

**信息科学技术学院**

**《大模型的代码缺陷检测能力分析报告》**

**学生姓名** 魏鹏超

**学生学号** 2208010423

计算机科学与技术 **专业** 224 **班**

2025 **年** 03 **月** 18 **日**

**摘要**

本次测试旨在评估**GPT-4o**、**DeepSeek-V3**和**文心X1**等常见大模型在代码缺陷检测任务中的表现，重点关注它们在不同难易程度和代码量下的检测能力，包括语法错误和逻辑错误等方面。

本次选取了以下三种测试模型：

* **GPT-4o**: OpenAI 的大模型，以强大的自然语言理解和代码生成能力著称，本次测试不采用**深度思考**。
* **DeepSeek-V3**: 专注于代码分析和优化的模型，擅长处理复杂代码逻辑，本次测试不采用**深度思考**。
* **文心X1**: 百度推出的**深度思考**多模态大模型，具备较强的中文理解和代码处理能力。

每个模型测试 3 段不同难易程度和代码量的代码片段：

* **简单代码**: 10 行左右的代码，包含明显的语法错误或逻辑错误。
* **中等代码**: 50 行左右的代码，包含较为复杂的逻辑错误或潜在的性能问题。
* **复杂代码**: 100 行以上的代码，包含多个模块、依赖关系，以及隐藏的逻辑错误。

对于每个代码片段, 我们预设一定数量的代码缺陷, 根据大模型的回答结果对大模型进行打分, 最终得到模型能力对比图。

**关键词**: GPT-4o、DeepSeek-V3、文心X1、软件测试、代码缺陷检测、语法错误、逻辑错误

目录

[大模型的代码缺陷检测能力分析报告 1](#_Toc193207693)

[1引言 1](#_Toc193207694)

[2测试方法 1](#_Toc193207695)

[2.1测试代码设计 1](#_Toc193207696)

[2.2测试步骤 4](#_Toc193207697)

[3测试结果 5](#_Toc193207698)

[3.1 模型A (GPT-4o) 测试结果 5](#_Toc193207699)

[3.1.1 简单代码测试结果 5](#_Toc193207700)

[3.1.2 中等代码测试结果 5](#_Toc193207701)

[3.1.3 复杂代码测试结果 6](#_Toc193207702)

[3.2 模型B (DeepSeek-V3) 测试结果 8](#_Toc193207703)

[3.2.1 简单代码测试结果 8](#_Toc193207704)

[3.2.2 中等代码测试结果 8](#_Toc193207705)

[3.2.3 复杂代码测试结果 9](#_Toc193207706)

[3.3 模型C (文心X1) 测试结果 10](#_Toc193207707)

[3.3.1 简单代码测试结果 11](#_Toc193207708)

[3.3.2 中等代码测试结果 11](#_Toc193207709)

[3.3.3 复杂代码测试结果 11](#_Toc193207710)

[4结果分析 13](#_Toc193207711)

# **大模型的代码缺陷检测能力分析报告**

# 1引言

随着人工智能技术的快速发展，大模型在代码生成、代码分析和缺陷检测等领域的应用日益广泛。OpenAI 的 GPT-4o、DeepSeek-V3 以及百度的文心X1等大模型，凭借其强大的自然语言处理和代码理解能力，正在成为开发者辅助工具的重要组成部分。然而，不同模型在代码缺陷检测任务中的表现存在显著差异，尤其是在处理不同难易程度和代码量的任务时，其能力边界尚未被充分探索。

本次测试旨在评估 **GPT-4o**、**DeepSeek-V3**和**文心X1**在代码缺陷检测任务中的表现，重点关注以下两个方面：

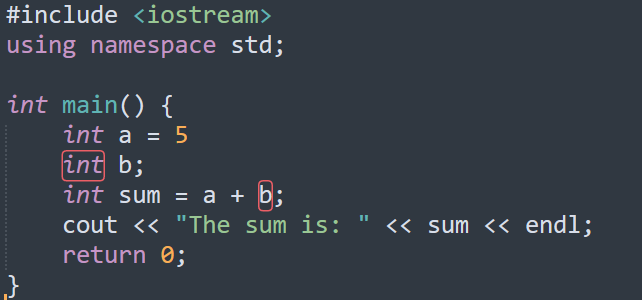
* **语法错误检测**: 模型能否快速识别代码中的语法错误并提供修复建议。
* **逻辑错误检测**: 模型能否发现代码中的逻辑缺陷，尤其是在复杂逻辑场景下的表现。

# 2测试方法

## 2.1测试代码设计

为了全面评估模型的缺陷检测能力，测试代码分为三个不同难易程度和代码量的类别，并且我们设置好对应的**预期缺陷**。

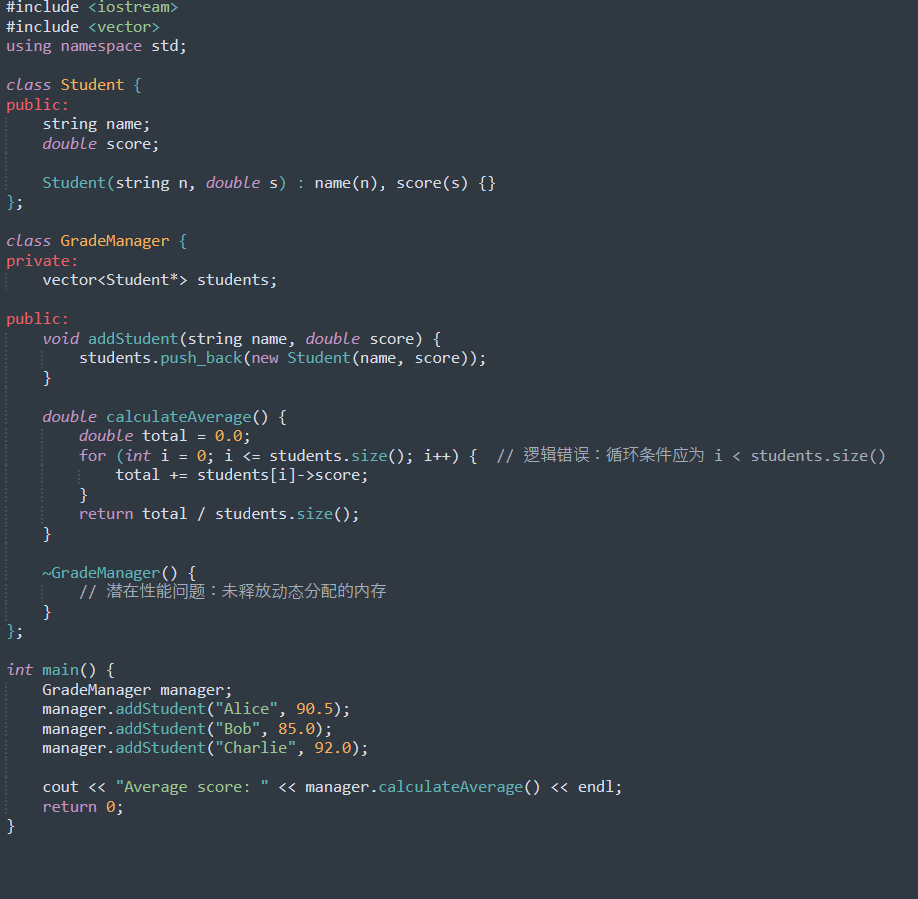
1. **简单代码**: 计算两个数的和并输出结果。



**预期缺陷**:

* 第 5 行：缺少分号（int a = 5 后面缺少分号）。
* 第 6 行：变量 b 未初始化

1. **中等代码**: 实现一个简单的学生成绩管理系统，包含添加成绩和计算平均分的功能。



**预期缺陷**:

* 第 18 行：循环条件错误（i <= students.size() 应为 i < students.size()）。
* 第 24 行：未释放动态分配的内存（应在析构函数中释放 students 中的指针）。

1. **复杂代码**: 实现一个简单的银行账户管理系统，包含存款、取款和转账功能。



**预期缺陷**:

* 第 50 行：隐藏的逻辑错误（转账时未检查账户余额是否足够）。
* 第 34 行：依赖关系问题（未处理并发访问，可能导致数据竞争）。

## 2.2测试步骤

**1. 输入代码**: 将代码片段分别输入到 GPT-4o、DeepSeek-V3 和 文心X1 中。使用统一的提示词（Prompt），例如：“请检测以下代码中的语法错误和逻辑错误，并给出修复建议。”

2. 记录结果 : 记录每个模型的检测结果，包括：正确指出代码缺陷的数量, 正确修正代码缺陷的数量, 错误指出代码缺陷的数量, 错误修正代码缺陷的数量。

# 3测试结果

## 3.1 模型A (GPT-4o) 测试结果

### 3.1.1 简单代码测试结果

### 

### 3.1.2 中等代码测试结果

### 

### 3.1.3 复杂代码测试结果



## 3.2 模型B (DeepSeek-V3) 测试结果

### 3.2.1 简单代码测试结果

### 

### 3.2.2 中等代码测试结果

### 

### 3.2.3 复杂代码测试结果



## 3.3 模型C (文心X1) 测试结果

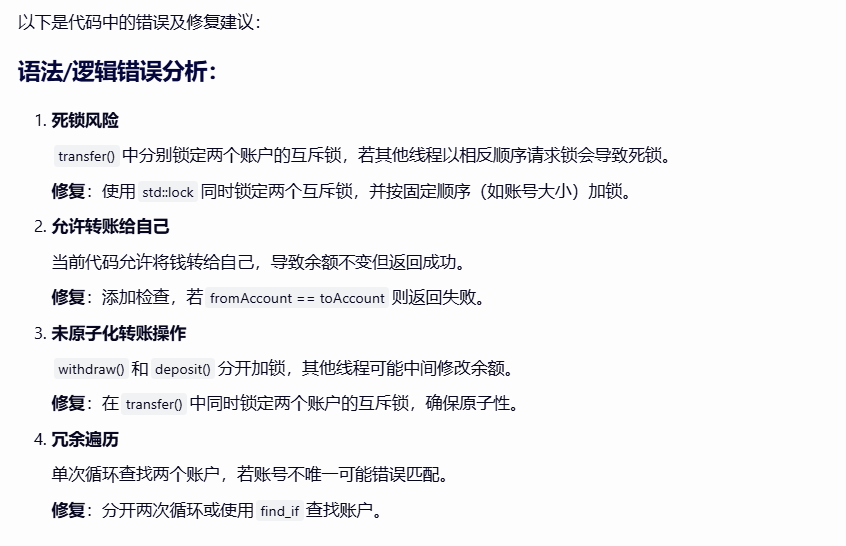
### 3.3.1 简单代码测试结果

### 

### 3.3.2 中等代码测试结果

### 

### 复杂代码测试结果

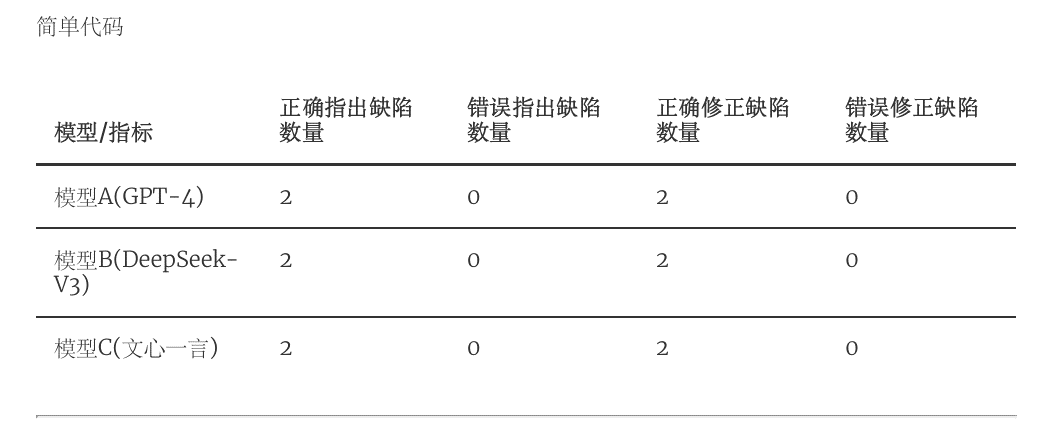


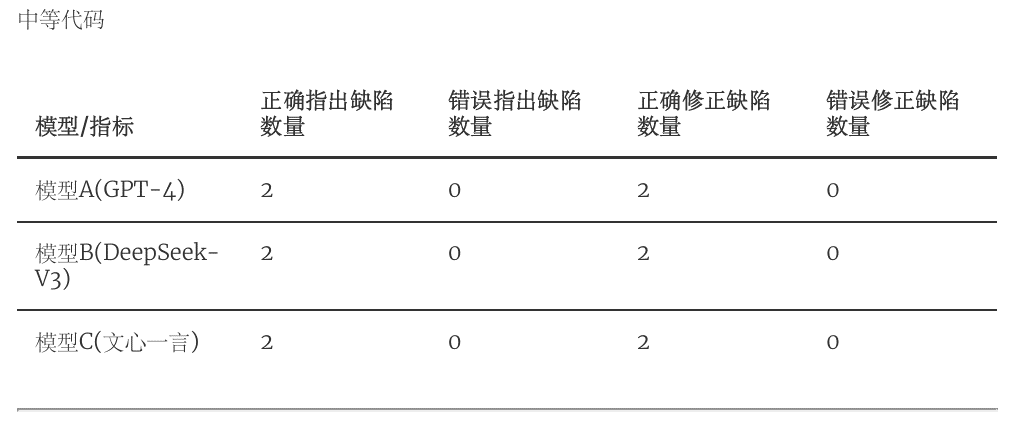
# 4结果分析

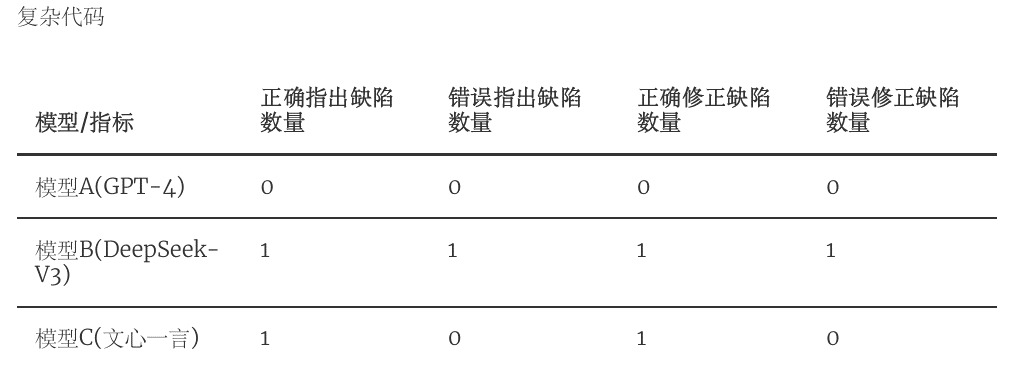
对于大模型给出结果中, 我们忽略一些**建议性的意见,** 因为这些意见不影响代码的实际运行。

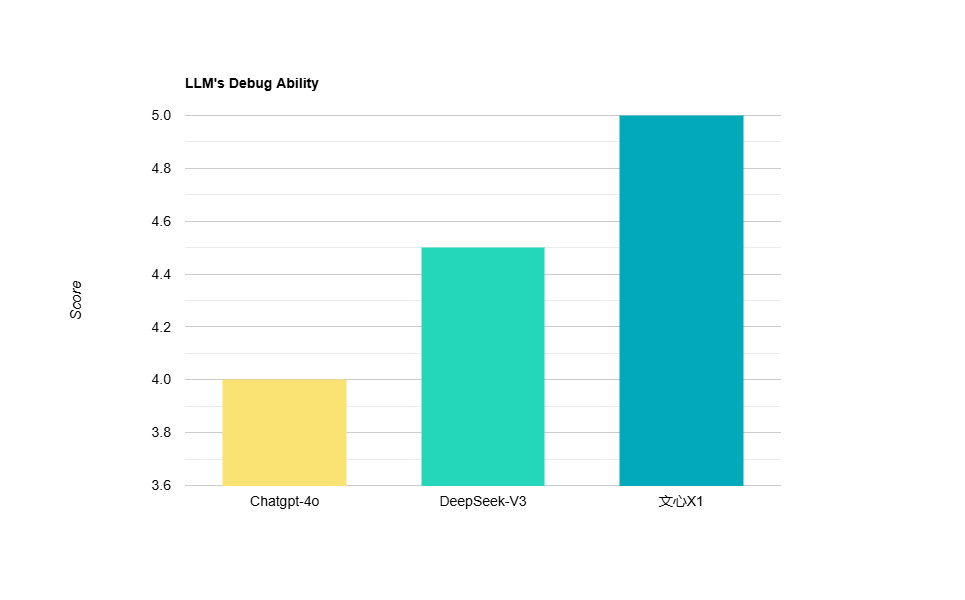
设大模型正确指出的缺陷数量为 , 正确修正的缺陷数量为 , 错误指出的缺陷数量为 , 错误指出的缺陷数量为 ，分别对应 chatgpt-4o, DeepSeek-V3, 文心X1的得分, 且 。

最终, 根据如下数据, 得出 Y1 = 4.0, Y2 = 4.5, Y3 = 5.0。









综上, 可以得出结论, **带有深度思考的文心X1模型**在总体表现上略优于目前因为服务器过载而下调了能力的**DeepSeek-V3模型**, 同时 DeepSeek-V3 模型的总体表现略优于 **Chatgpt-4o 模型**。

不过需要指出的是, 三种 LLM 在中低难度代码的纠错上都得到满分, 只是在复杂任务中表现出微小差距。