## Malicious code detection based on CNNs and multi-objective algorithm

在这一部分，介绍和讨论了恶意软件检测的相关研究和现有的检测方法。基于特征分析的检测方法分为两类，静态方法和动态方法。静态方法是一种常用的现有方法。Brian等人[5]开发了一种通过静态分析发现源代码中安全缺陷的方法。他们还证明了该方法可用于识别现有程序中的真实漏洞。然而，这种方法的一个问题是静态分析的有效性受到混淆技术的很大影响[27]。

为了处理代码混淆，Sharif等人[35]提出了一种用于检测可执行文件中恶意模式的框架，该框架能够抵抗常见的混淆转换。研究证明了该架构的有效性；然而，该方法仅限于指令级的特征提取和分析，并且存在许多技术容易欺骗静态方法。

动态分析根据对访问私有数据和使用受限制的API调用等行为的评估，监视和分析应用程序的运行时特征。给定这些信息，就建 立了一个行为模型来检测恶意代码。这种技术已经显示出改进的检测性能，但是仍然受到为产生不可靠结果而开发的各种对策的挑战[30]。此外，动态分析非常耗时，因为需要大量的计算，导致在面对大型数据集时效率低下。

如前所述，恶意代码可视化的方法已经被成功使用[34，39]。Yoo等人[31]利用自组织图来可视化计算机病毒。Trinius等人[10]提出了一种类似但扩展的方法，他们使用了两种可视化技术，树图和线程图来检测和分类恶意软件。Goodall等人[17]没有获得单个检测结果，而是将不同恶意软件分析工具的结果汇总到一个可视化环境中，从而增加了单个恶意软件工具的检测覆盖漏洞。

上面讨论的研究主要集中在恶意软件行为的可视化上[38，41，46]，然而，软件源代码可能会提供更有意义的模式。如前所述，Nataraj等人[29]提出了一种基于二值纹理分析的恶意软件检测可视化新方法。首先，他们将恶意软件可执行文件转换成灰度图像，然后根据这些图像的纹理特征识别恶意软件。与动力分析方法相比，这种方法产生了等价的结果[29]。在类似的工作中，Han等人[15]将恶意软件二进制信息转换成彩色图像矩阵，并通过使用图像处理方法对恶意软件家族进行分类。但是他们使用的方法有一个缺点就是资源消耗高。一旦恶意软件被可视化为灰度图像，恶意软件检测就可以转化为图像识别问题。在[30]中，Nataraj等人使用了广义搜索树(GiST)算法来提取恶意软件图像的特征。然而，GiST算法的操作是耗时的，并且最近已经提出了更强大的图像处理技术。

对于图像融合，苗等[26]提出了一种基于剪切波和遗传算法的图像融合算法。在他们的方法中，使用遗传算法来优化融合规则中的加权因子。实验结果表明，该方法能获得比其他方法更好的融合质量。

Cui等人[10]开发了一种将CNN与Bat算法相结合的方法，以提高模型的准确性。利用Bat算法对图像数据集进行平衡，利用卷积神经网络对图像进行识别和分类。但是他们使用一个评估标准来测试模型。正如我们在第1节中所述，单一评估标准不能代表所构建模型的性能。

深度学习[33，47]是近年来从人工神经网络的工作中出现的机器学习研究领域。神经网络可以通过学习深层非线性网络结构来利用近似复杂函数来解决复杂问题。近年来，多目标问题的算法得到了很好的发展。提出了基于深度学习和多目标算法的恶意软件检测方法。神经网络可以方便地训练识别出的模型。NSGA-ⅱ可以处理数据集的不平衡问题。这样可以有效避免以上总结的问题。

## Neural malware analysis with attention mechanism

在本节中，作为相关研究，我们通过将二进制数据转换为图像来调查现有的恶意软件分类方法。我们还回顾了使用深度学习的现有恶意软件分类方法，以及有时用于恶意软件分析的二进制比较方法。

2.1.通过将二进制数据转换为图像进行恶意软件分类

Nataraj等人(201 1a，b)提出了通过将二进制数据转换成图像的恶意软件分类。在将标记的恶意软件样本转换为图像后，使用k-最近邻方法对未标记的恶意软件样本进行分类。更具体地说，给定一个未标记的恶意软件样本，他们的方法将样本转换成图像，并获得如下特征向量。

1.读取目标恶意软件样本的二进制数据的每个字节，并将其存储在数字数组中(范围为0–255)。

2.根据恶意软件样本文件大小确定图像宽度，并将阵列转换为灰度图像。

3.将获得的图像拉伸到64 ×64的分辨率。

4.从拉伸的图像中提取320维的胃肠道间质瘤特征(Oliva和Torralba，2001)，用作特征向量

然后，使用带有特征向量和标签的k-最近邻方法预测恶意软件家族。作为使用各种恶意软件数据集的实验评估的结果，Nataraj等人(201 1b)得出结论，该方法实现了比现有方法更高的分类精度，即使样本包括由称为打包器的obfus-阳离子程序产生的混淆的二进制数据。

Kirat等人(2013)提出了另一种基于图像的恶意软件检测方法，并证明他们的方法比使用n -gram二进制数据的方法和基于控制流图的方法具有更高的检测精度。Ahmadi等人(2016)提出了一种使用梯度增强算法的分类方法，该算法利用了包括基于图像的特征在内的多个特征，并基于实验得出结论，基于图像的特征对于提高包括打包样本在内的数据集的准确性是有效的

一般来说，被打包器混淆的二进制数据有望成为一个近乎随机的二进制序列。因此，用打包样本获得的二进制文件或图像直接训练分类器可能看起来没有意义。然而，当同一家族中的恶意软件样本被打包者使用基于字典的压缩方法(例如UPX 2)压缩时，结果文件通常包括公共序列(Jacob等人，2012)。这是因为基于字典的打包器使用类似的字典混淆文件，产生有助于识别混淆恶意软件样本的恶意软件系列的特征。

然而，使用从图像中提取的手工特征的方法(Ahmadi等人，2016年；Kirat等人，2013年；Nataraj等人，201 1b)简单地检测或分类恶意软件样本，并且对于使手动分析高效无效。这种方法不能明确二进制数据中的哪个区域影响了结果，因为在特征提取中丢失了位置信息。因此，他们不能通过指出重要的部分来帮助人类分析师调查样本的功能性。

最近，提出了两种将深度学习应用于成像二进制数据的方法(Kalash等人，2018年；Le等人，2018年)。Kalash等人(2018年)提出了一种恶意软件分类方法，即使用与Nataraj等人(201 1b)相同的转换方法，将CNN应用于从二进制数据获得的图像，并报告称，该方法显示出比Ahmadi等人(2016年)提出的方法更高的准确性。Le等人(2018)还提出了一种结合CNN和长短期记忆(LSTM)的恶意软件分类方法，并证明了该组合方法比基于CNN的方法具有更高的准确性。然而，这些方法仍然不具备识别与恶意软件功能相关的重要区域的手段。

2.2.使用深度学习的恶意软件分类

除了第2.1节中提到的方法，已经提出了许多方法来将深度学习应用于恶意软件检测和恶意软件分类(Dahl等人，2013；大卫和内塔尼亚胡，2015年；Hardy等人，2016年；黄和斯托克斯，2016；Kolosnjaji等人，2016年；Pascanu等人，2015年；Raff等人，2018年；萨克斯和柏林，2015；Tobiyama等人，2016年)。例如，Saxe和Berlin (2015)提出了一种利用深度神经网络(DNN)的恶意软件检测方法，该方法基于从报头中提取的字节序列和元数据的直方图。Pascanu等人(2015年)使用API调用历史和基于从RNN获得的特征向量的神经网络的预先形成的检测来训练递归神经网络(RNN)。黄和斯托克斯(2016)使用由API调用n -gram组成的词包特征作为多任务的输入，该输入同时预测给定样本是否是恶意软件以及该样本属于哪个家族。Raff等人(2018)将一维CNN应用于二进制数据的原始字节，用于恶意软件分类。

还提出了其他针对检测恶意安卓应用的方法(McLaughlin等人，2017年；苏等，2016；袁等，2014；2016 ).Su等人(2016)提出了一种基于深度信念网络的应用元数据检测方法。袁等(2014，2016)也将特定API调用的出现作为的输入。麦克劳克林等人(2017)提出了一种检测方法，该方法将一维CNN应用于Java虚拟机的指令序列。

这些方法分为两类:静态分析和动态分析。这里，静态分析是指仅使用从恶意软件样本获得的信息而不执行样本的方法，如Saxe和Berlin (2015)、Hardy等人(2016)、Raff等人(2018)、Su等人(2016)和McLaughlin等人(2017)所检查的。动态分析是指利用Dahl等人(2013年)、Pascanu等人(2015年)、David和Ne- tanyahu (2015年)、黄和斯托克斯(2016年)、Kolosnjaji等人(2016年)、Tobiyama等人(2016年)、袁等人(2014年)和袁等人(2016年)调查的通过执行恶意软件获得的信息的方法。这些方法使用从虚拟环境或反恶意软件引擎中提取的应用编程接口调用信息。

与静态分析相比，动态分析利用了与目标相关的更详细的信息，尽管正如Chen等人(2008)和Oyama (2017)指出的那样，许多恶意软件样本通过检测虚拟机监视器和用于记录API调用信息的其他系统的存在来改变其行为，以干扰分析。因此，存在恶意软件样本由于其反分析机制而无法检测或无法直接分类的可能性。

此外，尽管使用静态分析的方法(Hardy等人，2016年；McLaughlin等人，2017年；Raff等人，2018年；萨克斯和柏林，2015；Su等人，2016)已经尝试确定样本是否是恶意软件，这些方法不提供关于恶意软件家族的信息。因此，这些方法不能通过寻找家庭特有的信息来有效地进行人工分析

此外，Raff等人(2018)使用类别激活图(Zhou等人，2016)来检查二进制数据中的哪个部分(例如，代码部分或数据部分)影响区分。然而，在这种方法中，能够被检测到的位置的数量受到网络结构的限制，尽管事实上合法行为在典型的恶意软件样本中的许多功能上被实现。因此，尽管这有助于讨论所提出的方法是如何工作的，但其使手动分析高效的能力仍未得到证实.

2.3.用于恶意软件比较的相似序列检测方法

当目标恶意软件样本的家族已知，并且同一家族的另一个恶意软件样本的人工注释分析结果可用时，人工分析师有时会应用二进制代码比较工具(Alrabaee等人，2015；杜林和罗尔斯，2005年；法尔哈迪等人，2014年；高等，2008)。分析者通过参考被分析样本的相似部分的注释来寻找揭示目标样本的功能的线索。例如，法尔哈迪等人(2014)提出了一种从分解的指令中生成和比较特征向量的检测方法。其他(Alrabaee等人，2015年；杜林和罗尔斯，2005年；Gao等人，2008)根据分解的指令计算控制流图，并将这些图进行比较以找到相应的节点。

但是，所有这些方法都只引用恶意软件示例的代码部分中的信息。因此，他们不能使用数据部分的信息。因此，如果样本被打包，并且与其行为相关的重要信息被放在数据部分，那么这些方法就不能利用这些信息。

为了比较包括数据段在内的恶意软件样本，我们可以使用一种方法来比较所有二进制数据，如珀西瓦尔(2006)所提出的。然而，该方法只是输出所有的差异，包括编译器差异产生的小的修改。因此，它们不适合用于恶意软件比较。

## Multiclass malware classification via first- and second-order texture statistics

使用静态或动态特征的基于可视化的检测包括基于操作码级别的图像(Manavi和Hamzeh，2017；Ni等人，2018年)、基于系统调用的行为图(Park等人，2010年)、基于API调用的恶意行为可视化(Shaid和Maarof，2014年；Kim等人，2019)和拆解的基于代码的广度优先搜索(Kim和Youn，2017)。树图和线程图(Trinius等人，2009)分别可视化了分段式API调用和操作码序列，以跟踪恶意操作。

转换成图像的恶意软件二进制文件已经基于基于小波分解的GIST特征进行了检测(Nataraj等人，201 1 a)，在Mal- img数据集上产生了97.18%的准确性。然而，该过程使用了320个高维特征，54 ms用于提取，1.4 s用于分类；此外，这些小于二元模型分析的参数，二元模型分析使用5 s进行提取，使用56 s进行分类。该方法还采用了分析前拆包。基于熵图的分类(Han等，2015)也未能对包装样本进行分类。已经使用了另一种基于GIST描述符的检测(Makandar和Patrot，2015年；Nataraj和Manjunath，2016年；周等，2017)；Nataraj和Manjunath (2016)在整个Malimg获得了97.4%的准确率。局部二值模式(LBPs)(Roo和Lo，2017)的性能优于基于GIST特征的分类，强度和小波特征(Kancherla和Mukkamala，2013)的性能也略好于计算量大的基于Gabor的特征。基于稀疏表示的恶意软件检测(Nataraj等人，2015)已经使用随机投影提出，并且它获得了92.83%的准确性，但是依赖于增加维度到512个特征，这花费了24秒，同时使用正交匹配追踪(OMP)来识别样本。纹理和全局颜色特征与局部特征相结合已被用于增强检测(傅等人，2018年)。然而，该技术需要在分析之前解析恶意软件的结构和解密或解压缩。基于深度学习的卷积神经网络(CNNs)(崔等，2018；Gibert等人，2019)应用于Malimg分别获得了94.5%和98.48%的准确率，运行时间分别为0.02秒和0.001秒。然而，Gibert等人(2019年)使用了较大的图像尺寸，增加了计算时间。CNN的另一种方法(Mourtaji等人，2019年)获得了97.02%的准确率。使用CNN和双向门控递归单元(BiGRU)的混合深度学习成本敏感模型(Venkatraman等人，2019)在应用于Malimg时获得了91.6%的F1测量值。另一个美国有线电视新闻网和长短期记忆(LSTM)网络模型(Akarsh等人，2019年)的准确率为95.5%。选择方法(Naeem等人，2019)已被用于降低恶意模式的维度，获得98%的准确率。然而，这类模型速度较慢。使用硬分配进行聚类的局部特征选择速度较慢，使用512个特征和4.27秒进行分类。一个生成性对抗网络(GAN)模型相对于85%的准确性，使用一个跨越Malimg的变分自动编码器(VAE)模型达到了90%的准确性(Burks等人，2019年)。Hashemi和Hamzeh (2019)使用二值纹理分析，从二值变换图像中提取微观模式。二元纹理分析与动态分析的比较如下(Nataraj等人，201 1 b)。前者的准确率更高，分类速度快40 0 0倍。瓦格纳等人(2015年)发表了一份综述基于可视化的恶意软件分类技术的调查。

与现有的可视化技术不同，该技术使用统计方法进行二值纹理分析。据我们所知，它为Windows恶意软件分类提取了新的特征:基于一阶和灰度共生矩阵的二阶纹理统计。所提出技术的关键成就是在大约37毫秒的较短时间内提取的35个特征的维数相对较低，导致0.01秒的平均分类时间和更好的分类效率。

恶意软件检测和分类机器学习的兴起:研究进展、趋势和挑战

4.传统的机器学习方法

在过去十年中，机器学习解决方案的研究和部署有所增加，以解决恶意软件检测和分类的任务。如果没有三个最新发展的融合，机器学习方法的成功和巩固是不可能的:

1。第一个发展是恶意软件的标记源的增加，这意味着标记恶意软件第一次不仅可用于安全社区，也可用于研究社区。这些反馈的规模从有限的高质量样本，如微软(Ronen et al .，2018)提供的样本，到数据创新者收集反恶意软件预测挑战，再到病毒共享(2011)的样本。

2.第二个发展是计算能力迅速提高，同时变得更便宜，更接近大多数研究人员的预算。因此，它允许研究人员加快迭代训练过程，并使更大和更复杂的模型适应不断增加的数据。

3.第三，在过去的几十年中，机器学习领域以更快的速度发展，在广泛的任务上，如计算机视觉、语音识别和自然语言处理，在准确性和可扩展性方面取得了突破性的成功。

在机器学习中，工作流是一个迭代过程，包括收集可用数据、清理和准备数据、构建模型、验证和部署到生产中。见图2。在处理原始恶意软件时，传统机器学习方法的数据准备过程包括对可执行文件进行预处理，以提取一组提供软件抽象视图的特征。然后这些特征被用来训练一个模型来解决手头的任务。由于恶意软件功能的多样性，不仅要检测恶意软件，而且要区分不同类型的恶意软件，以便更好地了解它们的功能，这一点很重要。用于恶意软件检测或分类的机器学习解决方案之间的主要区别在于所实现的系统返回的输出。一方面，恶意软件检测系统输出单个值y = f(x)，范围从0到1，这表明可执行文件的恶意性。另一方面，分类系统输出给定可执行文件属于每个输出类或系列的概率，y ∈ ℝN，w h e r e N表示不同系列的数量。

图3提供了特征的分类。因此，特征的类型可以被分成两组，像恶意软件分析方法的类型一样:(1)静态特征和(2)动态特征。每个功能类型分别如下。

4.1.静态特征

静态特征是从一段程序中提取出来的，不涉及它的执行。在Windows可移植可执行文件中，静态特征基本上来源于两个信息源，即可执行文件的二进制内容或对二进制可执行文件进行反编译和反汇编后得到的汇编语言源文件。另一方面，在安卓应用程序中，这些特征是通过分解APK提取的。要提取某些给定软件的汇编语言源代码，您可以使用自己选择的反汇编工具。对于Windows，您可以使用IDA Pro或Radare2。表2和表3总结了所审查的静态方法。下面你会发现图3中每个静态特征类型的描述。

4.1.1.字符串分析

字符串分析是指提取可执行文件或程序中的每个可打印字符串。字符串是指一系列字符。搜索字符串是获取程序功能线索的最简单方法。可以在这些字符串中找到的信息可以是，例如，程序连接到的URL、程序访问/修改的文件的文件位置或文件路径、应用程序菜单的名称等。名为“字符串”的实用程序可用于在可执行文件中搜索ASCII和Unicode字符串，忽略上下文和格式。虽然有研究使用字符串分析来检测恶意软件(Konopisky，2018；Lee等人，2011)，字符串分析通常与其他静态或动态技术一起使用，以减少其缺陷。(叶等，2008年a)。开发了一个恶意软件检测系统，该系统基于从API执行调用和反映攻击者意图和目标的语义字符串中提取的可解释字符串。该系统由一个为每个PE文件提取可解释字符串的解析器和一个用bagging构造检测器的SVM系综组成。在金山反病毒实验室收集的数据集上评估了系统的性能。

4.1.2.字节和操作码n-grams

恶意软件检测和类别识别最常见的功能类型是N-Grams。n-gram是给定文本序列中n个项目的连续序列。n-gram可以从代表恶意软件二进制内容的字节序列和汇编语言源代码中提取。通过将文件视为字节序列，通过将每n个连续字节的唯一组合视为一个单独的特征来提取字节n-grams。另一方面，汇编语言指令序列也可以从汇编语言源代码中提取。在这种情况下，只有指令的助记符，即“ADD”、“MUL”、“PUSH”等。，是保留的。因此，操作码或助记符n-grams是指每n个连续操作码的唯一组合，作为一个单独的特征。

Moskovitch等人(2008)提出了一种基于文本分类技术的恶意软件分类方法。首先，他们从训练数据中提取了所有的n-grams，n的范围从3到6。第二，他们根据文档频率(DF)分数选择了前5500个特征，后来将Fisher Score特征选择技术应用于此。然后，使用结果特征作为输入，他们训练各种算法，例如人工神经网络(ANN)、支持向量机(SVM)、朴素贝叶斯(NB)和决策树(DT).

Jain和Meena (2011)提出了一种从已知恶意样本中提取字节n-gram特征的方法，n的范围从1到8，以帮助未知可执行文件的分类。由于唯一n-grams的数量非常大，他们使用了一种称为分类文档频率的技术来减少特征空间。最后，使用不同的分类器如朴素贝叶斯、基于实例的学习器、决策树、Adaboost和随机森林来准备不同的N-gram模型.

福永等(2017)提出了一种计算训练样本中每个字节n-gram的信息增益，并选择信息增益最大的K个n-gram作为特征的方法。然后，他们分别计算恶意软件和良性样本的特征向量的每个属性的平均值。最后，根据未知样本的特征向量和两个类别的平均向量之间的相似性，将新的软件分配给两个类别中的一个。

Santos等人(2013)提出了一种基于操作码序列出现频率及其相关性的恶意软件检测技术。每个程序被表示为一个特征向量，其中每个特征对应一个不同的1g或2g。为了减少2g特征的数量，他们应用信息增益来选择前1000个特征。他们的方法在17000个恶意程序和1000个良性程序上进行了验证，结果表明，以皮尔逊七世为核的支持向量机分类器取得了较高的准确率。

Shabtai等人(2012)提出了一种基于操作码n-gram特征的恶意软件检测框架，n的范围从1到6。他们进行了一系列实验，以:(1)确定最佳术语表示，无论是术语频率(TF)还是术语频率逆文档频率，(2)确定n-gram大小，(3)找到最佳的K个顶级n-gram和特征选择方法，以及(4)评估各种机器学习算法的性能。

Hu等人(2013)提出了一种基于操作码N-gram特征的聚类方法MutantX-S，该特征是从反汇编过程后获得的恶意软件的汇编语言源代码中提取的。通过应用哈希技巧和近似线性聚类算法，MutantX S提高了处理大量具有高维特征的恶意软件的可扩展性。该算法不是处理大量数据，而是仅在原型上执行凝聚层次聚类。

或者，Yuxin等人(2019)使用深度信念网络(DBN)作为自动编码器来降低输入特征向量的维数。结果，在学习完成之后，DBN的最后一个隐藏层输出作为输入传递的N-gram向量的新表示或编码。通过用未标记数据训练DBN，它们的分类精度优于K-近邻、支持向量机和决策树算法。

尽管n-gram方法在检测恶意软件方面取得了成功，但仍有一些问题值得一提。首先，详尽地列举所有的n克是不切实际的，而且在计算上也是不可行的。当特征数量大于样本数量时，估计模型参数可能导致维数灾难。因此，必须采用特征选择和简化技术。其次，研究人员(Raff等人，2018b)得出结论，字节n-grams似乎主要是从可执行文件中的字符串内容学习，特别是来自PE头的项目。由于有数百万个潜在的n-grams(对于更大的n)，特征选择技术倾向于选择那些出现足够频繁的特征。这鼓励选择主要由字符串和填充组成的低熵特征。第三，不管学到了什么n-grams，我们在对新样本进行分类时都必须获得精确的匹配。因此，任何微小的改变都不会使特征出现，因此不会影响我们的模型。因此，这种缺乏普遍性是过度拟合的潜在来源。

4.1.3.API

应用编程接口及其函数调用被认为是非常有区别的特性。文献表明，应用编程接口函数调用可以用来模拟程序的行为。本质上，API函数和系统调用与操作系统提供的服务相关，如网络、安全、文件管理等。由于软件没有其他方法可以在不使用API函数的情况下访问系统资源，因此特定API函数的调用提供了表示恶意软件行为的关键信息。

Sami等人(2010)提出了一个基于API调用使用情况对PE文件进行分类的三步框架。首先，他们分析了可移植可执行文件，并提取了导入的应用编程接口调用列表。第二，他们使用Clospan算法减少了特征向量(Yan等人，2003)。最后，使用特征子集来学习使用随机森林的模型。

叶等人(2008b)提出了一种基于规则的恶意软件分类系统。该系统由三个主要部分组成:(1)对等解析器，(2)面向对象关联规则生成器和(3)恶意软件检测模块。PE解析器负责解析可执行文件，提取对应API函数的静态执行调用。然后，这些调用被用作对等文件的签名，并存储在签名数据库中。然后，应用面向对象分析算法生成存储在规则数据库中的类关联规则。最后，特征调用和规则被传递给恶意软件检测模块，以确定文件是良性的还是恶意的。

Ahmadi等人(2016)使用从对近500 K个恶意软件样本的分析中提取的794个API函数调用的子集的频率来构建多模态系统，以将恶意软件分类为家族。第节提供了他们研究的完整描述.