# 项目设计

## 问题描述

近些年来，随着深度学习技术的蓬勃发展，以及二进制恶意代码的不断发展扩张，手工分析已经无法满足对时刻变化的恶意代码的实时分析，网络安全的研究者们开始探索深度学习技术在二进制安全方向上的应用。二进制分析属于信息安全业界逆向工程中的一种技术，大多数情况下，研究人员无法获取软件程序的源代码，只能获取对应的二进制程序文件。在进行了获取二进制文件后，需要对文件进行反汇编，例如使用一些工具IDA,可以查看对应二进制的汇编代码，可以查看函数，并且对函数流程进行分析。本文需要对函数运行流程进行分析，查看函数运行中的脆弱部分，本文使用深度学习方法。

深度学方法在进行二进制文件分析时有一定的优势，深度学习擅长处理单一类型的数据，在本次实验对应输入的都是二进制文件。深度学习要求有足够多的样本，而样本越多准确的也越高。可以避免人工去选择特征，一开始设计好网络结构，深度学习会自动学习到重要的特征。

## 研究方案

二进制的指令流与自然语言处理有着共通点。将指令看做自然语言的词，指令流就是句子，对于二进制指令流的判断就可以看做是对自然语言的语义分析。所以我们提出了在深度学习的基础上，对本次运行路径进行判断。

首先对当前使用深度学习方面的论文进行具体分析，提取其中的可以进行利用的观点或方法，实现本次实验对二进制文件进行操作。

V-Fuzz：以缺陷为导向，将函数的控制流图结合尽可能多的信息保存，应用图嵌入转换为向量，对函数进行预测；NeuFuzz：利用自然语言处理的方法处理二进制指令，发现了24个CVE，本篇文章使用的算法为双向LSTM算法；Hecate：通过将程序中的函数进行剥离，减小攻击面；SAFE：应用self attention机制判断相同源码经过两个不同平台编译后，两个函数的相似度。

通过对这些当前二进制文件论文的分析，我们组总结了本次实验所使用的研究方案。

首先需要获取数据集，由于在二进制文件领域的数据集比较缺乏，所以我们使用了V-Fuzz这篇论文所使用的数据集，数据预处理需要使用IDA-python软件对数据进行反汇编，并使用python的代码对数据进行函数的划分。

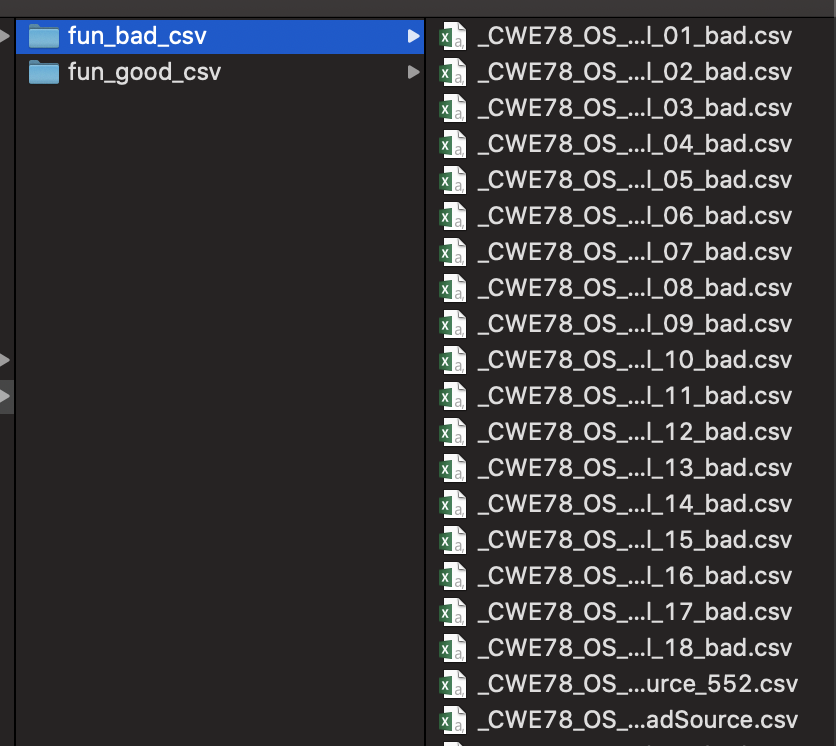
在选取算法方面，选取了LSTM这种方法，LSTM 有两个优点：1. 适合处理连续的信息，程序路径与自然语言处理的过程非常相似，代码是否有缺陷与上下文有很大的关系；2. 这个模型有记忆函数，适合于处理长相关的问题，缺陷函数的联系都处在一个相对较远的位置。

在前端框架方面，选取了PyQT来进行前端界面的撰写。首先，这种前端语言使用python库，使用方式简单，易于书写，界面简单。并且与本实验所使用的python代码易于链接，方便进行模型的测试。

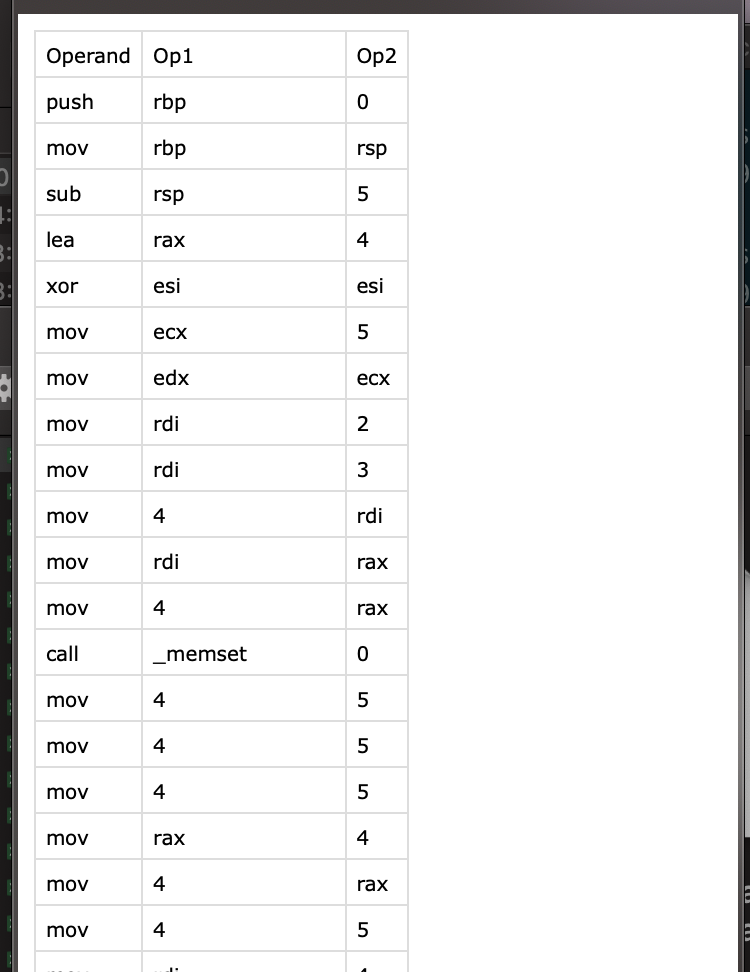
## 数据集

选用Juilet数据集，数据集中含有130多个CWE的漏洞类型，500多个人工生成程序

使用IDApython将函数进行遍历分析，将函数类型进行标注，并获取函数执行最长路径，得出下图结果：



下图显示为在IDApython处理后的结果：



需要继续对第三列的操作数进行归一化处理，在IDApython中将操作数分为八种不同的操作类型。需要对其进行转化，方便进行模型训练。下图为归一化结果：

|  |  |  |  |
| --- | --- | --- | --- |
| **Op** | | **Type** | |
| push\_rbp\_void  mov\_rbp\_rsp  sub\_rsp\_imm  mov\_rdi\_displ call\_\_printLongLongLine\_  void add\_rsp\_imm  pop\_rbp\_void  retn\_void\_void | 0 | |

## 4. 算法选择：

### 4.1算法安全性

即算法脆弱性问题，网络空间安全应用对算法的安全性要求极为严苛。而深度学习属于机器学习的范畴，因此深度学习算法与机器学习算法具有相同的脆弱性。深度学习算法存在的脆弱性会使算法存在受到对抗攻击和隐私窃取攻击的潜在风险，影响模型的完整性、机密性和健壮性。因此，我们认为这是深度学习的网络空间安全应用首先应解决的基础性问题。解决算法脆弱性问题，较好的一种办法是进行对抗训练。

### 4.2算法功能

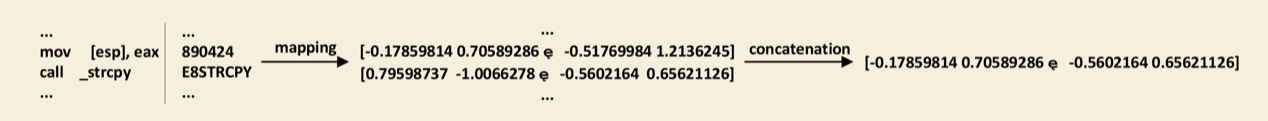
即序列化模型相关问题，在算法安全性得到保障的基础上，序列建模问题应该得到关注，因为基本上所有的网络空间安全数据都是序列化数据，所有的安全应用，如程序分析、漏洞挖掘和恶意代码检测等，均依赖于序列建模。

### 4.3算法性能

在算法安全性和算法功能实现的基础上，算法性能应得以关注，如算法自适应性、可解释性、特征选取、降低误报以及数据集均衡等问题。由于安全威胁会随着时间演变，使得一成不变的检测模型无法检测出最新的威胁，这就要求检测恶意行为的安全应用具有自适应性和自学习性，来适应变化的安全威胁和攻击技术。可选的措施是时间间隔性的模块化模型训练。

### 4.4 算法选择

**Word2vec：**word2vec是用一个一层的神经网络把one-hot形式的稀疏词向量映射称为一个n维(n一般为几百)的稠密向量的过程。下图为embedding过程：



**LATM：**深度神经网络训练模型采用LSTM（long short-term memory）

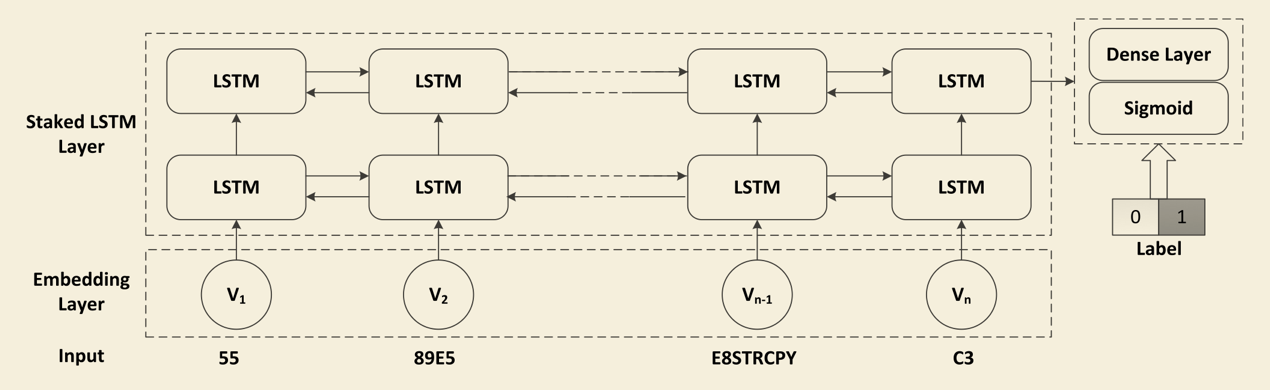
该神经网络有四层结构：

1）.嵌入层，句子中的元素与向量对应起来

2）.2-3层都是堆叠的LSTM层，每层有64个LSTM单元都具有双向模式

3）.最后一层选择了稠密输出，用于对输入路径进行判断

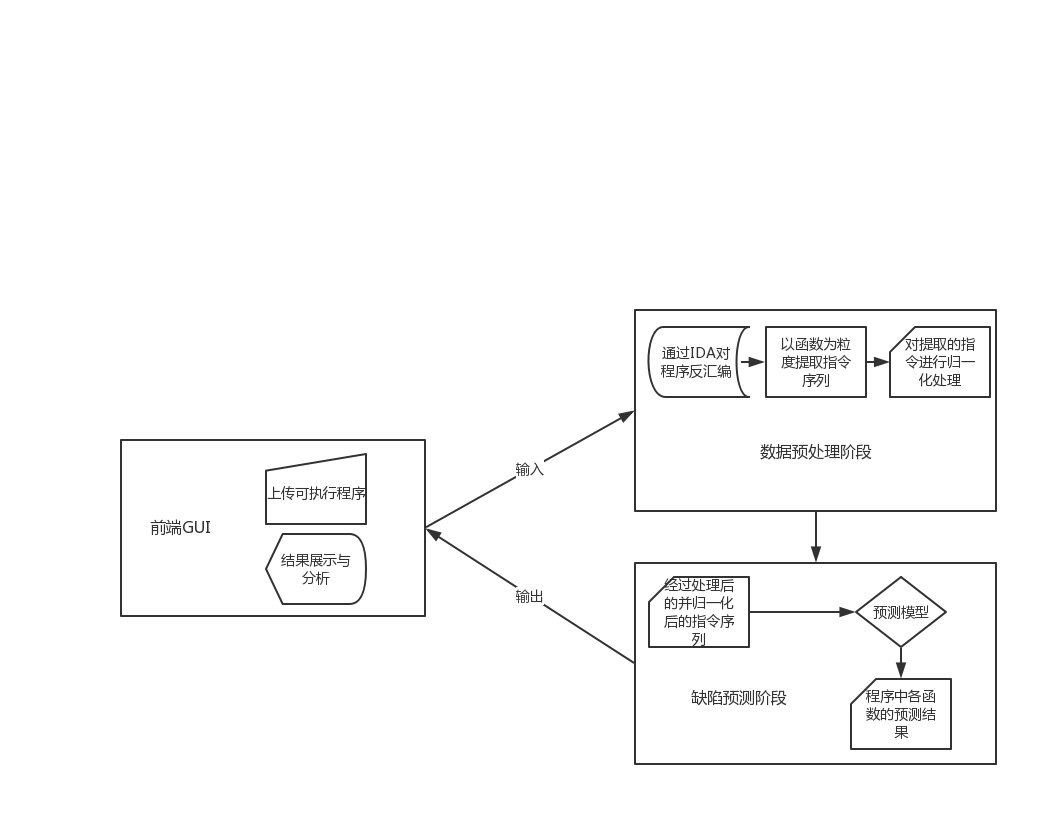
下图显示为LSTM架构图：



## 项目设计

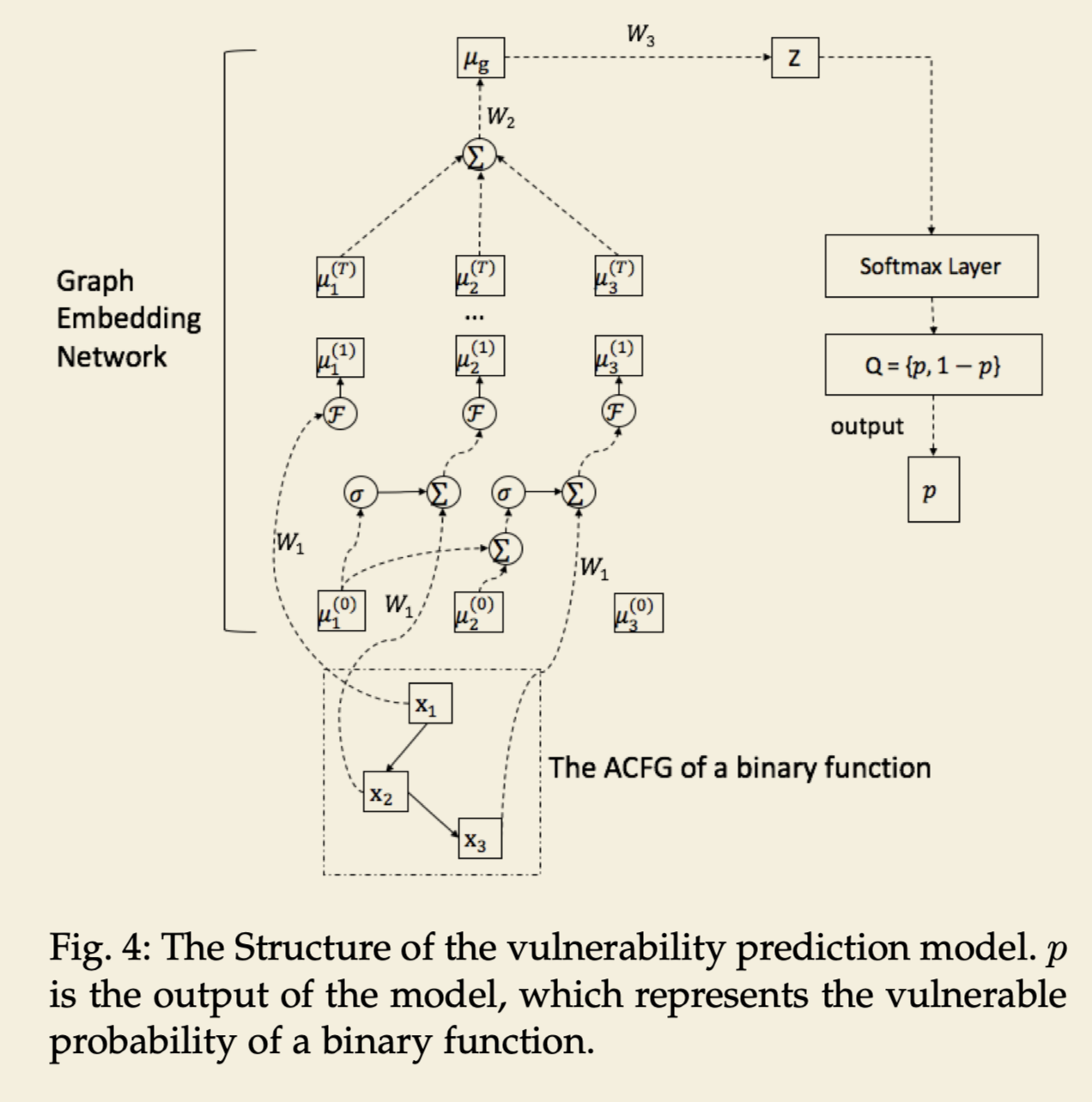
* 选取的数据集为V-Fuzz所使用的数据集，但是使用的方法与原论文中使用控制流图方法是不同的。
* 首先需要对二进制文件数据集进行预处理，将每个函数转换为对应的汇编调用流程，并将这些进行打标签操作，将其区分为良性数据集以及恶意数据集。
* 之后需要选取使用的深度学习算法，本次实验使用的是双向LSTM算法，对数据预处理后的结果进行建立深度学习训练模型。
* 需要使用前端界面进行展示，对于训练好的模型，可以传输可执行的Linux程序进行测试，可以得到对每个函数的预测结果，显示其中良性和恶意函数的·比例。

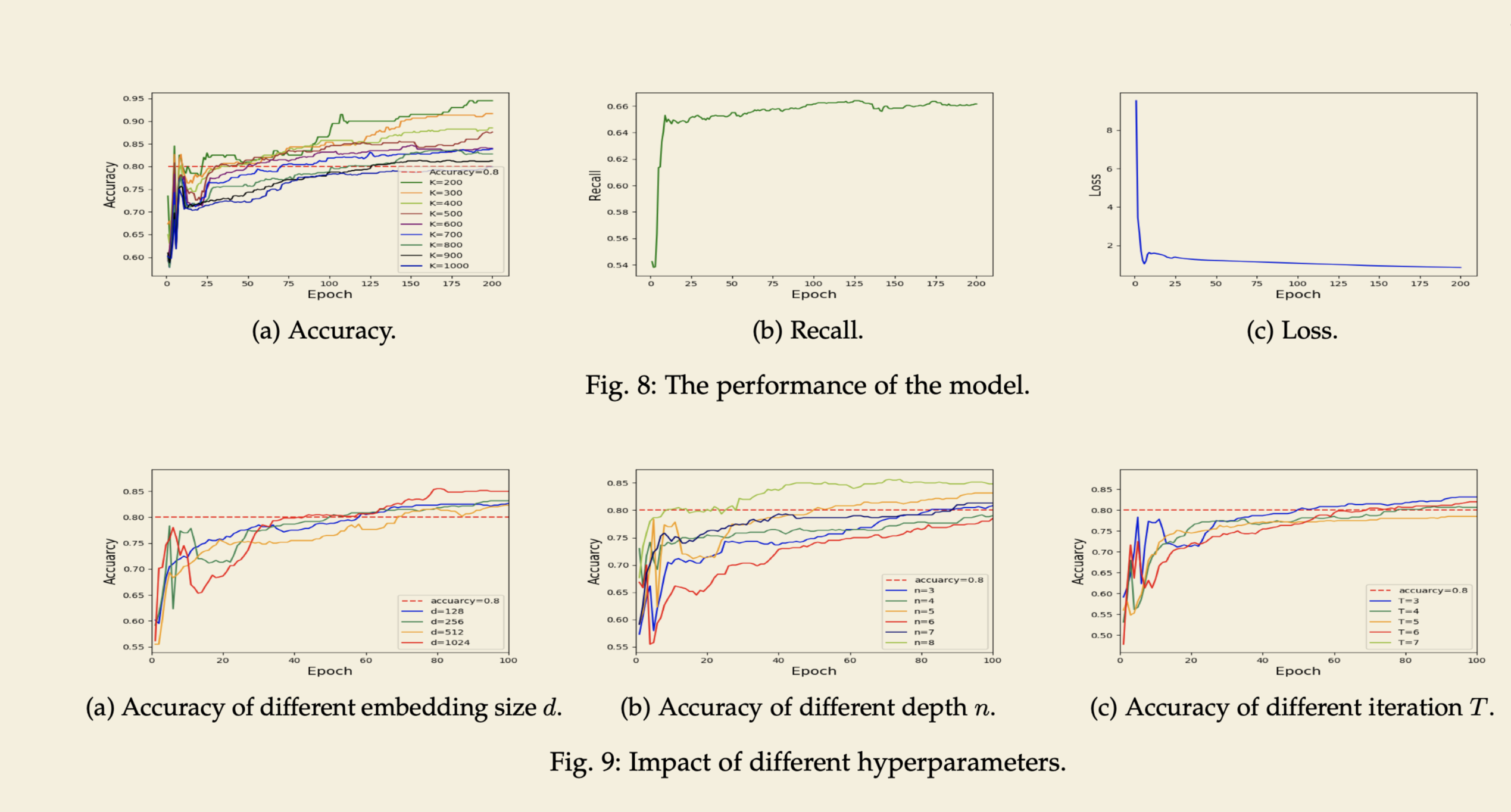
下图显示了本次实验的项目结构图：

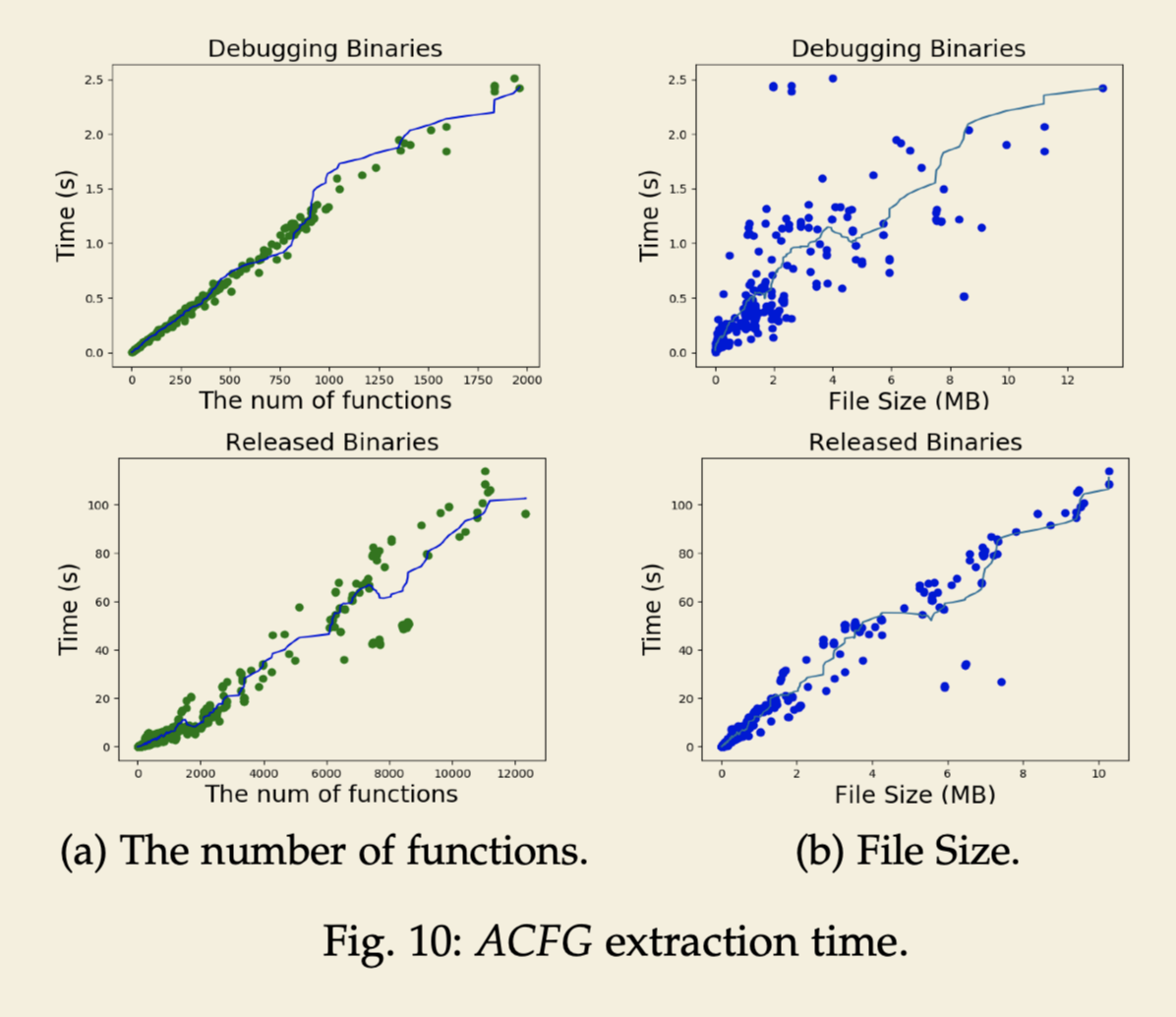


1. 研究成果对标对象

V-Fuzz中的一个主要组成部分就是对于函数缺陷的预测。在该模型中，使用了图嵌入的方法，用ACFG图表示程序流程，再将ACFG向量化从而携带了更多的信息。通过softmax将函数进行分类。

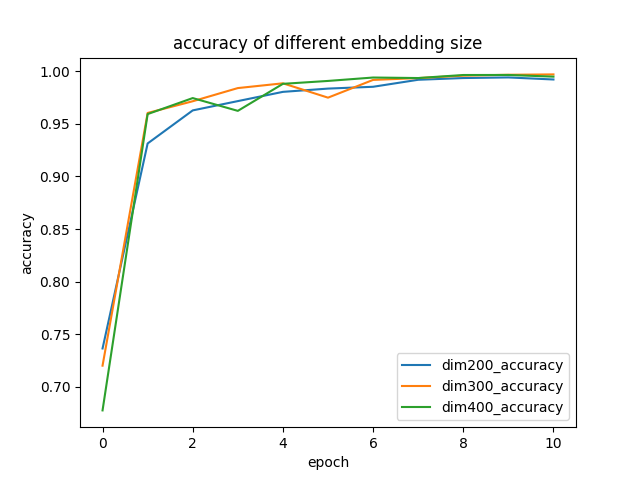






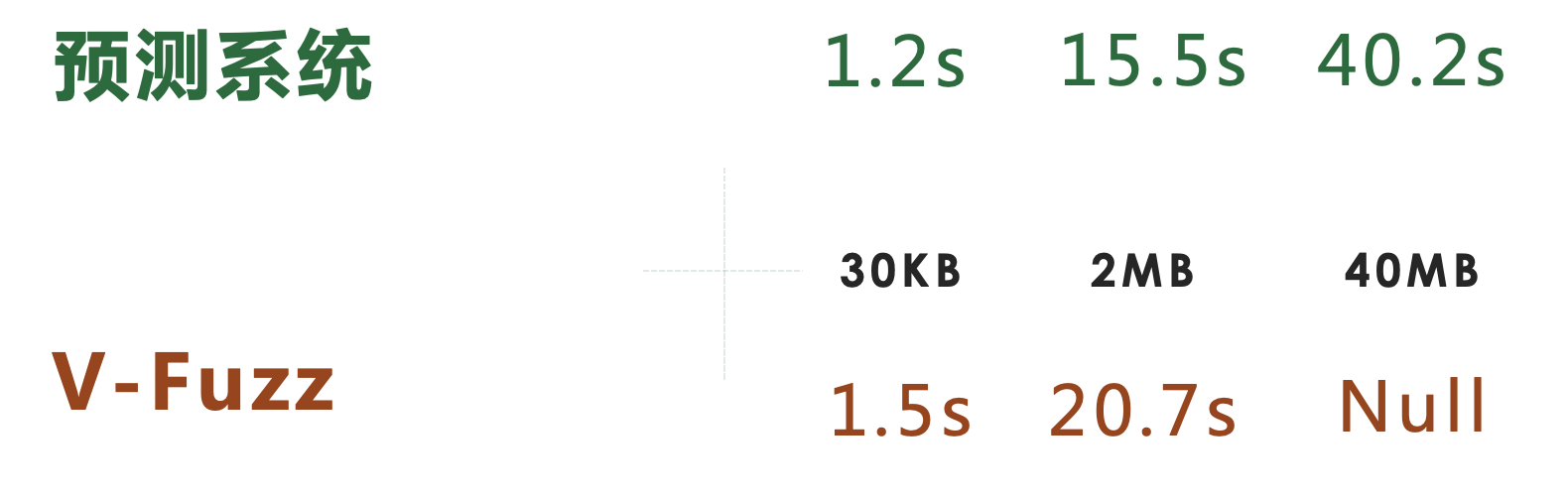
1. 研究成果评价指标

再上述研究成果对标的文章中，我们主要是研究程序预测在各种不同参数下的准确率准确率以及从不同文件大小程序中提取汇编指令的时间。



我们在对向量进行嵌入时，使用了不同的向量维度，在这三种维度中，300维时向量的拟合结果最快，并且准确率在到达95%后可以保持稳定。

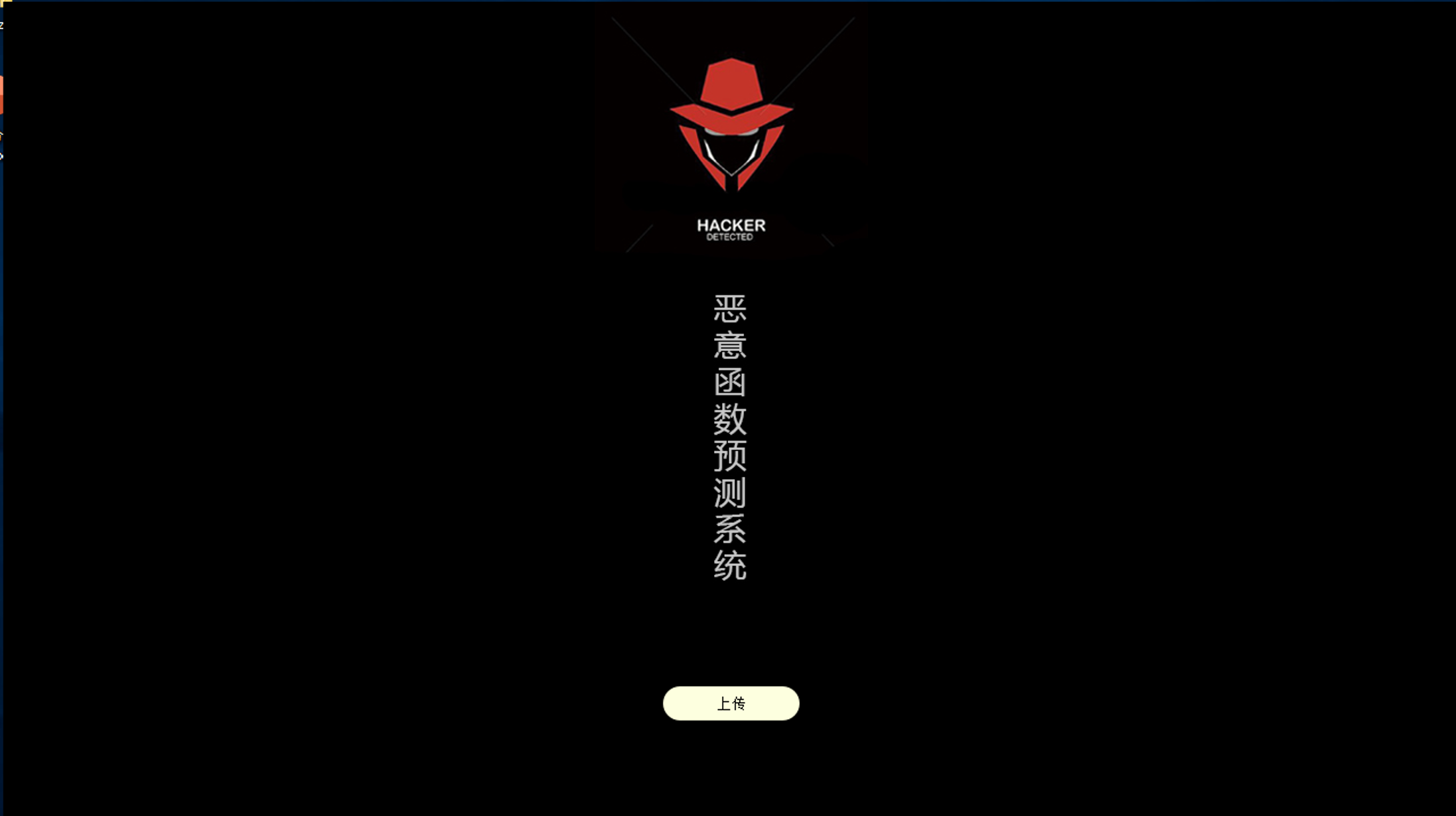
在提取时间上，由于提取指令的时间与机器的性能有很大关系，只能进行函数提取时间与文件大小趋势的比较。



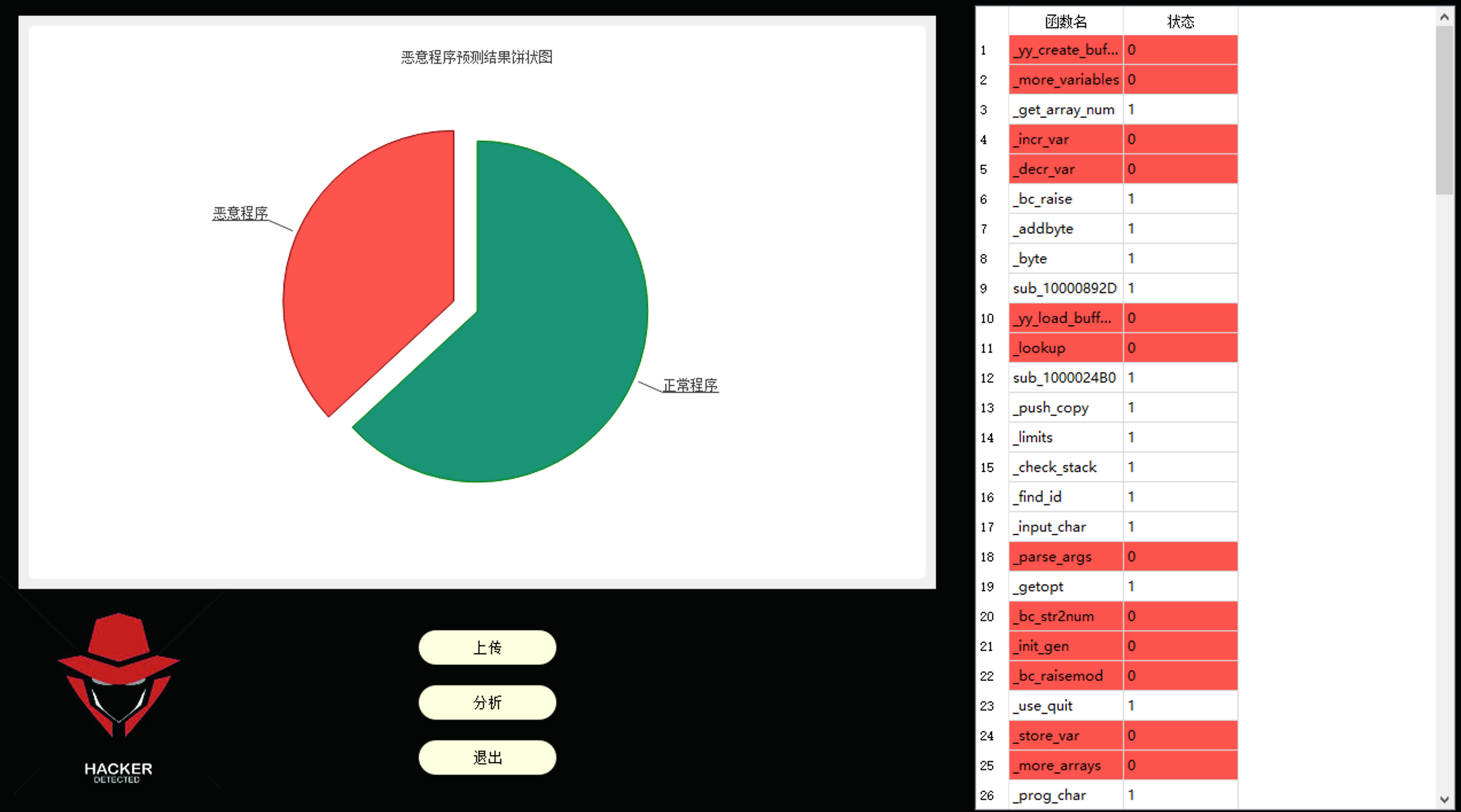
在预测系统的提取时间随文件大小增长趋势上比V-Fuzz是缓慢的，这主要是因为在我们的系统中只需要提取函数中的指令序列而不用提取函数中的CFG图并将必要的信息保存在CFG图中，这样就很大程度上能够提升在指令提取上的时间和系统开销。

1. 成果展示

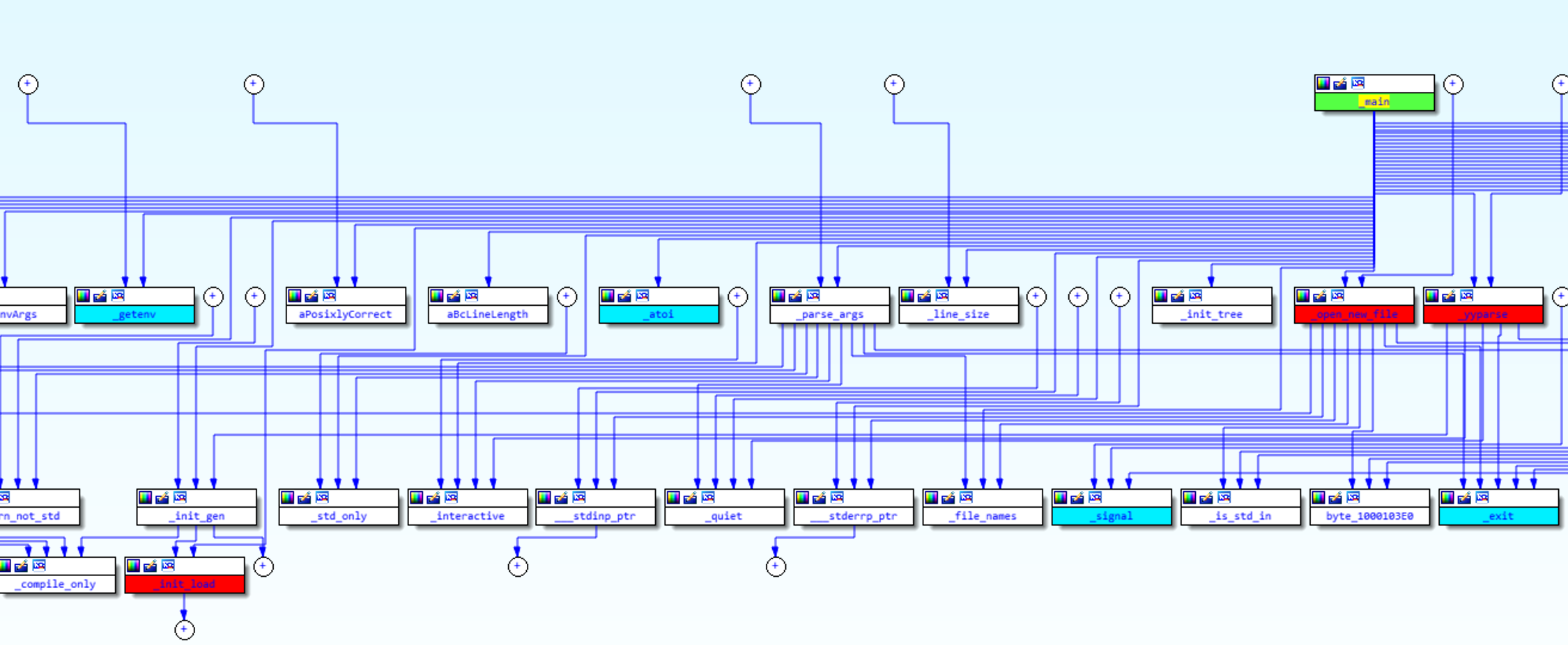
前端界面

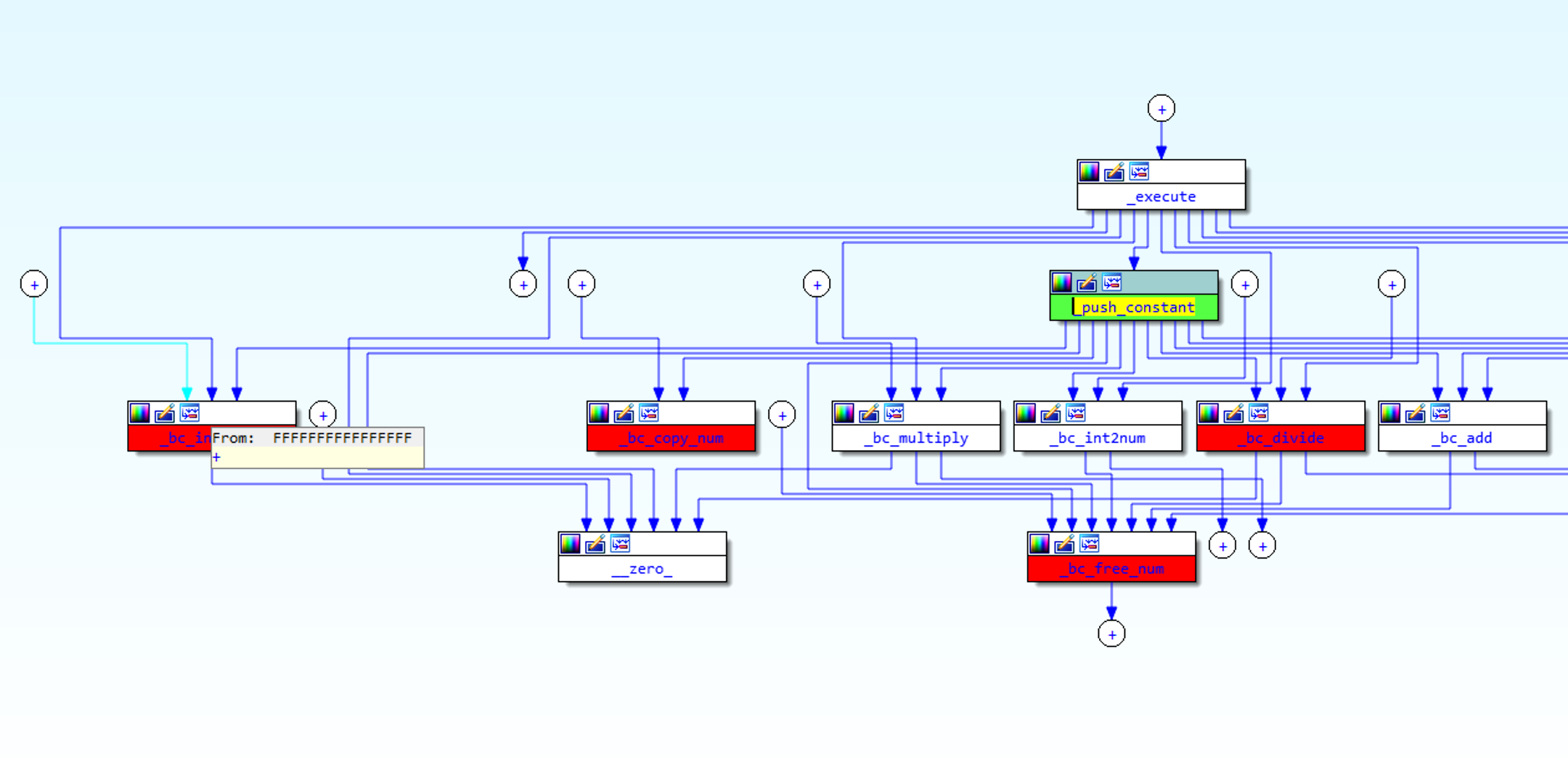


提交程序后的分析结果



对于分析结果的可视化展示





这是整个程序汇总函数的调用关系图，其中被红色标注的就是被预测为有缺陷的函数。

1. 不足之处
2. 现有的数据集并不能够多，漏洞类型不够完善，数据集中程序结构简单
3. 对于程序的静态分析不能够精确实现程序流程，动态分析与静态分析结合。
4. 提取函数指令通过静态方法与IDA配合效率低
5. 下一步工作规划
6. 能够使用动态执行的数据集，使用动态执行的方法对程序进行分析
7. 通过Intel PT技术提高程序指令提取，从CPU层面追踪指令，能够提升指令提取的效率和准确率
8. 可是尝试自然语言处理领域中其他算法，比如bert，或者attention机制
9. 总结

在这次的项目经历中，我们从深度学习在安全领域方向的研究为出发点，将现有的深度学习的算法运用在安全领域中。这让我们了解了深度学习的发展现状以及其在安全领域方向的拓展。之前已经有人在这方面做出了尝试，我们在前人的基础上，对二进制程序恶意检测进行了研究，将自然语言处理与二进制安全中的恶意代码识别做比较，发现他们之间还是有很多共同点。将二进制安全中的缺陷函数预测使用自然语言处理的方法就有了依据。在项目中，自主设计并完成了整个工程，从前端到后端到文档，我们的小组都能够认真完成任务，我们在这次的项目中有了很大的收获。也要感谢在这堂课中给予我们帮助的车老师和高老师，这次的项目也为我们后续的研究打下了基础。