QCon
全球软件开发大会

InfoQ 极客传媒

NES 模型训练的探索

逐云

算法工程师

目录

01 我们做了什么

02 数据工作

03 模型训练

04 模型评测

05 未来展望

06

InfoQ 极客传媒

北京

QCon 全球软件开发大会

会议时间：4月10-12日

- 大模型赋能 AIOps
- 越挫越勇的大前端
- 云上业务架构演进
- 多模态大模型及应用
- 海外 AI 应用创新实践

北京

AiCon 全球人工智能开发与应用大会

会议时间：6月27-28日

- 端侧智能
- AI Agent
- 多模态大模型
- 金融+大模型

上海

QCon 全球软件开发大会

会议时间：10月23-25日

- AI Agent
- 大模型训练推理
- 端侧 AI
- 搜推深度融合
- 数智金融

4月 — 5月 — 6月 — 8月 — 10月 — 12月

上海

AiCon 全球人工智能开发与应用大会

会议时间：5月23-24日

- 通用大模型
- AI Agent
- 垂直领域应用
- 模型可解释性

深圳

AiCon 全球人工智能开发与应用大会

会议时间：8月22-23日

- 模型效率与部署
- 多模态大模型
- 大模型安全
- 智能硬件

北京

AiCon 全球人工智能开发与应用大会

会议时间：12月19-20日

- 通用大模型
- 智能硬件
- LMOps
- 具身智能

极客邦科技 2025 年会议规划

促进软件开发及相关领域知识与创新的传播

参会咨询

查看会议

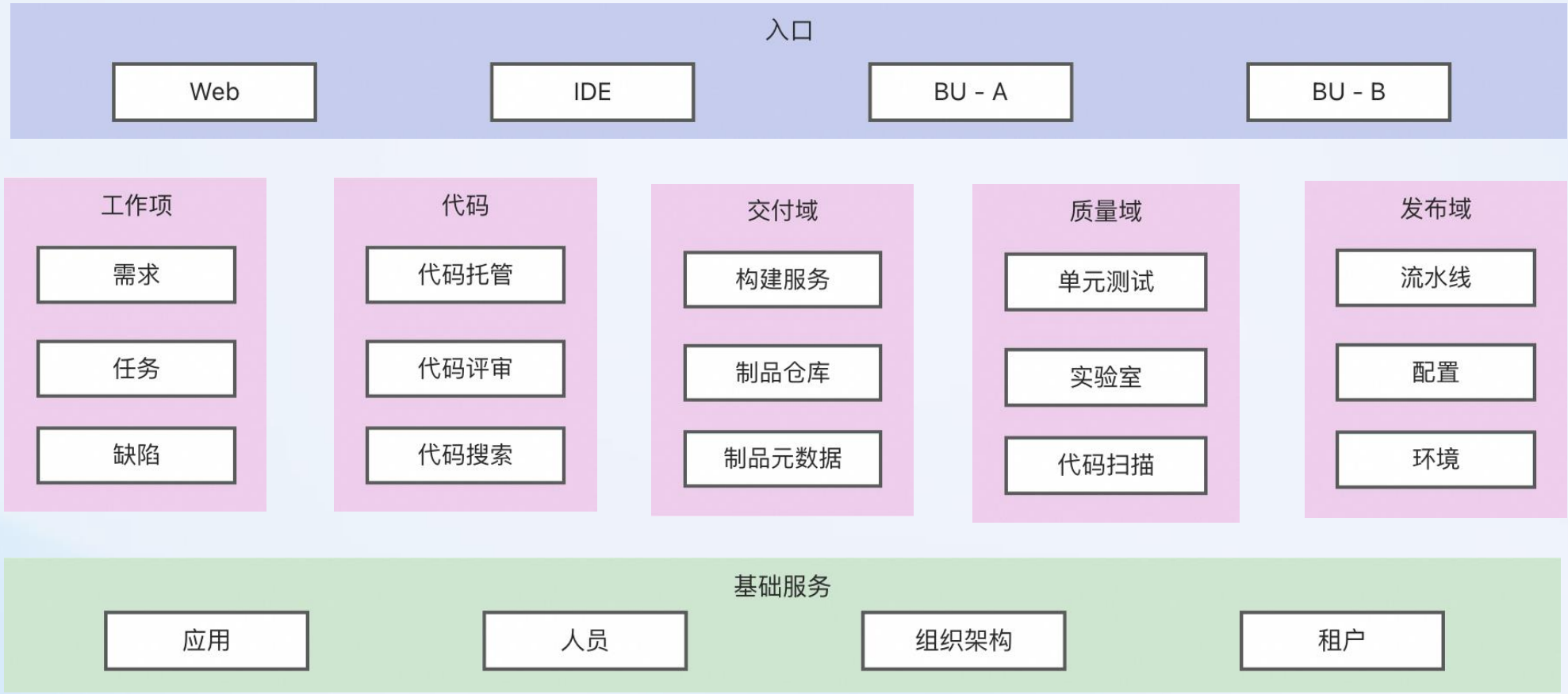
01

我们的工作

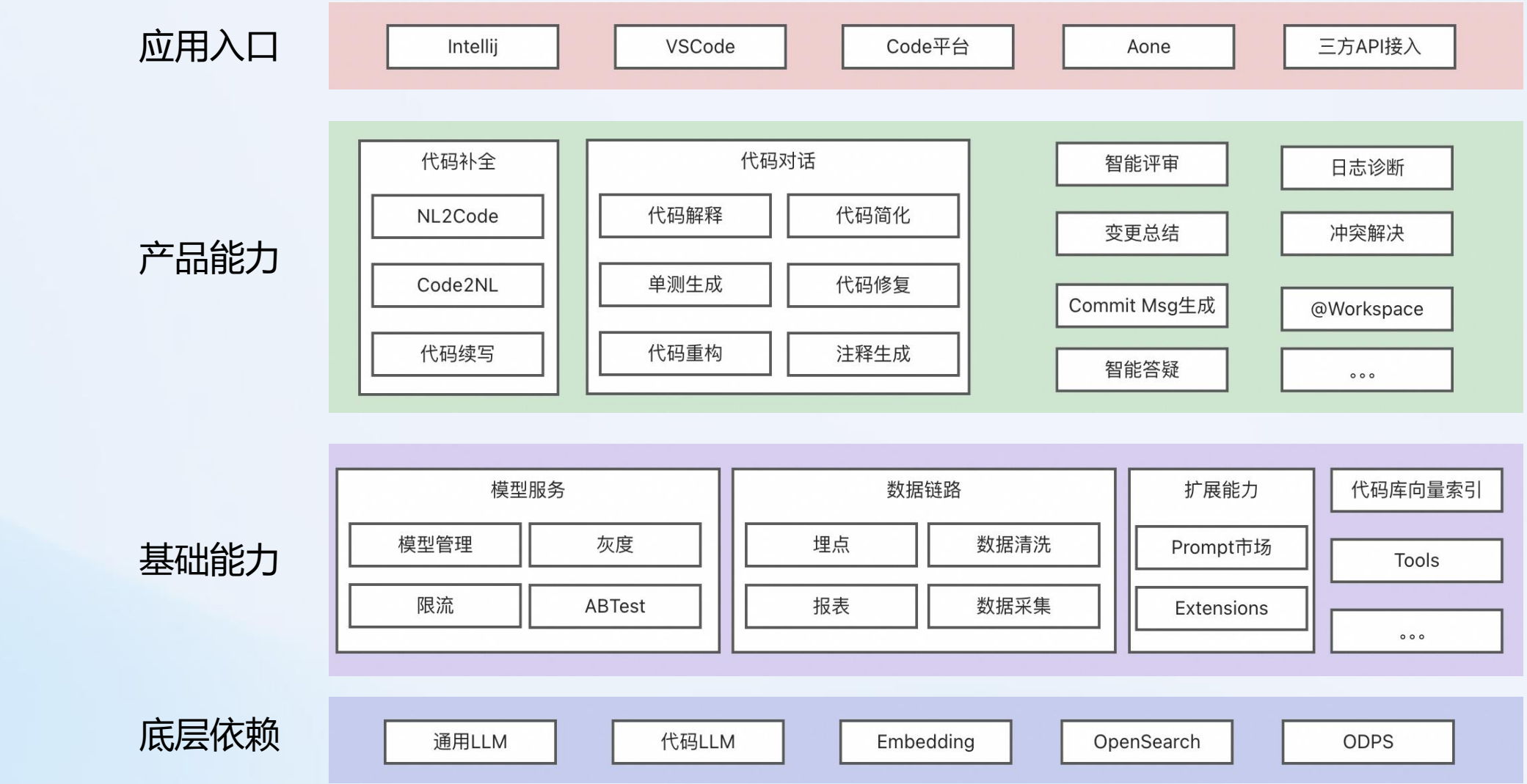
NES场景的探索

我们做了什么

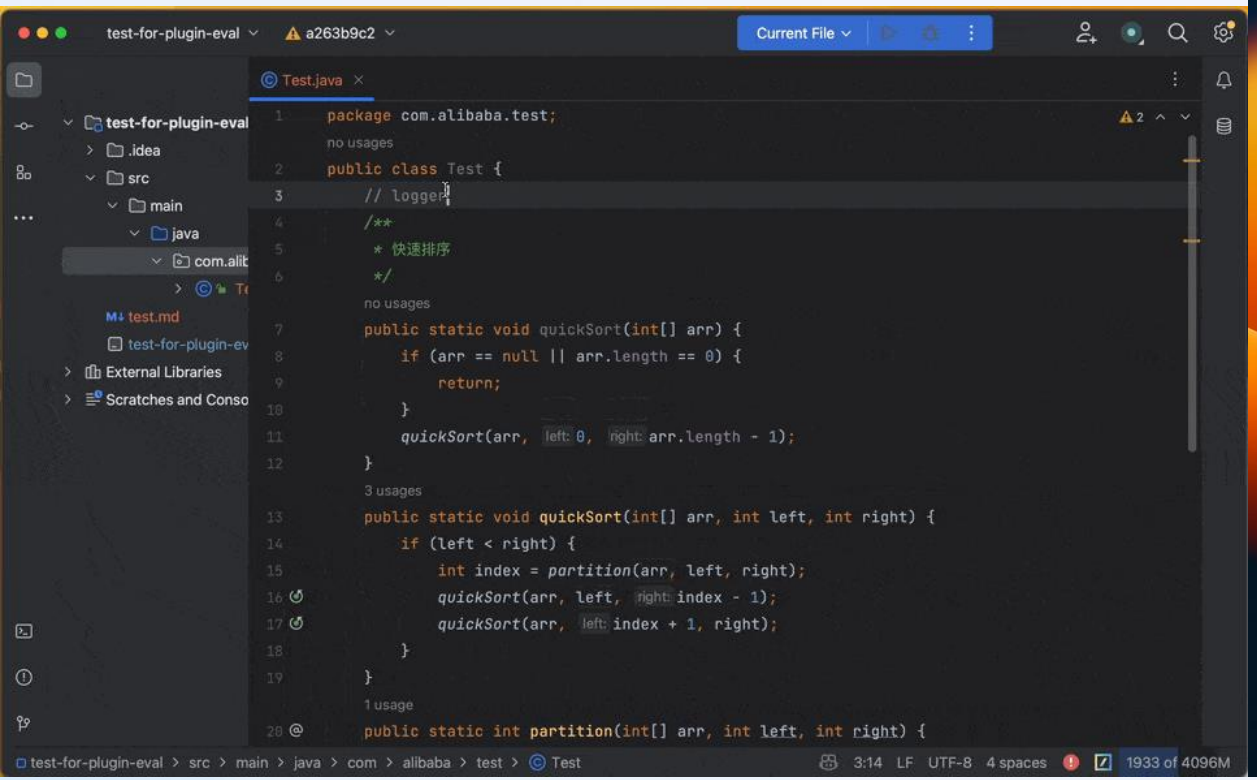
Aone大图

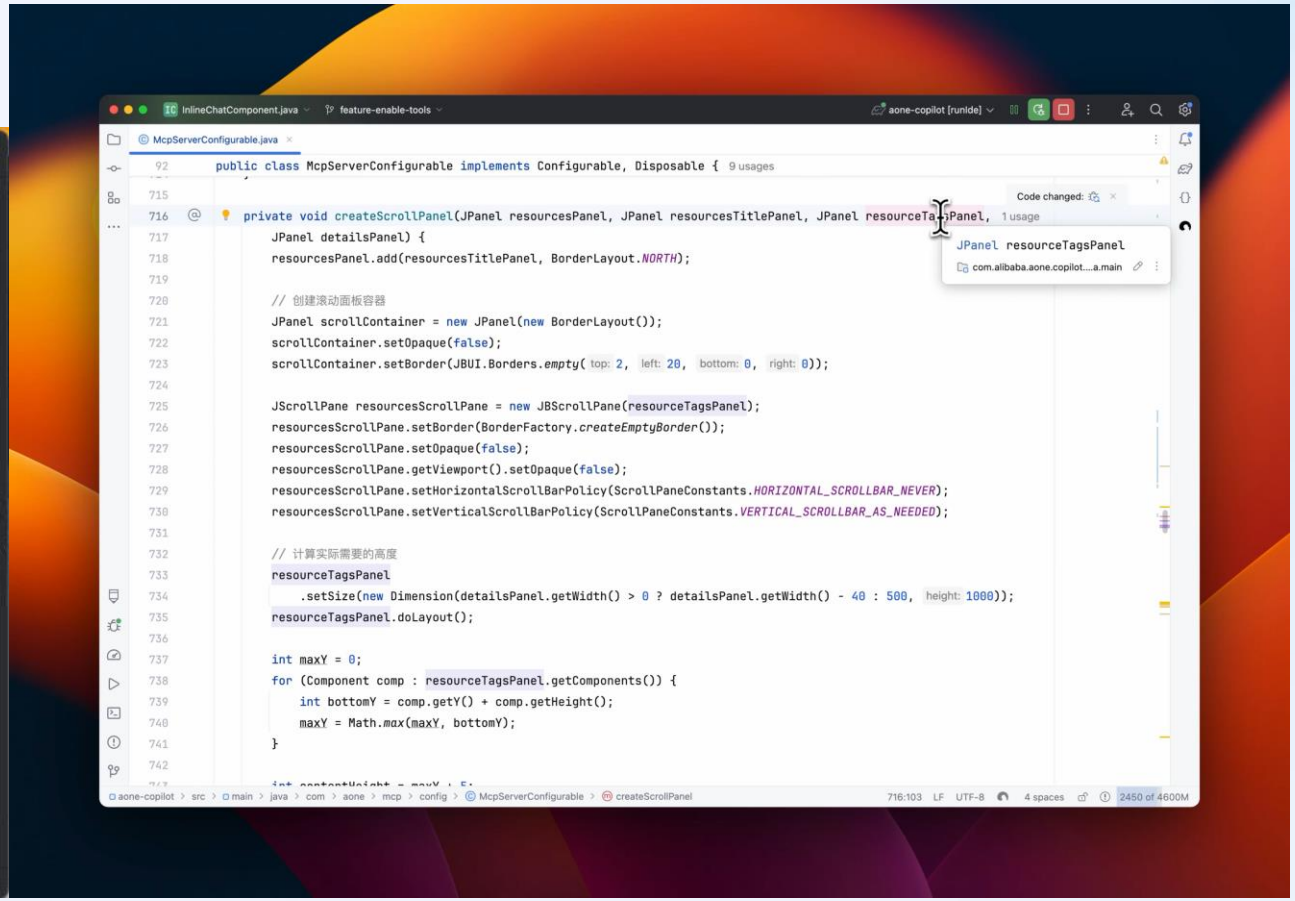


我们做了什么


应用入口
Intellij
VSCode
Code平台
Aone
三方API接入
产品能力
代码补全
NL2Code
Code2NL
代码续写
代码对话
代码解释
代码简化
单测生成
代码修复
代码重构
注释生成
智能评审
变更总结
Commit Msg生成
智能答疑
日志诊断
冲突解决
@Workspace
。。。
基础能力
模型服务
模型管理
灰度
限流
ABTest
数据链路
埋点
数据清洗
报表
数据采集
扩展能力
Prompt市场
Extensions
代码库向量索引
Tools
。。。
底层依赖
通用LLM
代码LLM
Embedding
OpenSearch
ODPS

常规补全和NES对比



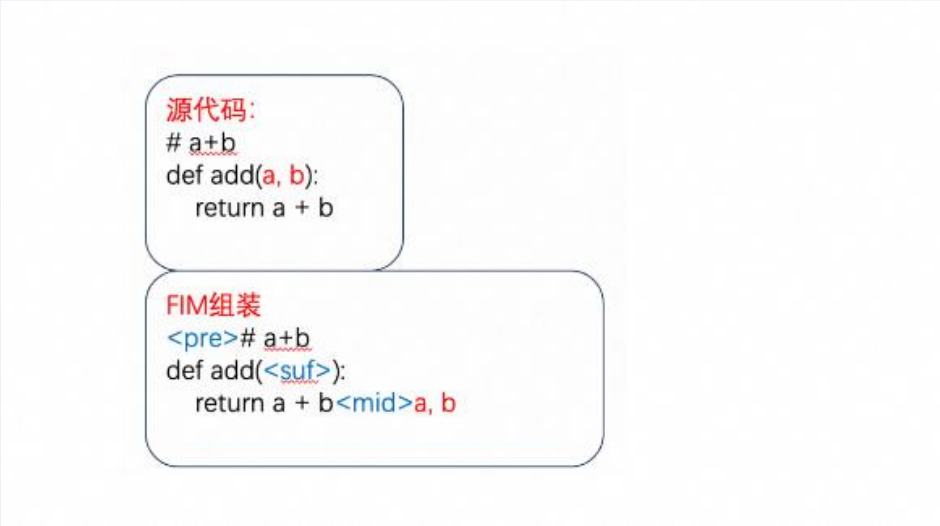


常规补全 ↑

NES ↑

补全的问题

- **无法修改既有代码**：补全范围被严格限制在光标位置，对于修改、重构前文代码的需求束手无策。
- **触发方式单一死板**：仅在用户输入字符或回车后触发。当用户希望在代码的任意位置（尤其是存在语法错误处）获得智能提示时，往往需要额外输入或删除字符来“唤醒”Copilot。
- **难以应对连续性任务**：当一项修改涉及多处代码时（例如，为方法增加一个参数，并更新所有调用点），传统补全需要用户手动跳转、多次触发，过程繁琐且效率低下。



数据格式探索

格式需要满足以下功能

- **修改范围：** 由指定位点->全文可控
- **能力范围：** 续写->增、删、改

待解决问题

- 有效信息片段选择
- 格式设计选择
- 输出样式满足修改范围和能力要求

输入格式：

```
### User Import Files
代码片段

### Similar snaps:
代码片段

### User Edits:
User edited "file_path_1":
Edit diff_1

### Lint Error:
Error信息

### User Excerpt:

```file path
---origin code
 <| editable region start |>
--编辑区域代码
-- cursor位置<cursor is here>
--编辑区域代码
 <| editable region end |>
----原始代码
```
Response:
```

输出格式：

```
<| editable region start |>

  Rewrite code snippet

<|editable region end|>
```

数据格式探索–输出格式

输出格式1：

<| editable region start |>

Rewrite code snippet

<|editable region end|>

输出格式2：

Code diff:

-start line num
-end line num
-code

输出格式3：

Rewrite code file

主要问题:

- 是否支持多处修改操作
- 模型推理要求
- 模型对数字敏感程度低

数据格式探索–输出格式

串联组合模型方案：

- 模型由location model + rewrite model

并联模型组合方案

- 模型由常规补全模型+NES模型

单模型方案

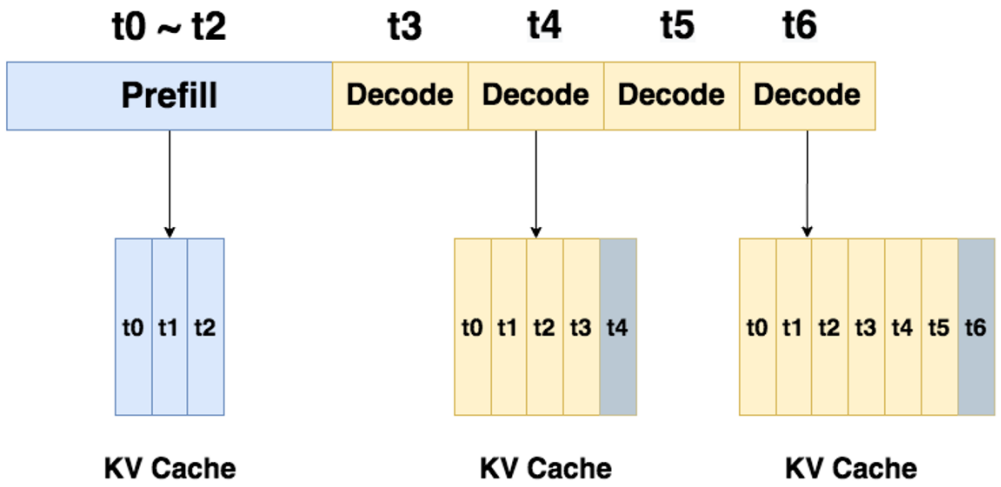
- 单一rewrite模型实现NES功能，同时具备单行补全和多行改写的能力。

	单行模型	NES模型
方式	FIM（fill in the middle）	Rewrite
能力	续写	增、删、改
输出token	avg 20token	avg 500token

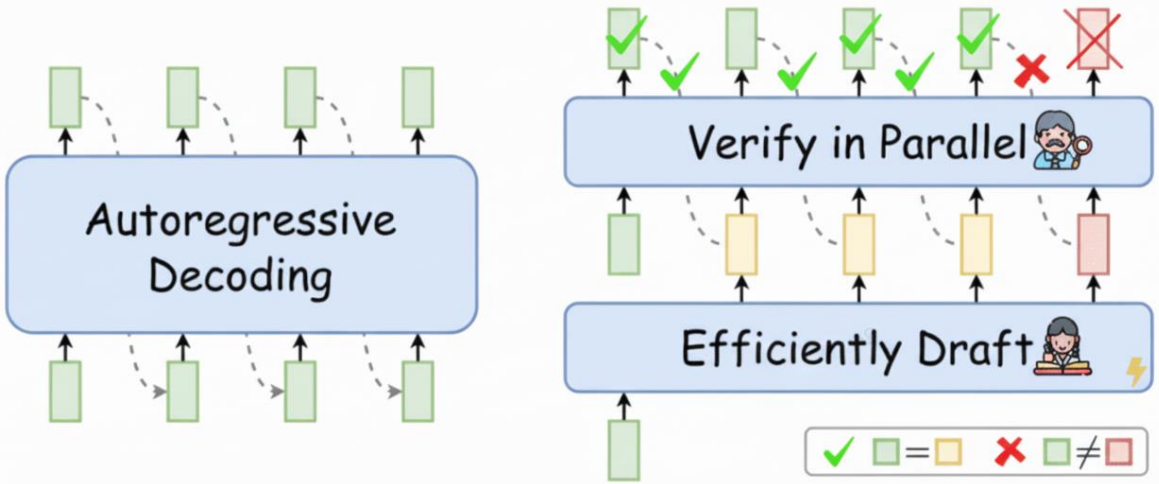
单行模型功能是NES功能的子集，续写是改写的特殊情况，其次效果上看，当前模型同时具备查找代码错误并修改代码错误的能力，且rewrite可避免行号偏移问题。

推理方式

LLM Inference



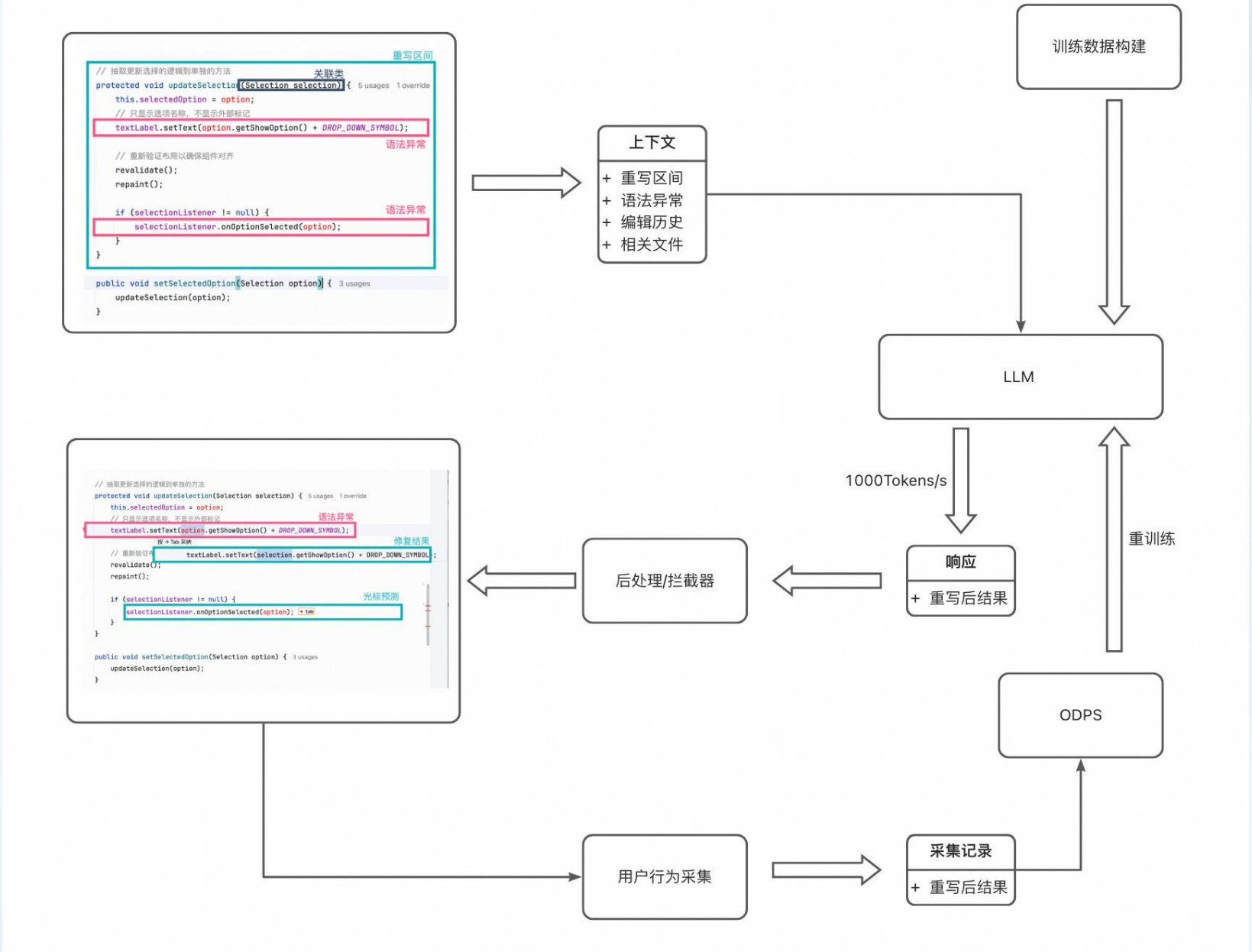
speculative decoding（投机采样）



- 推理速度1000token/s
- 基于字符串/token匹配的投机采样模型

方案确定

1. 有效信息
2. 推理速度
3. 迭代方案



02

数据构建

NES场景的探索

数据构建

NES（next edit suggestion）：根据用户输入的内容，预测代码文件中需要修改的内容，预测的动作是用户时间序列操作队列的延续，同时也是对用户编写代码中引入问题的修正，是一个功能更全面的代码辅助工作模型。

关键要素：

- 用户编辑历史（时间序列）
- 完整的代码独立单元（代码操作实现一个功能点）
- Repo级别代码（跨文件修改代码）

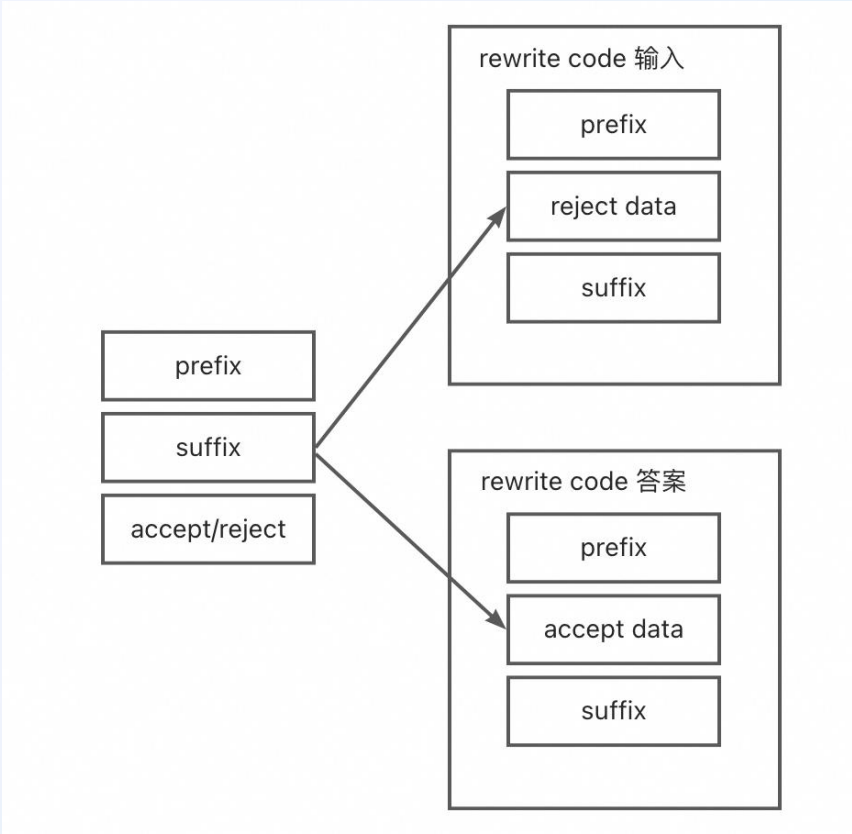
数据构建–单行补全扩充方案

1.用户pair数据：有用户拒绝数据和用户拒绝之后修改的数据，可以作为多行编辑的修改内容，用户从错误的数据修改为正确数据

2.用户采纳数据：

1. 以行为单位，如果采纳数据是原本数据扩写，我们可以认为是增删改中改场景。
2. 以行为单位，如果采纳数据是完整行生成，可以认为是NES中增场景

3.用户拒绝数据：可以作为删除场景的数据。



数据构建–AST构建方案

AST：抽象语法树（Abstract Syntax Tree，AST），或简称语法树（Syntax tree），是源代码语法结构的一种抽象表示。它以树状的形式表现语言的语法结构，树上的每个节点都表示源代码中的一种结构。

场景分类：

1. 变量名修改
2. 增加函数体
3. 函数入参变更

优点：快速、规则化

缺点：场景泛化能力不够，同意代码表现不佳

```
178
179  def check_data_formate():
180      """
181      检查数据
182      """
183      right_num = 0
184      with open("/home/data.jsonl", "r", encoding='utf-8') as infile:
185          datas = [json.loads(line) for line in infile]
186          for data in datas:
187              if data["output"] == "**YES**" :
188                  right_num += 1
189              elif data["output"] == "**NO**" :
190                  right_num += 1
191              #print(data)
192      print(f"right_num is {right_num}, total_num is {len(datas)}, accuracy is {right_num / len(datas)}")
193
```

Show Full Tree　　TreeSitter Output

```
- block body (179, 4)-(191,103)
  - expression_statement (179, 4)-(181,7)
    - string (179, 4)-(181,7)
        string_start (179, 4)-(179,7)
        string_content (179, 7)-(181,4)
        string_end (181, 4)-(181,7)
  - expression_statement (182, 4)-(182,17)
    - assignment (182, 4)-(182,17)
        identifier left (182, 4)-(182,13) Matched
        = (182, 14)-(182,15)
        integer right (182, 16)-(182,17)
  - with_statement (183, 4)-(190,24)
      with (183, 4)-(183,8)
    - with_clause (183, 9)-(183,66)
      - with_item (183, 9)-(183,66)
        - as_pattern value (183, 9)-(183,66)
```

数据构建–PR构建方案

PR数据：使用高质量pull request数据，每条数据满足单一功能开发需求。

问题：

- 如何构建用户编辑顺序
- 如何构建编码顺序

优点：方案简单高效、数据源头广

缺点：数据不真实、依赖模型能力和数据复杂度

数据构建–日志构建方案

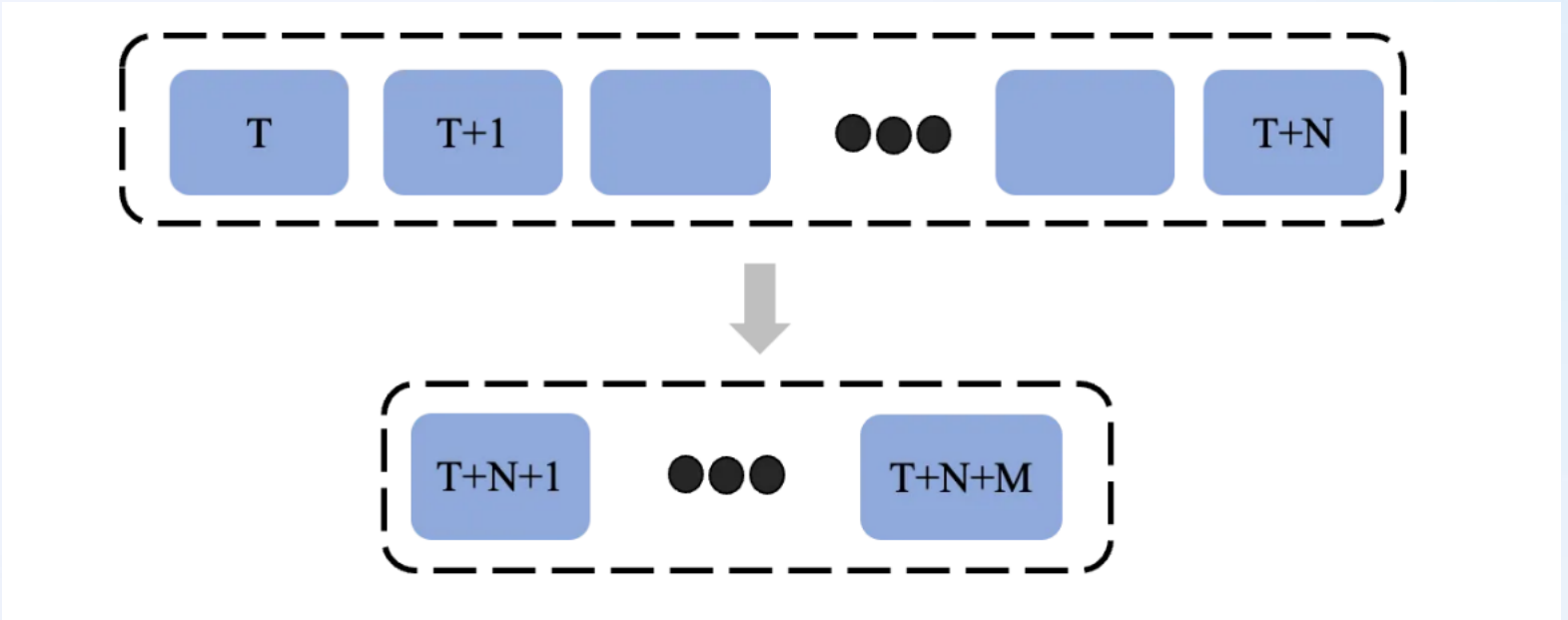
用户日志数据：数据包含用户编辑历史、代码lint error,可以获取repo级别用户日志数据。

优点：数据真实

缺点：依赖模型能力、工程链路复杂

数据构建–日志构建方案

用户日志数据：插件端打点用户编辑历史，记录用户编辑历史事件序列，聚合最小操作单位，截取片段时间序列作为输入，后续代码片段作为目标。



优点：数据真实

缺点：聚合逻辑复杂，用户type时机和跳转逻辑不符合用户编辑历史产出结果。

02

模型训练

NES场景的探索

模型训练


数据来源层
AST数据
PR数据
单行扩充数据
日志数据
数据处理层
规则筛选
数据配比实验
模型训练层
SFT训练
根据模型评测结果调整训练数据
DPO微调
针对Base Case产出pair数据
RL调整
优化训练
反馈循环层
用户线上采纳行为
KTO调整模型

模型训练–sft

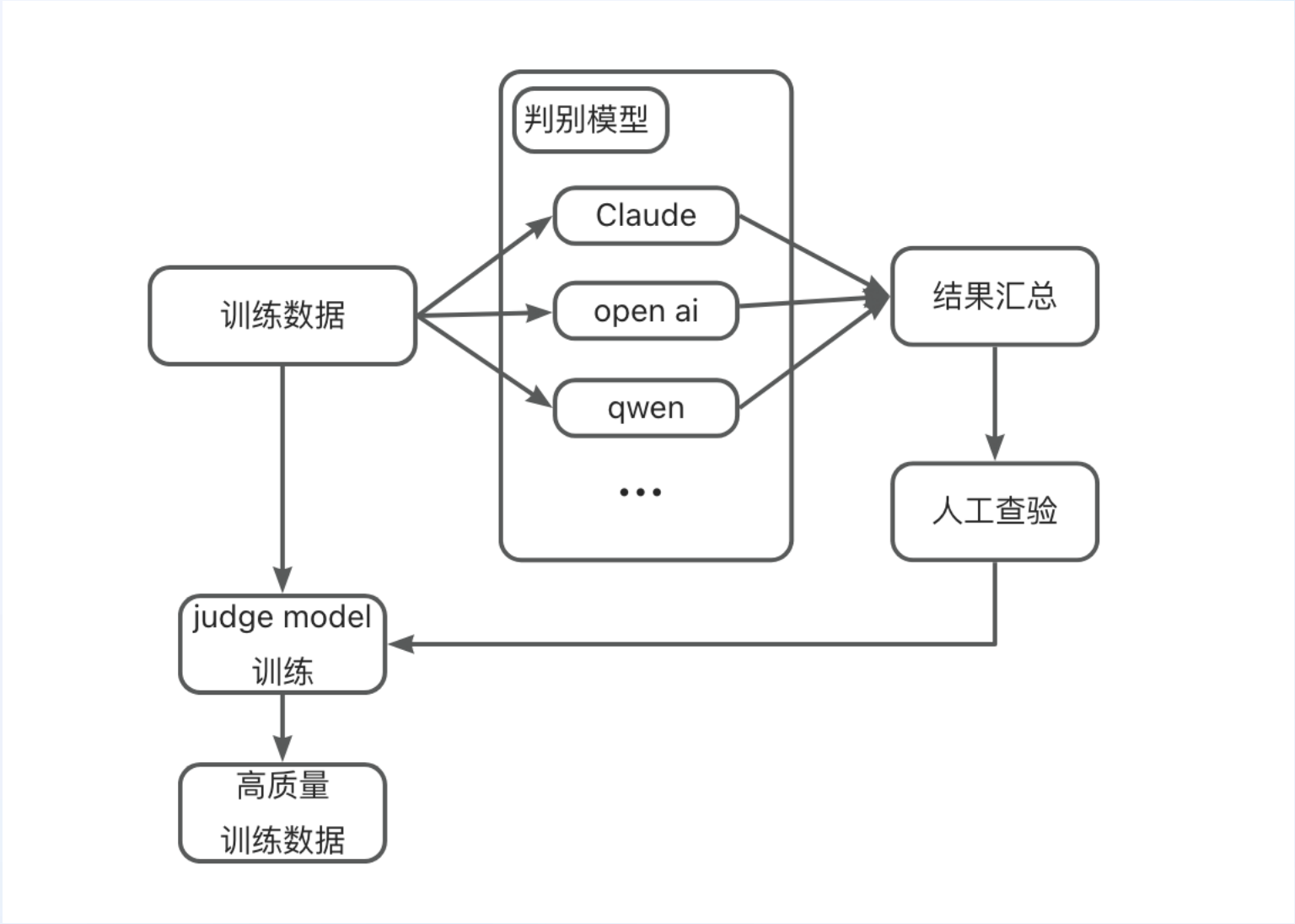
模型微调任务：

1. 输入信息是什么，模型要学习的知识是什么？信息和知识能否支撑起这个任务？
2. 如何在尽可能少的变动（训练）下完成任务，除非存在大量(100k+)的数据，比如模型输入输出format变化，对话格式变更等。
3. 模型训练中loss下降是必然的，算法收敛并不意味着现实中有效，反应业务实际能力的评测至关重要。
4. 数据顺序，数据质量 > 数据种类 > 数据数量，引入低质量数据杀伤力远大于高数据量，如果loss一直剧烈波动，有理由怀疑数据质量较差，数据方差太大。

模型训练–数据质量

数据质量：

1. LLM模型执行判别任务精度高于rewrite任务。
2. 结果汇总阶段，可以通过调整权重策略，控制数据精度
3. 自训练判别模型7B够用，经多组LLM判别之后数据的人工查验准确率高达90%，SFT之后的小模型准确率能到达92%



模型训练–SFT

Base model: Qwen coder

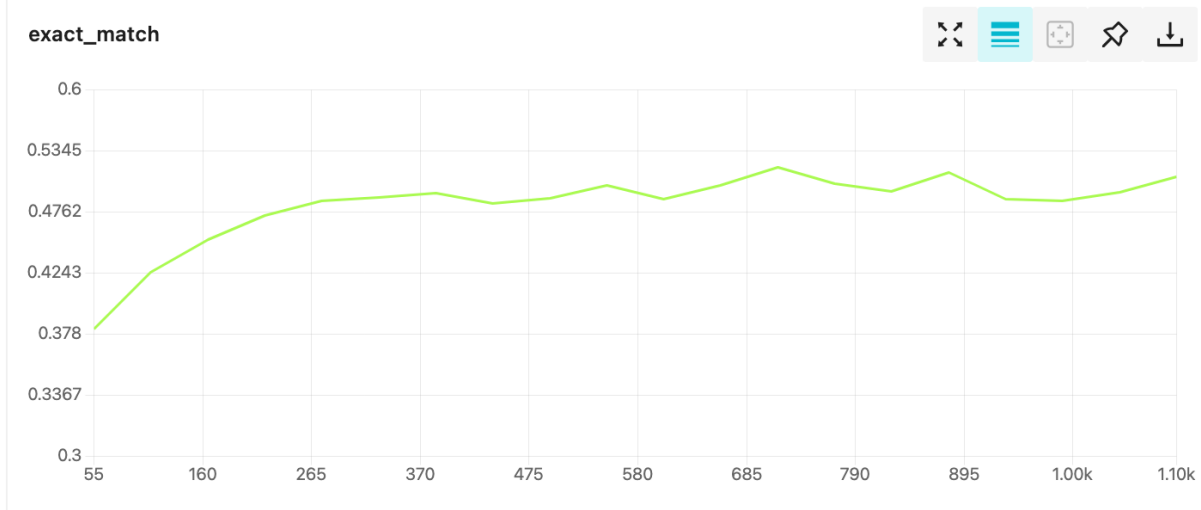
参数迭代方式：全量微调

数据量：20W

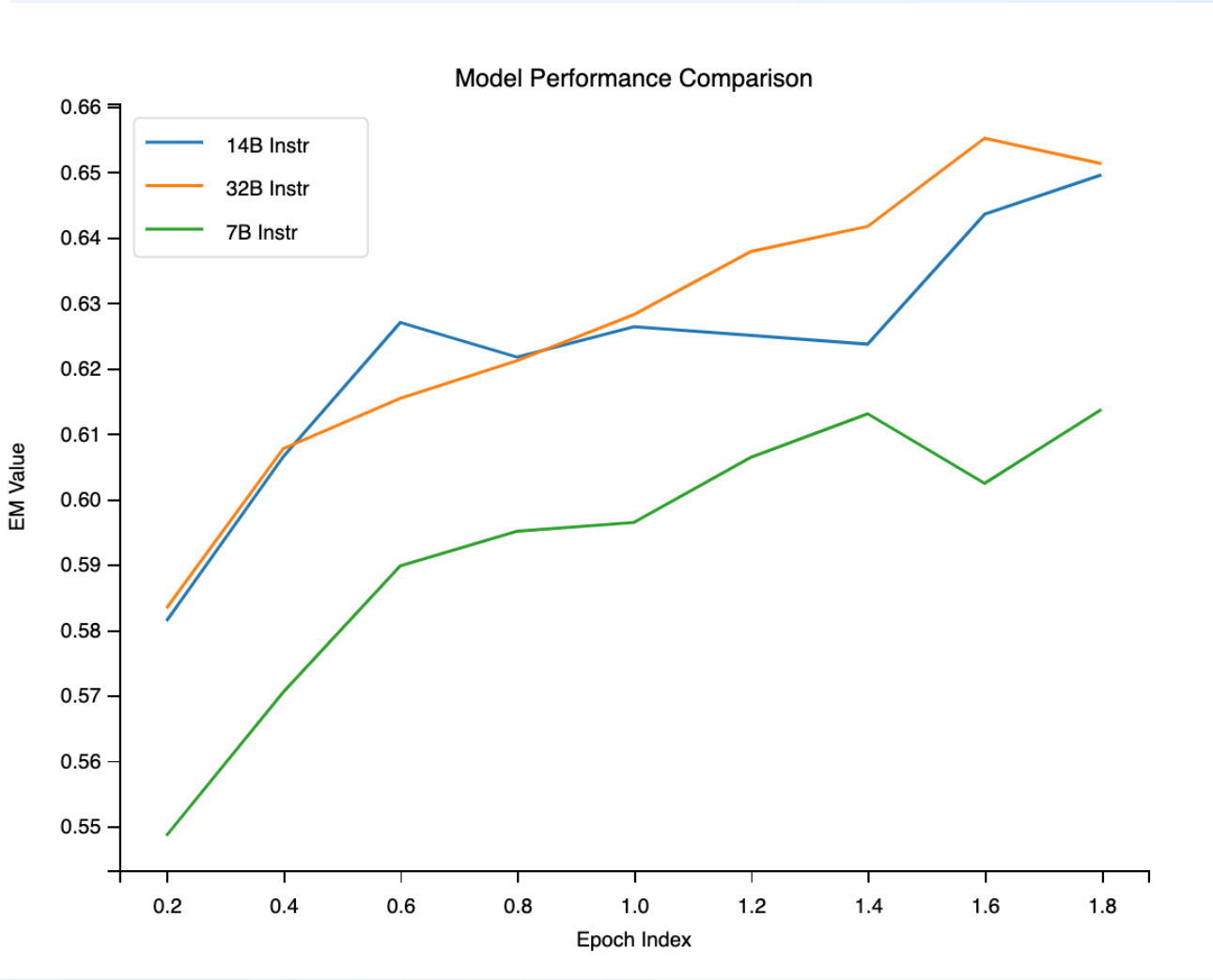
目的：模型输出符合设计样式，模型最终满足三个要求

1. 模型按照指定区域输出rewrite代码片段
2. 输出pattern满足设计需求
3. 离线指标符合要求（EM和ES）

模型训练–sft



1. Model size 不是越大越好。
2. 数据量少的时候，instruct模型效果要优于base，尤其是泛化能力会更强一些
3. 此场景epoch最优在5



模型训练

Bad case修复:

- 模型roll back问题（概率上符合模型当前推理内容，但是用户编辑历史中已经删除此处操作）
- 模型修改用户确信操作
- 不符合用户习惯数据

目的:

- DPO使用pair对数据，修复bad case，实现bad case 解决率提升50%，且原模型性能下降可控，3%以内
- KTO使用用户采纳数据和拒绝数据，拟合用户采纳习惯，采纳率提升10%

模型训练– RL

GRPO调整模型

规则设计

- Format一致性：根据special token的位置和出现准确率设置打分。
- Reward model 判别：判别模型能做到90%的准确率，可以使用判别模型作为打分依据。
- 语法正确性:使用AST工具，判断产出代码是否存在语法异常问题。

经验:

1. 冷启动SFT效果很重要，需要使模型具有格式和基本能力，保证sample多样性和输出稳定
2. Rollout采样数据尽量多一点。
3. 效率很重要，资源分配、异步等。

02

模型评测

NES场景的探索

模型评测

离线评测

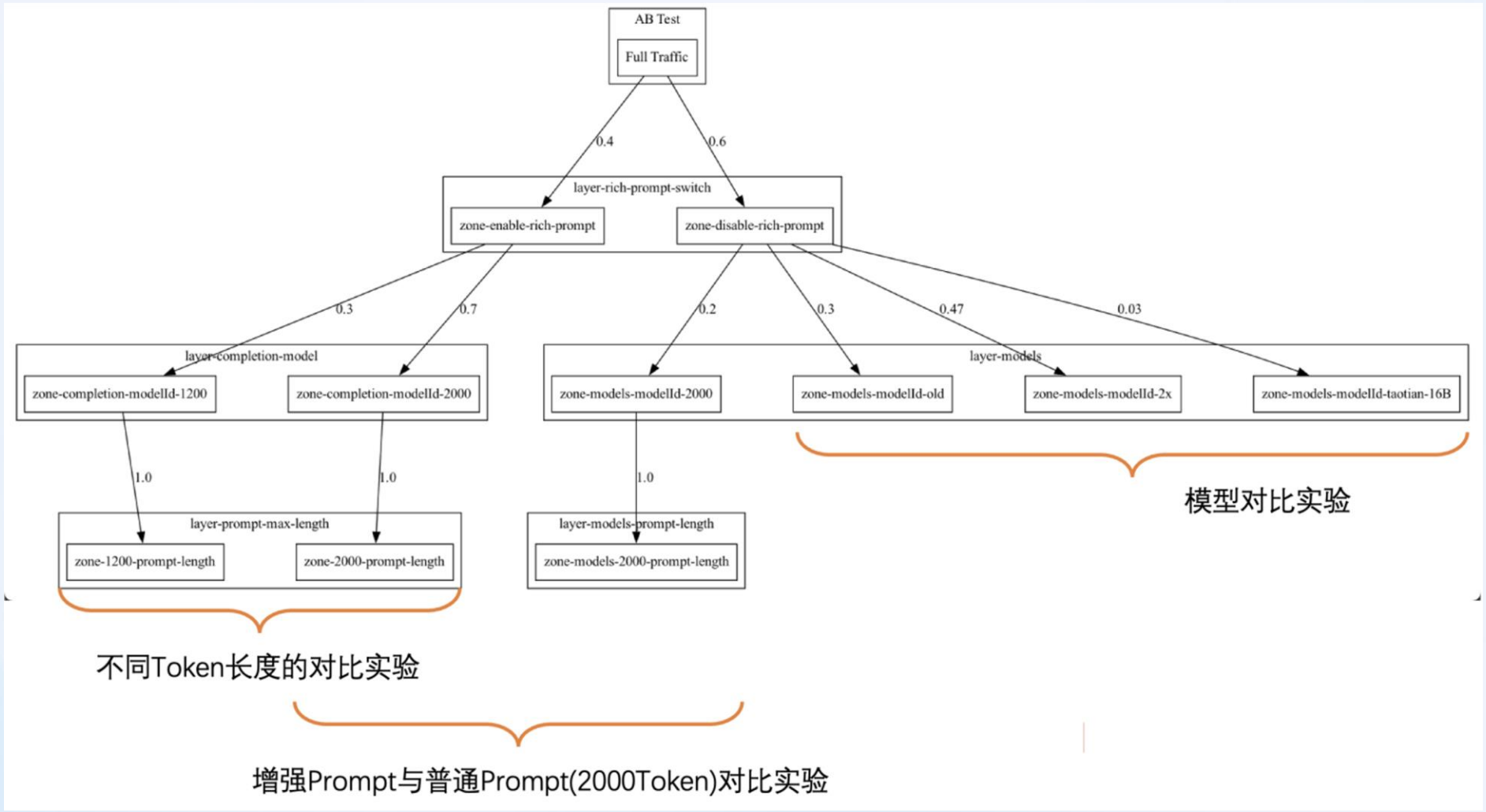
AB Test

线上评估

模型评测–离线评测

评测手段	评测方式	优点	缺点
模型评测	LLM对产出结果打分评测	快速高效	评测结果依赖LLM能力和inference质量
语法正确率	抽取评测集后检查补全结果的语法正确率	评测比较简单	上限很低
Exact Match&Edit Similarity	抽取评测集并标注正确结果后，检查补全结果是否一致/编辑距离	比较完善的评测补全效果，尤其是bad case 评测	尚无法反映文本不同但语义相同的写法
全链路复现	根据用户操作历史，在插件端回放输入，计算补全占比	完整端到端的评测，除了结果质量还考虑到延时的因素	影响结果的参数较多，无法完美复现

模型评测


AB Test
Full Traffic
0.4
0.6
layer-rich-prompt-switch
zone-enable-rich-prompt
zone-disable-rich-prompt
0.3
0.7
0.2
0.3
0.47
0.03
layer-completion-model
zone-completion-modelId-1200
zone-completion-modelId-2000
layer-models
zone-models-modelId-2000
zone-models-modelId-old
zone-models-modelId-2x
zone-models-modelId-taotian-16B
1.0
1.0
1.0
layer-prompt-max-length
zone-1200-prompt-length
zone-2000-prompt-length
layer-models-prompt-length
zone-models-2000-prompt-length
模型对比实验
不同Token长度的对比实验
增强Prompt与普通Prompt(2000Token)对比实验

模型评测

构建用户报表，持续跟踪模型能力，主要包含以下几点

- 采纳率：用户采纳请求数（去重）/展示请求数（去重）
- 采纳行：用户实际采纳的代码行数（包括修改和新增的行数
- 推理耗时：推理耗时在650ms，最符合用户习惯
- 用户完全采纳率：用户采纳之后没有修改，完全保留模型推荐数据
- 采纳行为分析：包含代码语言、代码行数、用户行为分类（增删改）。

02

未来展望

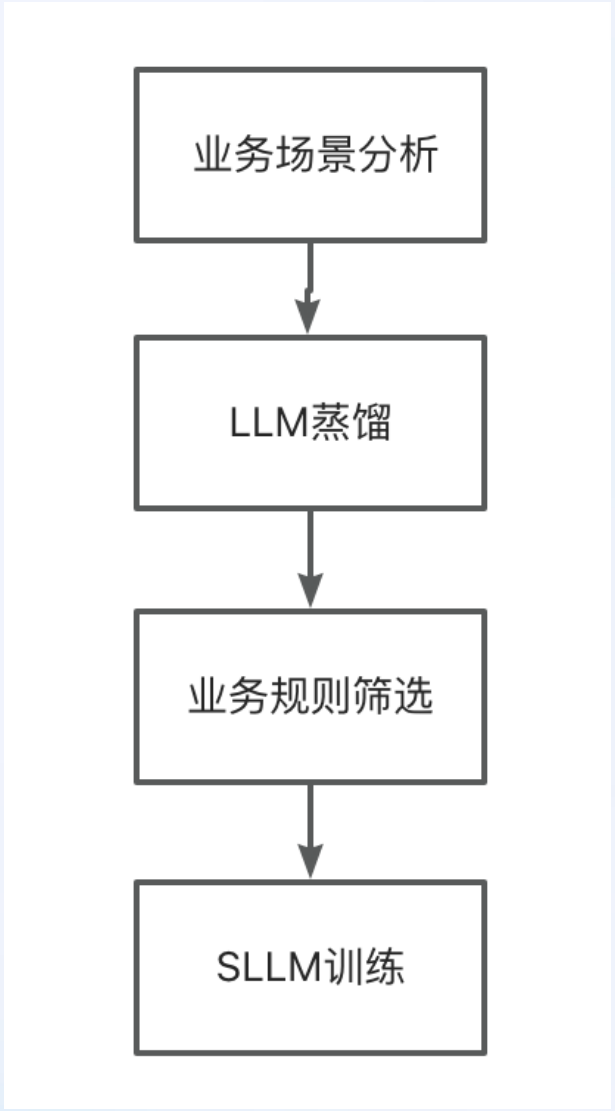
NES场景的机遇与挑战

SLLM VS LLM

业务满足：

1. LLM能力冗余，至少到60%以上能力
2. 业务有清晰规则可以验证结果准确性

在内部多个场景中验证，在1B ~ 14B的SLLM上相较于LLM均有50%以上的效果提升，且推理成本远低于大模型



业务模型

1. 连续过程行为数据：需要增加用户行为数据，训练中考虑用的采纳和拒绝等行为，增加模型的拒答能力，尤其是多次错误提示之后的拒答能力
2. 模型泛化能力提升，构建agentic RL环境。
3. 模型推理速度提升
    a) 增加数据格式探索，比如行号信息，模型能清晰输出需要修改的行号信息，降低推理负担。
    b) Diffusion模型推广：现阶段SP edit 的推理速度满足模型部署需求，但是会存在资源消耗量大等问题，需要在diffusion的基础上

问题

用户群体A:

深度使用LLM产品，认为模型只能完成自己70%的工作，尤其是复杂项目，模型很难一次性做好工作，会出现多个bug间反复拉扯的情况，甚至出现设定好的test case多次不通过之后被模型强行修改test case的情况。

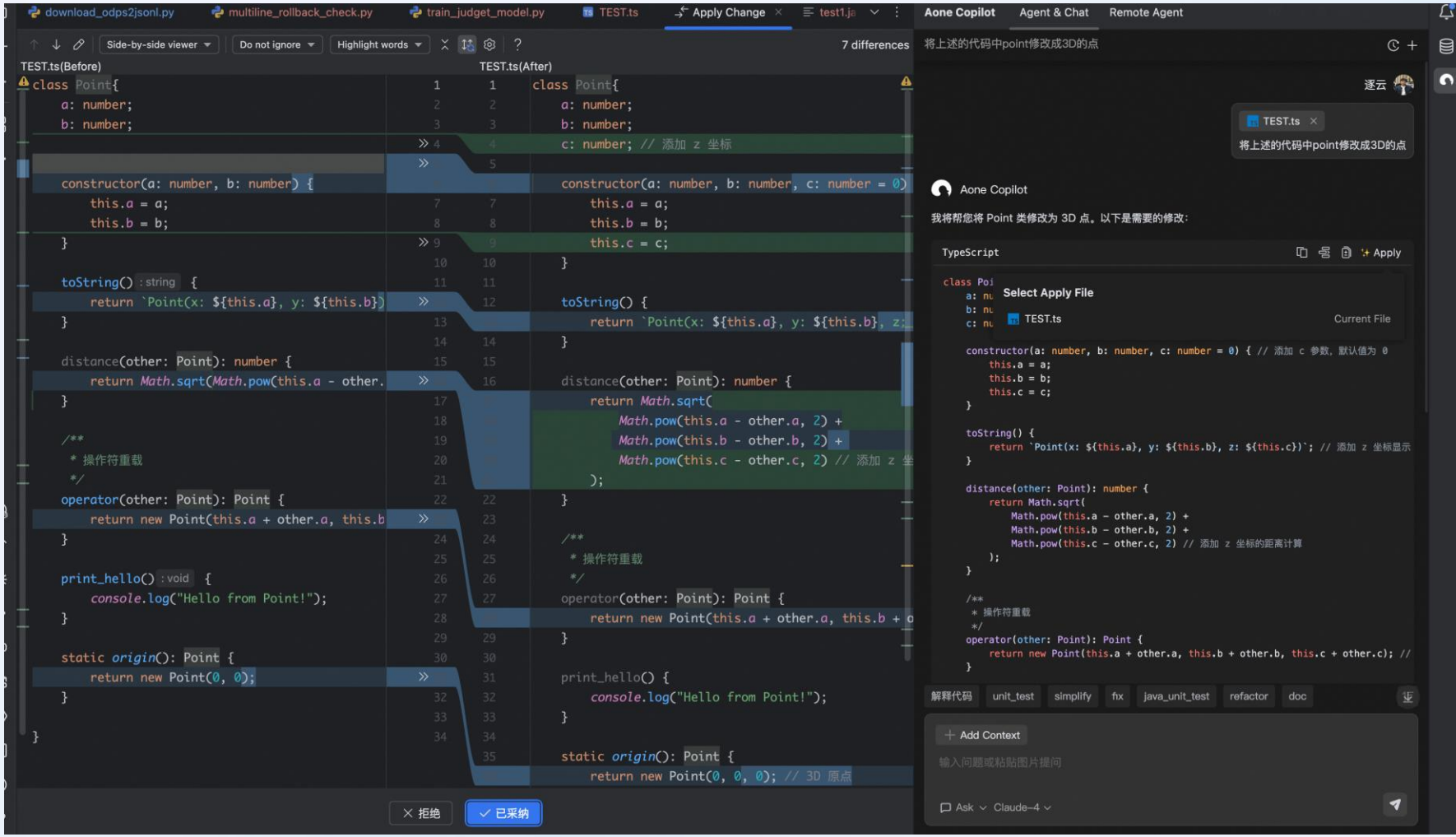
用户群体B:

现阶段已经很少用IDE写代码，使用Claude code、augment、Aone Agent等agent工具，能够做到vibe 编程，认为机写的代码比手写的代码好，且程序员的工作即将被LLM取代

最佳实践

Agent + NES 组合编码方式

- 针对大项目使用remote agent，创建项目并完成需求
- 小需求使用ide agent去产出初版修改
- 用户自身根据自己理解使用NES进一步迭代自己需求



InfoQ 极客传媒

北京

QCon 全球软件开发大会

会议时间：4月10-12日

- 大模型赋能 AIOps
- 越挫越勇的大前端
- 云上业务架构演进
- 多模态大模型及应用
- 海外 AI 应用创新实践

北京

AiCon 全球人工智能开发与应用大会

会议时间：6月27-28日

- 端侧智能
- AI Agent
- 多模态大模型
- 金融+大模型

上海

QCon 全球软件开发大会

会议时间：10月23-25日

- AI Agent
- 大模型训练推理
- 端侧 AI
- 搜推深度融合
- 数智金融

4月 — 5月 — 6月 — 8月 — 10月 — 12月

上海

AiCon 全球人工智能开发与应用大会

会议时间：5月23-24日

- 通用大模型
- AI Agent
- 垂直领域应用
- 模型可解释性

深圳

AiCon 全球人工智能开发与应用大会

会议时间：8月22-23日

- 模型效率与部署
- 多模态大模型
- 大模型安全
- 智能硬件

北京

AiCon 全球人工智能开发与应用大会

会议时间：12月19-20日

- 通用大模型
- 智能硬件
- LMOps
- 具身智能

极客邦科技 2025 年会议规划

促进软件开发及相关领域知识与创新的传播

参会咨询

查看会议

QCon
全球软件开发大会

InfoQ 极客传媒

THANKS

大模型正在重新定义软件

Large Language Model Is Redefining The Software