# 1.恶意文档检测器的设计与实现

在本节中，我们展示一个基于机器学习的恶意文档检测框架。我们的方法是在文档附加信息（metadata）和结构中提取特征。使用真实数据库(做测试)，（实验结果）展示了（1）如何使用文档属性去满足恶意软件检测要求；（2）这些特征对于未知样本的耐久性。由我们的分析可知，随机森林方法，一种从各单独分类树中随机选取特征的整体分类器，即使在未知样本下，依然取得最高分类准确度。我们的分类准确率在99%以上，同时误报率控制在0.01%内。

在本节中，我们展示一个基于机器学习的恶意文档检测框架。实验中我们采用的数据有20W，其中包含了所有PDF的文件类型。我们主要对这些文件的内容和结构进行解析，选取具有良好分类效果的特征，然后对提取到的特征用机器学习的方法进行分类。实验结果表明，通过我们提取的特征和分类方法，可以使模型准确率在99%以上，同时误报率控制在0.01%内

## 3.1 数据集

目前收集的数据一共201368个，良性（28332）和恶意（173036）两大类，其中我们收集到的文件数据有167061个，其中有156035个是从VirusShare下载下来的，大小有约6.8G，另有9000个正常良性样本来自于Contagio，2026的良性数据集是在搜狗和百度上通过爬虫抓取下来的。

通过我们对对抗性学习的研究，使用VirusShare为源样本，又生成了7000个对抗样本，在最后的试验中用于测试。

还有mimicus中的数据集用于PDFRATE实验性评估，可供下载[4]。mimicus 开源数据集有2万的平衡样本，包括来自于contagio的5,000个良性样本和5,000个恶意样本，以及来自于google的5000个良性样本和VirusTotal的5000个恶意样本。

## 3.2 特征提取

有效的特征提取方法主要基于结构、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。我们通过计算样本集中每一个特征的平均值，发现正常样本与恶意样本的特征均值在某些特征中存在明显差异差（具体见表2）。

特征如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。

特征如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。

特征如count\_endobj 与count\_endstream：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。这是恶意文件最常使用的逃逸解析器的方法。

特征如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。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。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均值对比

|  |  |  |
| --- | --- | --- |
| Feature | Benign File | Malware File |
| **count\_font** | **14.64** | **0.55** |
| **count\_acroform\_obj** | **700** | **1400** |
| **count\_box\_a4** | **12001** | **200** |
| **count\_box\_legal** | **395040** | **0** |
| count\_box\_letter | 7291529 | 866773 |
| count\_box\_other | 32.18 | 1.74 |
| count\_box\_overlap | 1000 | 0 |
| **count\_endobj** | **95.80** | **9.68** |
| **count\_endstream** | **30.43** | **3.78** |
| **count\_page\_obj** | **8001** | **16003** |
| count\_image\_large | 110711 | 400 |
| count\_image\_med | 465247 | 6401 |
| **count\_image\_small** | **915892** | **12002** |
| count\_image\_total | 36.56 | 0.30 |
| count\_image\_xlarge | 300 | 0 |
| count\_image\_xsmall | 21.64 | 0.11 |
| **count\_js** | **0.71** | **1.01** |
| **count\_obj** | **100.96** | **12.01** |
| count\_objstm | 1.57 | 0.15 |

**Design and Implementation of malicious file detector**

In this session, we present a framework of detecting malicious documents based on machine learning. The main method we used is to extract features from file structure and metadata. We use real dataset for testing, and the experiment results reveal that how to satisfy the malware detection requirement by file attributes, and durability of those features to unknown samples. From the analysis, we have found that Random Forest, an ensemble classifier which can extract features randomly from each decision tree, can reach the highest accuracy rate even using unknown samples.

We adopt a dataset includes 200,000 samples which contain all kinds of PDF files. We parse the content and structure of these files and select features with good classification performs, and then classify the features by using machine learning. The experiment results reveal that the accuracy rate is over 99% and false positive rate is less than 0.01%.

3.1 Dataset

The PDF dataset, a total of 201368, is divided into two categories: 28332 benign samples and 173036 malicious samples. Among these samples, 156035 are downloaded from VirusShare, with a size of 6.8G; 9000 benign samples of Contagio dataset, 2026 are obtained from Sogou and Baidu (two search engine in China).

We use VirusShare as source samples to generate 7000 adversarial samples which are used in testing in the experiment.

Besides, we obtained the open source dataset from mimicus which is used for assessing PDFRATE[4]. This dataset contains 20,000 balanced samples, with 5,000 benign and 5,000 malicious subsamples of Contagio dataset, 5,000 benign samples obtained from Google and 5,000 malicious samples downloaded from VirusTotal.

3.2 Feature Extraction

Effective methods for extracting features are based on structure, metadata, content and Javascript. The experiment results reveal that features based on structure perform well in classification. We calculate the average value of each features in the dataset and find that the average values of some features in benign and malicious samples are different.

Features such as count\_font,count\_box: There are several objects like font, box contained in the benign samples as PDF file mainly uses these objects for description. However, malicious files do not describe information, they usually run the malicious code embedded in the file.

Features such as count\_page\_obj and count\_obj: Generally, obj in benign files are much more than those in malicious files. When calculating the obj in the same page, the amount of obj in benign file . Thus, if obj in the same page increases sharply, the file is a malicious one with high probability.

Features such as count\_endobj and count\_endstream: In benign files, endobj occurs when an object ends. Yet malicious PDF files seldom use endobj and endstream so as to confuse the parser. Thus, the parser fails to obtain the whole object when parsing malicious file, or fails to parse malicious file which can evade successfully finally. That is the common evasion of parser.

Features such as count\_js: The main tactic of malicious file is to embed JS code to execute malicious behavior. In this way, JS codes contained in a malicious file are much more than those in a benign file.

Features such as count\_acroform\_obs: AcroForm is introduced in PDF Specification 1.2, which is to collect information from users via interaction. The form can display data, capture data and edit data, etc. Moreover, it can conduct dynamic interaction including: from the interactive and editable forms which contain characteristics like dynamic calculation, verification and so on, to the forms generated by servers and filled in by machine. With those characteristics, the form is vulnerable to obscuration and encryption by attacker. As a result, the value of AcroForm in a malicious sample is double that of a benign sample.