PBCNN-论文笔记

2021-05-19 09:35:55

- 论文笔记
- 入侵检测
- 流量分析
- 小样本学习

PBCNN: Packet Bytes-based Convolutional Neural Network for Network Intrusion Detection

来源信息

- Computer Networks(**CCF B**)
- 机构：清华大学
- 作者：Lian Yu； Jingtao Dong； Lihao Chen； Mengyuan Li； Bingfeng Xu et al.

摘要

网络入侵检测系统(IDS)可保护目标网络免受数据泄露和人员隐私不安全的威胁。然而，现有的网络入侵检测研究大多不能有效地实现对目标的保护，特别是在很大程度上取决于利用领域专家的知识和经验人工设计的统计特性，并且未能解决少数样本数据问题。网络流量具有层次结构，即字节数据包流。本文提出了 一种基于字节的分层包CNN，称为PBCNN，第一级自动从原始Pcap文件中的字节中提取抽象特征，然后第二级进一步构建流或会话中的包表示，而不是使用已有的CSV文件中的特征，充分利用原始数据信息。多个卷积池模块与大小可变的卷积核级联，一层TextCNN获得流量流的表示，并将表示到3层全连接网络进行入侵分类。应用基于PBCNN的小样本学习，提高了少实例问题的网络攻击类别的检测可靠性。实验表明，CIC-IDS2017和CSE-CIC-IC-IDS2018数据集优于现有的研究。

介绍

现有的问题

- 使用过时的数据集

  63.8%的人使用了KDDCUP99数据集，这些数据集可能不包含当前互联网中网络入侵的某些特征。因此，所提出的方法可能不能很好地检测到当前的网络入侵。

- 使用传统的学习方法

  利用传统的机器学习方法，如SVM（支持向量机）和决策树(DT)来检测网络入侵。

- 依赖于人工或半人工获得的统计特征

  CIC-IDS2017和CSE-CIC-IDS2018数据集相对较新，具有更多的网络攻击类型。但是现有的研究9篇有8篇是基于CSV格式的数据集进行检测的，它们的研究高度依赖于提取的特征，而不是原始的网络流量数据，即CSV文件中统计特征的质量成为这些研究的上界。

- 未能解决少数样本数据问题

  实际上，尽管互联网上有大量的网络流量数据，但攻击流量数据的量非常小，而且这些数据往往难以标记。因此，异常类别或类的训练样本往往不足，而且大多数现有的检测方法可能很难检测到这类类型的攻击。

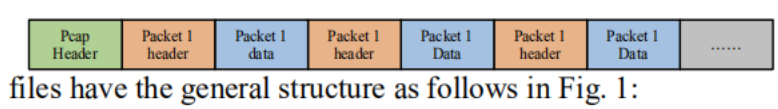
论文的贡献

- 提出了一种特征提取和分类模型：为了从原始流量数据中自动提取抽象特征并增强评价指标，提出了一种基于字节的分层卷积神经网络(PBCNN)，以适应网络数据流的层次，自动从包中的字节中提取抽象特征，并从会话或流中的分组中提取抽象特征，并进一步执行网络入侵检测的多种分类。实验表明，该模型优于使用CIC-IDS2017和CSECIC-IDS2018数据集的最新模型。
- 处理小样本：为了提高网络攻击类别的检测能力，本文提出了一种基于PBCNN的小样本学习，利用数据集CIC-IDC2017和CSE-CIC-IDS2018来检测异常网络流量，并与其他2D-CNN模型进行比较，证明了小样本能力的影响。
- 利用一个相对较新的数据集：本文利用了CIC-IDS2017年生成的CSE-IIC-CIC-IDS2017年和2018年生成的数据集，分别包含12种和14种网络攻击。根据官方网站的介绍，数据集是通过模拟当前互联网中的网络流量分布生成的，包含大量的原始流量，特别是CSE-CIC-IDS2018高达470GB。因此，它们相对更适合于网络入侵检测的研究。

网络流量数据预处理

分析网络流量

每一个pcap文件都是如下的结构，在解析的过程中就按照该结构进行分层解析。



files have the general structure as follows in Fig. 1:

在Pcap头之后，有许多捕获的包，每个包包括包头和包数据。以数据包的TCP/IP协议格式为例，开发的解析器根据需要根据协议提取每个字段。

```
###[ Ethernet ]###
  dst       = 02:9e:42:48:90:aa
  src       = 02:ca:69:c4:6d:06
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 358
     id        = 31440
     flags     = DF
     frag      = 0
     ttl       = 127
     proto     = tcp
     chksum    = 0x870
     src       = 18.218.115.60
     dst       = 172.31.69.28
     \options   \
###[ TCP ]###
        sport     = 51145
        dport     = http
        seq       = 1253654258
        ack       = 3372848359
        dataofs   = 5
        reserved  = 0
        flags     = PA
        window    = 2053
        chksum    = 0xd462
        urgptr    = 0
        options   = []
###[ Raw ]###
           load      = 'GET / HTTP/1.1\r\nHost: 18.218.83.150\r\nUser-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:
58.0) Gecko/20100101 Firefox/58.0\r\nAccept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\nAccep
t-Language: en-US,en;q=0.5\r\nAccept-Encoding: gzip, deflate\r\nConnection: keep-alive\r\nUpgrade-Insecure-Requests:
1\r\n\r\n'
```

Fig. 2. Example of an Extracted Information from a Packet in Pcap Files

组织网络流量

网络流量粒度对数据格式和数据分布方面的分析有影响。本文探讨了基于双流的检测方法，也称为会话，其中源IP地址-源端口、目标IP地址-目标端口可以成对交换。从网络数据包中提取五个数据(源IP地址-源端口、目的地IP地址-目的地端口、协议)，并将进一步的数据包组织成会话。当五元中的任何元素丢失时，数据包将不参与建模和分析。

在对网络流进行解析，得到时间戳和五元组信息后，按五元组信息进行分组，得到根据设计的分割策略进行分割的会话数据，得到会话样本。

Fig. 3. Pcap data processing

上面获得的会话可能包含大量的数据包。需要一种分割策略来分离会话数据，以便进一步进行建模和分析。会话分割策略：

- 使用TCP连接的相关标志来确定会话的开始和结束：例如，可以使用“三方握手”中的SYN作为开始标记，也可以使用“四手波”中的FIN作为结束标记，来分割一个会话。
- 对会话数据使用超时策略：当通信链路等待很长时间，即空闲时间-无数据包交互，超过预设阈值时，认为会话终止。
- 使用周期重置策略：如果通信链路上的信息交互时间超过预设阈值，则会话被迫截断，随后的数据包合并为新的会话。

本文采用的分割策略为超时策略（64s）和周期重置策略（120s）。

整个分割过程如下：

Algorithm 1: SessionSplit

Input: pcap file *pcap_file*
Output: session list *sessions*[1..*n*]

```
// sessions:  [session_1, session_2, ..., session_n]
// session_map:  {key1 : session_1, key2 : session_2, ...}
// A: packets list
// Parse():  sort packets by timestamp
// GetFiveTuple():  get the five tuple of a packet
// GetTime():  get the timestamp of a packet
// GetStartTime():  get the start time of a session
// GetLastTime():  get the last packet's time of a session
sessions ← [];
session_map ← {};
A ← Parse(pcap_file);
for i ← 1 to length(A) do
    if A[i] has five tuple and A[i] is IPv4 then
        session_id ← GetFiveTuple(A[i]);
        if session_map has session_id then
            time ← GetTime(A[i]);
            start_time ← GetStartTime(session_map[session_id]);
            last_time ← GetLastTime(session_map[session_id]);
            if time - start_time ≥ 120s or time - last_time ≤ 64s then
                append session_map[session_id] to sessions;
                remove session_map[session_id] from session_map;
                create new session as session_map[session_id];
                insert A[i] to session_map[session_id];
            else
                insert A[i] to session_map[session_id];
            end
        else
            create new session as session_map[session_id];
            insert A[i] to session_map[session_id];
        end
    end
end
for session in session_map do
    append session to sessions
end
```

如图所示，它首先从会话数据中读取数据包，并将数据包的时间戳信息与前数据包的时间戳信息进行比较。如果时间间隔时间不超过64秒，则它将继续与会话的第一个数据包的时间戳进行比较。如果间隔时间不超过120秒，则该数据包将包含在会话样本中。否则，只要不满足上述两个条件中的任何一个，就将分组分成新的会话样本，存储旧的会话样本，同时记录分组的时间戳信息，然后处理下一个分组，直到会话数据中的所有分组都被处理。

数据转换

转换的目的是将会话数据处理为深度学习模型所接受的输入格式。此步骤包括匿名化、数字编码，以及统一会话示例中的字节和数据包的数量。

匿名化：网络流量数据中的地址，包括MAC源和目标地址，以及IP源和目标地址。 数字编码：将数据包转换为每字节0-255的像素值作为模型输入。

Raw packet：

02d8a94e384202ca69c46d0608004500004c968140003f064478
0d3a6240ac1f4519b6dc00160973533b848f36d0801800d3efdd
00000101080abbda7e80004e997b5353482d322e302d7061726
16d696b6f5f322e302e300d0a

Anonymization：

00000000000000000000000008004500004c968140003f06447
80000000000000000b6dc00160973533b848f36d0801800d3ef
dd00000101080a0000000000000000005353482d322e302d70617
2616d696b6f5f322e302e300d0a

Digital encoding：

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8, 0, 69, 0, 0, 76, 150, 129, 64,
0, 63, 6, 68, 120, 0, 0, 0, 0, 0, 0, 0, 0, 182, 220, 0, 22, 9, 115,
83, 59, 132, 143, 54, 208, 128, 24, 0, 211, 239, 221, 0, 0, 1, 1,
8, 10, 0, 0, 0, 0, 0, 0, 0, 0, 83, 83, 72, 45, 50, 46, 48, 45, 112,
97, 114, 97, 109, 105, 107, 111, 95, 50, 46, 48, 46, 48, 13, 10]

Unifying the number of bytes：

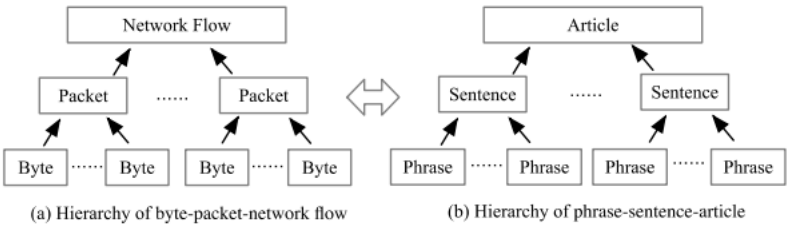
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 8, 0, 69, 0, 0, 76, 150, 129, 64,
0, 63, 6, 68, 120, 0, 0, 0, 0, 0, 0, 0, 0, 182, 220, 0, 22, 9, 115,
83, 59, 132, 143, 54, 208, 128, 24, 0, 211, 239, 221, 0, 0, 1, 1,
8, 10, 0, 0, 0, 0, 0, 0, 0, 0, 83, 83, 72, 45, 50, 46, 48, 45, 112,
97, 114, 97, 109, 105, 107, 111, 95, 50, 46, 48, 46, 48, 13, 10,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0…]

Fig. 4. Example of Data Transformation

为了优化梯度更新方向并加快训练，根据最大值和最小值对数据进行归一化。 标签编码：使用one-hot编码方式。

网络入侵检测建模

选择该网络结构的原因



(a) Hierarchy of byte-packet-network flow (b) Hierarchy of phrase-sentence-article

选择模型和确定神经网络结构的基础主要从两个方面来考虑：第一是数据处理、格式和特征；二是分类的性能和训练的效率。网络流在字节分组会话方面，具有分层结构，即，数据包由字节组成，而网络流或会话由数据包组成。它类似于自然语言处理(NLP)领域中的短语-句-文章的文本层次结构。CNN擅长捕捉局部上下文关系，如果多个卷积内核模块被级联以扩展接受域，CNN在获取远程信息依赖方面具有更大的潜力。一般来说，当有人认为某些信息属于上下文时，人们就会尝试使用CNN。此外，与RNN相比，CNN非常适合相似性，在GPU的加速方面具有显著的时间性能优势。

PBCNN模型结构：

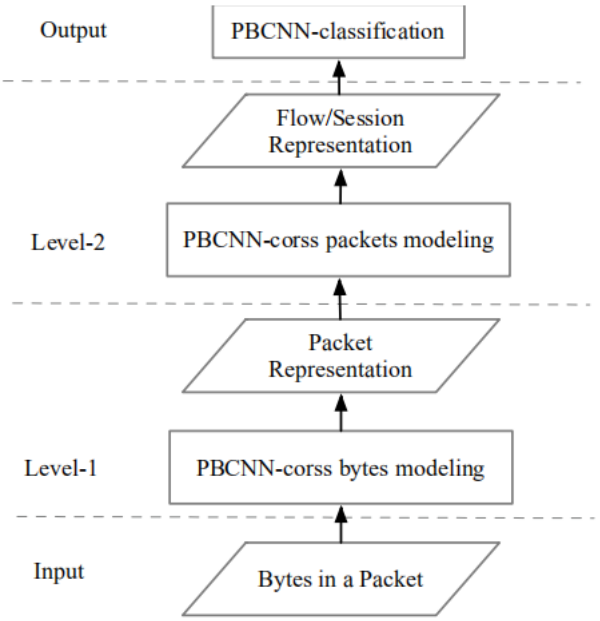
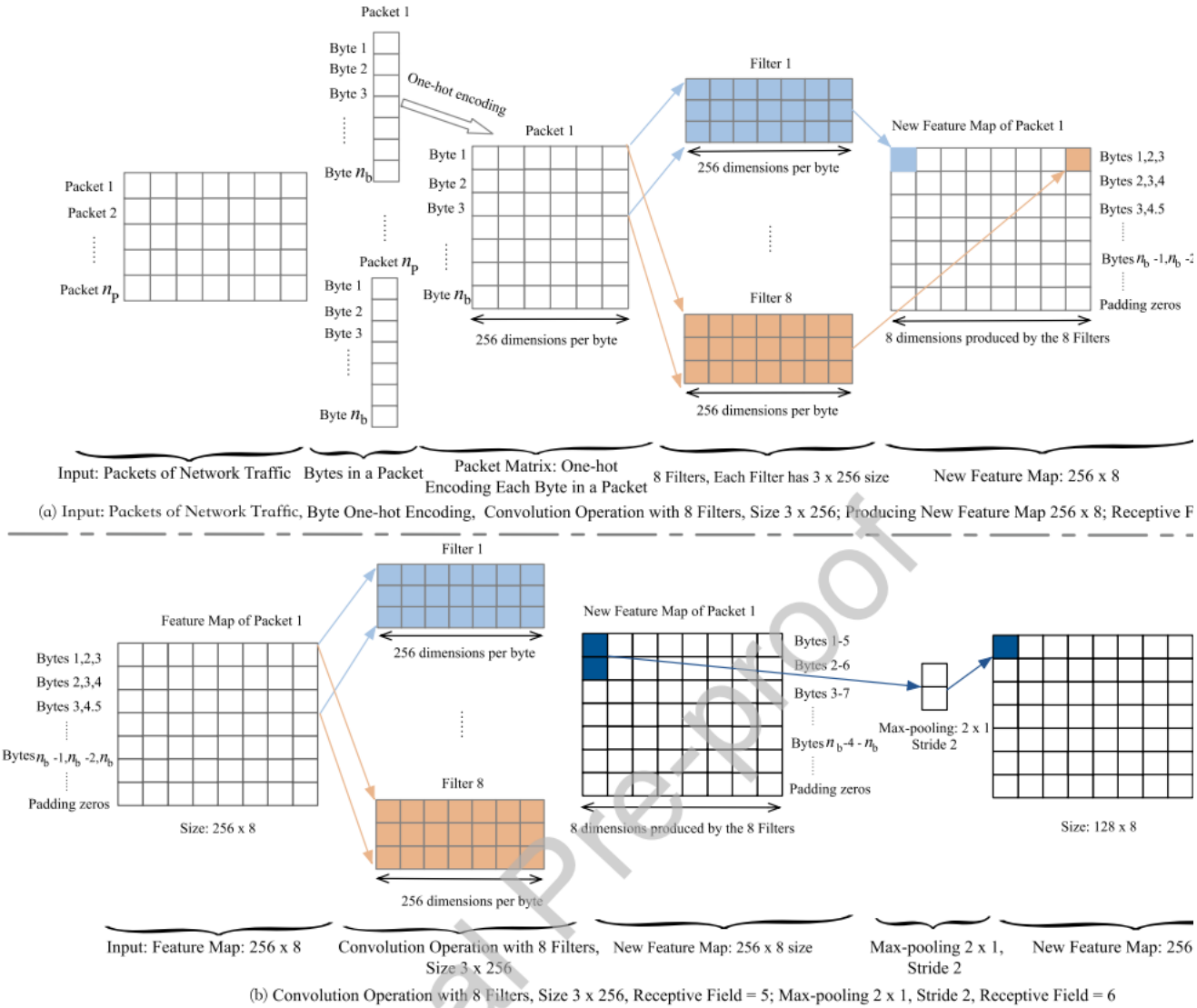


Fig. 6. Two-Level Hierachy of PBCNN Modelling

- Input：数据包中的字节
- Level-1：对字节进行建模，进行包的表示
- Level-2：对包进行建模，流/会话表示
- Output：输出分类的结果

由于会话中的数据包数量相对较小，就像处理TextCNN中NLP中的短语一样，不需要使用堆叠结构来扩展接受字段作为数据包中的处理字节。因此，只使用一层包含多个不同大小滤波器的卷积池模块从包向量学习表示来构建网络流，可以提高处理能力。

如下图为CNN模块：



(a) Input: Packets of Network Traffic, Byte One-hot Encoding, Convolution Operation with 8 Filters, Size 3 x 256; Producing New Feature Map 256 x 8; Receptive F

(b) Convolution Operation with 8 Filters, Size 3 x 256, Receptive Field = 5; Max-pooling 2 x 1, Stride 2, Receptive Field = 6

Fig. 7. Byte One-Hot Encoding as 1×256, Two Convolution Operations with $|W_1| = 8$ Filters, Size 3×256 (h=3), on Byte Vectors, and One Ma

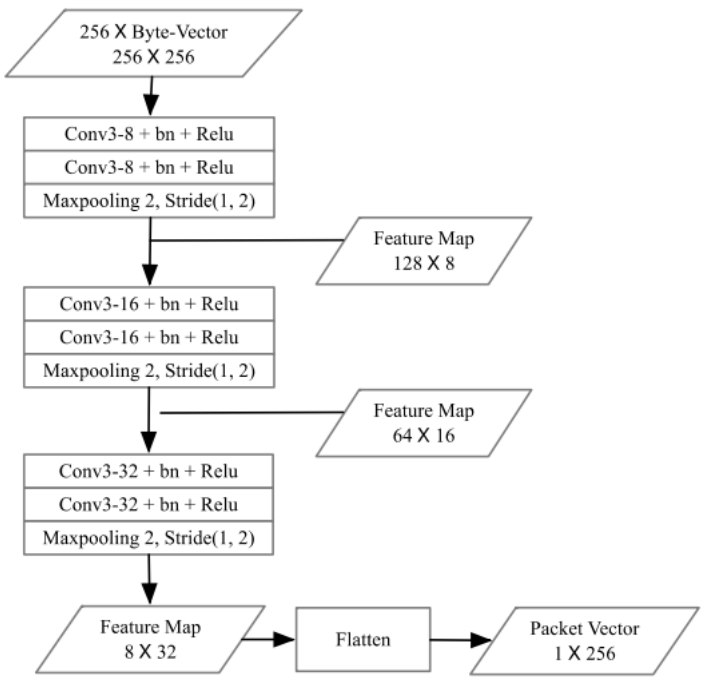


Fig. 8. Stacked Multiple Convolutional Operation and Pooling Modules

小样本入侵检测

使用的方法：孪生网络。

来自同一类别的数据对的两个样本是标签为1的正样本；来自不同类别的两个样本是标签为0的负样本，正负样本比为1：1.5。在训练阶段，它将多分类任务转换为二元分类匹配任务。

孪生网络的底层使用PBCNN来提取网络流量的特征，网络的上层使用一个小的神经网络来学习两个样本之间的距离。由PBCNN提取的两个样本的表示。遍罗网络上层的输入包括两个样本，以及另外两个可以在一定程度上表示样本距离的特征。这两个特征是欧几里得距离和余弦相似性。

实验

CSE-CIC-IDS2018数据集按比例分为训练集、验证集和测试集，训练集用于训练模型，验证集选择参数，测试集用于模型的效果测试。本文对各类数据分层进行采样。根据6：1：3的比例，确保每种样本的比例一致，防止该类别中缺少一类数据。

- 分层实验对比

TABLE VII. MICRO- AND AMCRO- AVERAGES IN BASELINE EXPRIMENTS

Model		Micro-Ave	Macro-Ave
		P/R/F1	P/R/F1
2D-CNN	VGG	0.918/0.928/0.919	0.460/0.434/0.431
	MobileNet	0.943/0.944/0.941	0.539/0.515/0.521
	ResNet	0.957/0.953/0.953	0.716/0.745/0.728
Hierarchical-Model	Hierararchical-CNN-LSTM	0.965/0.912/0.513	0.519/0.509/0.513
	PBCNN	**0.982/0.983/0.983**	**0.746/0.748/0.747**

- 针对每一个类别的检测精度

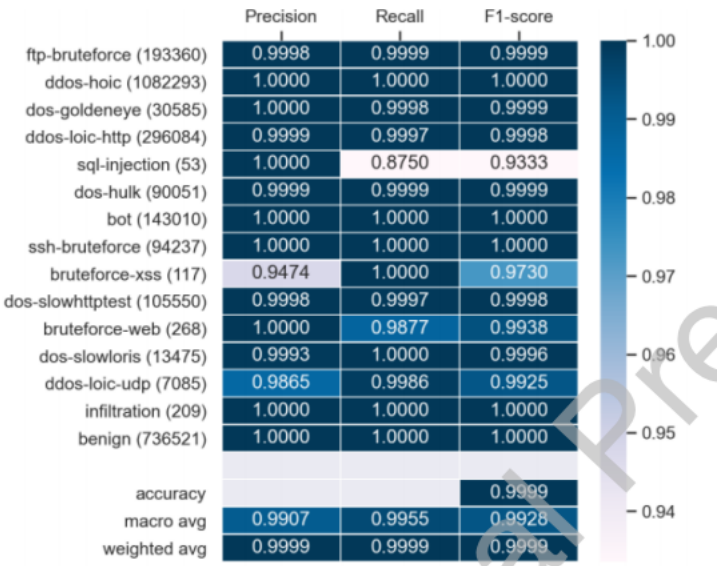


Fig. 16. Heat map: classification results by PBCNN for the 15 categories

- 与其他算法的精确度对比

TABLE VIII. COMPARED WITH [8] IN TERMS OF ACCURACY

Model	Accuracy (%)
PBCNN	99.99
Adaboost	99.69
Decision Tree	99.66
Random Forest	99.21
KNN	98.52
Gradient Boosting	99.11
Linear Discriminant Analysis	90.80

- 与传统机器学习算法对比

TABLE X. COMPARED WITH TRADITIONAL MACHINE LEARNING

Algorithm	Precision	Recall	F1-score	Time(Sec)
KNN	0.96	0.96	0.96	1908.23
RF	0.98	0.97	0.97	74.39
ID3	0.98	0.98	0.98	235.02
Adaboost	0.77	0.84	0.77	1126.24
MLP	0.77	0.83	0.76	575.73
Naive-Bayes	0.88	0.04	0.04	14.77
QDA	0.97	0.88	0.92	18.79
PBCNN	**0.982**	**0.983**	**0.983**	478.40

- 在CIC IDS2017数据集上的实验

TABLE XII. BENCHMARK EXPERIMENTS WITH TRADITIONAL MACHINE LEARNING APPROACHES USING CIC-IDS 2017 DATASET

Category	VGG	MobileNet	ResNet	Hierarchical-CNN-LSTM	PBCNN
	P/R/F1	P/R/F1	P/R/F1	P/R/F1	P/R/F1
Sql Injection/5	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
Infiltration/11	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
Bot/99	0/0/0	0/0/0	0.958/1.000/0.947	0/0/0	**1.000/1.000/1.000**
Web Attack XSS /12	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
Brute Force/116	0/0/0	0/0/0	**0.931**/0.964/**0.947**	0/0/0	0.903/**0.966**/0.933
PortScan/100	0/0/0	0.282/0.440/0.344	0.920/**1.000**/0.958	0/0/0	**0.960**/0.960/**0.960**
Macro-Ave	0/0/0	0.047/0.073/0.057	0.468/**0.494**/0.475	0/0/0	**0.477**/0.488/**0.482**

总结

优势

- 使用分层的CNN来重新提取特征，相比直接使用CSV的现有特征，
- 应用小样本检测算法，可以提高检测未知类的精度。

思考

- 数据转换中，既然IP和Mac地址都转换成了0.0.0.0和00.00.00.00.00.00。已经没有了任何的信息，为何不选择直接进行移除，直接删除该字段可以减少数据包的长度，缩短token的长度。
- 在数据转化过程中，按照每个字节转化成0-255的像素值的话，有的字段是4个字节或者2个字节，按照单个字节转换的话，就会丧失该字段的语义信息。