(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 111753290 A
(43)申请公布日 2020.10.09

(21)申请号 202010454339.4

(22)申请日 2020.05.26

(71)申请人 郑州启明星辰信息安全技术有限公司
地址 450000 河南省郑州市郑州高新技术产业开发区科学大道与黄河路交叉口赛微大数据产业园1号楼11、12层
申请人 北京启明星辰信息安全技术有限公司
启明星辰信息技术集团股份有限公司

(72)发明人 刘洋 卞超轶

(74)专利代理机构 北京安信方达知识产权代理有限公司 11262
代理人 王素燕 栗若木

(51)Int.Cl.
G06F 21/53(2013.01)
G06F 21/56(2013.01)
G06K 9/62(2006.01)

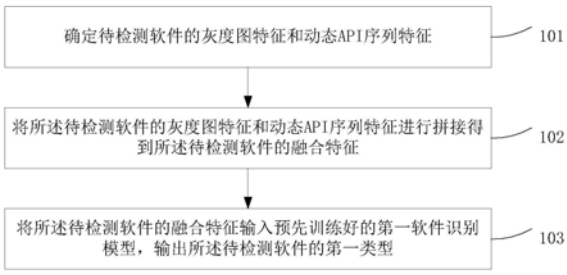
权利要求书2页 说明书18页 附图5页

(54)发明名称

软件类型的检测方法及相关设备

(57)摘要

本发明实施例公开了一种软件类型的检测方法及相关设备，其中方法之一包括：确定待检测软件的灰度图特征和动态API序列特征；将所述待检测软件的灰度图特征和动态API序列特征进行拼接得到所述待检测软件的融合特征；将所述待检测软件的融合特征输入预先训练好的第一软件识别模型，输出所述待检测软件的第一类型。如此，通过训练和识别软件的融合特征来进行检测，能够大大提高软件类型识别的准确率。



CN 111753290 A

1.一种软件类型的检测方法,包括:

确定待检测软件的灰度图特征和动态API序列特征;

将所述待检测软件的灰度图特征和动态API序列特征进行拼接得到所述待检测软件的融合特征;

将所述待检测软件的融合特征输入预先训练好的第一软件识别模型,输出所述待检测软件的第一类型;

其中,所述第一软件识别模型是对XGBoost模型通过第一软件样本进行训练得到的XGBoost二分类模型,以软件的融合特征作为输入,以软件的第一类型作为输出;所述第一软件样本为多个软件各自的已标注第一类型的融合特征。

2.根据权利要求1所述的检测方法,其特征在于,

所述第一类型为正常软件或者恶意软件。

3.根据权利要求2所述的检测方法,其特征在于,当所述待检测软件的第一类型为恶意软件时,该方法还包括:

将所述待检测软件的融合特征输入预先训练好的第二软件识别模型,输出所述待检测软件的第二类型;

其中,所述第二软件识别模型是对XGBoost模型通过第二软件样本进行训练得到的XGBoost多分类模型,以软件的融合特征作为输入,以软件的第二类型作为输出;所述第二软件样本为多个不同类型的恶意软件各自的已标注第二类型的融合特征。

4.根据权利要求3所述的检测方法,其特征在于,

所述第二类型包括以下全部或者部分类型:勒索病毒、挖坑程序、DDos木马、蠕虫病毒、感染型病毒、后门程序、木马程序。

5.根据权利要求1所述的检测方法,其特征在于,所述确定待检测软件的灰度图特征,包括:

对所述待检测软件进行反汇编得到其二进制代码文件;

将所述二进制代码文件转换成灰度图;

提取所述灰度图的特征向量作为待检测软件的灰度图特征。

6.根据权利要求1所述的检测方法,其特征在于,

所述确定待检测软件的动态API序列特征,包括:

将所述待检测软件在动态沙箱中模拟运行,得到运行后的动态API序列;

执行以下至少一种操作:

提取该动态API序列的特征向量;确定该动态API序列的N-gram特征向量;确定该动态API序列的分布式词向量;

所述动态API序列特征包括以下一种或者多种:

该动态API序列的特征向量、该动态API序列的N-gram特征向量、该动态API序列的分布式词向量。

7.一种软件类型的检测方法,包括:

确定待检测软件的灰度图特征和动态API序列特征;

将所述待检测软件的灰度图特征和动态API序列特征进行拼接得到所述待检测软件的融合特征;

将所述待检测软件的融合特征输入预先训练好的第三软件识别模型，输出所述待检测软件的第三类型；

其中，所述第三软件识别模型是对XGBoost模型通过第三软件样本进行训练得到的XGBoost多分类模型，以软件的融合特征作为输入，以软件的第三类型作为输出；所述第三软件样本为多个不同类型的软件各自的已标注第三类型的融合特征。

8.根据权利要求7所述的检测方法，其特征在于，

所述第三类型包括正常软件，以及以下全部或者部分类型：勒索病毒、挖坑程序、DDos木马、蠕虫病毒、感染型病毒、后门程序、木马程序。

9.一种电子装置，其特征在于，包括：存储器、处理器及存储在所述存储器上并可在所述处理器上运行的计算机程序，所述计算机程序被所述处理器执行时实现如权利要求1至8中任一项所述软件类型的检测方法。

10.一种计算机可读存储介质，其特征在于，所述计算机可读存储介质上存储有信息处理程序，所述信息处理程序被处理器执行时实现如权利要求1-8任一项中所述软件类型的检测方法。

软件类型的检测方法及相关设备

技术领域

[0001] 本发明实施例涉及网络安全技术，尤指一种软件类型的检测方法及相关设备。

背景技术

[0002] 近些年来，在大量不法资金的助力下，黑灰产业蓬勃发展，市值已高达千亿元规模。在此基础上，黑客根据现有反病毒软件存在的漏洞，研发出各种逃避反病毒软件的工具。而如果不能正确识别恶意软件，就可能会导致恶意软件大肆传播，就会给整个社会和国家带来非常巨大的危害。所以，检测恶意软件就显得至关重要。

[0003] 在检测恶意软件的应用中，绝大多数的方法是利用深度学习模型对恶意软件的单一特征进行训练从而对恶意软件进行检测和识别。例如，对恶意软件的图像样本数据训练卷积神经网络从而利用训练好的卷积神经网络识别恶意软件。又例如，利用机器学习算法对恶意软件的动态行为数据进行训练从而利用训练好的模型识别恶意软件。

[0004] 由此可见，现有技术中的恶意软件的检测方法都只是通过训练和识别软件的单一特征来进行检测，导致对恶意软件识别的准确率还不够高。

发明内容

[0005] 有鉴于此，本发明实施例提供了一种软件类型的检测方法，包括：

[0006] 确定待检测软件的灰度图特征和动态API序列特征；

[0007] 将所述待检测软件的灰度图特征和动态API序列特征进行拼接得到所述待检测软件的融合特征；

[0008] 将所述待检测软件的融合特征输入预先训练好的第一软件识别模型，输出所述待检测软件的第一类型；

[0009] 其中，所述第一软件识别模型是对XGBoost模型通过第一软件样本进行训练得到的XGBoost二分类模型，以软件的融合特征作为输入，以软件的第一类型作为输出；所述第一软件样本为多个软件各自的已标注第一类型的融合特征。

[0010] 本发明实施例还提供了一种软件类型的检测方法，包括：

[0011] 确定待检测软件的灰度图特征和动态API序列特征；

[0012] 将所述待检测软件的灰度图特征和动态API序列特征进行拼接得到所述待检测软件的融合特征；

[0013] 将所述待检测软件的融合特征输入预先训练好的第三软件识别模型，输出所述待检测软件的第三类型；

[0014] 其中，所述第三软件识别模型是对XGBoost模型通过第三软件样本进行训练得到的XGBoost多分类模型，以软件的融合特征作为输入，以软件的第三类型作为输出；所述第三软件样本为多个不同类型的软件各自的已标注第三类型的融合特征.

[0015] 本发明实施例还提供了一种电子装置，包括：存储器、处理器及存储在所述存储器上并可在所述处理器上运行的计算机程序，所述计算机程序被所述处理器执行时实现上述

任一项所述软件类型的检测方法。

[0016] 本发明实施例还提供了一种计算机可读存储介质，所述计算机可读存储介质上存储有信息处理程序，所述信息处理程序被处理器执行时实现上述任一项所述软件类型的检测方法。

[0017] 本发明实施例提供的技术方案，通过训练和识别软件的融合特征来进行检测，能够大大提高软件类型识别的准确率。

附图说明

[0018] 附图用来提供对本申请技术方案的理解，并且构成说明书的一部分，与本申请的实施例一起用于解释本申请的技术方案，并不构成对本申请技术方案的限制。

[0019] 图1为本发明一实施例提供的一种软件类型的检测方法的流程示意图；

[0020] 图2为本发明另一实施例提供的一种软件类型的检测方法的流程示意图；

[0021] 图3为本发明一实施例中提供的一种模型训练方法的流程示意图；

[0022] 图4为本发明一实施例中多层CNN的结构示意图；

[0023] 图5为本发明另一实施例提供的一种软件类型的检测方法的流程示意图；

[0024] 图6为本发明另一实施例提供的一种软件类型的检测方法的流程示意图；

[0025] 图7为本发明另一实施例提供的一种软件类型的检测方法的流程示意图；

[0026] 图8为本发明一实施例提供的一种软件类型的检测装置的流程示意图；

[0027] 图9为本发明另一实施例提供的一种软件类型的检测装置的流程示意图。

具体实施方式

[0028] 本申请描述了多个实施例，但是该描述是示例性的，而不是限制性的，并且对于本领域的普通技术人员来说显而易见的是，在本申请所描述的实施例包含的范围内可以有更多的实施例和实现方案。尽管在附图中示出了许多可能的特征组合，并在具体实施方式中进行了讨论，但是所公开的特征的许多其它组合方式也是可能的。除非特意加以限制的情况以外，任何实施例的任何特征或元件可以与任何其它实施例中的任何其他特征或元件结合使用，或可以替代任何其它实施例中的任何其他特征或元件。

[0029] 本申请包括并设想了与本领域普通技术人员已知的特征和元件的组合。本申请已经公开的实施例、特征和元件也可以与任何常规特征或元件组合，以形成由权利要求限定的独特的发明方案。任何实施例的任何特征或元件也可以与来自其它发明方案的特征或元件组合，以形成另一个由权利要求限定的独特的发明方案。因此，应当理解，在本申请中示出和/或讨论的任何特征可以单独地或以任何适当的组合来实现。因此，除了根据所附权利要求及其等同替换所做的限制以外，实施例不受其它限制。此外，可以在所附权利要求的保护范围内进行各种修改和改变。

[0030] 此外，在描述具有代表性的实施例时，说明书可能已经将方法和/或过程呈现为特定的步骤序列。然而，在该方法或过程不依赖于本文所述步骤的特定顺序的程度上，该方法或过程不应限于所述的特定顺序的步骤。如本领域普通技术人员将理解的，其它的步骤顺序也是可能的。因此，说明书中阐述的步骤的特定顺序不应被解释为对权利要求的限制。此外，针对该方法和/或过程的权利要求不应限于按照所写顺序执行它们的步骤，本领域技术

人员可以容易地理解,这些顺序可以变化,并且仍然保持在本申请实施例的精神和范围内。

[0031] 图1为本发明一实施例提供的一种软件类型的检测方法的流程示意图,如图1所示,该方法包括:

[0032] 步骤101,确定待检测软件的灰度图特征和动态API序列特征;

[0033] 步骤102,将所述待检测软件的灰度图特征和动态API序列特征进行拼接得到所述待检测软件的融合特征;

[0034] 步骤103,将所述待检测软件的融合特征输入预先训练好的第一软件识别模型,输出所述待检测软件的第一类型;

[0035] 其中,所述第一软件识别模型是对XGBoost模型通过第一软件样本进行训练得到的XGBoost二分类模型,以软件的融合特征作为输入,以软件的第一类型作为输出;所述第一软件样本为多个软件各自的已标注第一类型的融合特征。

[0036] 在一示例中,所述第一类型为正常软件或者恶意软件。

[0037] 在一示例中,当所述待检测软件的第一类型为恶意软件时,该方法还包括:

[0038] 将所述待检测软件的融合特征输入预先训练好的第二软件识别模型,输出所述待检测软件的第二类型;

[0039] 其中,所述第二软件识别模型是对XGBoost模型通过第二软件样本进行训练得到的XGBoost多分类模型,以软件的融合特征作为输入,以软件的第二类型作为输出;所述第二软件样本为多个不同类型的恶意软件各自的已标注第二类型的融合特征。

[0040] 在一示例中,所述第二类型包括以下全部或者部分类型:勒索病毒、挖坑程序、DDos木马、蠕虫病毒、感染型病毒、后门程序、木马程序。

[0041] 在一示例中,所述确定待检测软件的灰度图特征,包括:

[0042] 对所述待检测软件进行反汇编得到其二进制代码文件;

[0043] 将所述二进制代码文件转换成灰度图;

[0044] 提取所述灰度图的特征向量作为待检测软件的灰度图特征。

[0045] 在一示例中,所述确定待检测软件的动态API序列特征,包括:

[0046] 将所述待检测软件在动态沙箱中模拟运行,得到运行后的动态API序列;

[0047] 执行以下至少一种操作:

[0048] 提取该动态API序列的特征向量;确定该动态API序列的N-gram特征向量;确定该动态API序列的分布式词向量;

[0049] 所述动态API序列特征包括以下一种或者多种:

[0050] 该动态API序列的特征向量、该动态API序列的N-gram特征向量、该动态API序列的分布式词向量。

[0051] 本发明实施例提供的技术方案,通过训练和识别软件的融合特征来进行检测,能够大大提高软件类型识别的准确率。

[0052] 图2为本发明另一实施例提供得一种软件类型的检测方法的流程示意图,如图2所示,该方法包括:

[0053] 步骤201,对XGBoost模型通过第一软件样本进行训练得到第一软件识别模型;

[0054] 其中,所述第一软件识别模型是对XGBoost模型通过第一软件样本进行训练得到的XGBoost二分类模型,以软件的融合特征作为输入,以软件的第一类型作为输出。

[0055] 其中，该第一软件样本为为多个软件各自的已标注第一类型的融合特征。

[0056] 其中，该第一类型为正常软件或者恶意软件。

[0057] 其中，软件的融合特征是指由该软件的灰度图特征和动态API序列特征拼接组成的特征。该软件的灰度图特征是指该软件对应的灰度图的特征向量，该软件的动态API序列特征包括以下一种或者多种：该动态API序列的特征向量、该动态API序列的N-gram特征、该动态API序列的分布式词向量。

[0058] 其中，该第一软件识别模型的训练过程包括对第一软件样本的灰度图特征和动态API序列特征两个模态的融合学习。由于该两种模态的物理含义不同，所以无法直接将其一种模态转换成另外一种模态进行学习(例如把文本信息转换到图像，反之亦然)，例如，文本的维度是文本的长度*词向量的个数，而图像的维度是图像的长*高，两者如果不进行处理，是无法转换到一起的；如果强行把文本的维度缩放到图像的维度上，必然会造成大量的信息丢失；所以必须设计具体可行的方案才能够对两个模态的特征进行融合学习。另外，由于神经网络学习能力强，能够自动提取特征，但容易造成过拟合现象，并且具有不可解释性，不利于产品的迭代，而XGBoost是提升树模型，它具有不容易过拟合、可解释性、训练速度快等特点。其中XGBoost模型的可解释性极其重要，可以在训练后得到特征的重要性，从而反推得到模型的效果和稳定性。比如得到排名前十的特征，通过统计得到不同类别间这十个特征的差异性，如果差异较大，则说明选取到了具有区分性的特征，模型效果很好，反之亦然。

[0059] 因此，本实施例中，结合两者的优点，先通过CNN(卷积神经网络，Convolutional Neural Networks)网络提取不同模态中不同层次的特征向量，然后再将提取后的特征向量和其他维度的特征向量作为输入给XGBoost进行学习训练。例如，如图3所示，在灰度图模态学习中，先通过多层CNN网络提取特征向量，将倒数第二层即第二个全连接层的输入作为提取后的特征向量，即灰度图特征；在动态API序列模态学习中，由于动态API序列本质上是文本，所以先获取分布式词向量，基于分布式词向量使用单层的TextCNN网络提取特征向量，然后将两个模态提取后的特征向量和API序列的N-gram特征以及分布式词向量拼接到一起，然后将拼接后的融合特征输入XGBoost进行学习。例如，多层CNN提取的特征向量为512维，TextCNN提取的特征向量为128维，N-gram特征向量为90000维，分布式词向量的特征为300，则拼接后的特征向量为90000+300+128+512＝90940。之所以在API序列模态学习中，将上述的三种特征进行融合，是由于三种特征的主要信息不同，其中TextCNN中的卷积核的长度可以设置的较大(如10～20)，从而提取出对文本分类影响较大的长段落对应的特征，而N-gram由于N的限制(一般不超过3，否则会引起维度灾难)，表示的是文本分类影响较大的短段落的特征，而分布式词向量表示的是文本的整体语义，相比其它的特征来说它具有一定的抽象和概括能力。当然，也可以只选用API序列三种特征中的一种或者两种特征向量与灰度图特征向量进行拼接后输入到XGBoost。

[0060] 在一示例中，所述对XGBoost模型通过第一软件样本进行训练得到第一软件识别模型，包括：

[0061] 对所述第一软件样本中的每一个软件执行如下操作：

[0062] 确定该软件的灰度图和动态API序列；

[0063] 提取该灰度图的特征向量；

[0064] 提取该动态API序列的特征向量；

[0065] 将该灰度图的特征向量和该动态API序列的特征向量进行拼接得到该软件的融合特征；

[0066] 使用XGBoost模型对该软件的融合特征进行训练，将训练得到的XGBoost二分类模型作为所述第一软件识别模型。

[0067] 在另一示例中，所述对XGBoost模型通过第一软件样本进行训练得到第一软件识别模型，包括：

[0068] 对所述第一软件样本中的每一个软件执行如下操作：

[0069] 确定该软件的灰度图和动态API序列；

[0070] 提取该灰度图的特征向量；

[0071] 提取该动态API序列的特征向量；

[0072] 确定该动态API序列的N-gram特征；

[0073] 将该灰度图的特征向量和该动态API序列的特征向量和N-gram特征进行拼接得到该软件的融合特征；

[0074] 使用XGBoost模型对该软件的融合特征进行训练，将训练得到的XGBoost二分类模型作为所述第一软件识别模型。

[0075] 在另一示例中，所述对XGBoost模型通过第一软件样本进行训练得到第一软件识别模型，包括：

[0076] 对所述第一软件样本中的每一个软件执行如下操作：

[0077] 确定该软件的灰度图和动态API序列；

[0078] 提取该灰度图的特征向量；

[0079] 提取该动态API序列的特征向量；

[0080] 确定该动态API序列的N-gram特征；

[0081] 确定该动态API序列的分布式词向量；

[0082] 将该灰度图的特征向量和该动态API序列的特征向量、N-gram特征和分布式词向量进行拼接得到该软件的融合特征；

[0083] 使用XGBoost模型对该软件的融合特征进行训练，将训练得到的XGBoost二分类模型作为所述第一软件识别模型。

[0084] 在一示例中，确定软件的灰度图，包括：

[0085] 对该软件进行反汇编得到其二进制代码文件；

[0086] 将该二进制代码文件转换成灰度图。

[0087] 其中，由于软件的每个字节的范围是在十六进制00-FF之间，正好对应灰度图十进制0-255(0为黑色，255为白色)，其恰好覆盖了整个灰度值得范围。因此，将软件的二进制代码文件按照字节进行分割，然后基于十六进制码与图像灰度值的对应关系得到每个字节对应的灰度值，如此就可以将二进制代码文件转换成灰度图。

[0088] 在一示例中，确定软件的动态API序列，包括：

[0089] 将该软件在动态沙箱中模拟运行，得到运行后的动态API序列。

[0090] 其中，沙箱(Sandboxie)是一个虚拟系统程序，允许在沙盘环境中运行浏览器或其他程序，运行所产生的变化可以随后删除。它创造了一个类似沙盒的独立作业环境，在其内

部运行的程序并不能对硬盘产生永久性的影响。在网络安全中，沙箱指在隔离环境中，用以测试不受信任的文件或应用程序等行为的工具。动态沙箱可以包括虚拟操作系统层，首先将待检测文件在动态沙箱的虚拟操作系统层中运行，然后在运行过程中模拟对操作系统的应用程序编程接口API(application programming interface)进行调用操作，提取产生的动态行为信息即为动态API序列，该动态API序列为一种文本信息，例如读文件、写注册表等动态行为信息。

[0091] 其中，在API序列模态学习中，由于API序列本质上是文本，所以使用单层的TextCNN网络提取特征向量。

[0092] 其中，N-Gram是一种基于统计语言模型的算法，其基本思想是将文本里面的内容按照字节进行大小为N的滑动窗口操作，形成了长度是N的字节片段序列。每一个字节片段称为gram，对所有gram的出现频度进行统计，并且按照事先设定好的阈值进行过滤，形成关键gram列表，也就是这个文本的向量特征即N-gram特征，列表中的每一种gram就是一个特征向量维度。

[0093] 在一示例中，确定动态API序列的分布式词向量，包括：利用词向量算法训练该动态API序列的分布式词向量。

[0094] 其中，该词向量算法包括以下一种或者多种：Word2vec、Glove、Fasttext。例如使用Word2vec、Glove、Fasttext三种方法训练词向量，并将该动态API序列对应的三种词向量合并到一起作为的该动态API序列的分布式词向量。

[0095] 步骤202，确定待检测软件的灰度图特征和动态API序列特征；

[0096] 其中，确定该待检测软件的灰度图特征，包括：

[0097] 对该软件进行反汇编得到其二进制代码文件；

[0098] 将该二进制代码文件转换成灰度图；

[0099] 提取所述灰度图的特征向量作为待检测软件的灰度图特征。

[0100] 在一示例中，可以通过卷积神经网络(Convolutional Neural Network，CNN)提取所述灰度图的特征向量。例如，采用图4所示的多层CNN网络提取特征向量，将倒数第二层即第二个全连接层的输入作为提取后的特征向量，即灰度图特征。

[0101] 其中，确定该待检测软件的动态API序列特征，包括：

[0102] 将该软件在动态沙箱中模拟运行，得到运行后的动态API序列；

[0103] 执行以下至少一种操作：

[0104] 提取该动态API序列的特征向量；

[0105] 和/或，确定该动态API序列的N-gram特征；

[0106] 和/或，确定该动态API序列的分布式词向量。

[0107] 其中，所述动态API序列特征包括以下一种或者多种：

[0108] 该动态API序列的特征向量、该动态API序列的N-gram特征向量、该动态API序列的分布式词向量。

[0109] 在一示例中，可以使用单层的TextCNN网络提取该动态API序列的特征向量。

[0110] 步骤203，将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征；

[0111] 在一示例中，当该软件的动态API序列特征包括：动态API序列的特征向量时，所述

将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征，包括：

[0112] 将该待检测软件的灰度图特征和该待检测软件的动态API序列的特征向量进行拼接得到该软件的融合特征。

[0113] 例如，灰度图特征即通过多层CNN提取的特征向量为512维，动态API序列的特征向量即通过TextCNN提取的特征向量为128维，则拼接后的特征向量为512+128＝640。

[0114] 在一示例中，当该软件的动态API序列特征包括：动态API序列的特征向量和N-gram特征向量时，所述将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征，包括：

[0115] 将该待检测软件的灰度图特征和该待检测软件的动态API序列的特征向量以及N-gram特征向量进行拼接得到该软件的融合特征。

[0116] 例如，灰度图特征即通过多层CNN提取的特征向量为512维，动态API序列的特征向量即通过TextCNN提取的特征向量为128维，N-gram特征向量为90000维，则拼接后的特征向量为512+128+90000＝90640。

[0117] 在一示例中，当该软件的动态API序列特征包括：动态API序列的特征向量、N-gram特征向量和分布式词向量时，所述将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征，包括：

[0118] 将该待检测软件的灰度图特征和该待检测软件的动态API序列的特征向量、N-gram特征向量以及分布式词向量进行拼接得到该软件的融合特征。

[0119] 例如，多层CNN提取的特征向量为512维，TextCNN提取的特征向量为128维，N-gram特征向量为90000维，分布式词向量的特征为300，则拼接后的特征向量为90000+300+128+512＝90940。

[0120] 步骤204，将所述待检测软件的融合特征输入所述第一软件识别模型，输出所述待检测软件的第一类型。

[0121] 本发明实施例提供的技术方案，预先训练第一软件识别模型，然后利用该第一软件识别模型对待检测软件的类型进行检测，不仅能够大大提高软件类型识别的准确率，而且提升了检测效率。

[0122] 图5为本发明另一实施例提供得一种软件类型的检测方法的流程示意图，如图5所示，该方法包括：

[0123] 步骤501，对XGBoost模型通过第一软件样本进行训练得到第一软件识别模型；

[0124] 具体的训练过程与上一实施例相同，在此不再赘述。

[0125] 步骤502，对XGBoost模型通过第二软件样本进行训练得到第二软件识别模型；

[0126] 其中，所述第二软件识别模型是对XGBoost模型通过第二软件样本进行训练得到的XGBoost多分类模型，以软件的融合特征作为输入，以软件的第二类型作为输出。

[0127] 其中，所述第二软件样本为多个不同类型的恶意软件各自的已标注第二类型的融合特征。

[0128] 其中，所述第二类型包括以下全部或者部分类型：勒索病毒、挖坑程序、DDos木马、蠕虫病毒、感染型病毒、后门程序、木马程序。

[0129] 其中，该第二软件识别模型以恶意软件的融合特征作为输入，以该软件的恶意软

件类型作为输出。软件的融合特征是指由该软件的灰度图特征和动态API序列特征拼接组成的特征。该软件的灰度图特征是指该软件对应的灰度图的特征向量，该软件的动态API序列特征包括以下一种或者多种：该动态API序列的特征向量、该动态API序列的N-gram特征、该动态API序列的分布式词向量。

[0130] 其中，该第二软件识别模型的训练过程包括对第二软件样本的灰度图和动态API序列两个模态的融合学习。另外，由于神经网络学习能力强，能够自动提取特征，但容易造成过拟合现象，并且具有不可解释性，不利于产品的迭代，而XGBoost是提升树模型，它具有不容易过拟合、可解释性、训练速度快等特点。因此，本实施例中，结合两者的优点，先通过CNN网络提取不同模态中不同层次的特征向量，然后再将提取后的特征向量和其他维度的特征向量作为输入给XGBoost进行学习训练。例如，如图3所示，在灰度图模态学习中，先通过多层CNN网络提取特征向量，将倒数第二层即第二个全连接层的输入作为提取后的特征向量，即灰度图特征；在动态API序列模态学习中，由于动态API序列本质上是文本，所以先获取分布式词向量，基于分布式词向量使用单层的TextCNN网络提取特征向量，然后将两个模态提取后的特征向量和API序列的N-gram特征以及分布式词向量拼接到一起，然后将拼接后的融合特征输入XGBoost进行学习。例如，多层CNN提取的特征向量为512维，TextCNN提取的特征向量为128维，N-gram特征向量为90000维，分布式词向量的特征为300，则拼接后的特征向量为90000+300+128+512＝90940。之所以在API序列模态学习中，将上述的三种特征进行融合，是由于三种特征的主要信息不同，其中TextCNN中的卷积核的长度可以设置的较大(如10～20)，从而提取出对文本分类影响较大的长段落对应的特征，而N-gram由于N的限制(一般不超过3，否则会引起维度灾难)，表示的是文本分类影响较大的短段落的特征，而分布式词向量表示的是文本的整体语义，相比其它的特征来说它具有一定的抽象和概括能力。当然，也可以只选用API序列三种特征中的一种或者两种特征向量与灰度图特征向量进行拼接后输入到XGBoost。

[0131] 在一示例中，所述对XGBoost模型通过第二软件样本进行训练得到第二软件识别模型，包括：

[0132] 对所述第二软件样本中的每一个软件执行如下操作：

[0133] 确定该软件的灰度图和动态API序列；

[0134] 提取该灰度图的特征向量；

[0135] 提取该动态API序列的特征向量；

[0136] 将该灰度图的特征向量和该动态API序列的特征向量进行拼接得到该软件的融合特征；

[0137] 使用XGBoost模型对该软件的融合特征进行训练，将训练得到的XGBoost多分类模型作为所述第二软件识别模型。

[0138] 在另一示例中，所述对XGBoost模型通过第二软件样本进行训练得到第二软件识别模型，包括：

[0139] 对所述第二软件样本中的每一个软件执行如下操作：

[0140] 确定该软件的灰度图和动态API序列；

[0141] 提取该灰度图的特征向量；

[0142] 提取该动态API序列的特征向量；

[0143] 确定该动态API序列的N-gram特征；

[0144] 将该灰度图的特征向量和该动态API序列的特征向量和N-gram特征进行拼接得到该软件的融合特征；

[0145] 使用XGBoost模型对该软件的融合特征进行训练，将训练得到的XGBoost多分类模型作为所述第二软件识别模型。

[0146] 在另一示例中，所述对XGBoost模型通过第二软件样本进行训练得到第二软件识别模型，包括：

[0147] 对所述第二软件样本中的每一个软件执行如下操作：

[0148] 确定该软件的灰度图和动态API序列；

[0149] 提取该灰度图的特征向量；

[0150] 提取该动态API序列的特征向量；

[0151] 确定该动态API序列的N-gram特征；

[0152] 确定该动态API序列的分布式词向量；

[0153] 将该灰度图的特征向量和该动态API序列的特征向量、N-gram特征和分布式词向量进行拼接得到该软件的融合特征；

[0154] 使用XGBoost模型对该软件的融合特征进行训练，将训练得到的XGBoost多分类模型作为所述第二软件识别模型。

[0155] 在一示例中，确定软件的灰度图，包括：

[0156] 对该软件进行反汇编得到其二进制代码文件；

[0157] 将该二进制代码文件转换成灰度图。

[0158] 其中，由于软件的每个字节的范围是在十六进制00-FF之间，正好对应灰度图十进制0-255(0为黑色，255为白色)，其恰好覆盖了整个灰度值得范围。因此，将软件的二进制代码文件按照字节进行分割，然后基于十六进制码与图像灰度值的对应关系得到每个字节对应的灰度值，如此就可以将二进制代码文件转换成灰度图。

[0159] 在一示例中，确定软件的动态API序列，包括：

[0160] 将该软件在动态沙箱中模拟运行，得到运行后的动态API序列。

[0161] 其中，沙箱(Sandboxie)是一个虚拟系统程序，允许在沙盘环境中运行浏览器或其他程序，运行所产生的变化可以随后删除。它创造了一个类似沙盒的独立作业环境，在其内部运行的程序并不能对硬盘产生永久性的影响。在网络安全中，沙箱指在隔离环境中，用以测试不受信任的文件或应用程序等行为的工具。动态沙箱可以包括虚拟操作系统层，首先将待检测文件在动态沙箱的虚拟操作系统层中运行，然后在运行过程中模拟对操作系统的应用程序编程接口API(application programming interface)进行调用操作，提取产生的动态行为信息即为动态API序列，该动态API序列为一种文本信息，例如读文件、写注册表等动态行为信息。

[0162] 其中，在API序列模态学习中，由于API序列本质上是文本，所以使用单层的TextCNN网络提取特征向量。

[0163] 其中，N-Gram是一种基于统计语言模型的算法，其基本思想是将文本里面的内容按照字节进行大小为N的滑动窗口操作，形成了长度是N的字节片段序列。每一个字节片段称为gram，对所有gram的出现频度进行统计，并且按照事先设定好的阈值进行过滤，形成关

键gram列表，也就是这个文本的向量特征即N-gram特征，列表中的每一种gram就是一个特征向量维度。

[0164] 在一示例中，确定动态API序列的分布式词向量，包括：利用词向量算法训练该动态API序列的分布式词向量。

[0165] 其中，该词向量算法包括以下一种或者多种：Word2vec、Glove、Fasttext。例如使用Word2vec、Glove、Fasttext三种方法训练词向量，并将该动态API序列对应的三种词向量合并到一起作为的该动态API序列的分布式词向量。

[0166] 步骤503，确定待检测软件的灰度图特征和动态API序列特征；

[0167] 其中，确定该待检测软件的灰度图特征，包括：

[0168] 对该软件进行反汇编得到其二进制代码文件；

[0169] 将该二进制代码文件转换成灰度图；

[0170] 提取所述灰度图的特征向量作为待检测软件的灰度图特征。

[0171] 在一示例中，可以通过卷积神经网络(Convolutional Neural Network，CNN)提取所述灰度图的特征向量。例如，采用图3所示的多层CNN网络提取特征向量，将倒数第二层即第二个全连接层的输入作为提取后的特征向量，即灰度图特征。

[0172] 其中，确定该待检测软件的动态API序列特征，包括：

[0173] 将该软件在动态沙箱中模拟运行，得到运行后的动态API序列；

[0174] 执行以下至少一种操作：

[0175] 提取该动态API序列的特征向量；

[0176] 和/或，确定该动态API序列的N-gram特征；

[0177] 和/或，确定该动态API序列的分布式词向量。

[0178] 其中，所述动态API序列特征包括以下一种或者多种：

[0179] 该动态API序列的特征向量、该动态API序列的N-gram特征向量、该动态API序列的分布式词向量。

[0180] 在一示例中，可以使用单层的TextCNN网络提取该动态API序列的特征向量。

[0181] 步骤504，将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征；

[0182] 在一示例中，当该软件的动态API序列特征包括：动态API序列的特征向量时，所述将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征，包括：

[0183] 将该待检测软件的灰度图特征和该待检测软件的动态API序列的特征向量进行拼接得到该软件的融合特征。

[0184] 例如，灰度图特征即通过多层CNN提取的特征向量为512维，动态API序列的特征向量即通过TextCNN提取的特征向量为128维，则拼接后的特征向量为512+128＝640。

[0185] 在一示例中，当该软件的动态API序列特征包括：动态API序列的特征向量和N-gram特征向量时，所述将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征，包括：

[0186] 将该待检测软件的灰度图特征和该待检测软件的动态API序列的特征向量以及N-gram特征向量进行拼接得到该软件的融合特征。

[0187] 例如，灰度图特征即通过多层CNN提取的特征向量为512维，动态API序列的特征向量即通过TextCNN提取的特征向量为128维，N-gram特征向量为90000维，则拼接后的特征向量为512+128+90000＝90640。

[0188] 在一示例中，当该软件的动态API序列特征包括：动态API序列的特征向量、N-gram特征向量和分布式词向量时，所述将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征，包括：

[0189] 将该待检测软件的灰度图特征和该待检测软件的动态API序列的特征向量、N-gram特征向量以及分布式词向量进行拼接得到该软件的融合特征。

[0190] 例如，多层CNN提取的特征向量为512维，TextCNN提取的特征向量为128维，N-gram特征向量为90000维，分布式词向量的特征为300，则拼接后的特征向量为90000+300+128+512＝90940。

[0191] 步骤505，将所述待检测软件的融合特征输入所述第一软件识别模型，输出所述待检测软件的第一类型；

[0192] 步骤506，当该待检测软件的第一类型为恶意软件时，将该待检测软件的融合特征输入所述第二软件识别模型，输出该待检测软件的第二类型。

[0193] 其中，所述第二类型包括以下全部或者部分类型：勒索病毒、挖坑程序、DDos木马、蠕虫病毒、感染型病毒、后门程序、木马程序。

[0194] 本发明实施例提供的技术方案，预先训练第一软件识别模型和第二软件识别模型，然后利用该第一软件识别模型对待检测软件的类型进行检测；并当待检测软件为恶意软件时，再利用第二软件识别模型检测恶意软件的具体类型；不仅能够检测软件是否为恶意软件，而且还可以检测出恶意软件的具体类型，大大提高了软件类型识别的准确率，而且提升了检测效率。

[0195] 图6为本发明另一实施例提供的一种软件类型的检测方法的流程示意图，如图6所示，该方法包括：

[0196] 步骤601，确定待检测软件的灰度图特征和动态API序列特征；

[0197] 步骤602，将所述待检测软件的灰度图特征和动态API序列特征进行拼接得到所述待检测软件的融合特征；

[0198] 步骤603，将所述待检测软件的融合特征输入预先训练好的第三软件识别模型，输出所述待检测软件的第三类型；

[0199] 其中，所述第三软件识别模型是对XGBoost模型通过第三软件样本进行训练得到的XGBoost多分类模型，以软件的融合特征作为输入，以软件的第三类型作为输出；所述第三软件样本为多个不同类型的软件各自的已标注第三类型的融合特征。

[0200] 在一示例中，所述第三类型包括正常软件，以及以下全部或者部分类型：勒索病毒、挖坑程序、DDos木马、蠕虫病毒、感染型病毒、后门程序、木马程序。

[0201] 本发明实施例提供的技术方案，直接利用训练好的第三软件识别模型对待检测软件的具体类型进行检测，大大提高了软件类型识别的准确率，而且提升了检测效率。

[0202] 图7为本发明另一实施例提供得一种软件类型的检测方法的流程示意图，如图7所示，该方法包括：

[0203] 步骤701，对XGBoost模型通过第三软件样本进行训练得到第三软件识别模型；

[0204] 其中，所述第三软件识别模型是对XGBoost模型通过第三软件样本进行训练得到的XGBoost多分类模型，以软件的融合特征作为输入，以软件的第三类型作为输出。

[0205] 其中，所述第三软件样本为多个不同类型的软件各自的已标注第三类型的融合特征。

[0206] 其中，所述第三类型包括正常软件，以及以下全部或者部分类型：勒索病毒、挖坑程序、DDos木马、蠕虫病毒、感染型病毒、后门程序、木马程序。

[0207] 其中，该第三软件识别模型以软件的融合特征作为输入，以该软件的第三类型作为输出。软件的融合特征是指由该软件的灰度图特征和动态API序列特征拼接组成的特征。该软件的灰度图特征是指该软件对应的灰度图的特征向量，该软件的动态API序列特征包括以下一种或者多种：该动态API序列的特征向量、该动态API序列的N-gram特征、该动态API序列的分布式词向量。

[0208] 其中，该第三软件识别模型的训练过程包括对第二软件样本的灰度图和动态API序列两个模态的融合学习。另外，由于神经网络学习能力强，能够自动提取特征，但容易造成过拟合现象，并且具有不可解释性，不利于产品的迭代，而XGBoost是提升树模型，它具有不容易过拟合、可解释性、训练速度快等特点。因此，本实施例中，结合两者的优点，先通过CNN网络提取不同模态中不同层次的特征向量，然后再将提取后的特征向量和其他维度的特征向量作为输入给XGBoost进行学习训练。例如例如，如图3所示，在灰度图模态学习中，先通过多层CNN网络提取特征向量，将倒数第二层即第二个全连接层的输入作为提取后的特征向量，即灰度图特征；在动态API序列模态学习中，由于动态API序列本质上是文本，所以先获取分布式词向量，基于分布式词向量使用单层的TextCNN网络提取特征向量，然后将两个模态提取后的特征向量和API序列的N-gram特征以及分布式词向量拼接到一起，然后将拼接后的融合特征输入XGBoost进行学习。例如，多层CNN提取的特征向量为512维，TextCNN提取的特征向量为128维，N-gram特征向量为90000维，分布式词向量的特征为300，则拼接后的特征向量为90000+300+128+512＝90940。之所以在API序列模态学习中，将上述的三种特征进行融合，是由于三种特征的主要信息不同，其中TextCNN中的卷积核的长度可以设置的较大(如10～20)，从而提取出对文本分类影响较大的长段落对应的特征，而N-gram由于N的限制(一般不超过3，否则会引起维度灾难)，表示的是文本分类影响较大的短段落的特征，而分布式词向量表示的是文本的整体语义，相比其它的特征来说它具有一定的抽象和概括能力。当然，也可以只选用API序列三种特征中的一种或者两种特征向量与灰度图特征向量进行拼接后输入到XGBoost。

[0209] 在一示例中，所述对XGBoost模型通过第三软件样本进行训练得到第三软件识别模型，包括：

[0210] 对所述第三软件样本中的每一个软件执行如下操作：

[0211] 确定该软件的灰度图和动态API序列；

[0212] 提取该灰度图的特征向量；

[0213] 提取该动态API序列的特征向量；

[0214] 将该灰度图的特征向量和该动态API序列的特征向量进行拼接得到该软件的融合特征；

[0215] 使用XGBoost模型对该软件的融合特征进行训练，将训练得到的XGBoost多分类模

型作为所述第三软件识别模型。

[0216] 在另一示例中，所述对XGBoost模型通过第三软件样本进行训练得到第三软件识别模型，包括：

[0217] 对所述第三软件样本中的每一个软件执行如下操作：

[0218] 确定该软件的灰度图和动态API序列；

[0219] 提取该灰度图的特征向量；

[0220] 提取该动态API序列的特征向量；

[0221] 确定该动态API序列的N-gram特征；

[0222] 将该灰度图的特征向量和该动态API序列的特征向量和N-gram特征进行拼接得到该软件的融合特征；

[0223] 使用XGBoost模型对该软件的融合特征进行训练，将训练得到的XGBoost多分类模型作为所述第三软件识别模型。

[0224] 在另一示例中，所述对XGBoost模型通过第三软件样本进行训练得到第三软件识别模型，包括：

[0225] 对所述第三软件样本中的每一个软件执行如下操作：

[0226] 确定该软件的灰度图和动态API序列；

[0227] 提取该灰度图的特征向量；

[0228] 提取该动态API序列的特征向量；

[0229] 确定该动态API序列的N-gram特征；

[0230] 确定该动态API序列的分布式词向量；

[0231] 将该灰度图的特征向量和该动态API序列的特征向量、N-gram特征和分布式词向量进行拼接得到该软件的融合特征；

[0232] 使用XGBoost模型对该软件的融合特征进行训练，将训练得到的XGBoost多分类模型作为所述第三软件识别模型。

[0233] 在一示例中，确定软件的灰度图，包括：

[0234] 对该软件进行反汇编得到其二进制代码文件；

[0235] 将该二进制代码文件转换成灰度图。

[0236] 其中，由于软件的每个字节的范围是在十六进制00-FF之间，正好对应灰度图十进制0-255(0为黑色，255为白色)，其恰好覆盖了整个灰度值得范围。因此，将软件的二进制代码文件按照字节进行分割，然后基于十六进制码与图像灰度值的对应关系得到每个字节对应的灰度值，如此就可以将二进制代码文件转换成灰度图。

[0237] 在一示例中，确定软件的动态API序列，包括：

[0238] 将该软件在动态沙箱中模拟运行，得到运行后的动态API序列。

[0239] 其中，沙箱(Sandboxie)是一个虚拟系统程序，允许在沙盘环境中运行浏览器或其他程序，运行所产生的变化可以随后删除。它创造了一个类似沙盒的独立作业环境，在其内部运行的程序并不能对硬盘产生永久性的影响。在网络安全中，沙箱指在隔离环境中，用以测试不受信任的文件或应用程序等行为的工具。动态沙箱可以包括虚拟操作系统层，首先将待检测文件在动态沙箱的虚拟操作系统层中运行，然后在运行过程中模拟对操作系统的应用程序编程接口API(application programming interface)进行调用操作，提取产生的

动态行为信息即为动态API序列，该动态API序列为一种文本信息，例如读文件、写注册表等动态行为信息。

[0240] 其中，在API序列模态学习中，由于API序列本质上是文本，所以使用单层的TextCNN网络提取特征向量。

[0241] 其中，N-Gram是一种基于统计语言模型的算法，其基本思想是将文本里面的内容按照字节进行大小为N的滑动窗口操作，形成了长度是N的字节片段序列。每一个字节片段称为gram，对所有gram的出现频度进行统计，并且按照事先设定好的阈值进行过滤，形成关键gram列表，也就是这个文本的向量特征即N-gram特征，列表中的每一种gram就是一个特征向量维度。

[0242] 在一示例中，确定动态API序列的分布式词向量，包括：利用词向量算法训练该动态API序列的分布式词向量。

[0243] 其中，该词向量算法包括以下一种或者多种：Word2vec、Glove、Fasttext。例如使用Word2vec、Glove、Fasttext三种方法训练词向量，并将该动态API序列对应的三种词向量合并到一起作为的该动态API序列的分布式词向量。

[0244] 步骤702，确定待检测软件的灰度图特征和动态API序列特征；

[0245] 其中，确定该待检测软件的灰度图特征，包括：

[0246] 对该软件进行反汇编得到其二进制代码文件；

[0247] 将该二进制代码文件转换成灰度图；

[0248] 提取所述灰度图的特征向量作为待检测软件的灰度图特征。

[0249] 在一示例中，可以通过卷积神经网络(Convolutional Neural Network，CNN)提取所述灰度图的特征向量。例如，采用图3所示的多层CNN网络提取特征向量，将倒数第二层即第二个全连接层的输入作为提取后的特征向量，即灰度图特征。

[0250] 其中，确定该待检测软件的动态API序列特征，包括：

[0251] 将该软件在动态沙箱中模拟运行，得到运行后的动态API序列；

[0252] 执行以下至少一种操作：

[0253] 提取该动态API序列的特征向量；

[0254] 和/或，确定该动态API序列的N-gram特征；

[0255] 和/或，确定该动态API序列的分布式词向量。

[0256] 其中，所述动态API序列特征包括以下一种或者多种：

[0257] 该动态API序列的特征向量、该动态API序列的N-gram特征向量、该动态API序列的分布式词向量。

[0258] 在一示例中，可以使用单层的TextCNN网络提取该动态API序列的特征向量。

[0259] 步骤703，将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征；

[0260] 在一示例中，当该软件的动态API序列特征包括：动态API序列的特征向量时，所述将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征，包括：

[0261] 将该待检测软件的灰度图特征和该待检测软件的动态API序列的特征向量进行拼接得到该软件的融合特征。

[0262] 例如，灰度图特征即通过多层CNN提取的特征向量为512维，动态API序列的特征向量即通过TextCNN提取的特征向量为128维，则拼接后的特征向量为512+128＝640。

[0263] 在一示例中，当该软件的动态API序列特征包括：动态API序列的特征向量和N-gram特征向量时，所述将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征，包括：

[0264] 将该待检测软件的灰度图特征和该待检测软件的动态API序列的特征向量以及N-gram特征向量进行拼接得到该软件的融合特征。

[0265] 例如，灰度图特征即通过多层CNN提取的特征向量为512维，动态API序列的特征向量即通过TextCNN提取的特征向量为128维，N-gram特征向量为90000维，则拼接后的特征向量为512+128+90000＝90640。

[0266] 在一示例中，当该软件的动态API序列特征包括：动态API序列的特征向量、N-gram特征向量和分布式词向量时，所述将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征，包括：

[0267] 将该待检测软件的灰度图特征和该待检测软件的动态API序列的特征向量、N-gram特征向量以及分布式词向量进行拼接得到该软件的融合特征。

[0268] 例如，多层CNN提取的特征向量为512维，TextCNN提取的特征向量为128维，N-gram特征向量为90000维，分布式词向量的特征为300，则拼接后的特征向量为90000+300+128+512＝90940。

[0269] 步骤704，将所述待检测软件的融合特征输入预先训练好的第三软件识别模型，输出所述待检测软件的第三类型。

[0270] 本发明实施例提供的技术方案，直接利用训练好的第三软件识别模型对待检测软件的具体类型进行检测，大大提高了软件类型识别的准确率，而且提升了检测效率。

[0271] 在本发明的另一实施例中，在上述实施例的基础上，为了提高神经网络学习的效果，抑制过拟合现象，采用标签平滑和mixup技术。标签平滑是把标签进行转换，其中转换前的标签为one-hot表示：

[0272] $$y=\begin{cases}1,\text{真实类别}\\0,\text{其它类别}\end{cases}$$

[0273] 转换后的类别如下所示，其中∈为较小数，如0.001。

[0274] $$y=\begin{cases}1-\epsilon,\text{真实类别}\\\epsilon/(n-1),\text{其它类别}\end{cases}$$

[0275] 而mixup是通过已有的训练数据生成新的训练数据。随机选两个数据i和j，生成随机数λ∈[0,1]，则新的训练数据则为：

[0276] $x=\lambda x_i+(1-\lambda)x_j\ \ y=\lambda_{yi}+(1-\lambda)_{yi}$

[0277] 如此，采用标签平滑和mixup技术预先对样本数据进行处理，能够抑制过拟合现象，提高了神经网络学习的效果。

[0278] 在本发明的另一实施例中，在上述实施例的基础上，在对待检测软件进行检测的过程中，包括在线预测和离线预测两个部分。在线预测指的是使用服务的方式对线上的每个样本进行实时的预测。而离线预测中包括两种模式，一种是基于批量测试数据重新训练

模型，另外一种是基于现有模型直接预测。之所以提出基于批量测试数据重新训练模型，是因为当训练集和测试集分布差异较大时，就会导致训练出来的模型在测试集上效果不佳。所以需要先通过模型来判断训练集和测试集是否具有较大的差异。例如，将训练集和测试集数据放到一起；然后定义二分类任务，判断样本属于训练集还是测试集，并使用上面提到的学习过程，只是把XGBoost的学习目标修改为二分类，即输出为预测样本属于测试集的概率；观察上述二分类任务的学习效果，如果模型对应的AUC(AreaUnderCurve,ROC曲线下的面积)超过一定阈值(如0.8)，则表明训练集和测试集的差异较大。如果训练集和测试集的差异较大，将训练集样本按照上述任务中属于测试集的概率进行排序，例如，将前20％的样本作为重新训练的验证集，剩余80％作为训练集样本。由于验证集的选取对参数的选取至关重要，所以按照上述的方法重新生成训练集和验证集进行学习，就能在一定程度上提高测试集的效果。

[0279] 本发明的一实施例还提供了一种软件类型的检测装置，如图8所示，该装置包括：

[0280] 确定单元，设置为确定待检测软件的灰度图特征和动态API序列特征；

[0281] 拼接单元，设置为将所述待检测软件的灰度图特征和动态API序列特征进行拼接得到所述待检测软件的融合特征；

[0282] 第一检测单元，设置为将所述待检测软件的融合特征输入预先训练好的第一软件识别模型，输出所述待检测软件的第一类型；

[0283] 其中，所述第一软件识别模型是对XGBoost模型通过第一软件样本进行训练得到的XGBoost二分类模型，以软件的融合特征作为输入，以软件的第一类型作为输出；所述第一软件样本为多个软件各自的已标注第一类型的融合特征。

[0284] 在一示例中，所述第一类型为正常软件或者恶意软件。

[0285] 在一示例中，该装置还包括：

[0286] 第二检测单元，设置为当所述待检测软件的第一类型为恶意软件时，将所述待检测软件的融合特征输入预先训练好的第二软件识别模型，输出所述待检测软件的第二类型；

[0287] 其中，所述第二软件识别模型是对XGBoost模型通过第二软件样本进行训练得到的XGBoost多分类模型，以软件的融合特征作为输入，以软件的第二类型作为输出；所述第二软件样本为多个不同类型的恶意软件各自的已标注第二类型的融合特征。

[0288] 在一示例中，所述第二类型包括以下全部或者部分类型：勒索病毒、挖坑程序、DDos木马、蠕虫病毒、感染型病毒、后门程序、木马程序。

[0289] 在一示例中，所述确定单元，设置为对所述待检测软件进行反汇编得到其二进制代码文件；

[0290] 将所述二进制代码文件转换成灰度图；

[0291] 提取所述灰度图的特征向量作为待检测软件的灰度图特征。

[0292] 在一示例中，所述确定单元，设置为将所述待检测软件在动态沙箱中模拟运行，得到运行后的动态API序列；

[0293] 执行以下至少一种操作：

[0294] 提取该动态API序列的特征向量；确定该动态API序列的N-gram特征向量；确定该动态API序列的分布式词向量；

[0295] 所述动态API序列特征包括以下一种或者多种：

[0296] 该动态API序列的特征向量、该动态API序列的N-gram特征向量、该动态API序列的分布式词向量。

[0297] 在一示例中，该装置还包括显示单元；

[0298] 所述显示单元，设置为显示所述待检测软件的第一类型或者第二类型。

[0299] 本发明实施例提供的技术方案，通过训练和识别软件的融合特征来进行检测，能够大大提高软件类型识别的准确率。

[0300] 本发明的另一实施例还提供了另一种软件类型的检测装置，如图9所示，该装置包括：

[0301] 确定单元，设置为确定待检测软件的灰度图特征和动态API序列特征；

[0302] 拼接单元，设置为将所述待检测软件的灰度图特征和动态API序列特征进行拼接得到所述待检测软件的融合特征；

[0303] 第三检测单元，设置为将所述待检测软件的融合特征输入预先训练好的第三软件识别模型，输出所述待检测软件的第三类型；

[0304] 其中，所述第三软件识别模型是对XGBoost模型通过第三软件样本进行训练得到的XGBoost多分类模型，以软件的融合特征作为输入，以软件的第三类型作为输出；所述第三软件样本为多个不同类型的软件各自的已标注第三类型的融合特征

[0305] 在一示例中，所述第三类型包括正常软件，以及以下全部或者部分类型：勒索病毒、挖坑程序、DDos木马、蠕虫病毒、感染型病毒、后门程序、木马程序。

[0306] 在一示例中，该装置还包括显示单元；

[0307] 所述显示单元，设置为显示所述待检测软件的第三类型。

[0308] 本发明实施例提供的技术方案，通过训练和识别软件的融合特征来进行检测，能够大大提高软件类型识别的准确率。

[0309] 本发明实施例还提供了一种电子装置，包括：存储器、处理器及存储在所述存储器上并可在所述处理器上运行的计算机程序，所述计算机程序被所述处理器执行时实现上述任一项所述软件类型的检测方法。

[0310] 本发明实施例还提供了一种计算机可读存储介质，所述计算机可读存储介质上存储有信息处理程序，所述信息处理程序被处理器执行时实现上述任一项中所述软件类型的检测方法。

[0311] 本领域普通技术人员可以理解，上文中所公开方法中的全部或某些步骤、系统、装置中的功能模块/单元可以被实施为软件、固件、硬件及其适当的组合。在硬件实施方式中，在以上描述中提及的功能模块/单元之间的划分不一定对应于物理组件的划分；例如，一个物理组件可以具有多个功能，或者一个功能或步骤可以由若干物理组件合作执行。某些组件或所有组件可以被实施为由处理器，如数字信号处理器或微处理器执行的软件，或者被实施为硬件，或者被实施为集成电路，如专用集成电路。这样的软件可以分布在计算机可读介质上，计算机可读介质可以包括计算机存储介质(或非暂时性介质)和通信介质(或暂时性介质)。如本领域普通技术人员公知的，术语计算机存储介质包括在用于存储信息(诸如计算机可读指令、数据结构、程序模块或其他数据)的任何方法或技术中实施的易失性和非易失性、可移除和不可移除介质。计算机存储介质包括但不限于RAM、ROM、EEPROM、闪存或其

他存储器技术、CD-ROM、数字多功能盘(DVD)或其他光盘存储、磁盒、磁带、磁盘存储或其他磁存储装置、或者可以用于存储期望的信息并且可以被计算机访问的任何其他的介质。此外，本领域普通技术人员公知的是，通信介质通常包含计算机可读指令、数据结构、程序模块或者诸如载波或其他传输机制之类的调制数据信号中的其他数据，并且可包括任何信息递送介质。

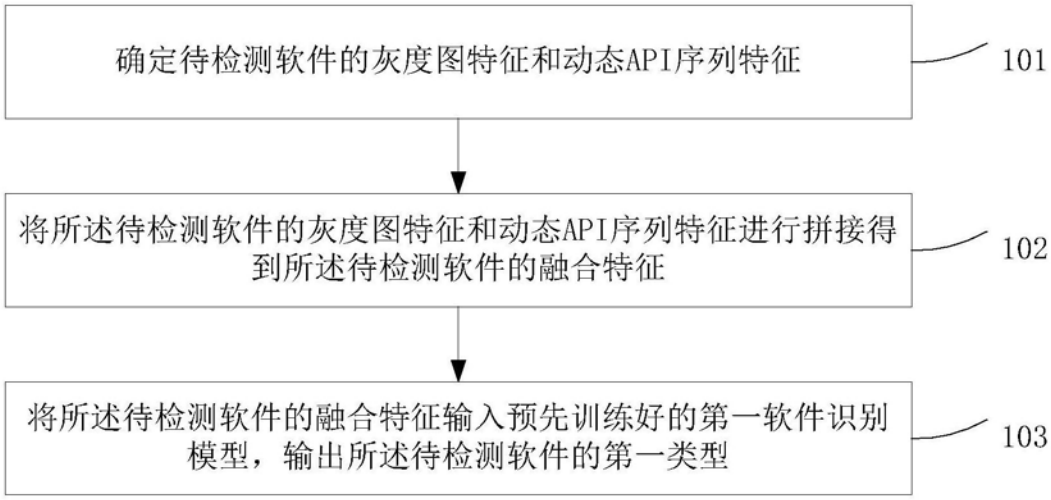

确定待检测软件的灰度图特征和动态API序列特征
101
将所述待检测软件的灰度图特征和动态API序列特征进行拼接得到所述待检测软件的融合特征
102
将所述待检测软件的融合特征输入预先训练好的第一软件识别模型，输出所述待检测软件的第一类型
103

图1

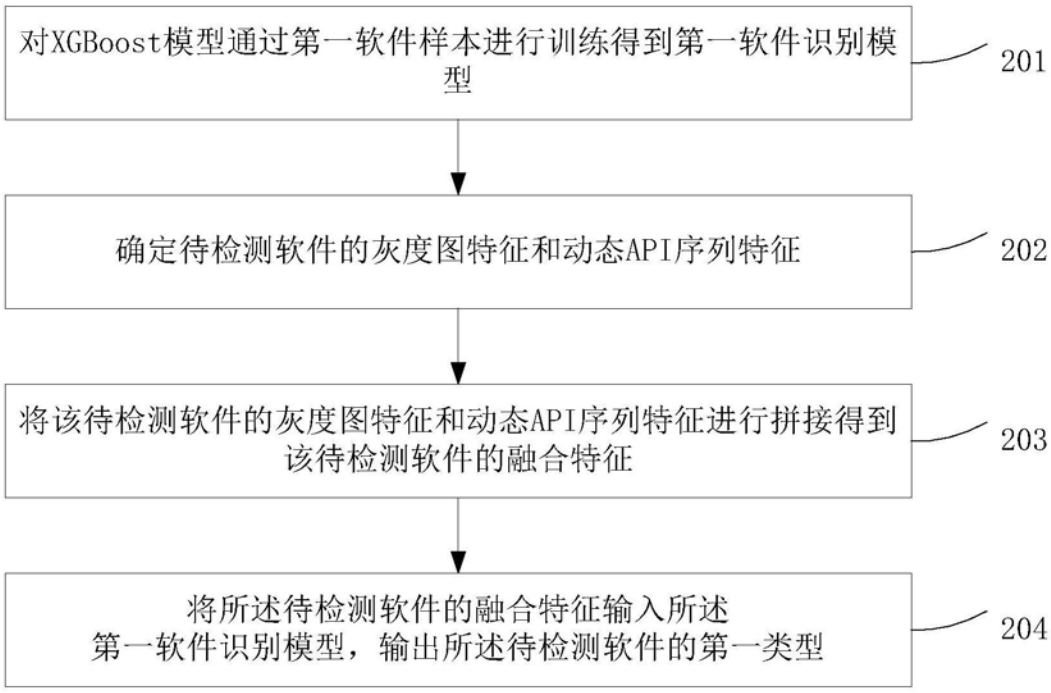

对XGBoost模型通过第一软件样本进行训练得到第一软件识别模型
201
确定待检测软件的灰度图特征和动态API序列特征
202
将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征
203
将所述待检测软件的融合特征输入所述第一软件识别模型，输出所述待检测软件的第一类型
204

图2

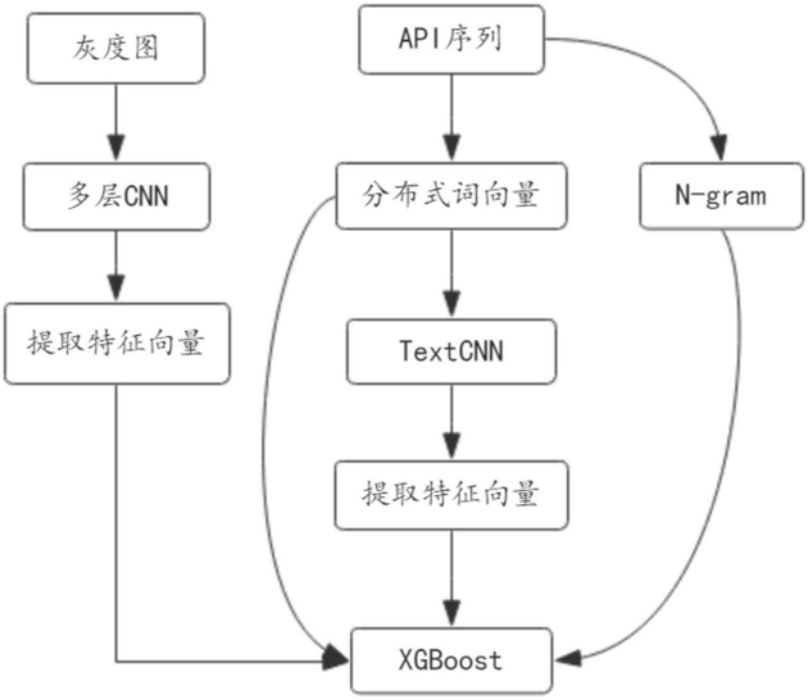

灰度图
API序列
多层CNN
分布式词向量
N-gram
提取特征向量
TextCNN
提取特征向量
XGBoost

图3

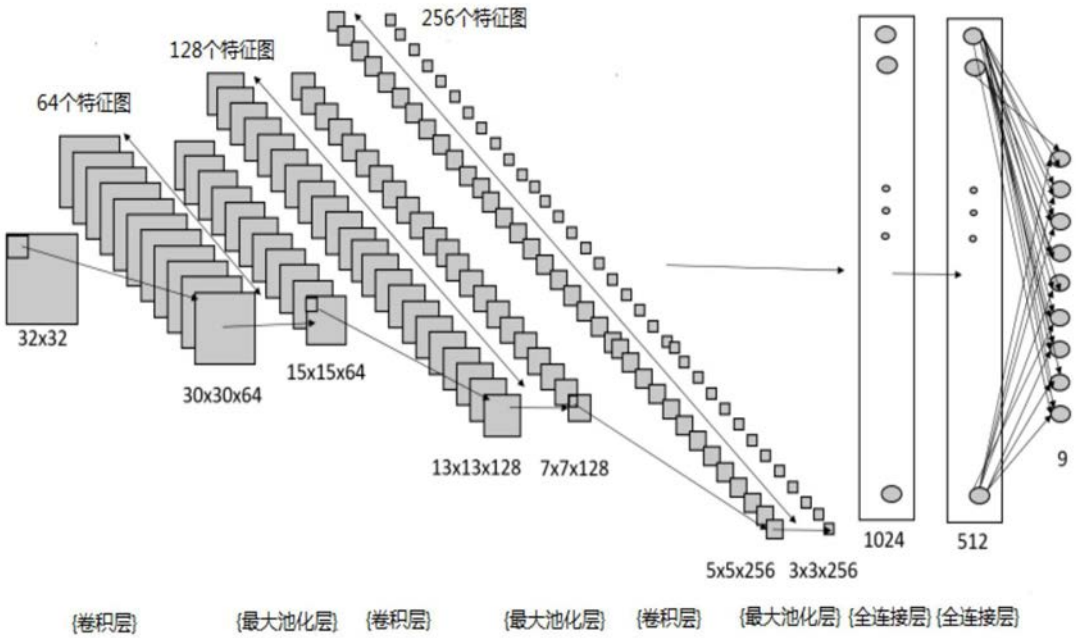

256个特征图
128个特征图
64个特征图
32x32
30x30x64
15x15x64
13x13x128
7x7x128
5x5x256
3x3x256
1024
512
9
{卷积层}
{最大池化层}
{卷积层}
{最大池化层}
{卷积层}
{最大池化层}
{全连接层}
{全连接层}

图4

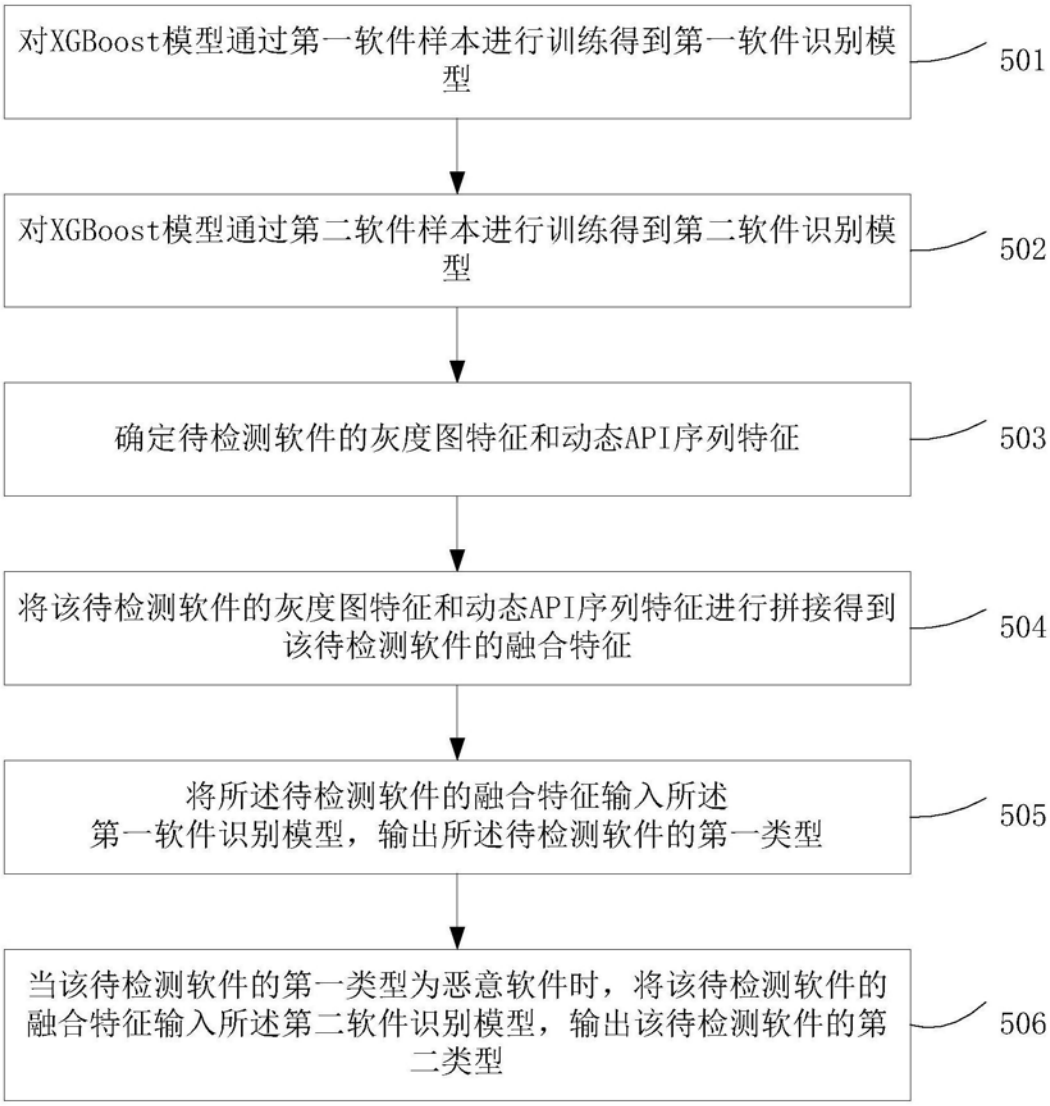

对XGBoost模型通过第一软件样本进行训练得到第一软件识别模型
501
对XGBoost模型通过第二软件样本进行训练得到第二软件识别模型
502
确定待检测软件的灰度图特征和动态API序列特征
503
将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征
504
将所述待检测软件的融合特征输入所述第一软件识别模型，输出所述待检测软件的第一类型
505
当该待检测软件的第一类型为恶意软件时，将该待检测软件的融合特征输入所述第二软件识别模型，输出该待检测软件的第二类型
506

图5

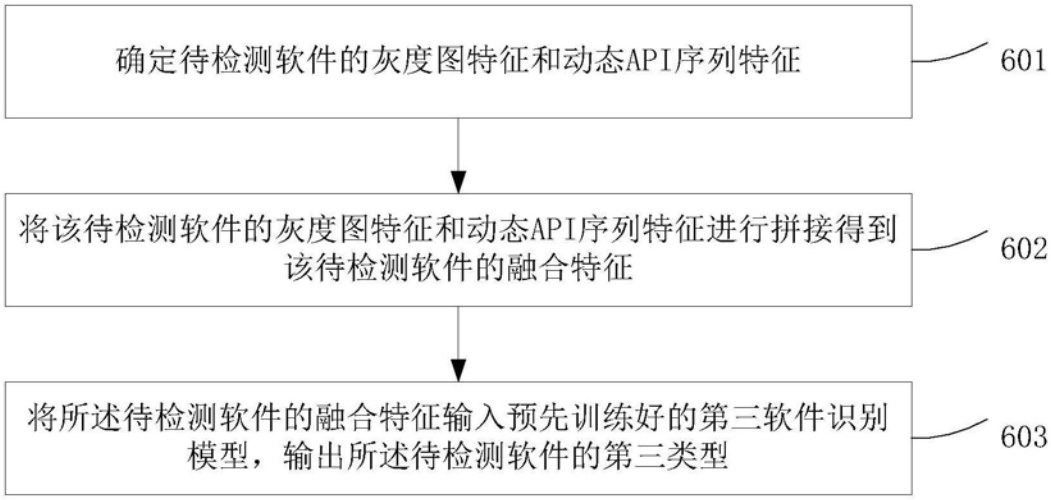

确定待检测软件的灰度图特征和动态API序列特征
601
将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征
602
将所述待检测软件的融合特征输入预先训练好的第三软件识别模型，输出所述待检测软件的第三类型
603

图6

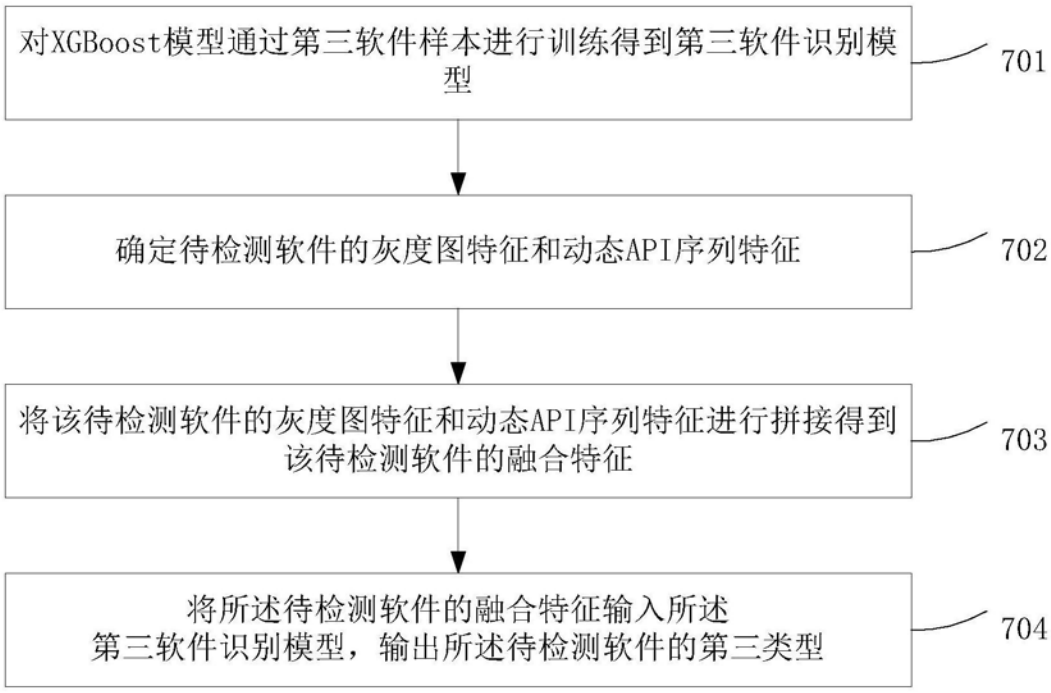

对XGBoost模型通过第三软件样本进行训练得到第三软件识别模型
701
确定待检测软件的灰度图特征和动态API序列特征
702
将该待检测软件的灰度图特征和动态API序列特征进行拼接得到该待检测软件的融合特征
703
将所述待检测软件的融合特征输入所述第三软件识别模型，输出所述待检测软件的第三类型
704

图7

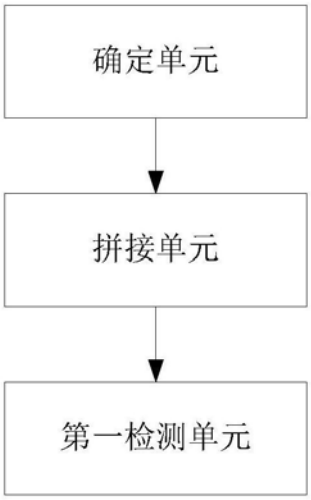


图8

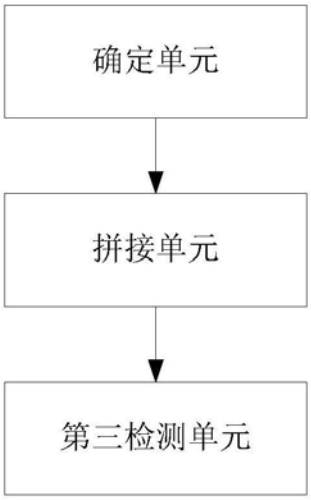


图9