代码补丁漏洞检测模型

问题

我们的问题是自动检测代码补丁中是否存在已知漏洞（如 CWE）。传统代码审查流程耗时且容易出现漏检，而自动化模型能够提高检测效率，降低错误率。本项目旨在通过分析代码补丁中的特征，利用机器学习技术实现对潜在漏洞的预测和分类。

挑战

1. 数据预处理复杂性：
    - 补丁文件存在多种编码格式，直接读取可能失败。
    - 部分补丁文件丢失或不完整。
2. 特征提取困难：
    - 补丁文件包含的上下文信息有限，需从中提取有效的特征。
    - 静态特征（如关键词统计）和动态特征（如TF-IDF）需要结合使用。
3. 标签质量：
    - 标签数据（漏洞类型）可能缺失或不均衡。
4. 模型泛化能力：
    - 模型需在不同代码库、不同语言和不同开发风格中保持良好表现。

解决方案

1. **数据处理**：
    - 使用 `chardet` 库自动检测补丁文件的编码，解决编码不一致的问题。
    - 对于丢失的补丁文件，记录日志以便进一步检查。
2. **特征提取**：
    - 静态特征：
        - 统计补丁中新增行（`+`开头）和删除行（`-`开头）的数量。
        - 统计特定编程语言关键词（如 `if`、`for`）的出现频率。
    - 动态特征：
        - 使用 `TF-IDF`（词频-逆文档频率）技术对补丁文本内容进行特征向量化。
        - 限制最大特征数（如 1000），降低维度，避免过拟合。
3. **模型训练**：
    - 使用 `RandomForestClassifier`（随机森林）作为分类模型，结合静态特征与动态特征进行训练。
    - 使用 `train_test_split` 方法将数据划分为训练集和测试集（7:3），保证模型训练和评估的公正性。
    - 保存训练后的模型和TF-IDF向量化器，以便后续推理。
4. **推理与评估**：
    - 在预测阶段加载预训练模型与向量化器

    - 使用训练好的模型对测试补丁提取相同类型的特征（静态+动态），并生成预测结果。
    - 将预测结果（包括漏洞类型、概率）保存为 CSV 文件，便于分析。

结果

- 性能：
    - 模型通过结合静态特征和动态特征，在训练集上实现了良好的拟合效果。
    - 测试集中，模型准确预测了大多数漏洞类型。
- 效率：
    - 自动化处理流程显著减少了人工审查时间。
    - 编码检测和错误日志记录提升了数据处理的鲁棒性。
- 输出：
    - 生成的 `result.csv` 文件包含每个提交的预测结果，包括漏洞类型和置信概率。

未来工作

- **改进特征提取**：增加函数调用图、变量分析等高级特征，提高模型表现。
- **多语言支持**：扩展关键词库，支持更多编程语言的补丁检测。
- **数据增强**：使用迁移学习方法，从公开漏洞数据集中学习通用特征。

这套模型结合静态和动态分析技术，为代码漏洞检测提供了一种高效且可扩展的解决方案。