



针对 BFCL 的性能优化

qwen 主要存在的问题

1. 输出格式不遵循指令
2. 参数值提取错误
2. 多个候选函数时选错函数
3. 并行调用能力缺失
4. 不会拒答

	qwen-1.5-7b-chat	GPT-4-0125-Preview (Prompt)	gorilla-openfunctions-v2 (FC)	Functionary-Small-v2.4 (FC)
simple_AST	0.525	0.8836	0.88	0.8218
multiple_AST	0.56	0.95	0.95	0.885
parallel_AST	0.35	0.92	0.875	0.82
parallel_multiple_AST	0.345	0.92	0.865	0.81
Relevance	0	0.7042	0.6125	0.6792

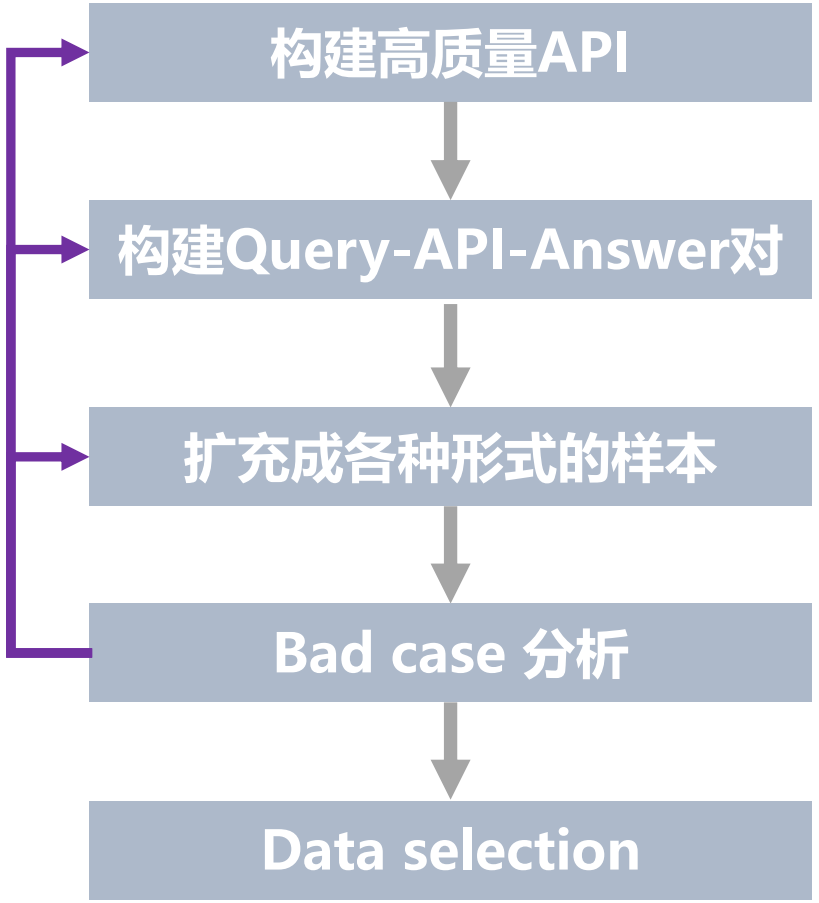
针对 BFCL 的性能优化

合成数据三大原则

Quantity

Quality

Diversity



改造 Gorilla OpenFunction v1 / APIGen

利用LLM / 原数据集生成

Multiple、Parallel、
Parallel multiple、Relevance

Less 方案筛选最有用的数据，质量保证

针对 BFCL 的性能优化

Quantity

自动化的数据生成和质量检查 pipeline

构建Query-API-Answer对

- 给定 API 信息，利用 Qwen-max 生成问答对
- 利用规则和 LLM 过滤低质量样本

扩充成各种形式的样本

- Parallel / Parallel_multiple：

组合不同的 Query-API-Answer 对，并利用 LLM 组合改写query

- Relevance:

随机组合无关function，利用 LLM构造样本，过滤函数样本

针对 BFCL 的性能优化

Quality

API的质量把控

- 删除缺少必需参数的 API
- 去除重名但描述不同的 API
- 利用 qwen-max 补充参数 type类型的字段
- 利用数据集中label 信息补充 required 字段信息

query-API-answer对的质量

- Format Checker
  jsonschema 检查参数类型、required字段

- Semantic Checker
  拆解打分细则, self-consistency 打分:

1. 函数名称是否和候选函数一致
2. 参数是否来自函数本身
3. 参数值提取是否正确
4. 函数能否解决问题

针对 BFCL 的性能优化

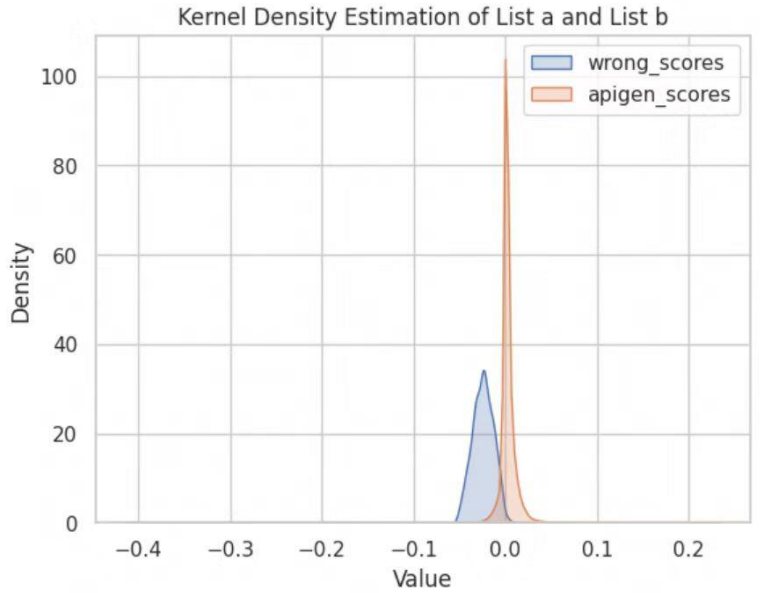
Quality

Data selection

- 通过优化器感知的方式从大量指令数据集中有效地 **选择部分有价值的数据用于目标指令微调**,
- 用影响力公式定义一条数据对模型训练影响的大小：其中z代表训练集，z'代表目标测试集。
- 发现低分的训练样本确实存在更多的错误数据

$$Inf_{SGD}(z, z') = \sum_{t=0}^{N-1} l(z'; \theta^t) - l(z'; \theta^{t+1})$$
$$= \sum_{t=0}^{N-1} \eta_t \cdot \langle \nabla l(z'; \theta^t), \nabla l(z; \theta^t) \rangle$$

10 % good	**0.81**
10% random	0.79
10% bad	0.74



针对 BFCL 的性能优化

Diversity

API 多样性保障

不同domain 采样

Query-API-Answer 多样性

给定不同的few-shot例子

样本的多样性

Simple：候选 function 1个，答案 function 1个
Multiple：候选 function 多个，答案 function 1个
Parallel：候选 function 1个，答案 function 多次
Parallel_multiple：候选function多个, 答案function 多个
Relevance: 候选中无相关 function

针对 BFCL 的性能优化

5k data Sft 实验效果: 在bfcl 上的效果

Model	AST Summary	Exec Summary	Relevance
GPT-4-1106-Preview (Prompt)	83.54	90.95	97.56
Gorilla-OpenFunctions-v2 (FC)	74.37	84.21	64.25
Llama 3.1(Prompt)	80.49	81.10	51.67
Functionary-Small-v3.2 (FC)	74.56	85.95	72.02
GLM-4-9B-Chat (FC)	33.58	41.55	69.61
qwen-1.5-7b-chat	45.36	53.19	0
qwen-1.5-7b-chat-sft	79.52	84.48	67.75

针对 BFCL 的性能优化

5k data 合并后在agent评测上实验效果

把数据合并入整体数据后对qwen2-7b对影响：

agent相关评测的结果显示：数据有泛化性，**在各类agent评测上都有显著提升。**

通用指标方面， 对其他通用能力没有显著影响

BFCL

模型名称	模型版本	推理参数	plugin_Overall 展开	code_interpreter_Overall 展开	plugin_gorilla-chat_prompt_Overall_withoutExec	plugin_gorilla-chat_prompt_Overall 展开	plugin_gorilla-function_call_Overall_withoutExec	plugin_gorilla-function_call_Overall 展开	nous_research_Overall 展开	plugin_nexus_chat-NexusRaven-V2-Overall 展开	plugin_teval_lite-ZH_Overall 展开	plugin_teval_lite-EN_Overall 展开
7b.qwen2												
实验 7b.15t--longcnt_mixv32-base100w-cpt32k-v10.15.32_fc0812_36e6bc	step3000	P0.8K20T0.7R1.1	75.67		78.20		79.42		86.00	34.75	76.51	80.26
基线 7b.15t--longcnt_mixv32-base100w-cpt32k-v10.15.32_trans-bsz1024	step3000	P0.8K20T0.7R1.1	75.07		70.50		60.65		84.90	34.82	74.64	80.90

实习项目

工作一：
针对 Berkeley Function-Calling Leaderboard 的性能优化
合成数据构造

工作二：
多轮对话 fun call benchmark构建
评测集的设计和构造

多轮对话 funcall benchmark构建

自建 benchmark 的必要性

实际业务场景多见

- User query 中未包含所有的信息

User: 请帮我预定酒店
Assistant: 请问您想预定的酒店名称？入住时间？

- 机器人助理-上下文意图相关

User: 请推荐一个北京的著名景点
Assistant: 圆明园
User：帮我查询该景点附近的酒店

当前的外部 Benchmark 存在问题

多轮对话 funcall benchmark构建

Benchmark 构造四大原则

能够反映真实问题：场景、指标设计

正确性

有区分度

稳定性：答案的唯一性

多轮对话 funcall benchmark构建

外部的benchmark不能够反映真实问题

- 不合理的prompt方案：在历史对话中上一步引导模型下一步的action，不符合实际场景

  "API request: "

  "What is the tool name to call at current step?"

  "What is the value of 'propertyId' required by the current tool?"

- 缺少对反问的评测
- 对话意图单一

	Prompt 方案	样本复杂度	评估指标		
	自主reasoning	历史对话存在意图转移	Function call	request	confirm
API Bank	✗	✗	✓	✗	✗
T-eval	✗	✗	✓	✗	✗
Ours	✓	✓	✓	✓	✓

多轮对话 fun call benchmark构建

反映真实问题

改造的数据集质量更高

Schema-Guided Dialogue Dataset

- **和真实对话场景更相似**

20000 多条带注释的**面向任务的人机对话**
20 个领域的服务和 **80 API** 的交互
历史对话存在**多个意图**
平均包含 **20.44 turn**

- **成功的参考**

MeetKai/functionary 的自用评测集改造
自SGD数据集

指标设计更符合实际需求

- **Request**

检测参数缺失时是否发起反问

- **Function call**

检测所有参数信息得到后是否调用function，
检测 function name，parameter value 是否正确

- **Confirm**

检测在调用执行类的function前是否向用户confirm 收集的参数，并检测参数的正确性

多轮对话 fun call benchmark构建

Request

检测参数缺失时是否发起反问

调用函数则accuracy = 0
反之，accuracy = 1


```
    {
        "role": "assistant",
        "content": "Do you want me to schedule an appointment at that location?"
    },
    {
        "role": "user",
        "content": "Yeah. Go ahead and make the appointment for me."
    }
],
"label": [
    "appointment_date",
    "appointment_time"
],
"example_label": "Can you please provide the preferred date and time for appointment?",
```

多轮对话 fun call benchmark构建

Function Call

未在message中加入引导function call 的内容
检测所有参数信息后是否调用function
检测 function name
检测parameter value 是否正确

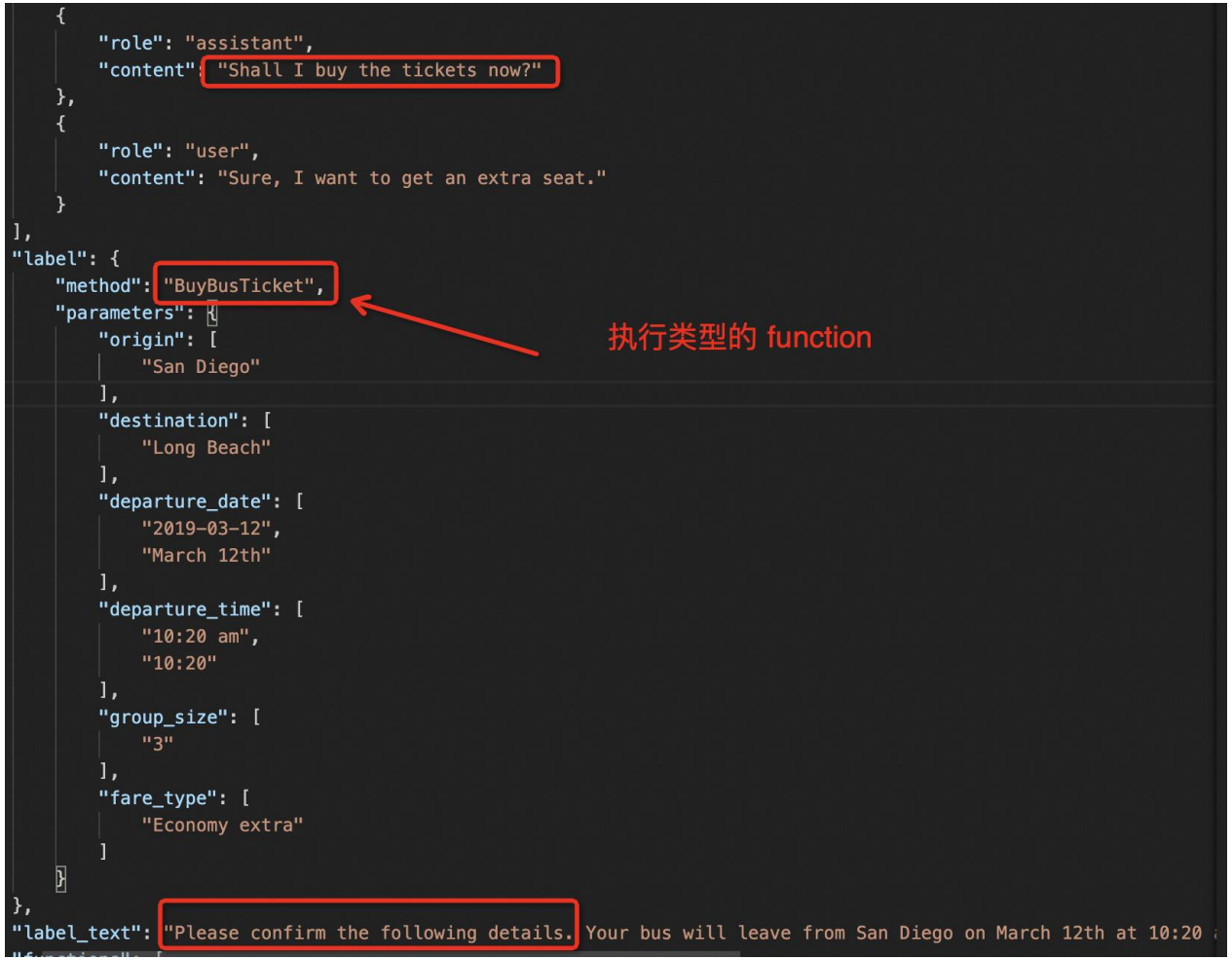
acc_strict: 样本正确
error_no_api：未调用api
error_wrong_format: 函数格式错误
error_wrong_fun_name：函数名称错误
error_wrong_fun_param：参数值错误


```
    {
        "role": "user",
        "content": "I'm leaving from Seattle and traveling to Portland, OR."
    },
    {
        "role": "assistant",
        "content": "Got it. When are you coming back?"
    },
    {
        "role": "user",
        "content": "I'm leaving on March 7th, and want to come back on March 11th."
    }
],
"label": {
    "method": "SearchRoundtripFlights",
    "parameters": {
        "airlines": [
            "United Airlines"
        ],
        "departure_date": [
            "March 7th",
            "2019-03-07"
        ],
        "destination_city": [
            "Portland, OR",
            "Portland"
```

多轮对话 fun call benchmark构建

Confirm

检测在调用执行类的function前是否向用户confirm 收集的参数，



多轮对话 fun call benchmark构建

提升标签的正确性

正确性

【问题】参数值答案不唯一

【解决方案】放宽字符串的匹配条件；时间类型增加格式规范；日期类型在prompt里增加当天日期。

稳定性

【问题】模棱两可的样本

实验发现 Gpt-4o 比较谨慎，倾向于再次向用户confirm 参数；Qwen2 则更倾向于编造参数值，直接调用函数

Eg. 前文提到我的朋友Mike来北京拜访我，后文预定饭店的时候标签中seat=2, 但 gpt-4 未直接调用函数，而是反问人数。

【解决方案】

利用gpt-4o function call 接口，过滤未调用函数的样本

多轮对话 fun call benchmark构建

区分度

样本的难易程度

【问题1】
发现历史对话中包含很多confirm信息，
导致参数抽取问题得到简化
"Please confirm these details:
Transfer $580 from savings to Raghav in
their checking account."

【解决方案】
采样不同的confirm程度的样本，构造三种难度的样本
Easy: 所有参数都经过 confirm
Mid: 部分参数经过confirm, 部分样本在后文被用户否定
Hard: 历史对话中未出现confirm

【问题2】
简单样本太多，区分度不大

【解决方案】
把qwen2-7b 和 llama3.1-8b都做对的样本看作“简单样本”，对其降采样

多轮对话 fun call benchmark构建

模型	Function Call			Request	Confirm
	Easy	Mid	Hard		
Gpt-4o	0.86	0.86	0.83	0.98	0.95
Gpt-4o-mini	0.75	0.57	0.49	0.99	0.96
Qwen2-72B	0.80	0.74	0.67	0.86	0.16
Qwen2-7B	0.72	0.66	0.61	0.47	0.05
Llama 3.1-70B		0.76		0.34	
Llama 3.1-8B				0.86	

区分度：样本的难易、模型的强弱
稳定性：在 gpt 和 Qwen2 上设置 T=0.7, p=0.8， 多次实验，波动在2～3个百分点

Gpt 系列对于function call 调用更加谨慎
qwen 更倾向于在未收集完全参数值的时候发起 function call

多轮对话 fun call benchmark构建

标签的正确性

扩充参数值的候选答案，如 New York = NYC

指标的细化

Request 的细化：提取关键词，检查是否命中候选tag

增加 Review 指标：总结工具执行结果

提升qwen多轮对话的能力

相似的方案构建训练数据集