需求三

**搭建深度学习平台Keras(基于TensorFlow)，根据需求二分析的特征字段将需求一提供的训练数据进行预处理和归一化操作。结合人工神经网络算法LSTM，进行攻击识别的训练。再将需求一提供的实际数据进行预处理和归一化操作后导入识别模型得到识别结果，并将结果输出至ELK日志分析平台进行日志分析，通过Kibana的可视化界面进行识别结果的展示。**

需求3.1：搭建深度学习平台Keras

先在Centos7系统的GPU上使用anaconda安装深度学习平台TensorFlow，再搭建Keras，用于之后恶意攻击识别模型的训练、检测。

一、基于GPU搭建深度学习平台TensorFlow

1、安装python3

tensorflow目前支持python3.5版本，注意查看版本。如果本机安装了python2，尽量不要管他，使用python3运行python脚本就好，因为可能有程序依赖目前的python2环境。

1）安装依赖环境

# yum -y install zlib-devel bzip2-devel openssl-devel ncurses-devel sqlite-devel readline-devel tk-devel gdbm-devel db4-devel libpcap-devel xz-devel

2）下载Python3，选择合适版本

https:// www.python.org/ftp/python/

3）安装python3

创建新的路径安装python

# mkdir -p /usr/local/python3

解压下载好的python

# tar -zxf Python-3.5.2.tgz

4）进入解压路径，编译安装

# cd Python-3.5.2/

# ./configure --prefix=/usr/local/python3

# make

# make install

5）建立软链接vim

# ln -s /usr/local/python3/bin/python3 /usr/bin/python3

6）添加环境变量

# vim ~/.bash\_profile

# .bash\_profile

# Get the aliases and functions

if [ -f ~/.bashrc ]; then

. ~/.bashrc

fi

# User specific environment and startup programs

PATH=$PATH:$HOME/bin:/usr/local/python3/bin

export PATH

编译环境变量

# source ~/.bash\_profile

检查python3及pip3是否正常使用

# python -V

Python 3.5.2 :: Anaconda 4.2.0 (64-bit)

# pip -V

pip 8.1.2 from /root/anaconda3/lib/python3.5/site-packages (python 3.5)

2、安装pip以及setuptools

1）安装setuptools

#wget--no-check-certificate https://pypi.python.org/packages/source/s/setuptools/setuptools-19.6.tar.gz#md5=c607dd118eae682c44ed146367a17e26

# tar -zxvf setuptools-19.6.tar.gz

# cd setuptools-19.6

# python3 setup.py build

# python3 setup.py install

2）安装pip

https://pypi.python.org/pypi/pip/

下载对应的安装包，版本需要在8.0版本以上

# tar -zxvf pip-8.0.2.tar.gz

# cd pip-8.0.2

# python3 setup.py build

# python3 setup.py install

3、安装anaconda

1）下载安装脚本

https://www.continuum.io/downloads，选择要安装的Python版本和系统架构：

# Python 3.5

$ wget https://repo.continuum.io/archive/Anaconda3-4.2.0-Linux-x86\_64.sh # 64位系统

$ wget https://repo.continuum.io/archive/Anaconda3-4.2.0-Linux-x86.sh # 32位系统

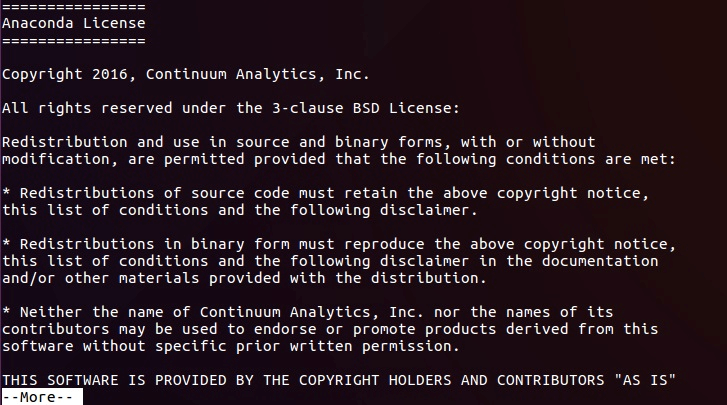
# Python 2.7

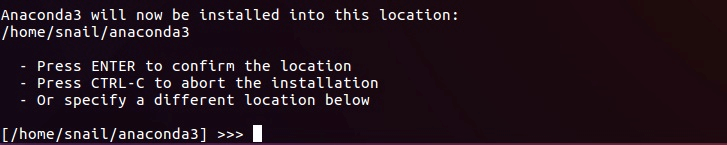
$ wget https://repo.continuum.io/archive/Anaconda2-4.2.0-Linux-x86\_64.sh # 64位系统

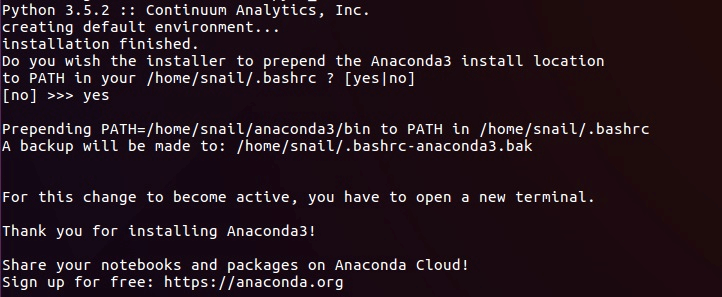
$ wget https://repo.continuum.io/archive/Anaconda2-4.2.0-Linux-x86.sh # 32位系统

2）运行安装向导

bash Anaconda3-4.2.0-Linux-x86\_64.sh







3）测试安装

# conda --version

conda 4.2.9

4、安装tensorflow

1、搭建tensorflow运算环境

# Python 2.7

$ conda create -n tensorflow python=2.7

# Python 3.5

$ conda create -n tensorflow python=3.5

2、激活tensorflow计算环境

#激活

$ source activate tensorflow

#关闭

$ source deactivate tensorflow

3、用pip安装tensorflow

下载地址：https://storage.googleapis.com/tensorflow/linux/cpu/tensorflow-1.4.0-cp35-cp35m-linux\_x86\_64.whl

(tensorflow)$ pip install --ignore-installed --upgrade tensorflow-1.4.0-cp35-cp35m-linux\_x86\_64.whl

4、打开python的shell，测试可否导入tensorflow

(tensorflow) [root@bogon ~]# python3

Python 3.5.4 |Continuum Analytics, Inc.| (default, Aug 14 2017, 13:26:58)

[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux

Type "help", "copyright", "credits" or "license" for more information.

>>> import tensorflow

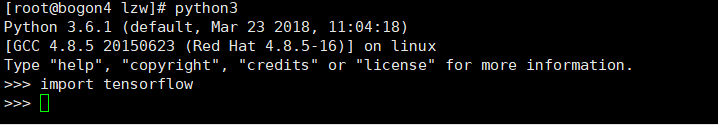
>>> exit()

5、conda安装spyder，同步python

(tensorflow) [root@bogon ~]# conda install spyder

6、打开spyder

(tensorflow) [root@bogon ~]# spyder

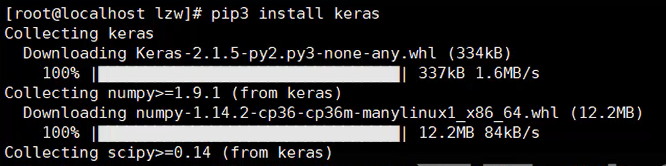


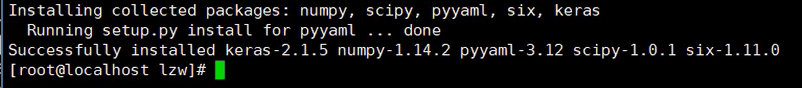
基于GPU的Tensorflow搭建完成。

二、基于GPU的Keras安装

1.执行指令以下指令,即可完成Keras的搭建

pip3 install keras





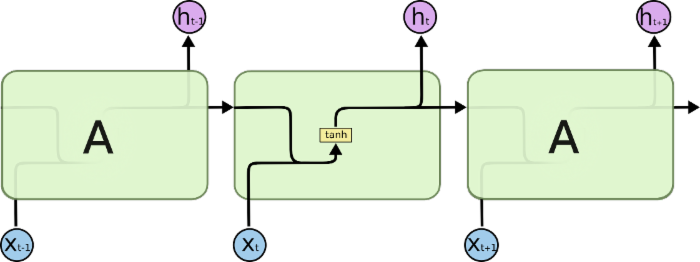
需求3.2：选择人工神经网络算法LSTM算法

在比较多种人工神经网络算法的优缺点之后，选择LSTM算法算法作为整个深度学习平台的核心，用于训练识别模型并最终实现对实际数据的识别检验。

**一．长短时记忆神经网络（LSTM算法）的介绍**

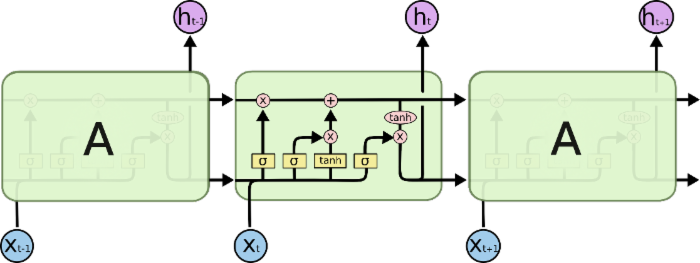
1. LSTM算法原理：

LSTM算法是一种特别的 RNN算法，比标准的 RNN算法在很多的任务上都表现得更好。LSTM算法通过刻意的设计来避免长期依赖问题。记住长期的信息在实践中是 LSTM算法的默认行为，而非需要付出很大代价才能获得的能力，所有 RNN算法都具有一种重复神经网络模块的链式的形式。在标准的 RNN算法中，这个重复的模块只有一个非常简单的结构，例如一个 tanh 层。



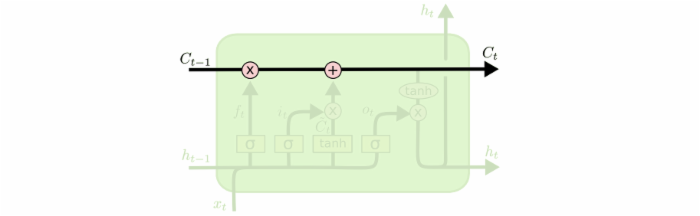
标准 RNN 中的重复模块包含单一的层

LSTM算法同样是这样的结构，但是重复的模块拥有一个不同的结构。不同于单一神经网络层，如下图所示是有四个，以一种非常特殊的方式进行交互。

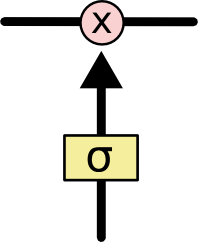


LSTM算法中的重复模块包含四个交互的层

1. LSTM算法的核心思想:

 LSTM算法的关键就是细胞状态，水平线在图上方贯穿运行。细胞状态类似于传送带。直接在整个链上运行，只有一些少量的线性交互。信息在上面流传保持不变会很容易。

LSTM算法有通过精心设计的称作为“门”的结构来去除或者增加信息到细胞状态的能力。门是一种让信息选择式通过的方法。他们包含一个 sigmoid 神经网络层和一个 pointwise 乘法操作。



Sigmoid 层输出 0 到 1 之间的数值，描述每个部分有多少量可以通过。0 代表“不许任何量通过”，1 就指“允许任意量通过”。LSTM算法拥有三个门，来保护和控制细胞状态。

1. LSTM算法的优点：

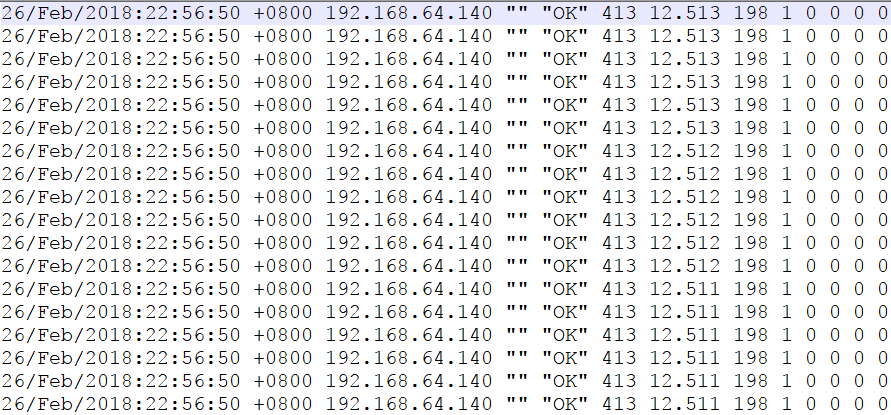
LSTM是RNN的一个优秀的变种模型，继承了大部分RNN模型的特性，同时解决了梯度反传过程由于逐步缩减而产生的Vanishing Gradient问题。LSTM在长远的更为复杂的任务上的潜力是CNN无法媲美的。它更真实地表征或模拟了人类行为、逻辑发展和神经组织的认知过程。它在序列建模上很强大，能够capture长远的上下文信息，此外还具备神经网络拟合非线性的能力。

需求3.3：数据预处理操作

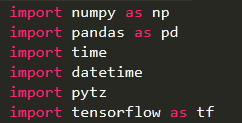
获取通过logstash统一数据格式的规格化数据，使用python第三方库（numpy、pandas等）对其进行解析，生成深度学习平台Keras可以读入的张量形式，从而进行可靠的深度学习。

一、数据预处理

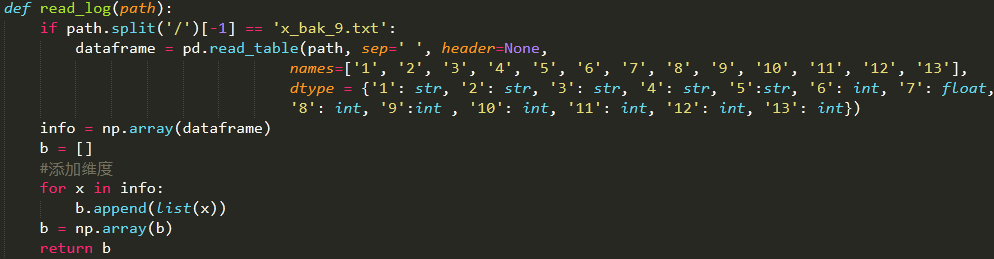
经过前期的数据采集、数据过滤、数据清洗等步骤，已经的得到了需要训练的数据集。数据的形式如下图所示：



根据七种维度不同的特征，将日志数据转化为tensoflow平台可以识别的矩阵形式。首先导入需要使用的第三方库：

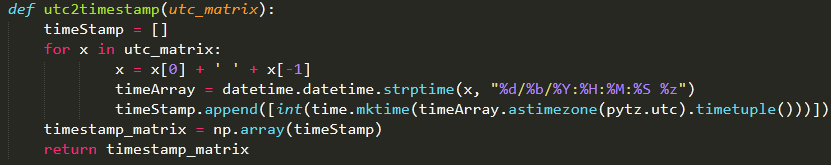


1、读取数据--def read\_log(path):



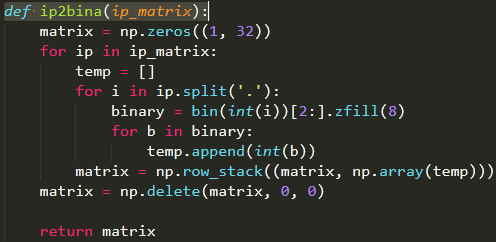
使用pandas库对文本数据进行格式化，将数据转化为n行13列的矩阵（包含5列特征值）。

2、日志时间转化为timestamp--def utc2timestamp(utc\_matrix):



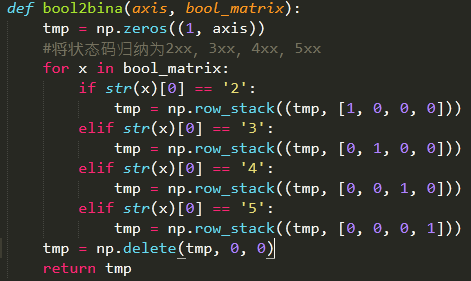
遍历每一条时间，匹配正则表达式“%d/%b/%Y:%H:%M:%S %z”，并调用pytz来实现标准时间的转化。

3、ip从点分十进制转化为二进制--def ip2bina(ip\_matrix):



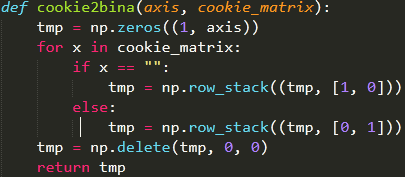
调用numpy库和python内置的bin()函数，实现将十进制转化为二进制并聚合成新矩阵。

4、将状态码转化为one-hot形式--def bool2bina(axis, bool\_matrix)



网页状态码可以分为2xx，3xx，4xx，5xx四种大类。识别四种情况转化为one-hot并返回。

5、将cookies值转化为二进制，判断有无--def cookie2bina(axis, cookie\_matrix)



在攻击情况下，cookie值一般为空，因此，只需要判断cookie值是否存在。One-hot两个维度即可。

6、将返回状态转化为二进制，表示有无--def completion2bina(axis, compt\_matrix)



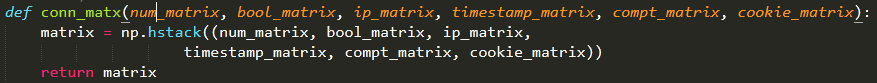
原理同上，慢链接情况下，$request\_completion为空，one-hot为两个维度。

7、数值数据归一化--def normalize(num\_matrix)：



使用线性函数归一化(Min-Max scaling)，将数值数据归一化到0-1之间。新数据=（原数据-极小值）/（极大值-极小值）。

8、矩阵拼接，对预处理数据进行合并--def conn\_matx(num\_matrix, bool\_matrix, ip\_matrix, timestamp\_matrix, compt\_matrix, cookie\_matrix):



将规格化的数据合并，整合成一张包含所有数据元素的矩阵。

9、调整矩阵维度，使之符合TensorFlow输入形式

C:\Users\admin\AppData\Roaming\Tencent\Users\576428168\QQ\WinTemp\RichOle\K3S_XM7}O_YGEO0GBQ7VLDM.png

需求3.4：识别模型训练、优化、检验

将归一化训练数据结合人工神经网络算法LSTM在GPU上进行模型训练，通过Tensorboard实现神经网络可视化，通过Matplotlib查看训练过程，检验神经网络的性能。得到训练模型后再进行调参、优化，提高模型识别正确率。最终将需求一提供的实际数据在经过规格化、归一化处理后导入训练模型进行识别检测。

一．LSTM算法神经网络训练--LSTM算法(data, target, num\_hidden)

1、建立序贯模型（Sequence）

C:\Users\admin\AppData\Roaming\Tencent\Users\576428168\QQ\WinTemp\RichOle\P4OO2}EBWIRYLL6~B$91X]H.png

2、添加LSTM层（调用LSTM函数）

C:\Users\admin\AppData\Roaming\Tencent\Users\576428168\QQ\WinTemp\RichOle\2SP}F}D0WU2E(98]J(QV51J.png

64 🡪 units：输出维度

0.5 🡪dropput的比率，防止过拟合

Input\_shape 🡪输入数据的维度

3、添加全连接层

C:\Users\admin\AppData\Roaming\Tencent\Users\576428168\QQ\WinTemp\RichOle\HX`0ZR9@)SOX`QV}[OQC0]W.png

5 🡪 输出数据的维度

Sigmoid 🡪 激活函数

4、编译模型

C:\Users\admin\AppData\Roaming\Tencent\Users\576428168\QQ\WinTemp\RichOle\QOAPYRV9D$U$`6P@V)`@O@T.png

binary\_crossentropy 🡪 损失函数

adam 🡪 优化器

[‘accuracy’] 🡪 指标列表

5、训练模型

C:\Users\admin\AppData\Roaming\Tencent\Users\576428168\QQ\WinTemp\RichOle\{52(M(I~R2HJ0IE9@HNTXVK.png

Validation\_split 🡪 在没有提供验证集的时候，按一定比例从训练集中取出一部分作为验证集。

Epoch 🡪 训练遍数

Batch\_size 🡪 批数量

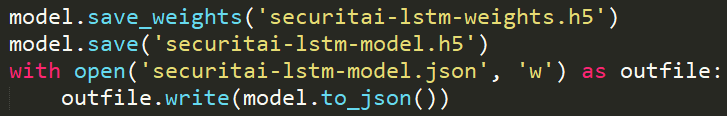
Callback 🡪 接入tensorboard函数

6、评估模型，计算正确率

C:\Users\admin\AppData\Roaming\Tencent\Users\576428168\QQ\WinTemp\RichOle\CHF3VPU_N`K@K(V@}(BM0KQ.png

Verbose 🡪 日志显示，0为不在标准输出流输出日志信息，1为输出进度条记录，2为每个epoch输出一行记录

7、保存模型



二、LSTM算法神经网络训练

1、加载模型

C:\Users\admin\AppData\Roaming\Tencent\Users\576428168\QQ\WinTemp\RichOle\7@)O@_IH[QVE}BF{QR_VDDG.png

2、编译模型

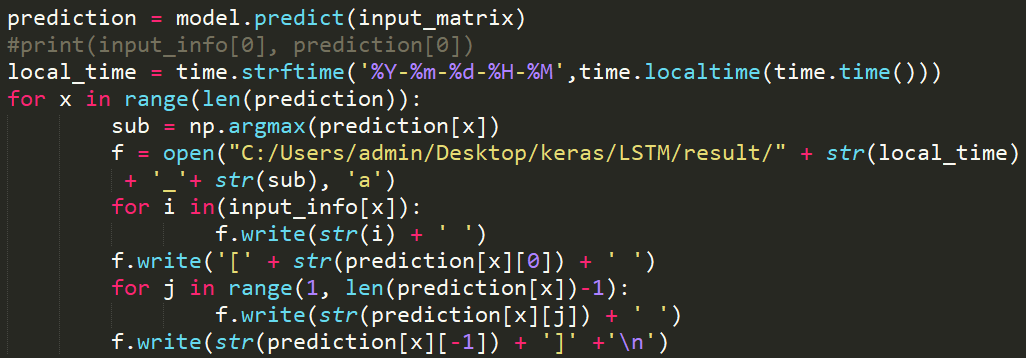
C:\Users\admin\AppData\Roaming\Tencent\Users\576428168\QQ\WinTemp\RichOle\`E)(~KEOEQ5E3SNUCZS($H7.png

参数同训练阶段

3、导入数据预测（数据预处理方式一致）

C:\Users\admin\AppData\Roaming\Tencent\Users\576428168\QQ\WinTemp\RichOle\48(E`J8JBK~BG33T7B7K3MT.png

4、按照格式（包括原日志数据）输出预测结果

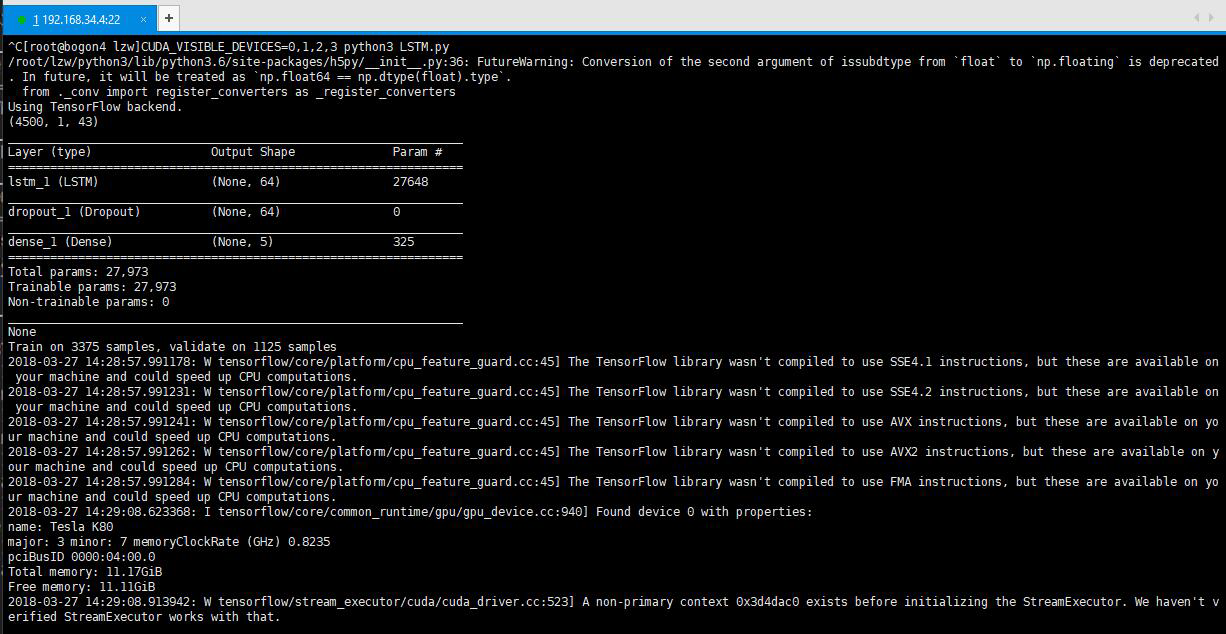


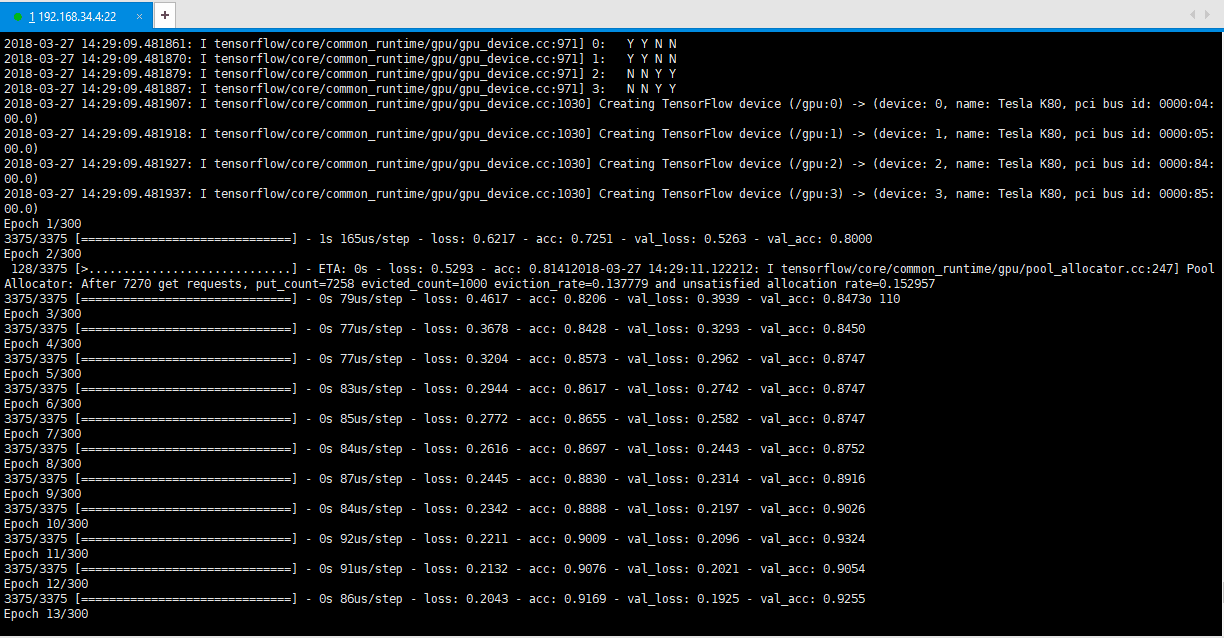
三．运行人工神经网络—GPU

1、训练数据集

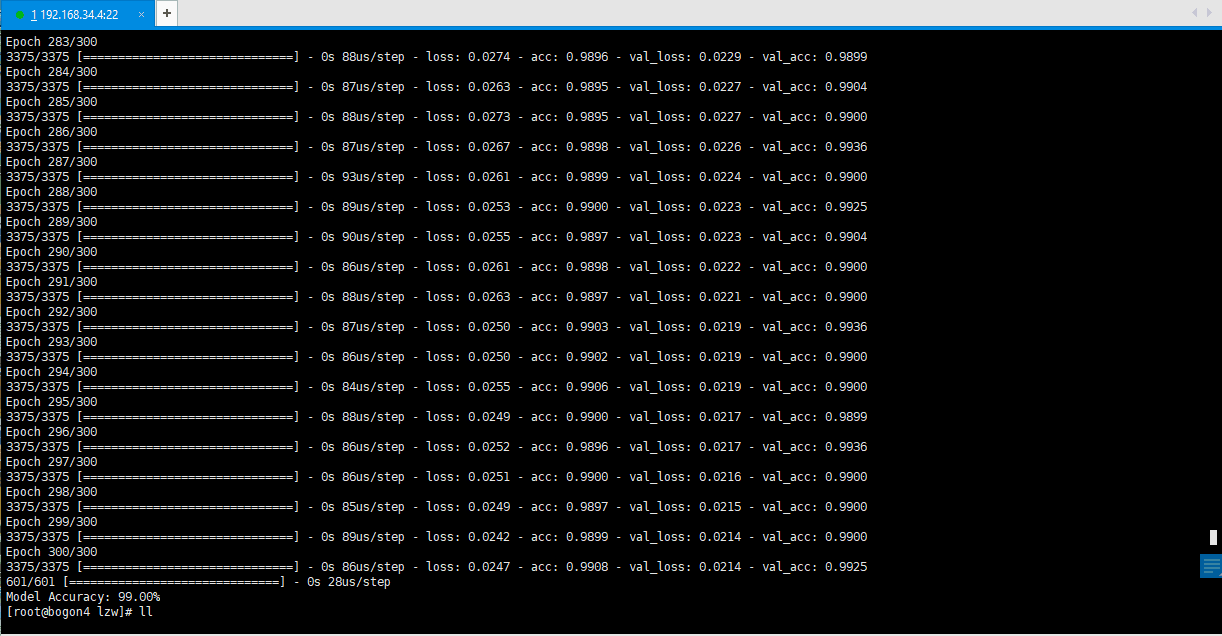
调用GPU训练人工神经网络

[root@localhost data]# CUDA\_VISIBLE\_DEVICES=0,1,2,3 python3 LSTM.py



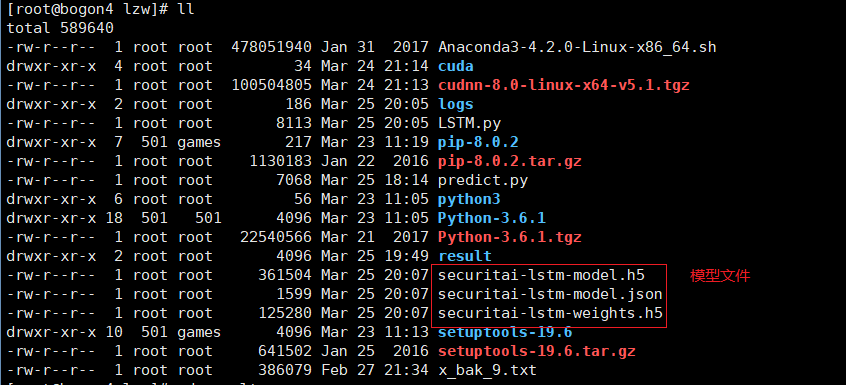


可以得到最后训练300遍的结果：



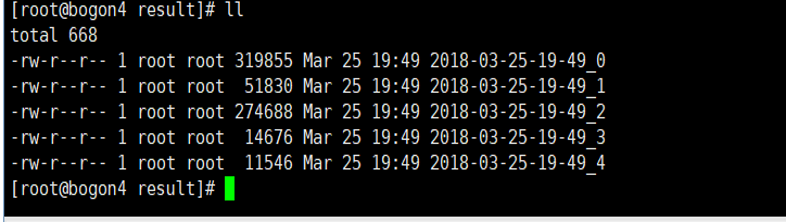
在最后可以看出LSTM算法训练最终的正确率为：99.00 %

在设置的训练模型存放路径下，显示出现的模型文件：

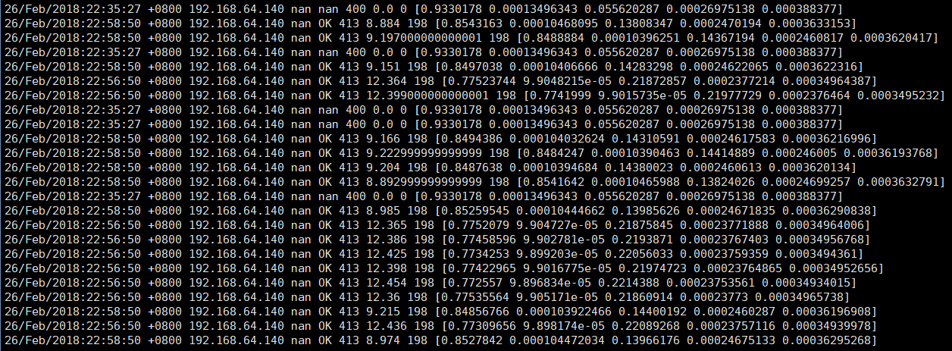


2、预测

使用实际数据做测试。最后得出的预测结果按照—“时间\_结果”的文件名存放在目录下。例如：“2018-03-25-19-49\_0”



文件中存放每条数据的预测结果。如下所示的就表明[1,0,0,0,0]的可能性最高。



四．神经网络可视化—Tensorboard

1、添加Tensorboard层

C:\Users\admin\AppData\Roaming\Tencent\Users\576428168\QQ\WinTemp\RichOle\3FT)[M1U]0)~94XWE@WW(E2.png

log\_dir 🡪 Tensorboard日志文件输出路径

2、Tensorboard对人工神经网络的展示

在命令行中输入如下命令：

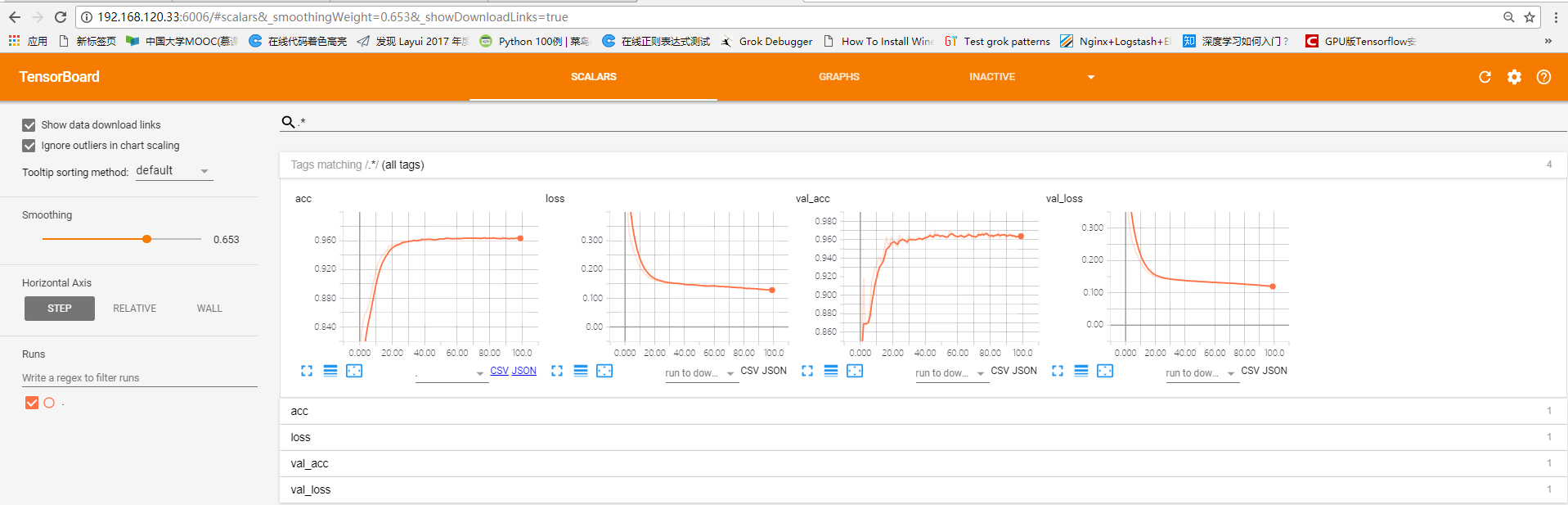
Log路径为代码中设置的路径。

(tensorflow) [root@localhost data]# tensorboard --logdir='/root/Desktop/data/log'

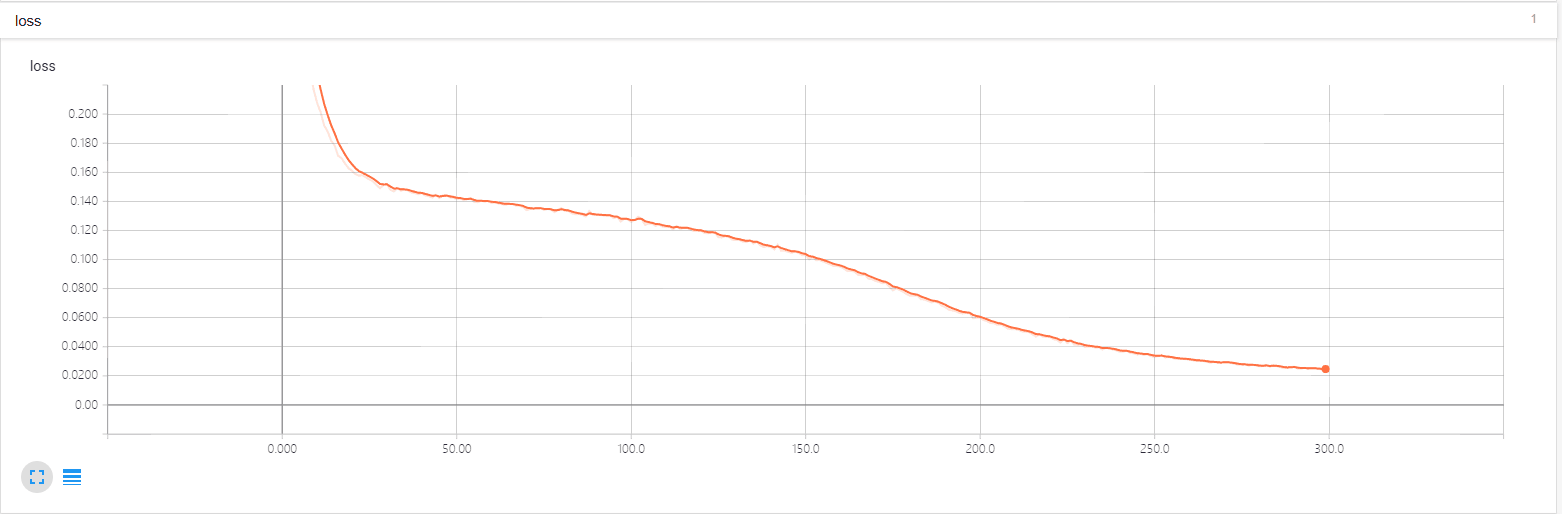
跳出：TensorBoard 0.4.0rc2 at http://localhost.localdomain:6006 (Press CTRL+C to quit)

在浏览器中输ip:6006端口即可进入Tensorboard窗口。

标量展示：



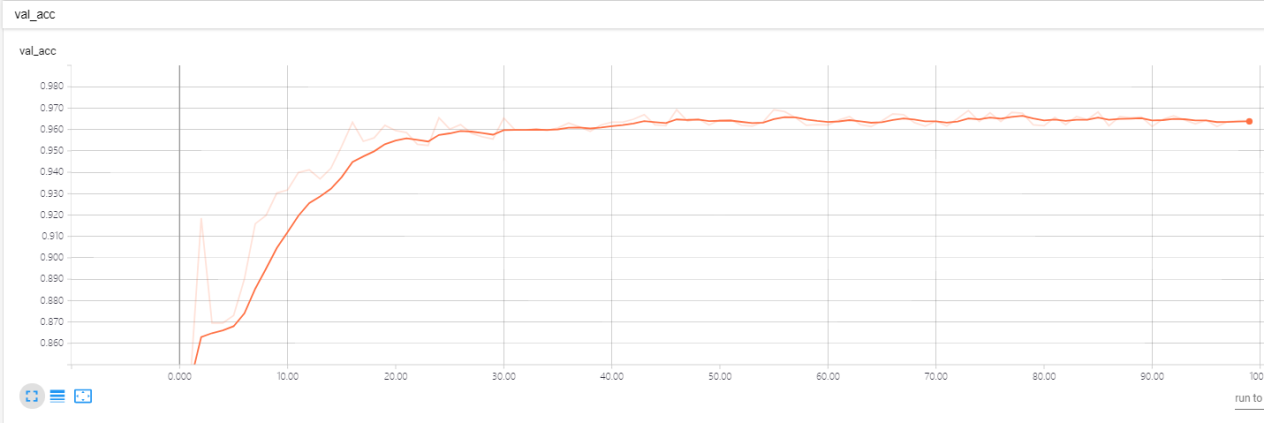
测试集的损失趋势



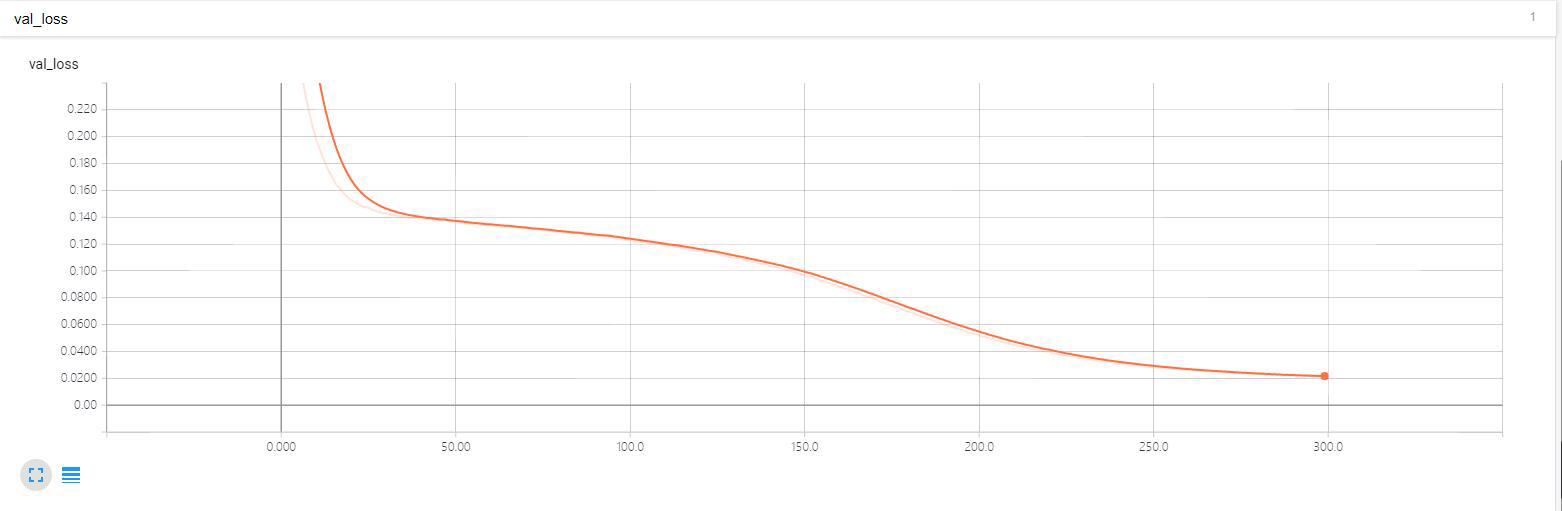
测试集的正确率趋势



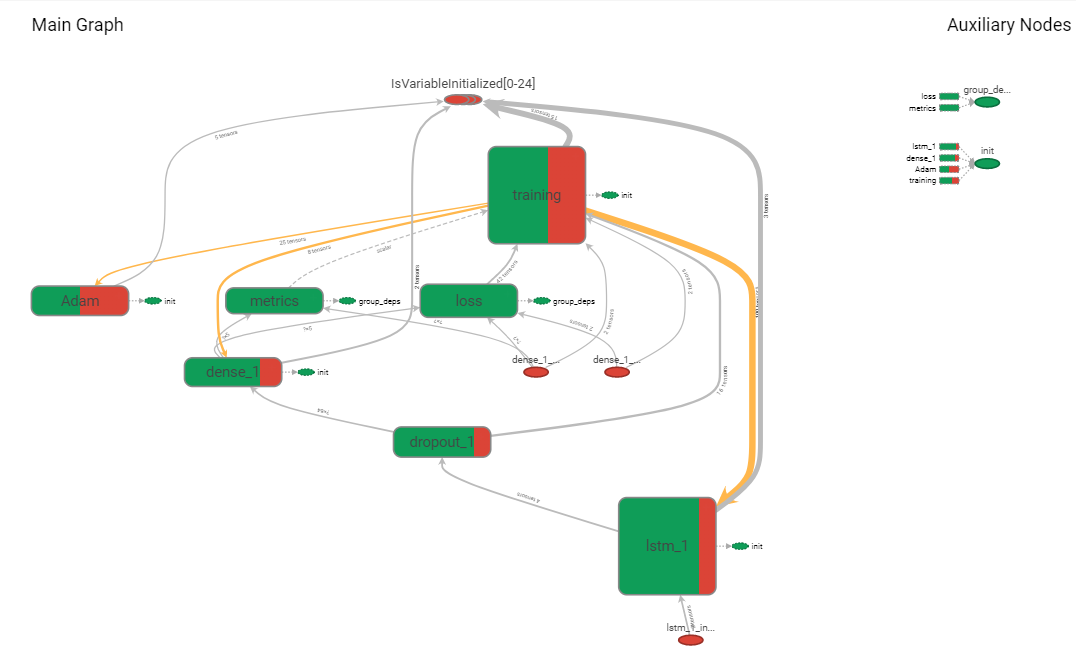
验证集的正确率趋势



验证集的损失趋势



人工神经网络结构图

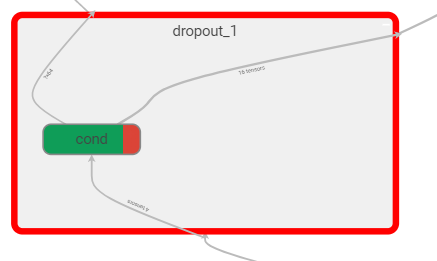


![C:\Users\admin\AppData\Roaming\Tencent\Users\576428168\QQ\WinTemp\RichOle\FSHLT_P($YLLOYQ[8@5](2I.png](data:image/png;base64,)

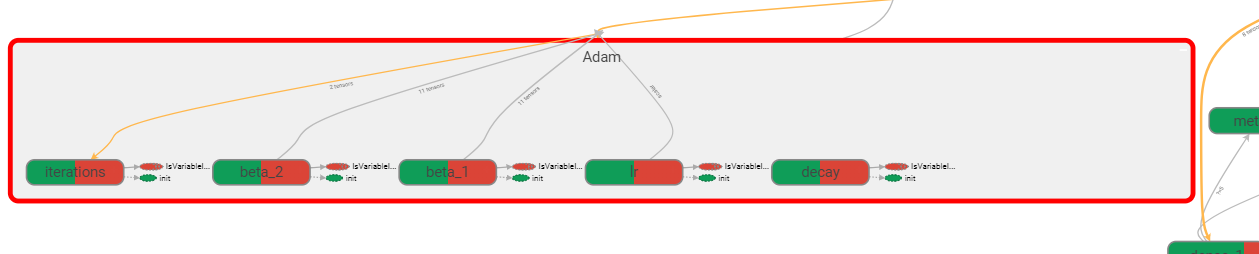
LSTM区域（部分）



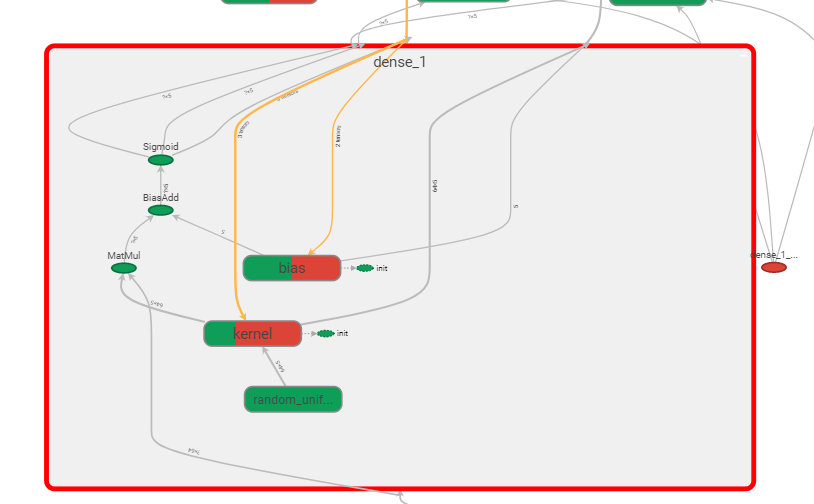
Dropout区域



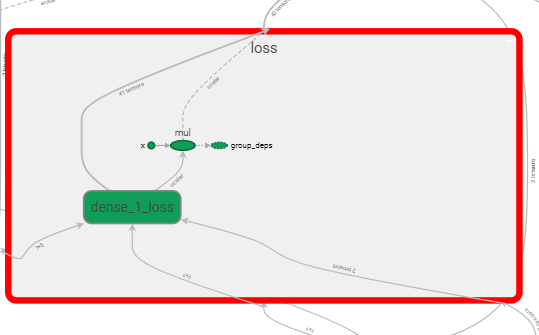
激活函数



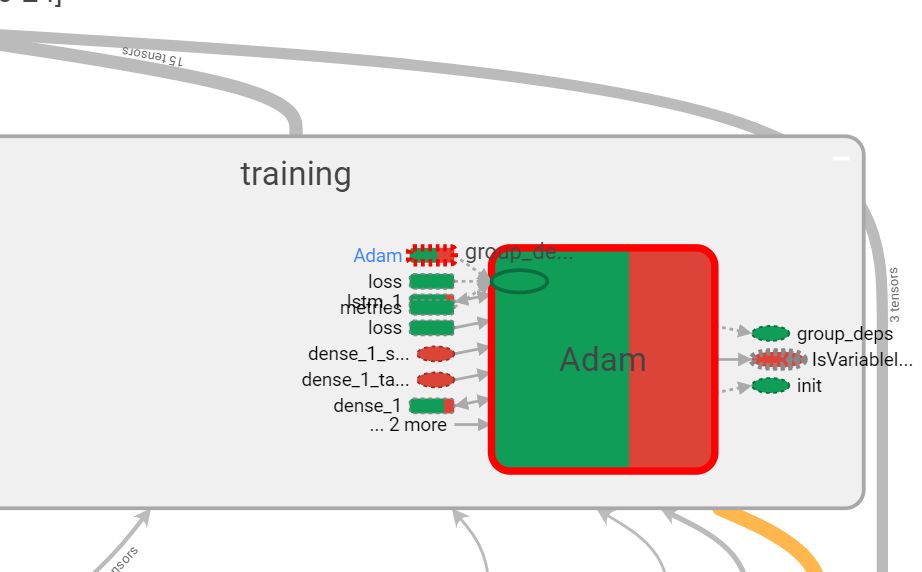
全连接层



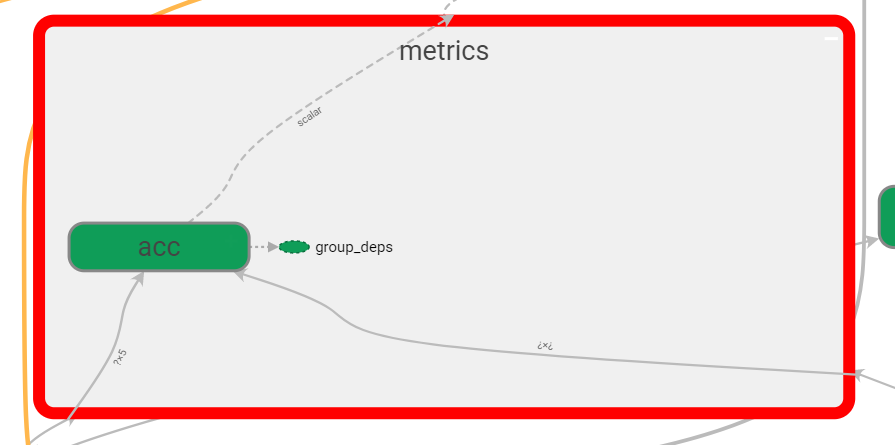
损失函数



训练区域



Metrics区域



五、 高维向量可视化—Matplotlib

高维向量的可视化效果可以直观展示出数据在神经网络训练过程中的整体变化情况，便于我们从图中找到不同数据类之间的相关性(correlation)，聚集(cluster)和异常值(outlier)等。同时，这种可视化也是一种理解和检验神经网络性能的方法，通过将输入和输出数据可视化可以看出神经网络的性能以及可能的问题。

要实现高维向量的可视化，我们需要考虑降低数据的维度，同时丢弃不相关的特征。当数据可以用较少的维度表示而不丢失信息时，就可以对数据绘图，可视化分析它的结构和离群点。

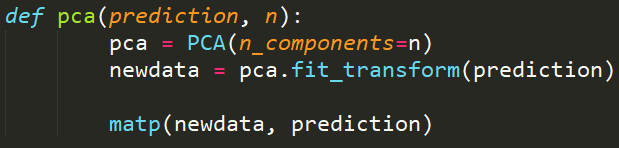
特征降维是指采用一个低纬度的特征来表示高纬度。特征降维一般有两类方法：特征选择（Feature Selection）和特征提取（Feature Extraction）。

1.特征选择是从高纬度的特征中选择其中的一个子集来作为新的特征。最佳子集是以最少的维贡献最大的正确率，丢弃不重要的维，使用合适的误差函数进行，方法包括在向前选择（Forword Selection）和在向后选择（Backward Selection）。

2.特征提取是指将高纬度的特征经过某个函数映射至低纬度作为新的特征。常用的特征抽取方法就是PCA（主成分分析）和LDA（线性判别分析）。

这里我们使用PCA（主成分分析法）来对训练结果降维，从而进行可视化展示。

1、PCA降维



prediction 🡪 预测结果

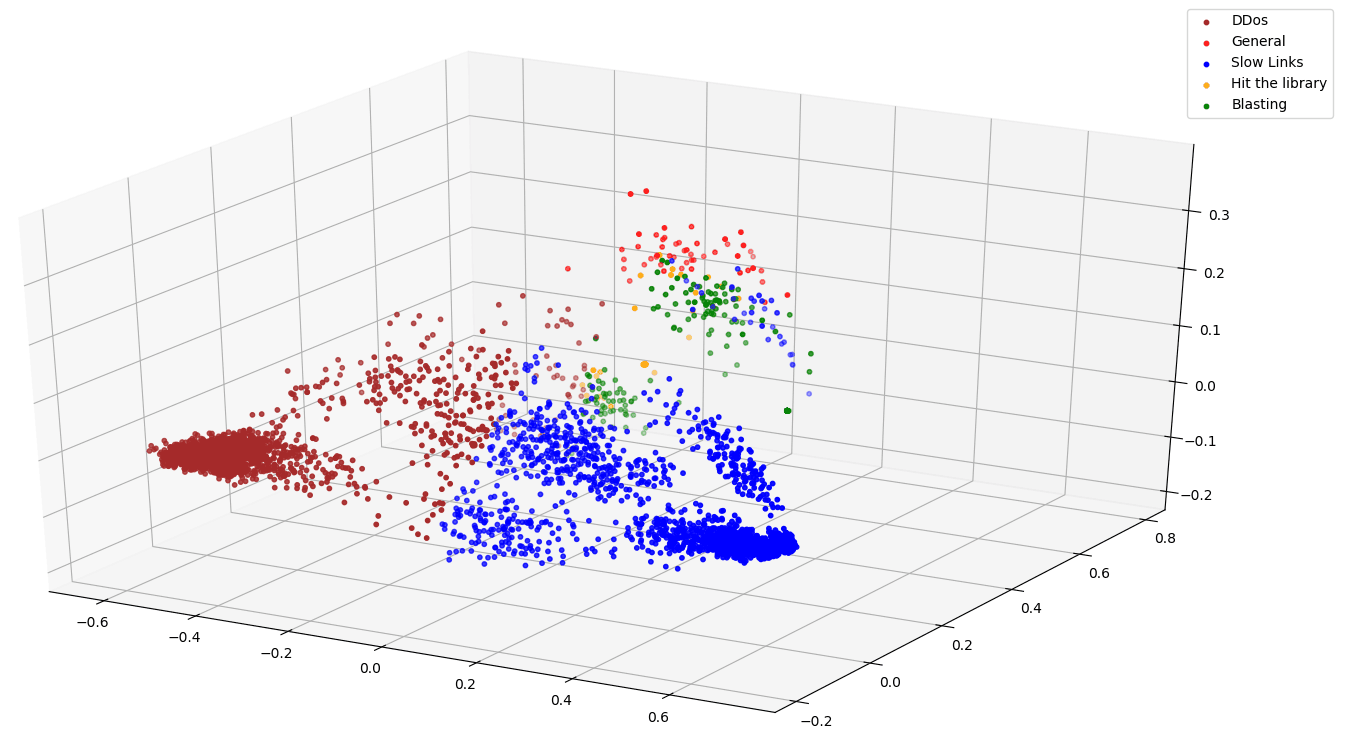
n 🡪 降低的维度（这里降成三维）

2、使用Matplotlib工具绘制预测结果图形

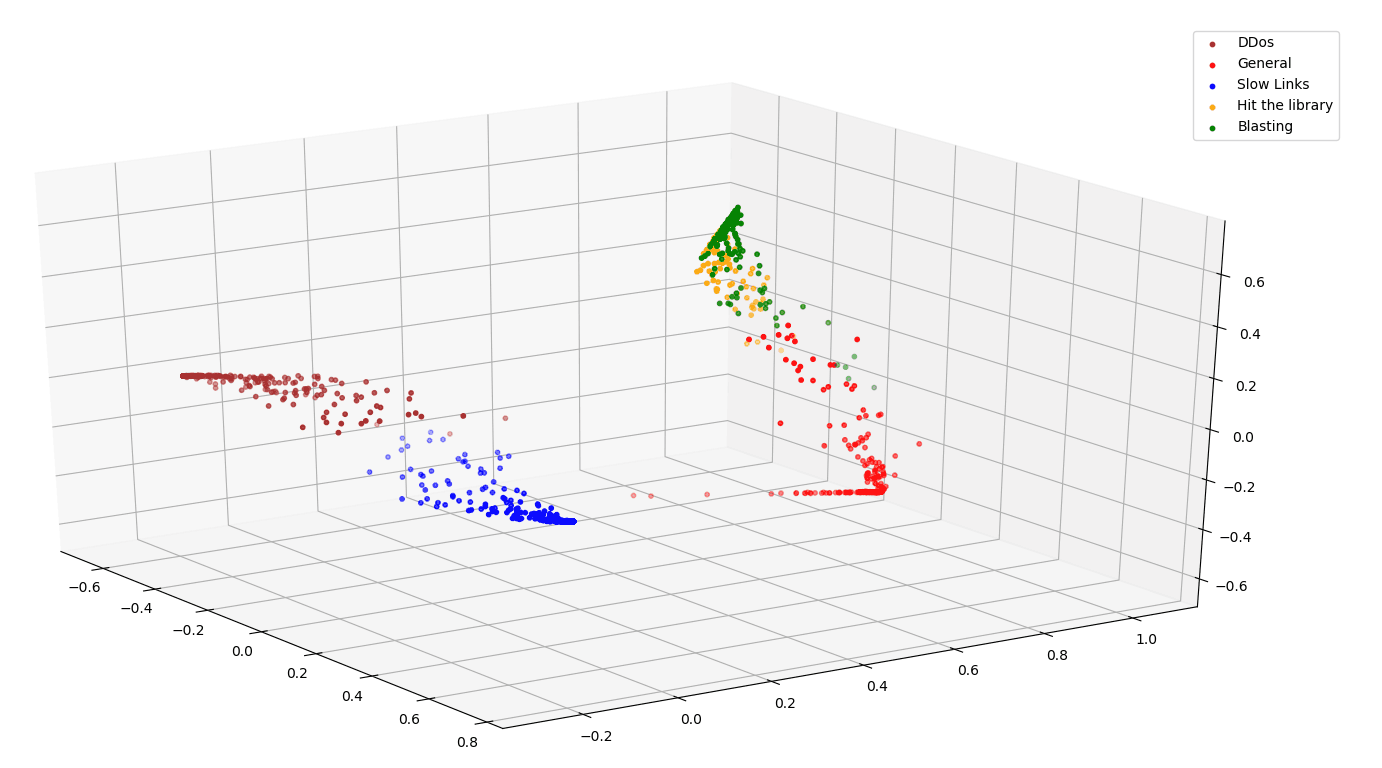


Data 🡪 降维后的数据

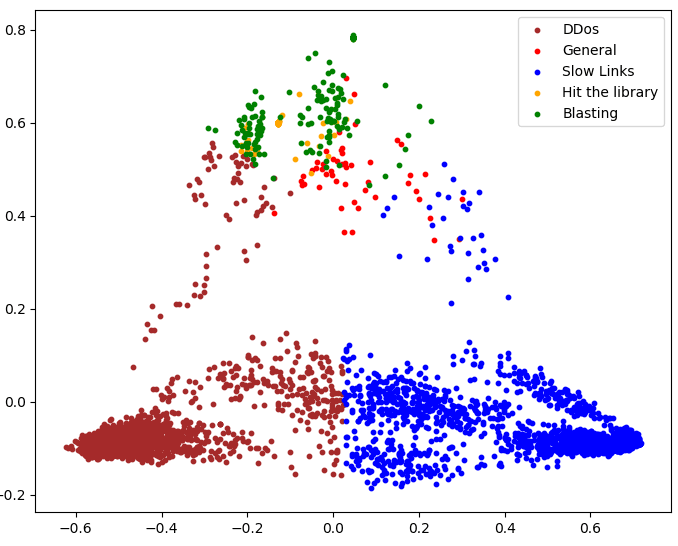
3、Matplotlib可视化结果



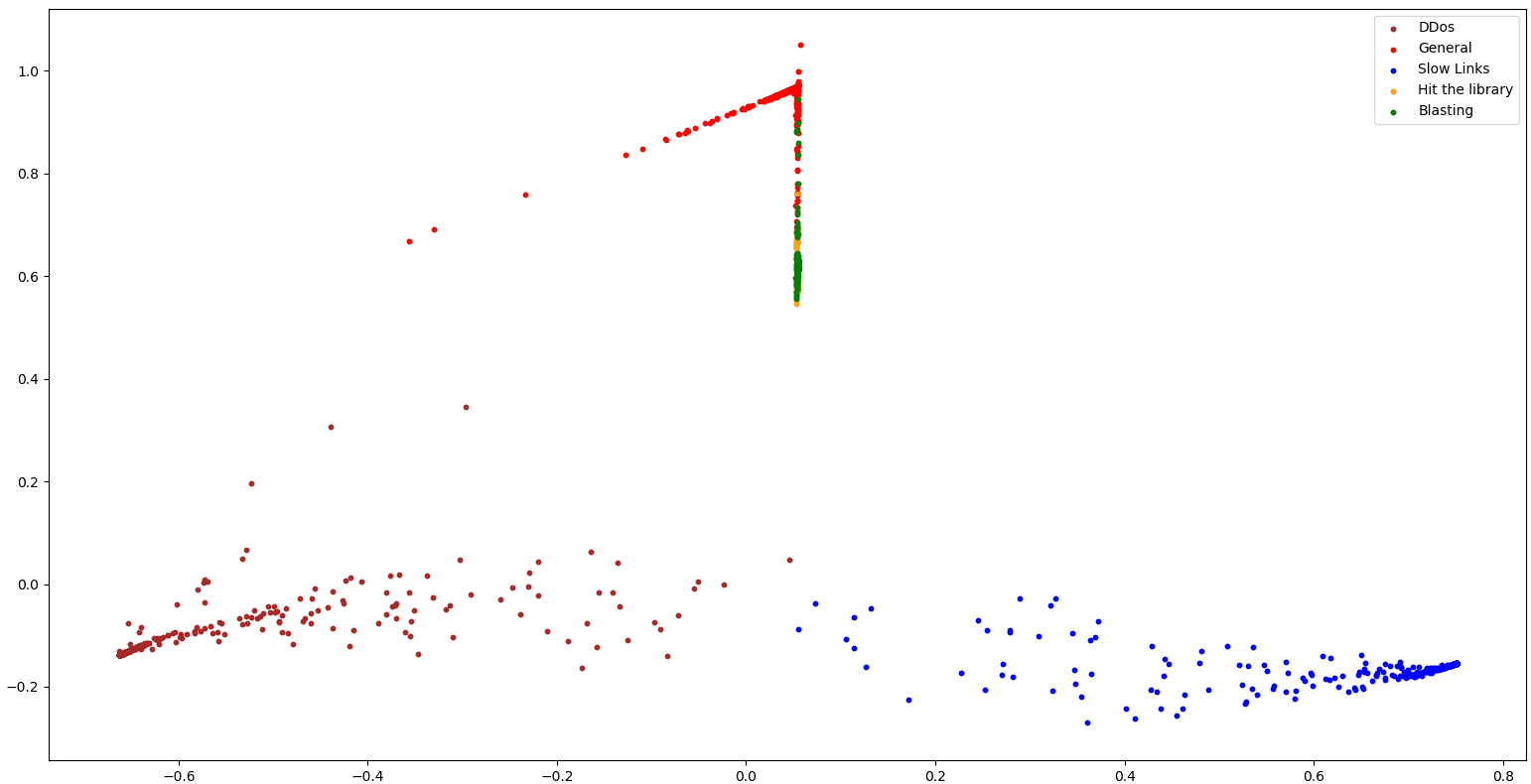
训练12遍时的三维效果图



训练300遍时的三维效果图



训练12遍时的二维效果图



训练300遍时的二维效果图

对比训练12遍和300遍时候的效果图可知，经过了300次的训练，长短时记忆神经网络（LSTM）对原始训练数据进行了学习，数据位置发生了明显的变化。从上图中可以看出，数据的分类逐步从一开始的界限模糊进而到经过训练之后可以明显地分清楚类别。此外，图中还可以看出，慢连接攻击（蓝色点）和DDOS攻击（褐色点）在某一维度上的值很相近，这也说明了慢连接攻击是DDOS攻击的一种。

与此同时，高维向量可视化效果图也足以检验神经网络的性能。通过对比训练12遍和300遍的时候，不同颜色数据点的数量、界限、位置、分布等指标，可以证明该神经网络模型已经能够精确识别出各种攻击。

需求3.5：识别结果可视化展示

将需求3.4输出的识别结果日志导入ELK日志分析平台进行日志分析，通过Kibana多维度可视化界面展示识别结果，识别日志的内容详细、可溯源，可以为之后采取相应防御方式提供依据。

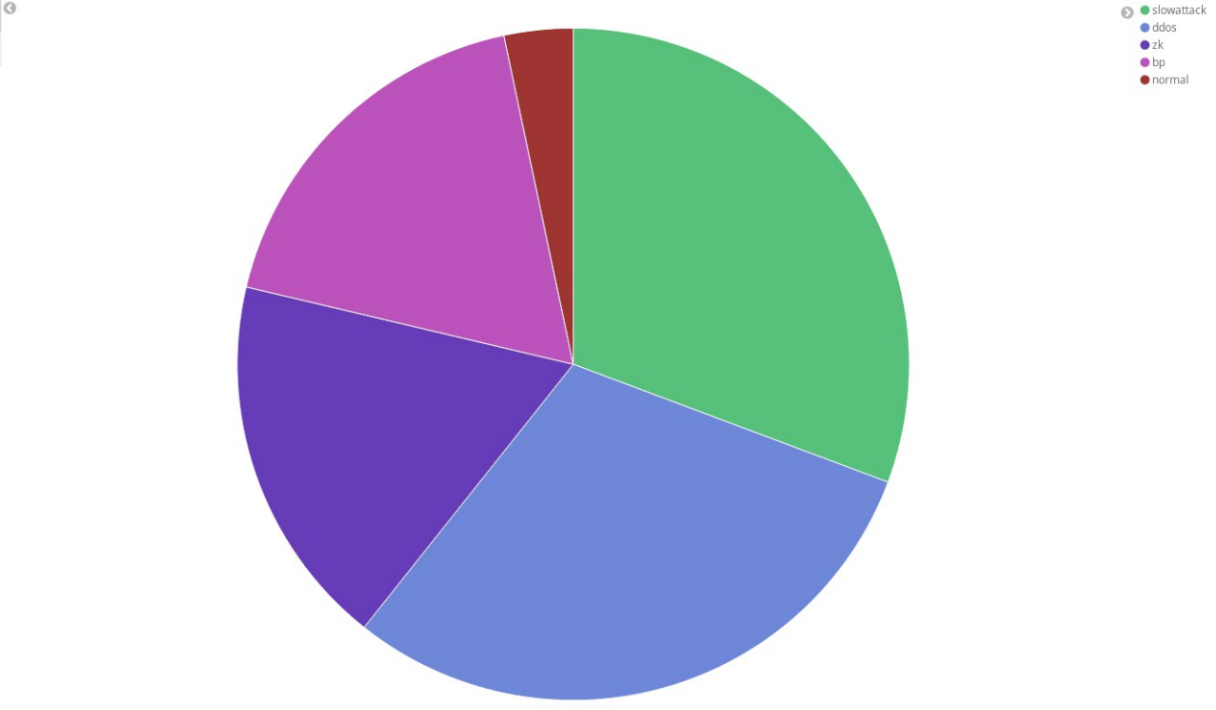
1. 识别结果表

表内列出了攻击日志的IP、识别类型、IP出现次数以及神经网络判定某种攻击的概率。



1. 识别结果饼图

饼图直观展示了识别结果中各种日志的数量比重



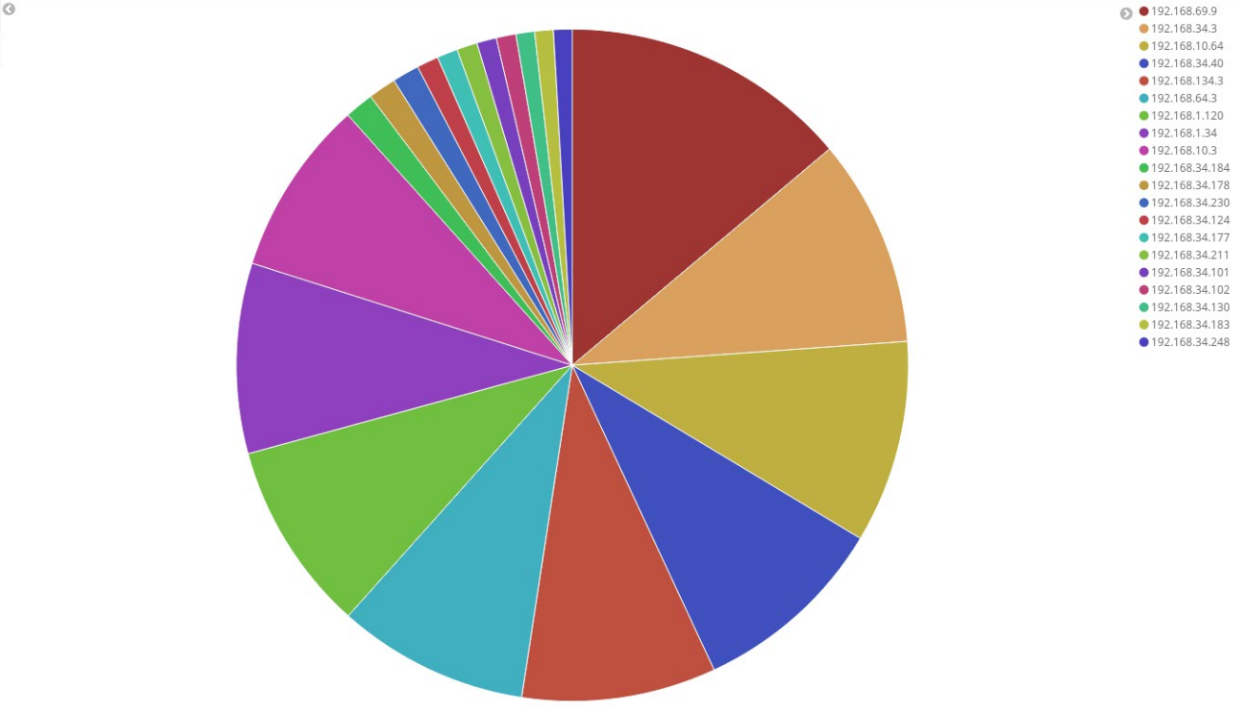
1. 日志信息表

详细展示了所有实际数据的全部日志信息以及识别结果，并切分了特征字段。

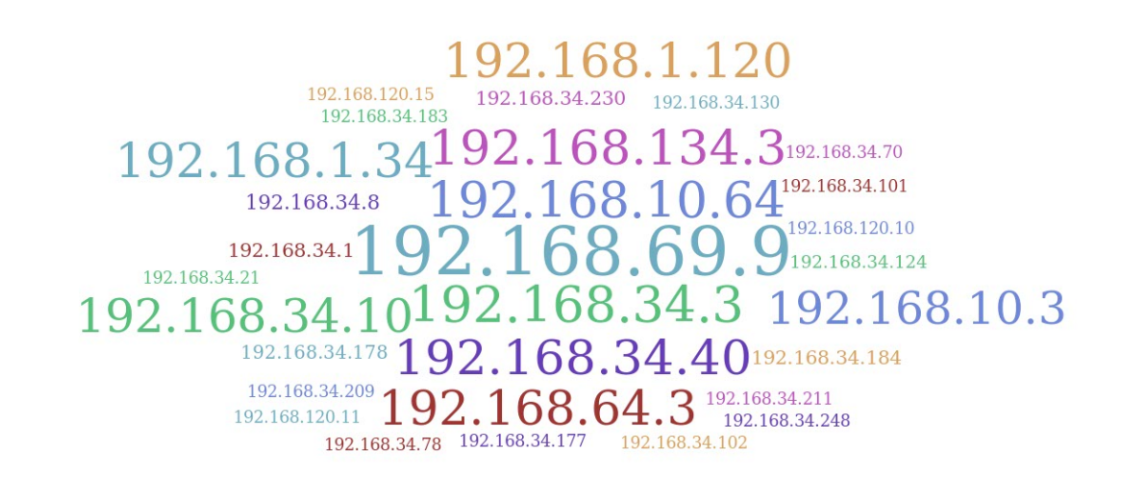


1. IP统计图

饼图、TOP10条形统计图、云图直观展示了IP数量的分布情况，为后续的防御措施如封锁IP提供依据。

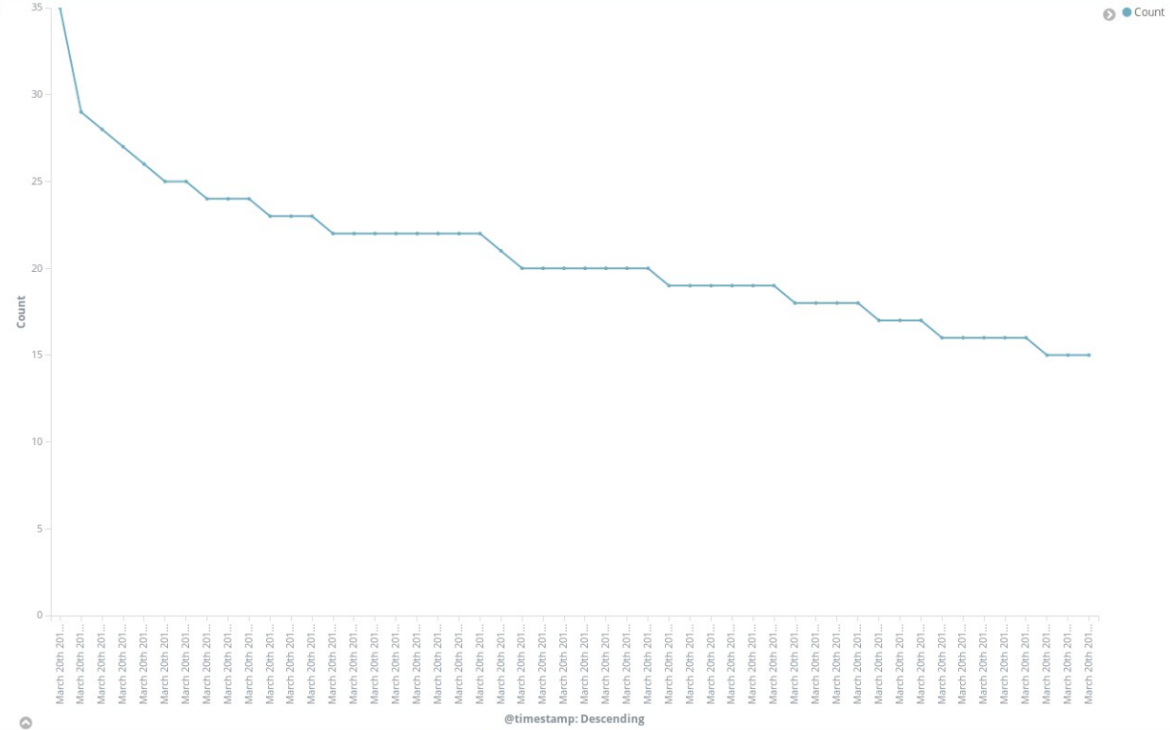






1. 时间折线图

时间折线图可以直观显示发生攻击的时间



6.DashBoard将所有图表汇总，多维度展示识别结果

