(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 105868108 B
(45)授权公告日 2018.09.07

(21)申请号 201610182075.5

(22)申请日 2016.03.28

(65)同一申请的已公布的文献号
申请公布号 CN 105868108 A

(43)申请公布日 2016.08.17

(73)专利权人 中国科学院信息工程研究所
地址 100093 北京市海淀区闵庄路甲89号
专利权人 国家计算机网络与信息安全管理中心

(72)发明人 石志强 刘中金 常青 陈昱 孙利民 朱红松 王猛涛 何跃鹰

(74)专利代理机构 北京君尚知识产权代理事务所(普通合伙) 11200
代理人 邱晓锋

(51)Int.Cl.
G06F 11/36(2006.01)
G06K 9/62(2006.01)

(56)对比文件
US 2014/0237277 A1,2014.08.21,
CN 102999615 A,2013.03.27,
惠国保等.基于改进的图像局部区域相似度学习架构的图像特征匹配技术研究.《计算机学报》.2015,第38卷(第6期),第1148-1161页.

审查员 陈沙沙

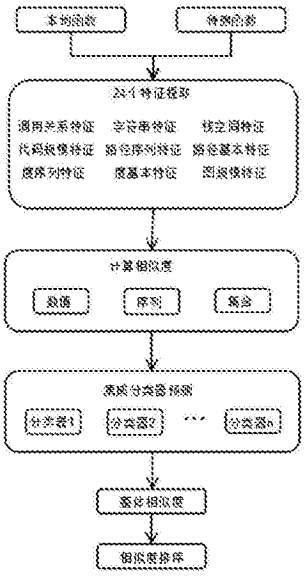
权利要求书2页 说明书9页 附图4页

(54)发明名称

基于神经网络的指令集无关的二进制代码相似性检测方法

(57)摘要

本发明涉及一种基于神经网络的指令集无关的二进制代码相似性检测方法,其主要步骤包括:对二进制文件进行逆向分析,提取函数的调用关系特征、字符串特征、栈空间特征、代码规模特征、路径序列特征、路径基本特征,度序列特征、度基本特征,图规模特征等9个方面共24个特征。基于特征的表现形式,采用3种相似度计算方法计算待比较的两个函数的24个特征的相似程度,作为集成神经网络分类器的输入向量,获得两个函数整体相似度的预测值并进行排序。本发明与现有的技术比,不依赖特定的指令集,可以实现不同指令集的二进制文件的相似度检测,准确率高,技术简单,易于推广。



CN 105868108 B

1.一种基于神经网络的指令集无关的二进制代码相似性检测方法，其特征在于，包括以下步骤：

1)选择同一份源码，选择不同的编译器和不同的优化选项，针对不同的架构进行编译，获得二进制可执行文件；

2)对二进制可执行文件进行逆向分析，提取每一个函数的调用关系特征、字符串特征、栈空间特征、代码规模特征、路径序列特征、路径基本特征，度序列特征、度基本特征，图规模特征共9个方面的特征；所述路径基本特征由平均路径和图直径构成；所述路径序列特征是函数所有基本块之间的路径长度的升序序列；所述度基本特征由图的熵、最大度和平均度构成；所述度序列特征由函数所有基本块的入度升序序列、出度升序序列和度升序序列构成；

3)基于步骤2)提取的9个方面的特征，采用相似度计算方法计算待比较函数的各特征的相似度；

4)将步骤3)提取的相似度作为神经网络的输入向量；如果两个函数名称相同，则标签作为正样本；如果两个函数名称不同，则标签作为负样本；

5)提取若干正样本和负样本以构造训练样本集，并初始化若干个神经网络，针对每个神经网络构造独立同分布的训练样本子集进行训练，根据预测结果调整参数直至预测结果符合要求，此时神经网络分类器训练完毕；

6)将已知的函数称为本地函数，将待检测的函数称为目标函数，根据需求构建本地函数库和目标函数库；对每个本地函数按步骤2)提取9个方面的特征，构造过滤器；对每个目标函数按步骤2)提取9个方面的特征；对每个本地函数，用其对应的过滤器对目标函数库进行过滤，构建该本地函数的候选集；按步骤3)计算每个本地函数与其对应候选集内每个目标函数的相似度值，作为测试样本；

7)将测试样本输入神经网络分类器中，将每个神经网络分类器输出的预测值取加权平均作为整体相似度预测值并进行排序。

2.如权利要求1所述的方法，其特征在于，所述9个方面的特征包括24个特征，具体如下：

a)4个调用关系特征：对二进制文件进行逆向分析，获得函数调用图，计算函数被其他函数调用的次数，该函数调用其他函数的次数，去重后的被其他函数调用的次数，以及去重后的该函数调用其他函数的次数，作为调用关系特征；

b)1个栈空间特征：对二进制文件进行逆向分析，获得函数的栈空间大小，作为栈空间特征；

c)2个字符串特征：统计函数调用的字符串个数和字符串集合，作为字符串特征；

d)3个代码规模特征：统计函数的指令个数、跳转指令个数和代码量，作为代码规模特征；

e)3个度序列特征：对二进制文件进行逆向分析，获得函数的控制流图，计算每个节点的出入度，构造CFG有向图邻接矩阵；将函数控制流图转化为无向图，计算每个节点的度，构造CFG无向图邻接矩阵；对CFG有向图邻接矩阵、CFG无向图邻接矩阵进行度分析；基于CFG有向图邻接矩阵计算入度升序序列、出度升序序列，基于CFG无向图邻接矩阵计算度升序序列，三者作为度序列特征；

f)3个度基本特征:基于度升序序列,计算最大度、平均度和度的概率序列,基于度的概率序列计算图的熵;将最大度、平均度和图的熵作为图基本特征;

g)1个路径序列特征:对CFG无向图邻接矩阵进行路径分析,计算入口节点到其他任意节点间的最小距离,构造路径序列,作为路径序列特征;

h)2个路径基本特征:由路径序列计算图的平均路径、图直径,作为路径基本特征;

i)5个图规模特征:对CFG有向图邻接矩阵进行基本属性分析,计算节点数、链路数、图密度、图聚类系数、链路效率,作为图规模特征。

3.如权利要求2所述的方法,其特征在于,通过Floyd算法或Dinkstra算法计算入口节点到其他任意节点间的最小距离,构造所述路径序列特征。

4.如权利要求2所述的方法,其特征在于,步骤3)基于9个方面共24个特征,对每两个函数计算每个特征的相似度作为神经网络的输入向量,这9个方面的相似度分别是:调用关系相似度、栈空间相似度、字符串相似度,代码规模相似度、路径序列相似度、路径基本属性相似度,度序列相似度、度基本属性相似度和图规模相似度,对应24个特征共计24个相似度。

5.如权利要求4所述的方法,其特征在于,步骤3)中,针对数值形式的特征,采用绝对值距离与最大值之差作为相似度;针对序列形式的特征,采用字符串编译距离作为相似度;针对集合形式的特征,采用Jaccard系数作为相似度。

6.如权利要求1所述的方法,其特征在于,步骤4)至步骤7)中,训练样本采用自编译的、多平台、带符号表的函数集;根据匹配模式不同,采用对应匹配模式的训练样本集训练集成神经网络,以提高准确率。

基于神经网络的指令集无关的二进制代码相似性检测方法

技术领域

[0001] 本发明涉及二进制程序漏洞挖掘与逆向分析领域，具体涉及一种基于神经网络的指令集无关的二进制代码相似性的检测方法，属于计算机程序检测技术领域。

背景技术

[0002] 随着开源软件的崛起，软件抄袭现象越来越多，检测代码是否抄袭的需求也就越来越大。实际应用中，大部分的商业软件均已二进制代码形式存在，源代码难以获取。因此，判断代码是否抄袭的方法主要采用二进制代码相似性检测技术。

[0003] 二进制代码相似性检测技术是依靠各种相似性计算方法，度量待比较的两份二进制代码的相似程度，可分为基于文本的相似性检测技术、基于图同构的相似性检测技术、基于结构化签名的相似性检测技术和基于语义信息的相似性检测技术。基于文本的相似性检测技术可分为二进制字节相似性检测技术和反汇编文本相似性检测技术，两者均是直接对内容做比较，依赖指令集和编码方式。基于图同构的相似性检测技术核心是基于指令的相似性，比较的是汇编指令语义级信息的相似性，进而将问题转化为图同构问题，同样依赖指令集。基于结构化签名的相似性检测技术是根据函数签名进行匹配，同时在函数控制流图上进行基点传播。函数签名往往取函数控制流图的节点数，边数等，对函数控制流图的变化缺乏鲁棒性。基于语义信息的相似性检测技术是依靠各种二进制代码分析平台对语义信息进行提取和比较，需要分析员对二进制代码分析平台熟悉，实现较为困难。

[0004] 二进制代码相似性检测技术不但可用于代码抄袭检测，还可用于补丁比对和二进制软件同源性检测等方面。随着代码相似性检测技术的不断发展，随之涌现大量的二进制代码相似性检测工具。2005年12月，iDefense公司发布IDA插件IDACompare；2006年11月，eEye发布开源的eEye Binary Diffing Suite(EBDS)套件；2007年9月，Sabre发布IDA插件Bindiff2。这些工具的出现和改进，使代码相似性检测技术得到越来越广泛的应用。

[0005] 目前，缺少一种实现简单的，指令集无关的二进制代码相似性检测技术。

发明内容

[0006] 本发明目的在于提供一种基于神经网络的指令集无关的二进制代码相似性检测方法。

[0007] 本发明涉及的方法流程主要包括：对二进制文件进行逆向分析，提取函数的调用关系特征、字符串特征、栈空间特征、代码规模特征、路径序列特征、路径基本特征，度序列特征、度基本特征，图规模特征共9个方面24个特征。通过计算待比较的两个函数的每个特征的相似程度，获得调用关系相似度、字符串相似度、栈空间相似度、代码规模相似度、路径序列相似度、路径基本属性相似度，度序列相似度、度基本属性相似度和图规模相似度9个方面共24个相似度，作为神经网络分类器的输入向量，带入集成神经网络分类器中进行预测，获得两个函数整体相似度并进行排序。

[0008] 本发明的技术创新点在于函数9个方面共24个特征的构造和基于集成神经网络分

类器的相似程度预测方法。24个特征主要从函数之间调用关系、函数的基本属性和函数控制流图的图属性三个方面进行提取，较完整地反映了一个函数的典型特征。特别的是，24个特征的提取不依赖特定的指令集，可以对两个不同指令集的二进制代码进行特征提取和相似程度预测。采用集成多个神经网络对相似程度进行预测，实现简单，易于推广。

[0009] 为实现上述目的，本发明采用如下技术方案：

[0010] 一种基于神经网络的指令集无关的二进制代码相似性检测方法，主要包含以下步骤：

[0011] 1)构造训练样本集。选择同一份源码，选择不同的编译器和不同的优化选项，针对不同的架构进行编译，获得二进制可执行文件。对二进制可执行文件进行逆向分析，提取每个函数的24个特征。基于特征，对每两个函数计算相似度共获得24个相似度作为神经网络的输入向量。如果两个函数名称相同，则标签为1，作为正样本，如果两个函数名称不同，则标签为0，作为负样本。

[0012] 2)构造神经网络分类器和子训练样本集。建立若干初始神经网络。从初始样本集中有放回的抽取一定比例(如80％)的样本构造若干独立同分布的子训练样本集，作为每个神经网络的训练样本。

[0013] 3)训练神经网络分类器。将对应的子训练样本集输入神经网络分类器进行训练，根据预测结果调整神经网络的参数直至预测结果符合要求，此时神经网络分类器训练完毕。

[0014] 4)构造测试集。将已知的函数称为本地函数，将待检测的函数称为目标函数，根据需求构建本地函数库和目标函数库。对每个本地函数提取24个特征，构造过滤器。对每个目标函数提取24个特征。对每个本地函数，用其对应的过滤器对目标函数库进行过滤，构建该本地函数的候选集。计算每个本地函数与其对应候选集内每个目标函数的24个相似度值，作为测试样本。

[0015] 5)将测试样本输入神经网络分类器中，用已经训练好的若干神经网络分类器进行预测得到若干预测值，取其加权平均作为整体相似度预测值并进行排序。

[0016] 进一步地，步骤1)和4)中，24个特征是调用关系特征、字符串特征、栈空间特征、代码规模特征、路径序列特征、路径基本特征，度序列特征、度基本特征和图规模特征。其中调用关系特征由该函数被调用的函数个数、被该函数调用的函数个数和去重后的个数构成；字符串特征是由函数调用的字符串数量和字符串集合构成；栈空间特征是该函数的栈空间大小；代码规模特征由指令个数、跳转指令个数、代码量构成；路径基本特征由平均路径和图直径构成；路径序列特征是该函数所有基本块之间的路径长度的升序序列。度基本特征由图的熵、最大度和平均度构成；度序列特征由该函数所有基本块的入度升序序列、出度升序序列和度升序序列构成。图规模特征由CFG有向图的节点数、链路数、链路效率、图密度、聚类系数构成。

[0017] 进一步地，步骤1)和4)中，24个相似度是由调用关系相似度、字符串相似度、栈空间相似度、代码规模相似度、路径序列相似度、路径基本属性相似度，度序列相似度、度基本属性相似度和图规模相似度构成。

[0018] 本发明可以获得以下有益效果：

[0019] 本发明在提取函数的24个特征时，主要从函数间的调用关系、函数的基本属性和

函数控制流图的图属性三个方面进行提取，较完整地反映一个函数的典型特征。这24个特征的提取不依赖特定的指令集，因此本发明可以对两个不同指令集的二进制代码进行特征提取和相似程度预测。

[0020] 本发明在构造、训练和测试神经网络时，训练样本集是由同一份源码，选择不同的编译器，不同的优化选项，针对不同的指令集进行编译获得的，较为完整地包含不同编译器、不同优化选项和不同指令集下的样本；采用的是若干神经网络集成的方法，对每个神经网络建立了独立同分布的子训练样本集进行训练；在计算预测结果时，采用的是若干神经网络预测值取加权平均的方法。这种集成分类器的方法，将提高预测结果的准确性和稳定性。

[0021] 本发明在构造测试集时，采用过滤器，用于本地函数候选集的构造。过滤器的使用将特征相差巨大，明显不相似的目标函数快速排除，提高了检索效率。

[0022] 本发明与现有的技术比，不依赖特定的指令集，可以实现不同指令集的二进制文件的相似度检测，准确率高，技术简单，易于推广。

附图说明

[0023] 图1.函数特征提取流程示意图。

[0024] 图2.神经网络模型结构示意图。

[0025] 图3.训练集成分类器方法。

[0026] 图4.候选集构造示意图。

[0027] 图5.集成分类器预测方法。

[0028] 图6.整体流程示意图。

具体实施方式

[0029] 为使本发明的上述目的、特征和优点能够更加明显易懂，下面通过具体实施例和附图，对本发明做进一步说明。

[0030] 本发明的一种基于神经网络的指令集无关的二进制相似代码检索方法，具体包括如下步骤：

[0031] 1)构造训练集样本。选择同一份源码，选择不同的编译器，不同的优化选项，针对不同的架构进行编译，获得二进制可执行文件。对二进制可执行文件进行逆向分析，对每个函数提取9个方面的特征。这9个方面的特征分别是调用关系特征、字符串特征、栈空间特征、代码规模特征、路径序列特征、路径基本特征，度序列特征、度基本特征，图规模特征，字符串特征。图1是函数特征提取流程示意图，具体的函数特征提取方法如下：

[0032] a).对二进制文件进行逆向分析，获得函数基本属性、函数调用图和函数控制流图。

[0033] b).分析函数调用图，计算该函数被其他函数调用的次数callto和计算该函数调用其他函数的次数callfrom，以及去重后的被其他函数调用的次数callto2和去重后的该函数调用其他函数的次数callfrom2，构成调用关系特征Call＝(callfrom,callto,callfrom2,callto2)。

[0034] c).分析函数基本属性，计算栈空间stack、跳转指令个数jump、指令个数inst，代

码量code，调用的字符串数量strnum和调用的字符串集合strset。构成栈空间特征Stack＝(stack)、字符串特征Str＝(strnum,strset)和代码规模特征Code＝(jump,inst,code)。

[0035] d).分析函数控制流图(CFG图)，计算每个节点的出入度，构造CFG有向图邻接矩阵如下：

[0036]
$$\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1k} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2k} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{t1} & x_{t2} & \cdots & x_{tk} & \cdots & x_{tn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nk} & \cdots & x_{nn} \end{bmatrix} \begin{matrix} \rightarrow o_1 \\ \rightarrow o_2 \\ \vdots \\ \rightarrow o_t \\ \vdots \\ \rightarrow o_n \end{matrix}$$
$$\begin{matrix} \downarrow & \downarrow & \cdots & \downarrow & \cdots & \downarrow \\ i_1 & i_2 & \cdots & i_k & \cdots & i_n \end{matrix}$$

[0037] 其中，$x_{tk}=\begin{cases}1 & \text{存在} n_t \rightarrow n_k \text{的有向边} \\ 0 & \text{不存在} n_t \rightarrow n_k \text{的有向边}\end{cases}$。其中，$n_t$、$n_k$表示CFG有向图的节点，$x_{tk}$表示节点$n_t$到节点$n_k$在有向图中的边连接情况，$o_1$～$o_n$表示出度，$i_1$～$i_n$表示入度。

[0038] 将函数控制流图转化为无向图，计算每个节点的度，构造CFG无向图邻接矩阵如下：

[0039]
$$\begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1k} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2k} & \cdots & y_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ y_{t1} & y_{t2} & \cdots & y_{tk} & \cdots & y_{tk} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nk} & \cdots & y_{nn} \end{bmatrix} \begin{matrix} \rightarrow d_1 \\ \rightarrow d_2 \\ \vdots \\ \rightarrow d_t \\ \vdots \\ \rightarrow d_n \end{matrix}$$

[0040] 其中，$y_{tk}=\begin{cases}1 & x_{tk}=1 \text{或} x_{kt}=1 \\ 0 & x_{tk}=0 \text{且} x_{kt}=0\end{cases}$。其中，$y_{tk}$表示节点$n_t$到节点$n_k$在无向图中的边连接情况。

[0041] e).对CFG有向图邻接矩阵和CFG无向图邻接矩阵进行度分析。基于CFG有向图邻接矩阵按公式(1)和(2)计算入度升序序列i＝(i_1,i_2,…,i_t,…,i_n)、出度升序序列o＝(o_1,o_2,…,o_t,…o_n)；基于CFG无向图邻接矩阵按公式(3)计算度升序序列d＝(d_1,d_2,…,d_t,…d_n)。构造度序列特征degreeList＝(i,o,d)。

[0042] 基于度升序序列d，按公式(4)计算平均度aveDegree，按公式(5)计算最大度maxDegree，并计算度的概率序列P＝(P_0,P_1,…$P_{maxDegree}$)，其中P_k表示出入度为k的节点占所有节点的比例。基于度的概率序列P按公式(6)计算图的熵entropy，构造度的基本属性特征degreeBasic＝(aveDegree,maxDegree,entropy)。

[0043] $$i_t=\sum_{k=1}^{n} x_{kt} \qquad (1)$$

[0044] $$o_t = \sum_{k=1}^{n} x_{tk} \quad (2)$$

[0045] $$d_t = \sum_{k=1}^{n} y_{tk} \quad (3)$$

[0046] $$aveDegree = \frac{\sum_{t=1}^{n} d_t}{n} \quad (4)$$

[0047] maxDegree＝max(d) (5)

[0048] $$entropy = -\sum_{k=1}^{maxDegree} P_k \log_2(P_k) \quad (6)$$

[0049] f).对CFG无向图邻接矩阵进行路径分析，计算入口基本块到任意其他基本块的最小距离，可以采用Floyd算法、Dinkstra算法等，构造升序距离序列如下：

[0050] path＝($minpath_1$,$minpath_2$,…,$minpath_{n-1}$)

[0051] 用python实现的Floyd算法如下：

[0052]
```
def countPath(A1):
    nodeNum = len(A1)
    c = list(sum(A1))
    s = c.index(0)
    A = A1
    for i in range(0,nodeNum):
        for j in range(0,nodeNum):
            if A1[i][j] == 0.0:
                A[i][j] = float('inf')
    dis = [A[s][i] for i in range(0, nodeNum)]
    flag = [0 for i in range(0, nodeNum)]
    for i in range(0, nodeNum):
        selected = -1;
        max_weight = 1E8;
        for j in range(0, nodeNum):
            if (flag[j] == 0) and (dis[j] <      max_weight):
```

```
                max_weight = dis[j]
                selected = j
        if selected != -1:
            flag[selected] = 1
            for j in range(0,nodeNum):
                if  j != selected:
                    if dis[j] > dis[selected] + A[selected][j]:
                        dis[j] = dis[selected] + A[selected][j]
    return dis
```

[0053]

[0054] 构造路径序列特征为pathList＝(path)。按公式(7)计算图的平均路径长度avePath。按公式(8)计算图直径diameter。构造路径基本属性特征pathBasic＝(avePath,diameter)。

[0055] avePath＝mean(path) (7)

[0056] diameter＝max{path} (8)

[0057] g).对CFG有向图邻接矩阵进行基本属性分析,计算节点数node,计算链路数edge。按公式(9)计算图的链路效率E(cfg),其中m表示有向CFG图的有向边数。按公式(10)计算图密度density,n表示有向CFG图的节点数。按公式(11)(12)计算图的聚类系数Cc(cfg),其中,c表示节点k的所有邻居节点构成的无向CFG图的子图边数,d_k表示节点k的度。构造CFG图基本属性特征cfgBasic＝(node,edge,E(cfg),density,Cc(cfg))。

[0058] $$E(cfg)=\frac{m-avePath}{m} \quad (9)$$

[0059] $$density=\frac{2m}{n(n-1)} \quad (10)$$

[0060] $$Cc_k=\frac{2c}{d_k(d_k-1)} \quad (11)$$

[0061] $$Cc(cfg)=\sum_{k=1}^{n}\frac{Cc_k}{n} \quad (12)$$

[0062] 按以上步骤操作,提取Call,Str,Stack,Code,degreeList,degreeBasic,pathList,pathBasic,cfgBasic9个方面共24个特征作为函数的特征。

[0063] 基于以上9个方面共24个特征,对每两个函数计算每个特征的相似度作为神经网络的输入向量。这9个方面的相似度分别是:调用关系相似度、栈空间相似度、字符串相似度,代码规模相似度、路径序列相似度、路径基本属性相似度,度序列相似度、度基本属性相似度和图规模相似度,共计24个相似度。计算方法如下:

[0064] 针对以下数值型特征:callfrom,callto,callfrom2,callto2,strnum,stack,jump,inst,code,aveDegree,maxDegree,entropy,avePath,diameter,node,edge,E(cfg),density,Cc(cfg),采用公式(13)进行相似度计算。

[0065] $$sim=\begin{cases}1.0 & a=0且b=0\\ 1.0-\dfrac{|a-b|}{\max(a,b)} & otherwise\end{cases} \quad (13)$$

[0066] 针对以下序列型特征：i，o，d，path，采用字符串编译距离算法进行计算。用python编译的字符串编译距离算法如下：

[0067]
```
1.  def countLevd(s1,s2):
2.      m = len(s1)
3.      n = len(s2)
4.      if m == 0 and n == 0:
5.          return 1.0
6.      elif m!=0 and n!=0:
7.          colsize = m + 1
8.          matrix = []
9.          for i in range((m + 1) * (n + 1)):
10.             matrix.append(0)
11.         for i in range(colsize):
12.             matrix[i] = i
13.         for i in range(n + 1):
14.             matrix[i * colsize] = i
15.         for i in range(n + 1)[1:n + 1]:
16.             for j in range(m + 1)[1:m + 1]:
17.                 cost = 0
18.                 if s1[j - 1] == s2[i - 1]:
```

```
                    cost = 0
                else:
                    cost = 1
                minValue = matrix[(i - 1) * colsize + j] + 1
                if minValue > matrix[i * colsize + j - 1] + 1:
                    minValue = matrix[i * colsize + j - 1] + 1
                if minValue > matrix[(i - 1) * colsize + j - 1] + cost:
                    minValue = matrix[(i - 1) * colsize + j - 1] + cost
                matrix[i * colsize + j] = minValue
        distance =  matrix[n * colsize + m]
        return 1.0 -float(distance) / max(m,n)
    else:
        if abs(m-n)<10:
            return 0.3
        else:
            return 0.0
```

[0068]

[0069] 针对以下集合型特征:strset,采用Jaccard系数进行相似度计算。计算方法见公式(14)。

[0070] $$sim=\begin{cases}0.1 & |A|=0且|B|=0\\ \dfrac{A\cap B}{A\cup B} & otherwise\end{cases} \qquad (14)$$

[0071] 如果两个函数名称相同,则标签为1,作为正样本,如果两个函数名称不同,则标签为0,作为负样本。

[0072] 2).构造神经网络分类器和训练样本集。建立若干初始神经网络,经验取值为10个神经网络。采用matlab神经网络工具箱实现,神经网络输入层的神经元个数为24个,代表以上计算所得的24个相似度。神经网络隐含层为1层,神经元个数经验取值为50。神经网络输出层的神经元个数为1个,代表输出的是整体相似度预测值。图2是1个神经网络结构示意图。训练样本采用自编译的、多平台、带符号表的函数集,根据匹配模式不同,可以采用对应匹配模式的训练样本集。从训练样本集中有放回的抽取80%的样本构造10个独立同分布的子训练样本集,作为每个神经网络的训练样本。

[0073] 3).训练神经网络分类器。图3是训练集成分类器方法。将对应的子训练样本集输入神经网络分类器进行训练,根据预测结果,调整神经网络的参数直至预测结果符合要求,此时神经网络分类器训练完毕。

[0074] 4).构造测试集。将已知的函数称为本地函数,将待检测的函数称为目标函数,根

据需求构建本地函数库和目标函数库。图4是候选集构造示意图。对每个本地函数提取24个特征，构造过滤器。过滤器是基于专业经验，对本地函数的某一特征，计算与其相似的函数的特征经验范围。如果待检测函数的这一特征超过了该特征范围，则认为该待检测函数与该本地函数不相似。对每个目标函数提取24个特征。对每个本地函数，用其对应的过滤器对目标函数库进行过滤，构建该本地函数的候选集，计算该本地函数与其对应候选集内每个目标函数的8个相似度值，作为测试样本。

[0075] 5).用神经网络分类器进行预测。图5是集成分类器预测方法示意图。将测试样本输入神经网络分类器中，获得预测值。

[0076] 6).根据预测值进行相似度排序，输出相似度前100的函数。上述步骤4)～6)的整体流程如图6所示。

[0077] 综上所述，本发明公开了一种基于神经网络的指令集无关的二进制相似代码检索方法。上面描述的应用场景及实施例，并非用于限定本发明，任何本领域技术人员在不脱离本发明的精神和范围内可作各种更动和润饰。因此本发明的保护范围视权利要求范围所界定。

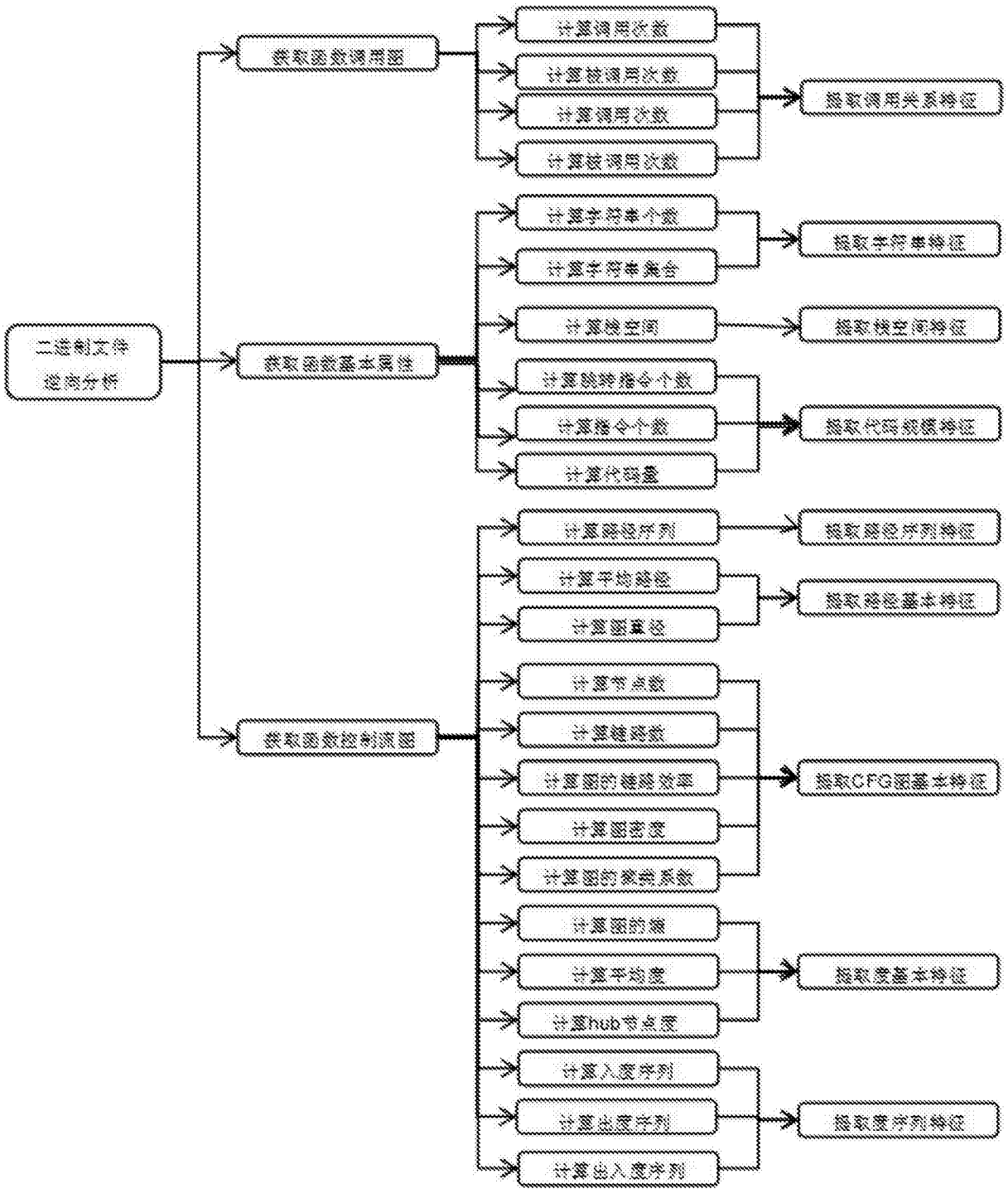

二进制文件
逆向分析
获取函数调用图
获取函数基本属性
获取函数控制流图
计算调用次数
计算被调用次数
计算调用个数
计算被调用次数
提取调用关系特征
计算字符串个数
计算字符串集合
提取字符串特征
计算栈空间
提取栈空间特征
计算跳转指令个数
计算指令个数
计算代码量
提取代码规模特征
计算路径序列
计算平均路径
计算圈复杂
提取路径基本特征
计算节点数
计算链路数
计算图的链路效率
计算图密度
计算图的聚类系数
提取CFG图基本特征
计算图的度
计算平均度
计算hub节点度
提取度基本特征
计算入度序列
计算出度序列
计算出入度序列
提取度序列特征

图1

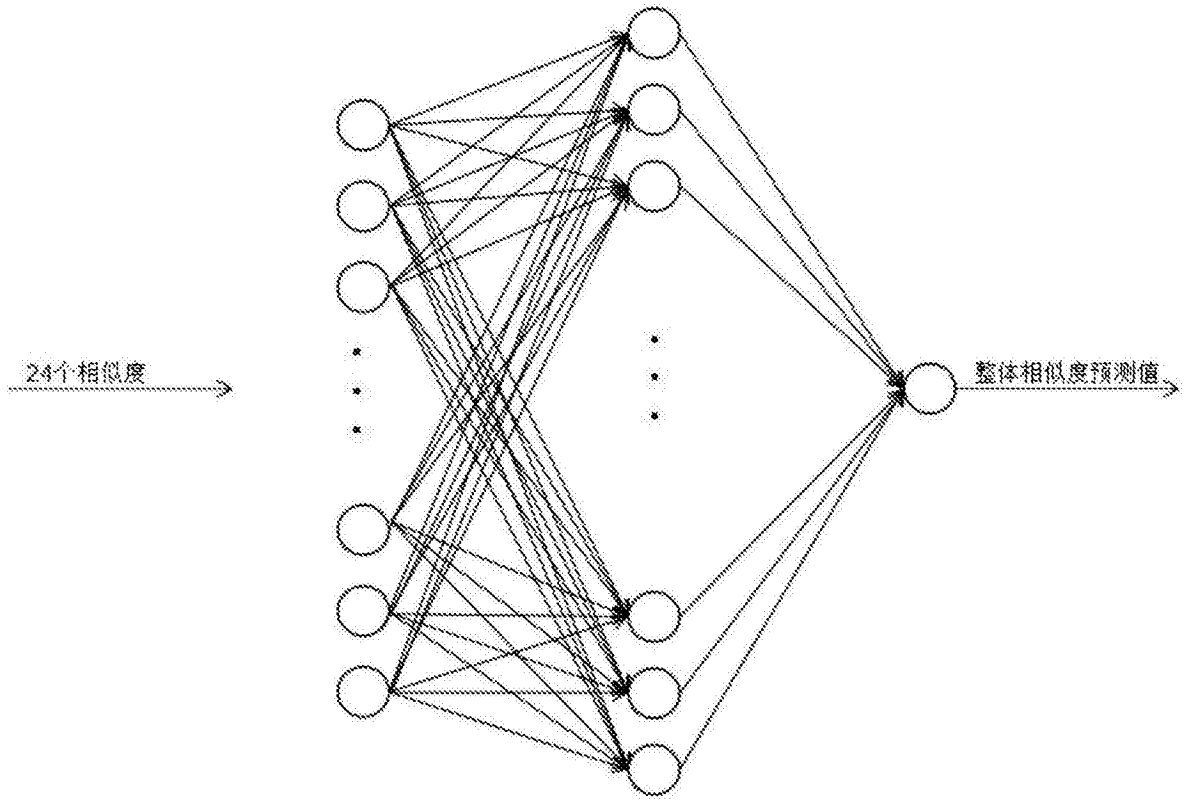

24个相似度
整体相似度预测值

图2

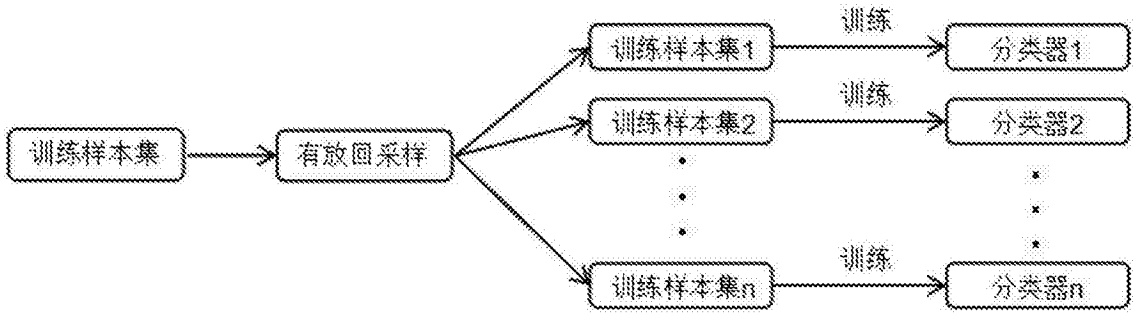

训练样本集
有放回采样
训练样本集1
训练样本集2
训练样本集n
训练
训练
训练
分类器1
分类器2
分类器n

图3

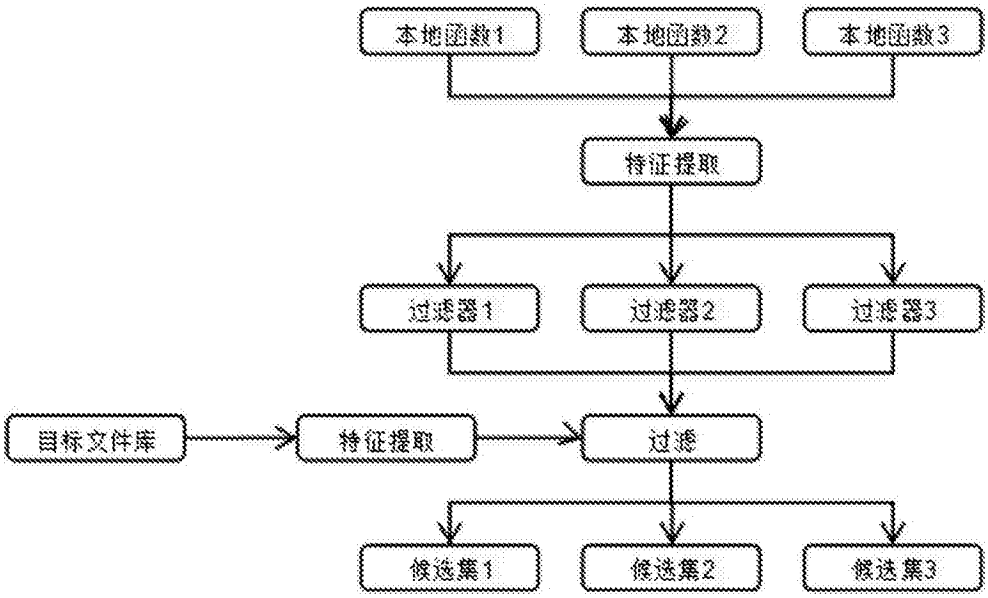

本地函数1
本地函数2
本地函数3
特征提取
过滤器1
过滤器2
过滤器3
目标文件库
特征提取
过滤
候选集1
候选集2
候选集3

图4

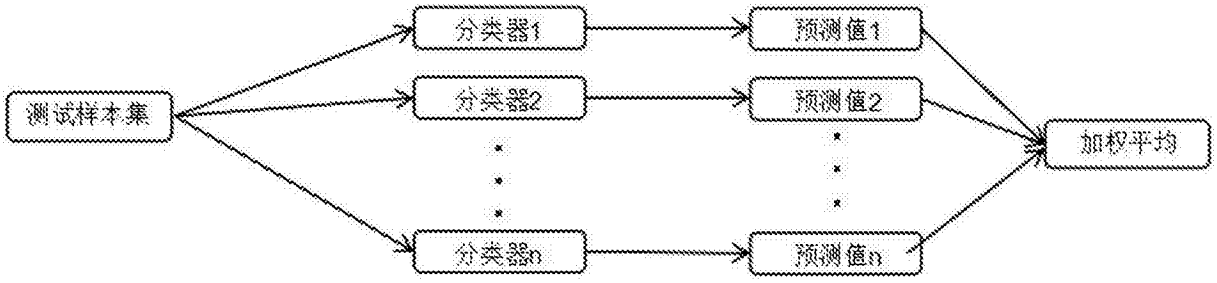

测试样本集
分类器1
分类器2
分类器n
预测值1
预测值2
预测值n
加权平均

图5

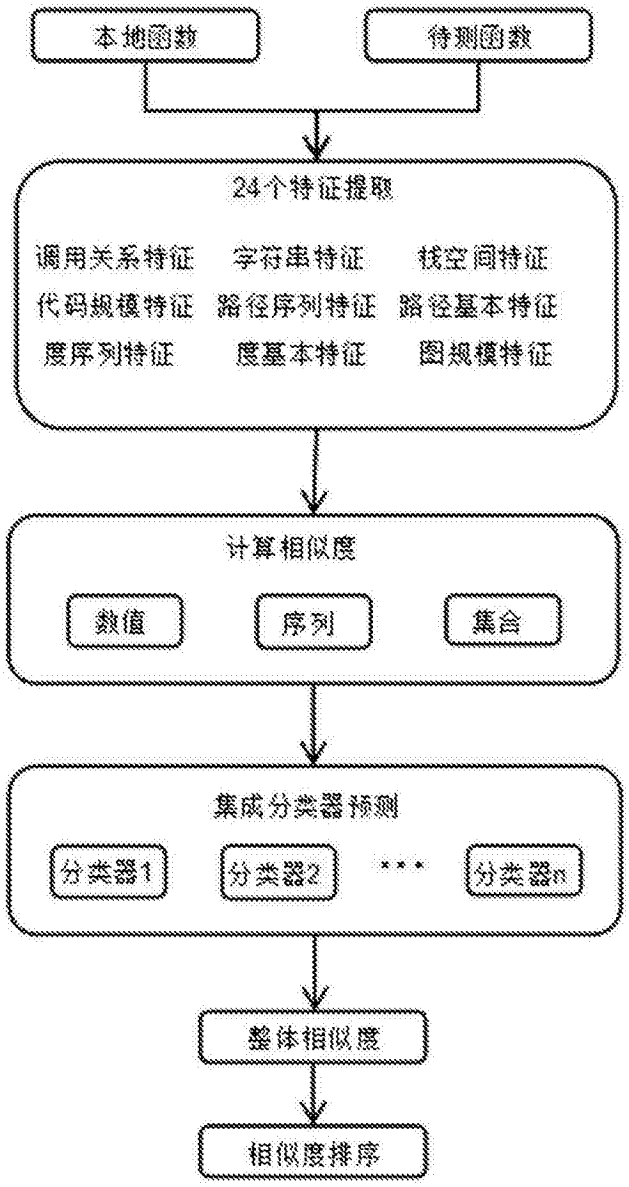

本地函数
待测函数
24个特征提取
调用关系特征
字符串特征
栈空间特征
代码规模特征
路径序列特征
路径基本特征
度序列特征
度基本特征
图规模特征
计算相似度
数值
序列
集合
集成分类器预测
分类器1
分类器2
...
分类器n
整体相似度
相似度排序

图6