**基于机器学习的恶意PDF检测与分类器对抗性学习研究**

王凤娇，江纬和杨育斌B2

（1.单位名1,城市名1 邮政编码1）

（2.单位名2,城市名2 邮政编码2）

**Abstract -** Nowadays, with the highly rapid development of information, it is becoming more and more important to detection based on malicious documents (such as PDF). A attack for document type is targeted and more vulnerable. Because of the diversity of its structure and form, attackers have various accesses and its technique is hard to be retrieved. Therefore, this research project aims to provide document (PDF) classifier based on AI for industry and academia. The study has collected 200,000 samples data and analyzed training process. This results show that the accuracy for training model highly rated 99.82% while the false positives only rated 0.01%; and through the study of antagonism, the model has certain ability to resist attacks and good robustness. This model can be widely used in the Terminal Security Products or mail servers when a file is detected at a millisecond level. It is a feasible model for AI is not limited to use in PDF document classifier, also can be extended the structure of documents such as word, excel etc.

**Key Words：**AI Security; Machine Learning; Maldoc Detection; Adversarial ML

**摘 要：**在当今信息安全领域，基于恶意PDF的检测越发重要，基于文档的攻击通常具有针对性（targeted attack），加上其文件结构的多样性，攻击手段变得丰富且隐蔽，因此更容易成功。。此研究的目的是为工业界和学术界提供一个基于AI的恶意文档（PDF）分类器原型。目前我们收集良性和恶意样本共20万个，经过解析（Parsing），训练生成AI模型。实验数据表明，模型的准确率达到99.82%，而误报率却只有0.01%，单个文件的平均检测时间仅需几毫秒。进一步地，我们研究了部分对抗性学习的方法，并用实验数据证明模型具有良好的抗攻击（抗样本逃逸）能力和鲁棒性。最后，我们通过实际应用，表明此模型可广泛部署在终端安全产品，邮件服务器上等，这些均是非常有意义的应用场景。

**关键词：** 人工智能之于信息安全；机器学习；恶意文件检测；对抗性学习

作者简介：作者1姓名（19XX－ ），性别，籍贯，职称，最后学位（或在读学历），主要研究方向，通信地址和邮编，手机，电话，Email;作者2姓名（19XX－ ），性别，籍贯，职称，最后学位（或在读学历），主要研究方向，通信地址和邮编，手机，电话，Email。项目基金：名称、编号

1. **简介**

随着时间的推移，PDF规格和样式变得更加丰富。新版本增加脚本的功能使文档与可执行文件几乎能以相同的方式工作，如连接到Internet，运行进程以及与其他文件/程序进行交互等。这种复杂性的增长为攻击者提供了更多的武器来发动攻击，并且能更灵活地隐藏恶意有效载荷，并逃逸检测。由于企业和个人普遍对此类安全漏洞反应迟缓，安全意识不足，导致大量的用户系统未采用最新版本修补程序进行更新，最终使这些攻击取得成功。

在2014年发现的24个0-day中，有16个是针对Adobe Reader和Flash Player的。在通用漏洞与披露（CVE）也可以明显观测到，从2015年开始关于Adobe Reader发现的漏洞呈现高增长态势，这给基于PDF的文档攻击敞开了大门。

针对于近几年遇到的多种基于文档的攻击，传统的PDF恶意软件检测方法有基于Shellcode的检测[15]、基于签名的检测[13]等。这些方法均存在识别率不高、无法及时更新恶意代码等普遍问题。基于机器学习的PDF恶意软件检测为此提供了崭新的方向，最开始使用机器学习的方法是2011年Nedim Srndic等人，主要对javascript进行提取分析，之后很多研究者基于内容和结构提取文件的静态特征[8],或是基于元数据与结构提取文件特征[24]使用SVM和决策树对文件进行分类，经过AI算法调优后，可以达到很好的效果。我们的方法结合他们所提到的一些特征进行提取，包含有结构，内容，javascript,元数据信息等。并通过研究发现，我们使用随机森林的算法比使用SVM 算法的准确率要高。同时我们也考虑到了AI模型的安全问题，Nedim Srndic等人在后来的研究中关注的不止是模型的准确率，而是AI模型的抗逃逸与鲁棒性，他们在IEEE会议上[4]针对AI模型的逃逸提出了几种假设，最后成功逃逸分类器，在我们的实验当中同时也使用了其中的4种方法来验证我们的模型是否具有这种抗逃逸的能力，我们通过对模型的一些特征选取和算法调优，发现可以使之前逃逸的一部分样本被检测出来，说明我们的模型有一定的抗逃逸能力。

本文的主要贡献如下：

* 一个PDF数据集，总样本数达到二十万，其中恶意样本173036个，正常样本28332个。无论在数量还是种类上，均达到业界领先水平；
* 精心选取了一套静态特征集（133个）以用于刻画PDF恶意软件形象（Profile），使其能快速有效的区分恶意与良性样本；
* 模型准确度高达99.82%，误报率0.01%，单个文件检测时间维持在毫秒水平，领先业界；
* 成功使用自己生成的变种病毒对分类器发动逃逸攻击，分类器根据攻击进行自我修复，重新训练得出一个鲁棒性强，抗逃逸能力强的模型

1. **相关工作**

如今PDF恶意文档检测技术可大约分为两大类：静态分析和动态分析。静态分析无需使样本运行，仅通过文件头部格式，二进制层N-gram等静态模式，即可对目标样本进行预测；动态分析，则通过使目标样本运行于受控环境内，以此捕捉其恶意行为。一般而言，静态分析的优点是速度易于部署更新，动态分析速度慢，消耗资源多，但人员参与精度很高。两种方法在业界均有大量成功应用案例，更高级的解决方案如可以把静态和动态分析结合，典型的工作如Maiorca et al. [9]。表1基于之前的研究列出了对现有方法的总结。

表1 PDF分析技术对比

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 方法 | 分析重点 | 检测技术 | 工作 | 年份 | 外置解析器 | ML | Detectable Discrepancy |
| 静态分析 | JavaScript | Lexical 分析 [5] | PJScan | 2011 | Y | Y | Y |
| JavaScript | Token 聚类 [12] | Vatamanu et al. | 2012 | Y | Y | Y |
| JavaScript | API 调用分类 [7] | Lux0r | 2014 | Y | Y | Y |
| JavaScript | Shellcode and opcode 签名 [13] | MPScan | 2013 | N | N | N |
| Metadata | Linearized object path [11] | PDF Malware Slayer | 2012 | Y | Y | Y |
| Metadata | 分层结构检测 [8, 1] | Srndic et al. | 2013 | Y | Y | Y |
| Metadata | 基于内容和Metadata [24] | PDFrate | 2012 | Y | Y | Y |
| Both | 基于结构和内容解析 [8] | Maiorca et al. | 2015 | Y | Y | Y |
| Both | 结合上述几种技术解析分类 [9] | Maiorca et al. | 2016 | Y | Y | Y |
| 动态分析 | JavaScript | Shellcode and opcode签名检测 [15] | MDScan | 2011 | Y | N | N |
| JavaScript | 已知的攻击模式 [16] | PDF Scrutinizer | 2012 | Y | N | N |
| JavaScript | 内存访问模式 [17] | ShellOS | 2011 | Y | N | Y |
| JavaScript | 常见 maldoc 行为分析 [18] | Liu et al. | 2014 | N | N | Y |
| JavaScript | 独立平台的tap point标识技术[2] | tap point | 2016 | N | N | Y |
| 文档类型 | 异常内存访问约束变量 [19] | CWXDetector | 2012 | N | N | N |
|  | 平台多样性 | 系统平台多样性利用[8] | PlatPal | 2017 | Y | N | Y |

由表1可见，静态分析一般聚焦于JavaScript本身或使用Metadata进行分析。代表性的检测技术有基于Shellcode和OPCode签名的MPScan[13]、基于结构与内容两者的分类[9]。动态分析技术一般聚焦于提取嵌入在PDF文档中的JavaScript代码，再通过实际试运行这些代码片段，检测出恶意行为。这类工作的代表有基于maldoc的行为分析[8]和基于平台多样性的Platpal[21]等。

在以上工作中，有12/15的工作使用外置的PDF解析器，这使得外置PDF解析器的健壮性成为研究焦点。这是因为外置PDF解析器一般设计和实现均较为简单，恶意样本经少量变异即能轻易逃逸此类解析器。这种攻击在Carmony et al.[20]的工作中被称为解析器混淆性攻击（Parser-Confusion Attacks）。

从表1可知，机器学习一般并不适合于动态分析，而几乎所有的静态分析工作，都在某种程度上使用了机器学习的技术。这部分的典型工作有PDFrate[7]、PDF Malware Slayer[11]等。这些工作均声称能在低功耗环境下达到很高的检测精度，但对模型本身的安全性，恶意样本逃逸分类器等对抗性学习的研究内容却鲜有提及。这种攻击在Xu et al[14]的工作中被提出，作者通过构建一个能自动生成恶意样本变种的框架，使得在每一次的样本变异迭代过程中，原始输入样本集会经过某种遗传算法把良性PDF对象加入到恶意样本集中。在不断的变异过程中系统一方面需保持恶意样本的恶意本来面目不变，另一方面则需要达到迷惑分类器的目的。这种专门针对分类器的攻击及其框架被称为自动化分类器逃逸攻击（Automatic Classifier Evasion Attack）

在上述工作中，有11/15的工作有如下假设：即恶意样本和良性样本间需具有明显特征分辨能力或分界线。换句话说，我们假设恶意样本和良性样本有一高平面能把其很好地在高纬度特征空间中分开。一些有趣的研究性问题是这样的：是否可以通过不改变原文件的恶意属性，用增加良性行为部分的方法，以成功逃逸分类器的检测？是否可以通过不改变原文件的善意行为，用增加恶意行为的方法，使这些样本通过隐藏方式，成功逃逸分类器？Srndic et al. [4]的工作从恶意样本着手，聚焦于前一种攻击，我们把其称为模拟性攻击（Mimicry Attack）；而Maiorca et al.[10]的工作从良性样本着手，我们在这里称其为反向模拟性攻击（Reverse Mimicry Attacks）。

综上所述：对于外置PDF解析器，现有攻击手段是解析器混淆攻击（Parser-Confusion Attacks）；对于机器学习模型，现有的攻击手段是自动化分类器逃逸攻击（Automatic Classifier Evasion Attacks）；对于假设性的“可检测的分辨力”(Detectable Discrepancy)，现有攻击手段为模拟和反向模拟攻击（Mimicry and Reverse Mimicry）。这些攻击手段对于模型的安全本身就提出了很大挑战，在我们的工作当中不仅生成了一个准确度高的模型，并且在模型安全本身也有所建树，可以在多中场景中进行推广使用。

# 模型构建

## 3.1 特征提取

有效的特征提取方法主要有基于结构、Metadata、内容和Javascript。实验数据表明，基于结构的特征具有很好的分类能力。我们结合以上特征，模型识别率高达99%，且特征提取方法的主要优势是在于：

1. 提取方式已经较为成熟，使用工具就可以很快将文件结构提取出来；
2. 提取文件结构的时间也比其他的提取方式要快，利用更少的时间就可以将文件的特征提取出来；
3. 以现有的技术提取出文件结构，已经可以达到一个很好的精确度，经过多次测试都达99%以上。

我们通过计算样本集中每一个特征的平均值，发现正常样本的特征均值与恶意样本的特征均值在某些特征中的差异明显（具体可看表2）。

差异明显的特征有如count\_font、count\_box：在正常样本中会有很多关于font ,box这些对象，是因为PDF文件主要功能在于用这些对象来描述信息。而恶意文档一般不把展示信息作为其首要功能，通常是直接把JS恶意代码嵌入到文档当中，以运行恶意代码。

特征count\_page\_obj和count\_obj：一般来说，良性样本的obj对象比恶意样本多很多，在统计同一个页面中obj对象的个数时，良性样本和恶意样本会存在约1倍差距，如果obj在同一个页面中突然增多，此文件为恶意文件的概率大增。

特征count\_endobj 与count\_endstream：这是基于结构的具有很强分辨能力的特征，良性PDF样本在每个对象结束时会有一个endobj，但PDF恶意文件为了混淆解析器，会近可能少地使用endobj和endstream。这就导致解析器在解析恶意PDF文件时不能完整获取整个对象，或者导致整个PDF文件解析失败，使恶意PDF文件成功逃逸AI模型。这是恶意文件最常使用的逃逸解析器的方法，在本文稍后部分中会提及如何防止恶意文件使用本方法进行逃逸。

另一个特征是count\_js。恶意文件的主要攻击手段是嵌入JS代码来执行恶意行为，或调用系统命令。因此，一个恶意文件所含JS代码量会比良性样本的代码量多。还有一部分用于混淆和加密的JS的大小与良性样本间也存在一定的差异。

还有一个重要的差异是特征count\_acroform\_obs。在PDF Specification 1.2 中引入AcroForm。这种表单从用户处通过交互方式收集信息。表单支持包括[数据表示](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E8%A1%A8%E7%A4%BA)、数据捕捉和数据编辑等功能。它还可以进行动态交互，从具有动态计算、验证及其他特性的交互式、可编辑的表单，到由服务器生成的、机器填充的表单等。同时动态布局表单可以自动重新调整自身以适应用户或外部数据源（如数据库服务器）提供的数据。[动态文档](https://baike.baidu.com/item/%E5%8A%A8%E6%80%81%E6%96%87%E6%A1%A3" \t "_blank)将基于数据内容进行装配，而大型生产运行包含数十万个事务。基于以上几个特点，表单很容易成为攻击者混淆和加密的地方，故在计算AcroForm值的时候，恶意样本比正常样本高约一倍。

表 2: 良性样本与恶意样本之间的特征比较

|  |  |  |
| --- | --- | --- |
| Features | Benign Files | Malware Files |
| **count\_font** | **14.646365** | **0.551510** |
| **count\_acroform\_obs** | **700** | **1400** |
| **count\_box\_a4** | **12001** | **200** |
| **count\_box\_legal** | **395040** | **0** |
| count\_box\_letter | 7291529 | 866773 |
| count\_box\_other | 32.184318 | 1.749550 |
| count\_box\_overlap | 1000 | 0 |
| **count\_endobj** | **95.800680** | **9.685137** |
| **count\_endstream** | **30.436044** | **3.785357** |
| **count\_page\_obj** | **8001** | **16003** |
| count\_image\_large | 110711 | 400 |
| count\_image\_med | 465247 | 6401 |
| **count\_image\_small** | **915892** | **12002** |
| count\_image\_total | 36.56366 | 0.30006 |
| count\_image\_xlarge | 300 | 0 |
| count\_image\_xsmall | 21.64216 | 0.11202 |
| **count\_js** | **0.714071** | **1.012402** |
| **count\_obj** | **100.969997** | **12.014803** |
| count\_objstm | 1.572057 | 0.153831 |

## 3.2 分类算法

首先对收集到的文件进行分类，将提取出来的特征作为训练数据集，在这个时候，随机森林（random forests）在分类上表现了很好的优势，有效且误报率极低，并易于使用，可以很快的对数据进行分类。随机森林分类方法给出的结果是基于很多棵分类树判断结果的集合展现，每一个决策树都是在训练数据中随机选择生成的，因此，随机森林总的来说是一个集成分类器(确定有集成分类器这一说法？)，他使用 bagged training data, 通过随机选择的特征子集,，并使用该节点的训练数据，确定每个节点处的最佳分割来创建树中的每个节点。此外，每棵树都是基于一个独立的特征子集，最后，在分类过程中每一个树的投票来确定最终结果。

AI引擎的重要组成部分之一是算法，我们选取了几个实用性较好的算法来比较包括KNN，邻近算法，NNET(神经网络)，RF（随机森林）SVM（支持向量机）经过多次的训练与分类实验，发现随机森林准确率高，误报率低，低延时鲁棒性良好，等和可解析性等优势，于是将其作为我们选定作为默认算法。。并且在特征发生改变的时候，随机森林准确率依然趋于一个稳定的值。

表3 算法准确率对比

|  |  |  |  |
| --- | --- | --- | --- |
| KNN | NNET | RF | SVM |
| 0.971292 | 0.824163 | 0.996411 | 0.752392 |

## 3.4 数据集

PDF数据集共分成良性和恶意两类，目前训练和测试样本一共20万，具体个数如表4所示。我们从VirusShare下载样本156035个，大小约6.8G。另有9000个良性样本来自开源数据集Contagio，还有2万样本来自mimicus[2] ，包括有contagio的5,000个良性文件和5,000个恶意文件，还有5000的google的良性样本，和VirusStotal的5000个恶意样本。其他的数据集是在搜狗和百度上通过爬虫爬取下来的。其中mimiucus中的数据集中的两个用于[4]中的PDFRATE实验性评估：Contagio可供下载。（姑娘，这段数据集描述非常散，可否重新整理，可读性较差）

表4 训练样本的版本分布情况

|  |  |  |
| --- | --- | --- |
| 恶意样本 | 良性样本 | 总数 |
| 173036 | 28332 | 201368 |

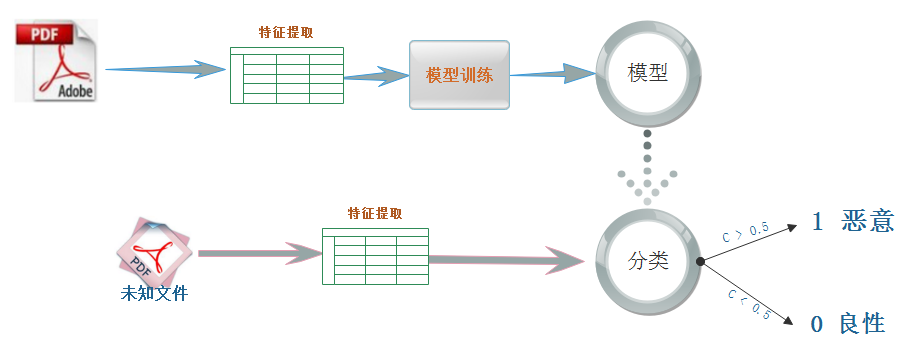
## 3.5 模型实现

所提出的基于机器学习的恶意PDF文档检测的方法包括以下两个步骤，如图1所示：

1. 提取文件特征。此为基本的预处理步骤，对PDF文件的结构、内容和元数据进行解析，并做一些相应的向量计算，提取为一个二维的特征集，使得这些特征可以进入到基于机器学习的模型中进行训练分类。

2. 学习和分类。我们会随机选取数据的80%进行训练，通过训练之后保存训练模型，然后使用20%的文件进行预测分类，从中计算出模型的准确率，误报率等信息。

图1 机器学习的基本框架



在我们的实验当中，一共对模型进行了4次更新，其中最开始的模型（model1）是使用peepdf为解析工具，然后通过一些特征计算与特征量化，使其可以用于机器学习训练与预测，当我们的特征数提取到133个的时候，这些特征包括有基于结构的（count\_font、size、count\_startxref），内容信息的（title\_oth、subject\_lc）和metadata(producer\_oth、producer\_len)的一些静态属性,并通过多次实验核对比较选取性能和精度最好的算法(随机森林RF)，此时准确率就达到了99.30%。可是经过我们的研究发现，由于在一开始使用peepdf进行解析的时候，并非所有文件都解析到，所以在第二阶段重新选取了解析器mimicus[2]，这个工具可以解决之前因为结构缺陷或混淆而不能正常解析的问题，目前这个工具已经开源[2]，迄今为止也是基于PDF文件检测使用最多的一个解析工具。我们使用了mimicus 对之前的文件进行解析，可以对所有的数据（20万）进行解析，并做特征提取。

于是在模型2的训练中，我们先使用平衡的数据集进行训练与预测，其中包含2万恶意样本与2万良性样本，在总的数据集中进行随机抽选一共4万个样本用于model2 的训练与预测。并且从Model2 开始我们就使用mimicus 对文件进行特征提取，一共提取特征135个。我们的机器学习的核心算法还是使用准确率最高的随机森林。通过算法调优后对数据进行训练与预测，使Model2的检测率提高到99.99%，误报率降低为0.012%，并经过多次随机选择数据进行训练预测后，准确率依然可以维持在99.99%。

经过Model2 的算法调优和数据筛选后，我们把训练数据增加到十万级别，来对模型进行训练。我们的训练是在本地进行的，在4核4G的CPU上，训练10万个样本仅需要56s, 当训练样本达到十万级别的时候，Model2.1准确率持续稳定在99.81%，误报率为0.086%。

# 对抗性学习

## 4.1 对抗性学习

我们对模型的鲁棒性做了一些测试与对抗性学习，具体来说就是假设攻击者已知信息，比如说攻击者知道我们所提取的特征，那么他就可以修改恶意文件的特征，使其看起来更像良性，从而逃逸分类器。又或者假设攻击者知道模型提取的特征和算法，通过加密、混淆等方法逃逸分类器；按照一定的逻辑关系，当攻击者知道模型的信息越多，越容易逃逸。我们主要利用Nedim Smdic [4]中提到的几种方法，对模型进行对抗性学习。

在通过一系列的对抗学习攻击之后，我们对模型重新训练，使模型对那些经过特定技术伪装的恶意文件具有一定的免疫能力，并且我们将Model3的训练数据升级到20万，添加了一些新奇的样本，如通过模仿良性样本的恶意文件，和反向模仿生成的变异文件，一同加入到了Model3 的训练当中，并对整个机器学习的参数重新调优，经过我们的多次训练，模型3 在数据达到20w 万的情况下，更新模型花费了3分42秒，预测时间需要2秒，最后使模型准确率达到99.82%，误报率小于0.01%。

为了研究分类器逃逸攻击（evasion attack），我们复现了Nedim Srndic[4]中提到的4种逃逸攻击。攻击方法描述了攻击者的可用信息。

* F（feature）：表示只有特征集可用于敌手；
* FT（feature and training）：除了已知的特征外，攻击者还可以利用目标分类器训练数据集的知识；
* FC（feature and classifier）：攻击者知道特征集以及关于分类器的一些细节，例如类型，参数或具体实现；
* FTC（all above）：如果知道所有三个分类器组件的细节，在这种情况下，攻击者可以在线下完全重现在线分类器，只有在找到足够好的规避样本时才提交攻击结果。离线模仿攻击或离线分类器特定攻击击败离线分类器也具有很强的击败在线分类器的可能性。那么对手就有最大的机会躲避目标分类器。

表5 不同攻击方法与精度

|  |  |  |  |
| --- | --- | --- | --- |
| 攻击方法 | 测试样本个数 | Model2准确度 | 修复后的检测率 |
| F | 2157 | 71.18% | 96.71% |
| FC | 240 | 2.92% | 12.5% |
| FT | 4196 | 84.25% | 96.76% |
| FTC | 600 | 15.83% | 18.71% |

我们通过使用这四种方法找到关于PDFrate的回避变体后，对我们的模型进行预测，发现仍然有很大一部分，既可以逃逸PDFrate ,也可以逃逸我们的分类器，其中在FC情况下比FTC情况的逃逸更高，这就说明了我们训练的模型与PDFrate的一个差异性：在特征提取方面比较接近，所以容易逃逸；而在后面算法训练这部分，就没有全部逃逸。

针对于这些逃逸样本，如果我们的特征选取已经被利用，那么我们可以通过改变特征集，修改权值，或删除重要特征等操作，将样本重新训练测试。或者我们可以使用多个算法来对文件分类，再计算不同算法在分类器中的权值，综合给出最后文件的预测结果。

## 4.2 模型更新

我们对训练后的特征权值进行统计 如图2 所示，是模型训练后，按照特征重要性排序的前30个特征，我们可以看到count\_font，count\_javascript，size，count\_obj，count\_endobj这几个特征在分类中占有比较多的权值，同时也是非常容易被攻击者利用来对解析器和分类器进行逃逸，于是我们在训练时就删除了这几个特征，然后对新文件预测，预测结果如表6 所示。

图2 前30个重要特征分布图



由表6 我们可以观察到，当分类器使用全部特征进行训练分类的时候，模型可以达到99.8%的准确率，而当我们将第一个重要的特征在训练的时候删去的时候，检测率基本没有太大的波动，当减到size，count\_obj，count\_endobj 这几个特征时，模型的准确率是趋于稳定的。这也说明了我们的模型可以对抗一些基于特征的攻击，即使对手知道我们分类器使用的特征，模型同样可以达到99%的精度。

表6 对前5个特征修改后的模型识别率

|  |  |
| --- | --- |
| Feature delete train | 识别率 |
| None | 0.9982 |
| count\_font | 0.9952 |
| count\_javascript | 0.9952 |
| size | 0.9964 |
| count\_obj | 0.9964 |
| count\_endobj | 0.9964 |

同时我们还通过重要特征自减来评估模型的鲁棒性，对特征有效性进行研究，将所有特征进行一个重要性排序，然后依次将最重要的特征删减后进行重新训练预测，图3是特征在自减之后的识别率的图示，我们可以观察到当特征减少了100个之后，模型检测率依然可以达到90%，说明模型具有很好的鲁棒性和抗攻击能力。

图3 特征自减后的识别率



## 4.3 模型评估

为了评估模型的预测性能，我们把数据集分为训练（90％）和测试（10％）两部分，并采用10-Fold Cross Validation（我们对每个组合重复该过程10次以上来验证结果。 这个方法有助于评估模型的鲁棒性）。

我们通过改变输出类别概率的阈值来生成ROC曲线。ROC曲线下面积约为1，表明有效的特征和鲁棒性模型。图4所示，我们的检测率超过99％，RF分类器的误判率低于0.01％。换句话说，静态分析在区分良性和恶意文件方面非常有效。我们可以使用静态分析有效识别未知文件的好坏，以发现可疑信息。

图4 ROC曲线图



性能：

在处理大量PDF数据时，高吞吐量是一个重要的考虑因素，就像V IRUS T OTAL 那样。我们的系统旨在处理这种负载，并利用现代计算机系统的并行处理能力。我们测量了使用随机森林，决策树和SVM 对十万的数据集进行特征提取，学习和分类所花费的时间如 表7 。训练预测是在一个4核4G的CPU的一个Ubuntu 的系统下进行的

特征提取是最耗时的操作，因为它需要从硬盘加载所有的PDF文件，并对文件进行逐个解析。于是我们将文件解析与训练分步处理，作为中间结果保存，不仅可以减少CPU 内存占用，同时也可以使模型在更新训练时更为快速。当训练样本达到十万级别的时候，解析一共耗时22分钟36秒，我们测试的数据有10万，一共5.8G。然后对文件进行训练和预测。如表7 是不同算法之间的训练时间和预测时间，以及精度的统计，由表我们可以看出，随机森林和决策树的预测时间都快于SVM，但是决策树的训练时间会比随机森林快很多，这个是由算法本身的因素决定的，并且随机森林的准确率是明显高于其他算法的。

表7 训练时间与预测时间

|  |  |  |  |
| --- | --- | --- | --- |
|  | Training | Classification | accuracy |
| Random Forest | 56s | 1s | 99% |
| Decision tree | 4s | 1s | 97% |
| SVM | 58m 18s | 12s | 75% |

1. **应用实例：蓝盾AI防火墙**

按照模块化的思想，我们把基于AI的文档分类器作为一独立检测模组，集成到边界安全产品如防火墙中。一个拥有30多年历史的老式安全产品，该如何为其插上AI的翅膀？

在当今边界安全产品中，如防火墙，对于网络应用层（第七层）的恶意文件扫描已经是国际标准。业界对此功能的需求极为严苛，一个优秀的功能模组通常要求单个文件检测（扫描）时间（即延迟）维持在毫秒级别，对PDF文件的检测准确度需在99%以上，而误报率则不能大于0.01%。

低延迟的功能要求是因为文件扫描功能，在硬件层，是串联于整个文件检测的流水线中，高延迟会显著增加丢包率，造成严重数据丢失，这对于安全设备是不可容忍的。这些年，随着恶意软件的急速增加，先前基于规则匹配的引擎从 标配 开始力不从心：一方面为了达到产品需求的高准确度，需要大量的安全分析人员编写规则和旧设备的规则库；另一方面随着核心规则库的扩大，规则匹配算法的时间也呈现指数级别增长。这两方面都促使我们探索更新更好的AI引擎。

1. **总结**

在本论文中，我们详细介绍了基于AI的PDF恶意文档分类器的设计与实现，并在此基础上，结合具体真实应用场景，成功搭建了一个基于动静态分析的恶意文档立体检测平台。实验数据表明，在十数万级文档数据集中，我们能达到大于99%的准确度和小于0.01%的误报率。更进一步的，且在实际运行时，CPU和内存的 时空效能比（time & space performance）比旧有基于规则的模型，有显著提升。

本文除了使用大量数据描述安全AI（apply AI tech. to security），把AI安全(securing the model & the associate) 也放在了同等重要的位置上。我们通过大量的实验，模拟了（1）攻击者通过对恶意样本的增删改（如添加好的PDF对象），以混淆分类器，大量逃逸的目的；（2）分类器经自身修正，通过重新训练模型，去除已被攻击者所利用的特征（feature set been exploited），以维持模型的健壮性。

基于AI的文档分类器是社会工程学、病毒分析等领域的重要研究课题。在未来，我们还会尝试解决以下研究问题：

* 基于深度学习的恶意PDF文档检测
* 动静态分析引擎的调优
* 对于Microsoft Office等其他文件格式的支持（如docx，pptx等）

参考文献:

1. Nedim ˇ Srndic and Pavel Laskov. Detection of Malicious Pdf Files Based on Hierarchical Document Structure. In 20th Network and Distributed System Security Symposium (NDSS), 2013
2. Nedim ˇ Srndic and Pavel Laskov. Mimicus: A Library for Adversarial Classifier Evasion. <https://github.com/srndic/mimicus>.
3. Nedim Šrndic and Pavel Laskov . Hidost: a static machine-learning-based detector of malicious files, Šrndi′c and Laskov EURASIP Journal on Information Security (2016) 2016
4. Nedim Srndic and Pavel Laskov. Practical Evasion of a Learning- Based Classifier: A Case Study. In Proceedings of the 35th IEEE Symposium on Security and Privacy (Oakland), San Jose, CA, May 2014
5. Pavel Laskov and Nedim Srndic. Static Detection of Malicious JavaScript-Bearing PDF Documents. In Proceedings of the Annual Computer Security Applications Conference (ACSAC), 2011
6. Davide Balzarotti, Marco Cova, Christoph Karlberger, Christopher Kruegel, Engin Kirda, and Giovann Vigna. Efficient Detection of Split Personalities in Malware. In Proceedings of the 17th Annual Network and Distributed System Security Symposium (NDSS), San Diego, CA, February–March 2010
7. Igino Corona, Davide Maiorca, Davide Ariu, and Giorgio Giacinto. Lux0R: Detection of Malicious PDF-embedded JavaScript Code through Discriminant Analysis of API References. In Proceedings of the Artificial Intelligent and Security Workshop (AISec), 2014.PDFrate
8. Davide Maiorca, Davide Ariu, Igino Corona, and Giorgio Giac- into. A Structural and Content-based Approach for a Precise and Robust Detection of Malicious PDF Files. In *Proceedings of the International Conference on Information Systems Security and Privacy (ICISSP)*, 2015.
9. DavideMaiorca, DavideAriu, IginoCorona, andGiorgioGiacinto. An Evasion Resilient Approach to the Detection of Malicious PDF Files. In Proceedings of the International Conference on Information Systems Security and Privacy (ICISSP), 2016.
10. Davide Maiorca,IginoCorona,andGiorgioGiacinto.Lookingat the Bag is not Enough to Find the Bomb: An Evasion of Structural Methods for Malicious PDF Files Detection. In Proceedings of the 8th ACM Symposium on Information, Computer and Commu- nications Security (ASIACCS), Hangzhou, China, March 2013.
11. Davide Maiorca,Giorgio Giacinto,and Igino Corona. APattern Recognition System for Malicious PDF Files Detection. In *Pro- ceedings of the 8th International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM)*, 2012.
12. Cristina Vatamanu, Drago¸s Gavrilu ¸ T, and R˘azvan Benchea. A Practical Approach on Clustering Malicious PDF Documents. Journal in Computer Virology, June 2012.
13. Xun Lu, Jianwei Zhuge, Ruoyu Wang, Yinzhi Cao, and Yan Chen. De-obfuscation and Detection of Malicious PDF Files with High Accuracy. In *Proceedings of the 46th Hawaii International Con- ference on System Sciences (HICSS)*, 2013.
14. Weilin Xu, Yanjun Qi, and David Evans. Automatically Evading Classifiers: A Case Study on PDF Malware Classifiers. In Proceedings of the 2016 Annual Network and Distributed System Security Symposium (NDSS), San Diego, CA, February 2016. http://evademl.org/
15. Zacharias Tzermias, Giorgos Sykiotakis, Michalis Polychronakis, and Evangelos P. Markatos. Combining Static and Dynamic Analysis for the Detection of Malicious Documents. In Proceedings of the 4th European Workshop on System Security (EUROSEC), 2011.
16. Florian Schmitt, Jan Gassen, and Elmar Gerhards-Padilla. PDF Scrutinizer: Detecting JavaScript-based Attacks in PDF Docu- ments. In *Proceedings of the 10th Annual International Confer- ence on Privacy, Security and Trust (PST)*, 2012.
17. Kevin Z. Snow, Srinivas Krishnan, Fabian Monrose, and Niels Provos. ShellOS: Enabling Fast Detection and Forensic Analysis of Code Injection Attacks. In *Proceedings of the 20th USENIX Security Symposium (Security)*, San Francisco, CA, August 2011.
18. DaipingLiu,HainingWang,andAngelosStavrou.DetectingMa- licious Javascript in PDF through Document Instrumentation. In *Proceedings of the 44th International Conference on Dependable Systems and Networks (DSN)*, Atlanta, GA, 2014.
19. Carsten Willems, Felix C. Freiling, and Thorsten Holz. Using Memory Management to Detect and Extract Illegitimate Code for Malware Analysis. In Proceedings of the Annual Computer Security Applications Conference (ACSAC), 2012.
20. Curtis Carmony, Mu Zhang, Xunchao Hu, Abhishek Vasisht Bhaskar, and Heng Yin. Extract Me If You Can: Abusing PDF Parsers in Malware Detectors. In Proceedings of the 2016 Annual Network and Distributed System Security Symposium (NDSS), San Diego, CA, February 2016
21. Meng Xu and Taesoo Kim, Georgia Institute of Technology:PlatPal: Detecting Malicious Documents with Platform Diversity . 26th USENIX Security Symposium 2017
22. VirusTotal. Free Online Virus, Malware and URL Scanner.https://www.virustotal.com/.
23. Stephan Chenette. Malicious Documents Archive for Signature Testing and Research - Contagio Malware Dump. http://contagiodump.blogspot.de/2010/08/ malicious-documents-archive-for.html.
24. Charles Smutz and Angelos Stavrou. Malicious PDF Detection using Metadata and Structural Features. In Proceedings of the Annual Computer Security Applications Conference (ACSAC), 2012.
25. <https://www.symantec.com/content/dam/symantec/docs/reports/istr-22-2017-en.pdf>
26. M.Polychronakis,K.Anagnostakis,andE.Markatos.Com- prehensive shellcode detection using runtime heuristics. In Annual Computer Security Applications Conference (AC- SAC), pages 287–296, 2010.