ISCAS 中国科学院软件研究所学术年会'2024
暨重点实验室科技活动周

学术论文

当大型语言模型面对存储库级别自动程序修复时：它们做得如何？

陈昱晓，吴敬征，凌祥，李长江，芮志清，罗天悦

ICSE 2024 Industry Challenge Track

陈昱晓，chenyuxiao2021@iscas.ac.cn，19818965315

研究背景 RESEARCH BACKGROUND

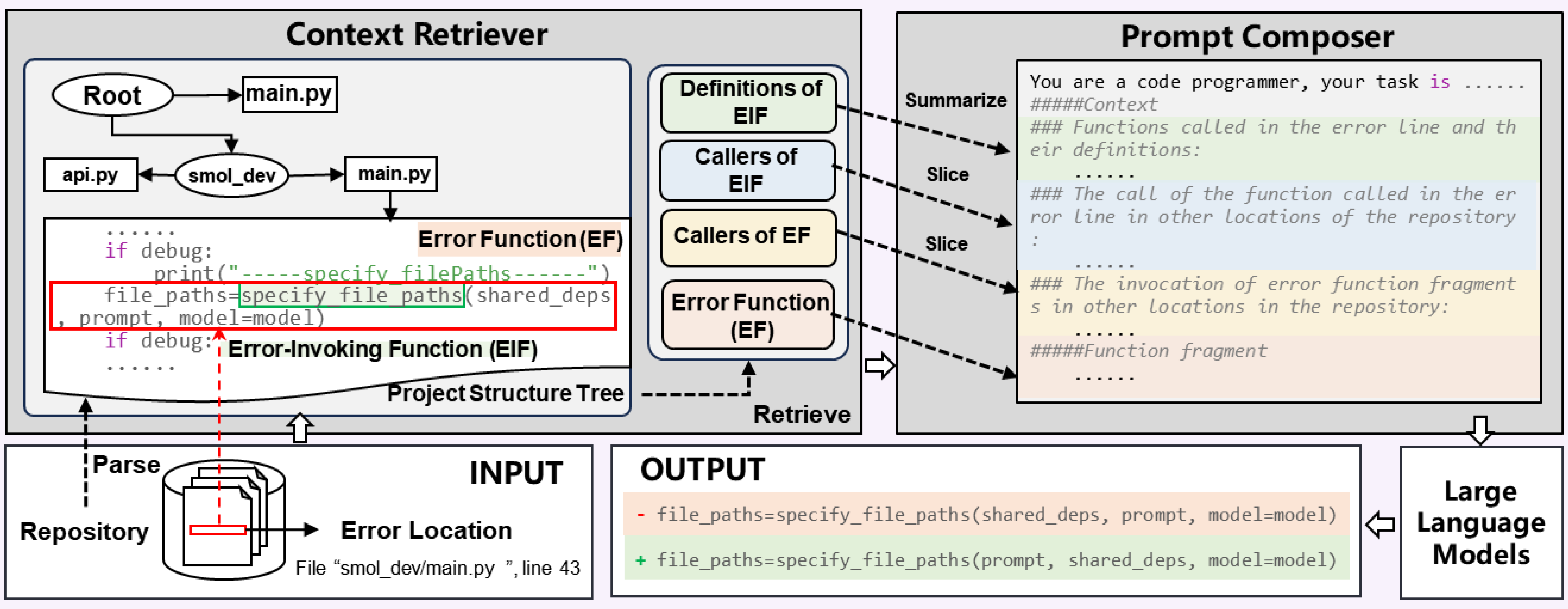
程序自动修复任务一直以来是软件工程中的一个重要挑战。最近，大语言模型出色的生成以及理解代码的能力为程序自动修复带来了新的潜在解决方案。相关研究表明，大语言模型在处理程序自动修复任务上具有很强的竞争力。程序自动修复任务可以根据修复bug所依赖的上下文规模分为函数级别以及存储库级别。当前基于大语言模型对的研究只局限于函数级别，而对于需要使用更广泛存储库级别的任务的表现仍然未知，并且缺乏合适的针对于存储库级别任务的数据集

主要贡献 MAIN CONTRIBUTIONS

- 率先研究了主流的大语言模型在处理存储库级别程序自动修复任务中的性能
- 构造了一个新的基于GitHub开源存储库的基准数据集，RepoBugs，包含124个典型的存储库级bug。据我们所知，这是第一个专门为存储库级程序修复而设计的基准数据集
- 提出了一种简单而通用的方法RLCE，它能为存储库级程序修复任务提供更精确的上下文，与原始方法相比，在所有实验大语言模型上的修复率都提高了100%以上

关键技术 KEY TECHNOLOGY

- **存储库级别上下文提取方法（RLCE）**
  - **上下文提取器（Context Retriever）**：主要解决从存储库中何处检索以及获取何种上下文的问题。它是一种静态代码分析工具。其总体结构如下图所示，主要由两个关键步骤组成：解析存储库文件并构建项目结构树；根据错误位置在项目结构树中进行检索，以获得所需的代码段。
  - **提示生成器（Prompt Composer）**：主要功能是接收上下文提取器的输出代码片段，并根据得到的代码片段做进一步的处理，然后将处理后的代码片段根据不同提示策略的模板嵌入，以生成大型语言模型的最终提示。对于不同上下文源的处理方法包括代码摘要、补充额外语义信息以及代码切片等操作。



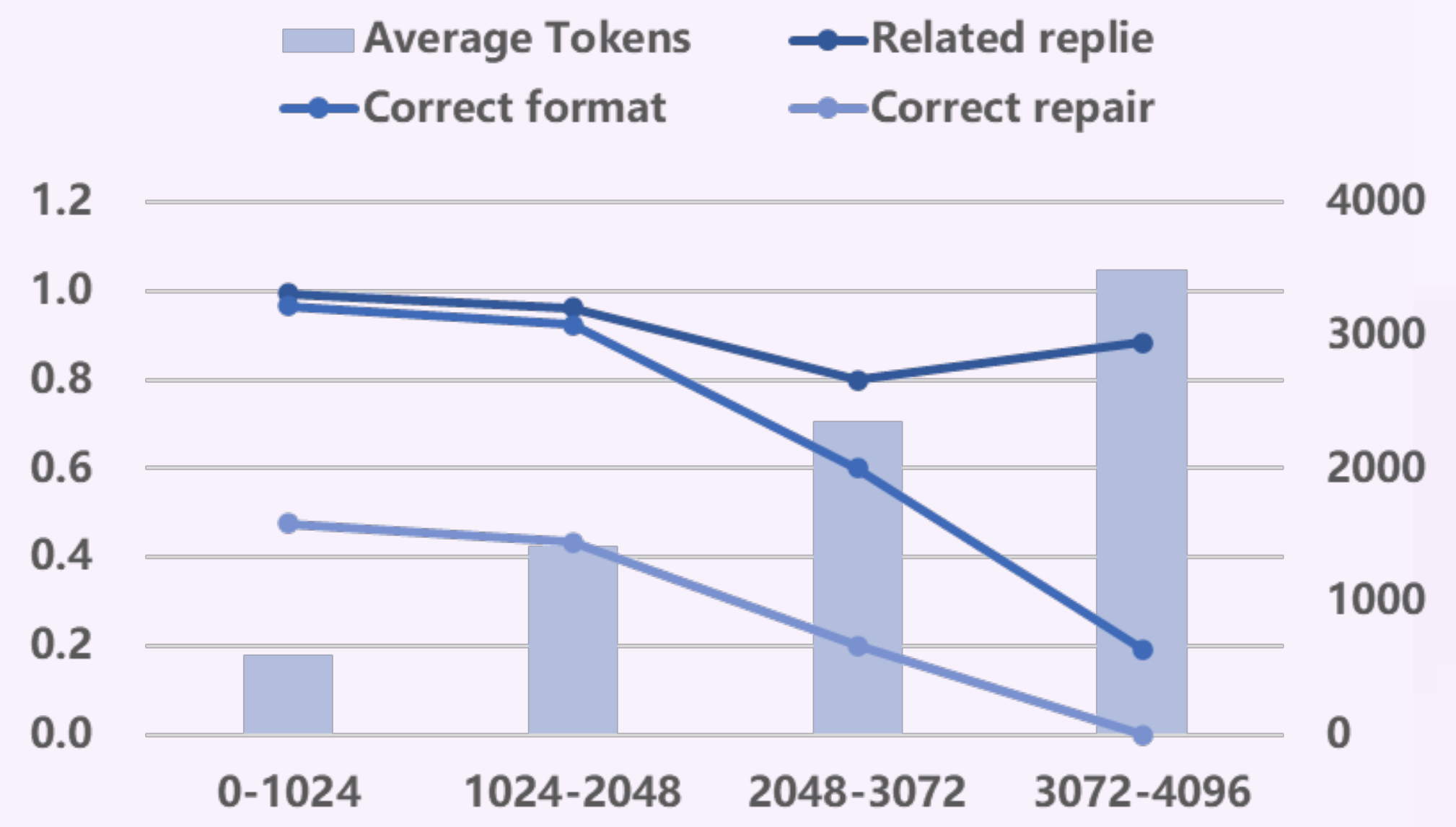
```
    ......
    if debug:
        print("-----specify filePaths------")
    file_paths=specify_file_paths(shared_deps
, prompt, model=model)
    if debug:
    ......
```

```
You are a code programmer, your task is ......
#####Context
### Functions called in the error line and th
eir definitions:
    ......
### The call of the function called in the er
ror line in other locations of the repository
:
    ......
### The invocation of error function fragment
s in other locations in the repository:
    ......
#####Function fragment
    ......
```

```
- file_paths=specify_file_paths(shared_deps, prompt, model=model)
+ file_paths=specify_file_paths(prompt, shared_deps, model=model)
```

实验效果 EXPERIMENTAL RESULTS

Model	Method					
	Preliminary	Slice-similarity	RLCE			
			Simple	Detail	One-shot	CoT
GPT3.5	0.2258	0.3387	0.4113	0.5645	**0.5968**	0.5161
PaLM2	0.2177	0.2419	**0.4272**	0.3952	0.4032	0.2742
GPT4	0.4113	0.4919	0.7742	0.7581	**0.8145**	0.75



- 与原始方法和切片相似性方法对比，RLCE均有了显著的改进，与原始方法相比修复率普遍提高了100%以上，GPT3.5最高，达到了160%
- 采用原始方法所有模型的修复效果都不佳，即使是最优的GPT4也只能获得41.43%的成功率，很难仅凭借函数级有限的上下文来完成存储库级的修复任务
- 切片相似性方法的修复率都没有超过我们的方法，切片相似性方法检索与错误位置相似的代码段，仅仅依赖于代码的相似性使得重建错误位置前后的实际执行的上下文环境具有挑战性，难以正确推断错误的原因
- 随着上下文规模的增大，修复效果反而出现了下降趋势，上下文应尽可能精确，减少冗余信息

This paper is supported by YuanTu Large Research Infrastructure