[5]R. Sommer and V.Paxson. Enhancing Byte-Level Network Intrusion Detection Signatures with Context[J].Proc. ACM Conference on Computer and Communications Security （CCS）, 2003.

基于 Python 的机器学习入侵检测的研究

◆孟　平　龙华秋

（五邑大学智能制造学部　广东　529020）

摘要：由于网络技术迅速发展，攻击成本低廉而与安全相关的产品却十分缺乏或落后，导致近年来安全问题频出。如何从海量的流量中检测出攻击威胁成了一种迫切需要而又困难重重的问题。恰逢机器学习的兴起，当下的硬件性能等条件可被满足的前提下，利用机器学习来判断数据报文是否恶意，成了一种理论上可行的方案。本篇包括流量特征的提取和相关机器学习算法的介绍，根据日常捕获的流量来提取相关的特征，再经由机器学习模型的判断，可迅速判断其是否具有威胁性。经过使用测试集对机器学习的模型进行评估，其准确率可以达到 99%。

关键词：Python；机器学习；入侵检测

如今互联网的发展速度和规模的扩张十分迅猛，IT技术、硬件产品和软件产品的迭代更新非常迅速。互联网+的时代，如此迅速的发展给我们的生活带来了巨大的便利，同时也带来了众多的安全隐患。在大多数人眼中，互联网上的安全问题，大多都是数据泄露，以及其他相关问题，这些问题影响的大多都是企业，造成的损失也基本是经济上的损失。然而近年来发生的多起工业控制系统相关的安全问题，充分证明了攻击者完全可以通过互联网威胁到人们的现实生活，比如近期国外发生的工控系统入侵事件导致的他国全国范围大部分地区停电的事件。当下互联网的安全问题，已经被国家高中重视，网络安全事关国家安全。

利用机器学习中的监督学习相关算法训练好的模型，在实际中应用时，可以快速对一个新提取的样本做判断，判断其是否恶意。比以往通过特征码等其他判断方式，少了查询“特征码集合”和比对的操作，其判断速度上要远快于传统方式。良好的训练集和合适的算法及相应的参数，可使训练出来的模型具有很好的泛化性能。

1 系统设计

计算机网络具有多层协议，在其七层模型（物理层、数据链路层、网络层、传输层、会话层、表示层、应用层）中，物理层和数据链路层太过底层，会话层、表示层和应用层涉及的协议众多不适合作为用于进行判断其是否恶意的基础。在网络层上可以开始不考虑具体的数据传输的实现，而传输层TCP和UDP协议是上层协议的基础且仅为实现连续和离散的传输方式。因此在本次研究中，判断流量是否恶意便是基于网络层和传输层的数据报文。

在整个机器学习入侵检测中，我们需要有“流量探针”为我们捕获流量，而捕获的数据报文要经过“特征提取”，提取的特征会组成一个“样本”，样本将会是我们直接用于训练模型和模型判断的目标。在本次研究中，由于服务器性能等问题，为了方便，我们直接使用KDD 99 比赛中提供的五百万现成的样本，作为训练模型和模型评估的数据集合。在整个系统中将分为三个部分：模型训练、样本采集和模型判断。

图1为系统流程图。

2 具体实现

如流程图中所示，整个系统中有三个部分相对独立：模型训练、检测判断、样本采集。本系统涉及机器学习的内容，要保证模型训练、样本采集和检测判断的部分的最终样本的维度特征的一致性，才能保证模型的使用。因本课题中，对样本的判断可以看作是分类，即是否具有攻击性和可能的攻击类别，因此在机器学习模型算法的选择上，我们将选择分类器相关的算法。

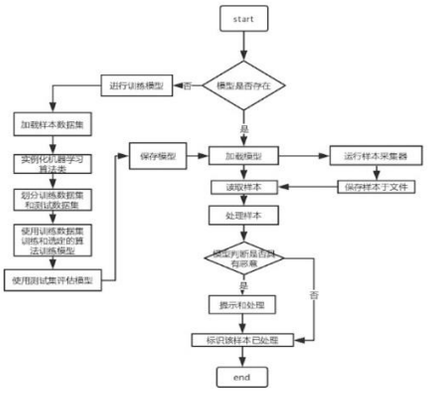

图 1 系统流程图

以下将对模型训练阶段的内容进行讲解：

2.1 机器学习算法—逻辑斯谛回归

在本次研究课题中，我们采用的是机器学习中监督学习的逻辑斯谛回归算法（Logistic Regression）。需要注意的是，逻辑斯谛回归算法并不是回归模型相关的算法，而是用于分类的分类器算法。

2.1.1 算法原理

逻辑斯蒂回归（Logistic Regression），是一种分类算法，其算法原理如下：

$$z = w^T x + b$$

不同类的 x 对应 z 的值为 0 或者 1，单位阶跃函数由于函数性质不可导不连续，所以引入 sigmoid 函数：$y = \frac{1}{1+e^{-z}}$。

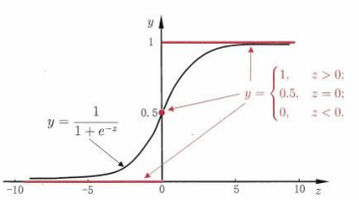

图 2 $y = \frac{1}{1+e^{-z}}$ 函数图

$$y=\frac{1}{1+e^{-z}}$$

所以令:

$$z=w^Tx+b$$

得到:

$$y=\frac{1}{1+e^{-(w^Tx+b)}}$$

化成对数形式:

$$\ln\frac{y}{1-y}=w^Tx+b$$

$y/(1-y)$ 代表了样本作为正例的可能性，$1-y$ 是反例。

所以 y=概率 $p(y=1|x)$

$1-y$=概率 $p(y=0|x)$

所以上式又可化为:

$$\ln\frac{p(y=1|x)}{p(y=0|x)}=w^Tx+b$$

解得:

$$p(y=1|x)=\frac{e^{w^Tx+b}}{1+e^{w^Tx+b}}$$

$$p(y=0|x)=\frac{1}{1+e^{w^Tx+b}}$$

将这 2 个等式化成一个等式:

$$p(y_i|x_i;w,b)=(\frac{e^{w^Tx_i+b}}{1+e^{w^Tx_i+b}})^y(\frac{1}{1+e^{w^Tx_i+b}})^{1-y}$$

对上式连乘，然后取对数，进行极大似然估计:

$$\lambda(w,b)=\sum_{i=1}^{m}\ln p(y_i|x_i;w,b)$$

最大化上式相等于最小化:

$$\ell(w)=\sum_{i=1}^{m}(-y_iw^Tx+\ln(1+e^{w^Tx}))$$

这是一个凸函数，可以使用梯度下降或者牛顿迭代（割线法）来求最优解。

梯度下降的迭代式为:

$$w=w+\alpha\nabla_\beta\zeta(w)$$

因为要往增高的方向，所以是+号，代表梯度的正方向。

2.1.2 逻辑斯蒂回归——多分类扩展

逻辑斯蒂回归是二分类器，本系统是多分类器，所以采用 OVR 的方式将二分类器的方式扩展到多分类器，OVR 训练多个分类器，每一个分类器将某一类别的视为正样本，其他视为负样本，训练出 n 个后，然后输出每个样本对应每个类别的概率，取最大的概率作为最终的输出结果。

2.2 特征工程

为了方便，我们在模型训练阶段直接使用了 KDD 99 比赛的数据集，以缩减前期数据采集的时间。该数据集采集自美国军方网络，其样本的特征基于计算机网络的网络层和传输层的，但在其提供的特征上做了适当的选择和特征的向量化，使其更加适用于本课题的研究。

具体特征内容如下：

（1）duration：TCP 以 3 次握手建立到 FIN/ACK 连接结束为止的时间；

（2）protocol type：协议类型；

（3）Service：网络服务类型；

（4）flag：连接正常或错误；

（5）src_bytes：源主机到目标主机数据字节数；

（6）dst_bytes：目标主机到源主机数据字节数；

（7）land：若连接来自或送达同一个主机则为 1，否则为 0；

（8）wrong_fragment：错误分段数量；

（9）urgent：加急包个数；

（10）count：与当前连接具有相同的目标主机的连接数；

（11）srv_count：与当前连接具有相同服务的连接数；

（12）serror_rate：在与当前连接具有相同目标主机的连接中，出现“SYN”错误的连接的百分比；

（13）srv_serror_rate：在与当前连接具有相同服务的连接中，出现“SYN”错误的连接的百分比；

（14）rerror_rate：在与当前连接具有相同目标主机的连接中，出现“REJ”错误的连接的百分比；

（15）srv_rerror_rate：在与当前连接具有相同服务的连接中，出现“REJ”错误的连接的百分比；

（16）same_srv_rate：在与当前连接具有相同目标主机的连接中，与当前连接具有相同服务的连接的百分比；

（17）diff_srv_rate：在与当前连接具有相同目标主机的连接中，与当前连接具有不同服务的连接的百分比；

（18）srv_diff_host_rate：在与当前连接具有相同服务的连接中，与当前连接具有不同目标主机的连接的百分比；

（19）dst_host_count：与当前连接具有相同目标主机的连接数；

（20）dst_host_srv_count：与当前连接具有相同目标主机相同服务的连接数；

（21）dst_host_same_srv_rate：与当前连接具有相同目标主机相同服务的连接所占的百分比；

（22）dst_host_diff_srv_rate：与当前连接具有相同目标主机不同服务的连接所占的百分比；

（23）dst_host_same_src_port_rate：与当前连接具有相同目标主机相同源端口的连接所占的百分比；

（24）dst_host_srv_diff_host_rate：与当前连接具有相同目标主机相同服务的连接中，与当前连接具有不同源主机的连接所占的百分比；

（25）dst_host_serror_rate：与当前连接具有相同目标主机的连接中，出现 SYN 错误的连接所占的百分比；

（26）dst_host_srv_serror_rate：与当前连接具有相同目标主机相同服务的连接中，出现 SYN 错误的连接所占的百分比；

（27）dst_host_rerror_rate：与当前连接具有相同目标主机的连接中，出现 REJ 错误的连接所占的百分比；

（28）dst_host_srv_rerror_rate：与当前连接具有相同目标主机相同服务的连接中，出现 REJ 错误的连接所占的百分比。

2.3 模型训练代码

```
def ReadData(path):
data=open(path).readlines()
data=np.array([i.split(',') for i in data])
data[:, -1] = [i.replace('\n', '').replace('.', '') for i in data[:, -1]]
data_r=np.zeros(shape=data.shape)
data_r[:,0]=[float(i) for i in data[:,0]]
for i in range(4,40):
data_r[:, i] = [float(j) for j in data[:, i]]
protocol_type={k:i for i,k in enumerate(set(data[:,1]))}
service={k:i for i,k in enumerate(set(data[:,2]))}
flag={k:i for i,k in enumerate(set(data[:,3]))}
label={k:i for i,k in enumerate(set(data[:,-1]))}
print(protocol_type)
print(service)
print(flag)
print(label)
data_r[:, 1] = [protocol_type[j] for j in data[:, 1]]
data_r[:, 2] = [service[j] for j in data[:, 2]]
data_r[:, 3] = [flag[j] for j in data[:, 3]]
```

```
    data_r[:, -1] = [label[j] for j in data[:, -1]]
    data_r=np.c_[data_r[:,0:9],data_r[:,22:42]]
    weight={}
    for j in range(len(label)):
    weight[j]=len(data)-len([i    for i in data_r if i[-1]==j])
    return data_r,weight

    def Classify(feature,weight):
    lr=LogisticRegression()
    X_train, X_test, y_train, y_test = train_test_split(feature[:,:-1],
feature[:,-1], test_size = 0.3, random_state = 42)
    w=[weight[i] for i in y_train]
    d = [weight[i] for i in y_test]
    lr.fit(X_train,y_train,sample_weight=w)
    print(str(lr.score(X_test,y_test,sample_weight=d)))
    joblib.dump(lr, 'kdd99lr')

    def train():
    feature,weight=ReadData(r'kddcup.data.corrected')
    Classify(feature,weight)

    train()
```

3 功能测试

3.1 数据采集功能（见图3）

```
0,tcp,private,OTH,1514,0,0,0,0,3,3,0.00,0.00,0.00
,0.00,1.00,0.00,0.00,3,3,1.00,0.00,0.75,0.00,0.00
,0.00,0.00,0.00normal.DONE
0,tcp,private,OTH,1514,0,0,0,0,4,4,0.00,0.00,0.00
,0.00,1.00,0.00,0.00,4,4,1.00,0.00,0.80,0.00,0.00
,0.00,0.00,0.00normal.DONE
0,tcp,private,OTH,1514,0,0,0,0,5,5,0.00,0.00,0.00
,0.00,1.00,0.00,0.00,5,5,1.00,0.00,0.83,0.00,0.00
,0.00,0.00,0.00normal.DONE
0,tcp,other,OTH,54,0,0,0,0,1,1,0.00,0.00,0.00,0.0
0,1.00,0.00,0.00,1,1,1.00,0.00,0.50,0.00,0.00,0.0
0,0.00,0.00normal.DONE
0,tcp,private,OTH,1514,0,0,0,0,6,6,0.00,0.00,0.00
,0.00,1.00,0.00,0.00,6,6,1.00,0.00,0.86,0.00,0.00
,0.00,0.00,0.00normal.DONE
```

图 3 数据采集功能

其中DONE表明该表明该样本已经过模型检测判断等处理。

3.2 模型检测

图4内容为已训练好的模型对样本采集器中样本的检测结果。

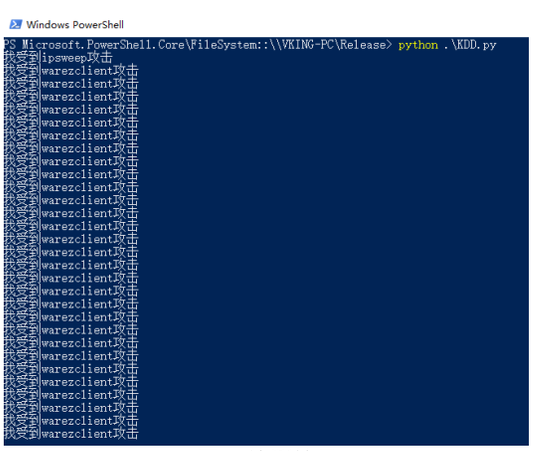

图 4 检测结果

3.3 模型性能评估（见表1）

表1 模型性能评估表

	precision	recall	f1-score	support
satan	1.00	0.90	0.95	4822
phf	0.00	0.00	0.00	1
ipsweep	0.98	0.20	0.33	3709
smurf	1.00	1.00	1.00	842033
pod	0.00	0.00	0.00	72
rootkit	0.00	0.00	0.00	3
neptune	0.99	1.00	0.99	321807
warezmaster	0.00	0.00	0.00	5
multihop	0.00	0.00	0.00	1
nmap	0.88	0.09	0.17	733
teardrop	1.00	0.99	0.99	279
back	0.00	0.00	0.00	664
spy	0.00	0.00	0.00	1
ftp_write	0.00	0.00	0.00	4
loadmodule	0.00	0.00	0.00	3
land	0.50	0.20	0.29	10
buffer_overflow	0.00	0.00	0.00	10
warezclient	0.00	0.00	0.00	303
guess_passwd	0.00	0.00	0.00	13
portsweep	0.25	0.00	0.00	3079
Imap	0.00	0.00	0.00	1
perl	0.00	0.00	0.00	1
multihop	0.98	1.00	0.99	291976
Micro-avg	0.99	0.99	0.99	1469530
Macro-avg	0.33	0.23	0.25	1469530
Weighted-avg	0.99	0.99	0.99	1469530

4 结束语

本文详细介绍了基于Python的机器学习入侵检测系统的设计与实现。项目总共有三大功能需要实现：模型训练、检测判断、样本采集。在本项目中，我们直接使用KDD 99 比赛中提供的五百万现成的样本，作为训练模型和模型评估的数据集，使用逻辑斯蒂回归算法对流量样本进行训练，从大量的流量数据集合中找到恶意样本。通过在实际环境进行的大量网络流量测试，与流量样本的测试，验证了该入侵检测系统的实用性。

参考文献：

[1]陈春玲，吴凡，余瀚.基于逻辑斯蒂回归的恶意请求分类识别模型[J].计算机技术与发展，2019，29（02）：124-128.

[2]王展鹏，吴红光，马蓓娇，周梦甜，张曼雨，周驰航.基于机器学习的工业物联网入侵检测技术研究[J].智能物联技术，2018，1（02）：13-17.

[3]戴梦杰，罗颖，刘真岩，彭夕茁，张金全.用机器学习方法检测基于 PHP 的 web shell 进展回顾[J].网络安全技术与应用，2019（05）：34-35.

[4]王展鹏，吴红光，马蓓娇，周梦甜，张曼雨，周驰航.基于机器学习的工业物联网入侵检测技术研究[J].智能物联技术，2018，1（02）：13-17.

[5]刘阚蓉，李丹，裴梦迪，张家熹.机器学习算法在网络入侵检测中的应用综述[J].赤峰学院学报（自然科学版），2018，34（12）：44-46.

[6]席海龙，刘海燕，张钰.应用于入侵检测的机器学习现状与发展分析[J].价值工程，2018，37（34）：269-272.

[7]V. Kanimozhi，T. Prem Jacob. Artificial Intelligence based Network Intrusion Detection with hyper-parameter optimization tuning on the realistic cyber dataset CSE-CIC-IDS2018 using cloud computing[J]. ICT Express，2019.

基金项目：2017 年省级高等学校大学生创新创业训练计划项目。