# 实验记录

## 1时间：20190601，18：00-？

## 2实验目的：

在探究epoch在50-100-150这些关系下的变化。【实验操作失误，只收集到了50-100这组数据】

## 3实验代码存放文件夹：

20190601中。

## 4实验运行时间：

1天及以上。

## 5实验使用数据信息：

药物作用位点预测，可能存在的药物作用位点标志为1，正样本。其余其他位点均为负样本。消除位于边界的负样本，包含正样本位点的负样本。

## 6.1，数据集正负样本设置，比例设置，

正负样本1：33。采用十则交叉验证模式构建10个数据集的数据，依次进行学习。目前没完成整体评估+存取模型+选取最优模型等步骤

## 6.2，网络设置：

def zyh\_CNN(trainX, trainY,valX=None, valY=None, compiletimes = 0, forkinas=False,transferlayer=1,compilemodels=None,earlystop=None,nb\_epoch=100,batch\_size = 8192,fildername = None ):

"""

:argument

:return:

"""

input\_row = trainX.shape[2]

input\_col = trainX.shape[3]

trainX\_t = trainX;

valX\_t = valX;

#print(input\_row)

#print(input\_col)

trainX\_t.shape = (trainX\_t.shape[0], input\_row, input\_col)

if (valX is not None):

valX\_t.shape = (valX\_t.shape[0], input\_row, input\_col)

#print(valX\_t)

if (earlystop is not None):

early\_stopping = EarlyStopping(monitor='val\_loss', patience=earlystop)

nb\_epoch = 1000; # set to a very big value since earlystop used

if compiletimes == 0:

input = Input(shape=(input\_row, input\_col))

filter1 = 64

filtersize1 = 2

dropout1 = 0.25

L1CNN = 0

nb\_classes = 2

batch\_size = batch\_size

actfun = "relu";

nadam = Nadam(lr=0.00001)

optimization = nadam

dense\_size1 = 128

dense\_size2 = 64

dense\_size3 = 8

dropout\_dense1 = 0.298224

dropout\_dense2 = 0

dropout\_dense3 = 0

input = Input(shape=(input\_row, input\_col))

x = conv.Conv1D(filter1, filtersize1, init='glorot\_normal', W\_regularizer=regularizers.l2(L1CNN),

border\_mode="same")(input)

x = Dropout(dropout1)(x)

x = Activation(actfun)(x)

x = core.Flatten()(x)

output = x

output = Dropout(dropout1)(output)

output = Dense(dense\_size1, init='glorot\_normal', activation='relu')(output)

output = Dropout(dropout\_dense1)(output)

output = Dense(dense\_size2, activation="relu", init='glorot\_normal')(output)

output = Dropout(dropout\_dense2)(output)

output = Dense(dense\_size3, activation="relu", init='glorot\_normal')(output)

output = Dropout(dropout\_dense3)(output)

out = Dense(nb\_classes, init='glorot\_normal', activation='softmax')(output)

cnn = Model(input, out)

cnn.compile(loss='binary\_crossentropy', optimizer=optimization, metrics=['accuracy'])

pass

else:

print("use old net")

cnn = compilemodels

pass‘

之后使用输出模式查看网络。

## 7实验变量：

网络设置中的epoch：分别设置为50,100

## 8实验运行记录

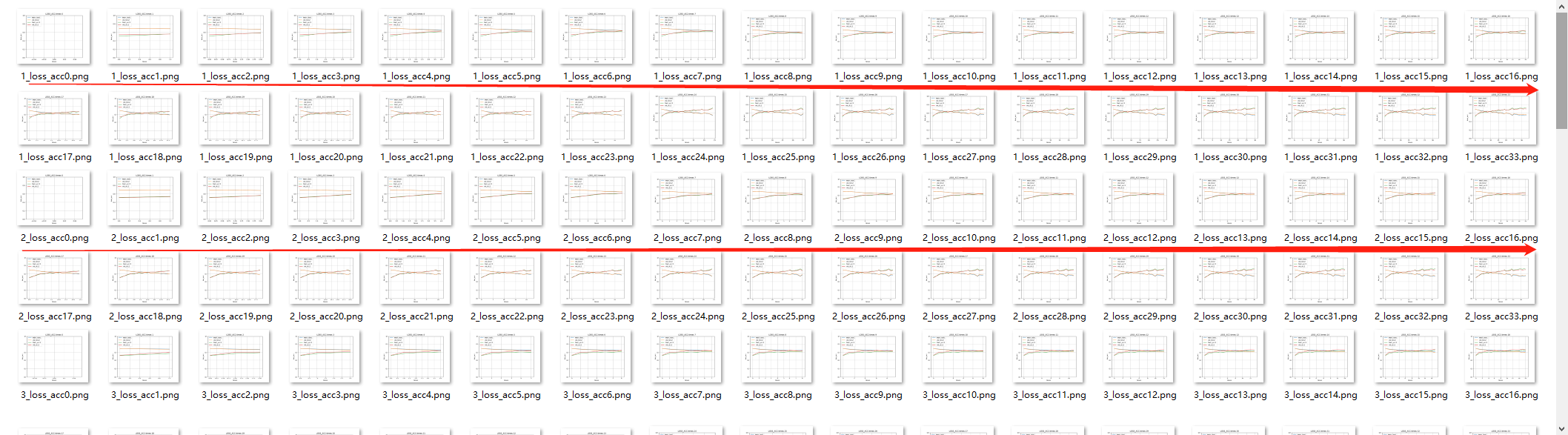
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 开始时间 | 输出文件 | 结束时间 | GPU编号（数量） | 运行id | Epoch |
| 1 | 下午 | 20190601\_01.ou |  | 7 | 18948 | 50 |
| 2 | 下午 | 20190601\_02.ou |  | 6 | 19207 | 100 |
| 3 | 下午 | 20190601\_03.ou |  | 5 | 19589 | 150 |
| 4 | 下午 |  |  | ----- |  |  |

## (好像是)

## 9实验结果存放位置

## 10结果分析

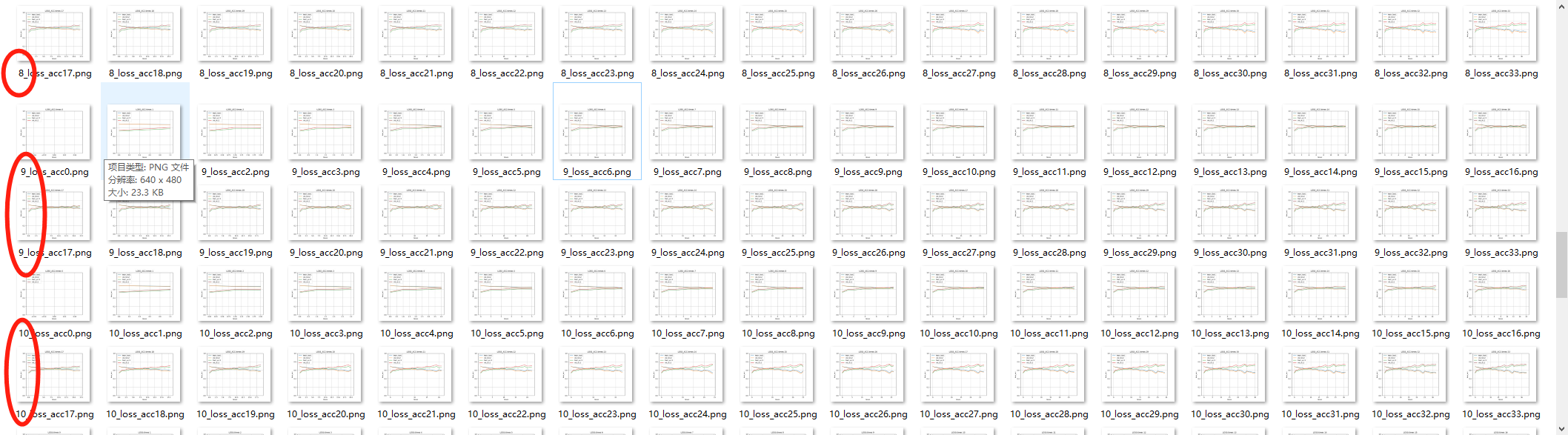
先看第一组实验，其中epoch为50，在第一折数据中，我们根据正样本对34组负样本依次画图，evaluate model得到的数据画图，可以看到这样的结果：



（第一个红色箭头上下两排是第一折的。第二个红色箭头上下两排是第二折的）

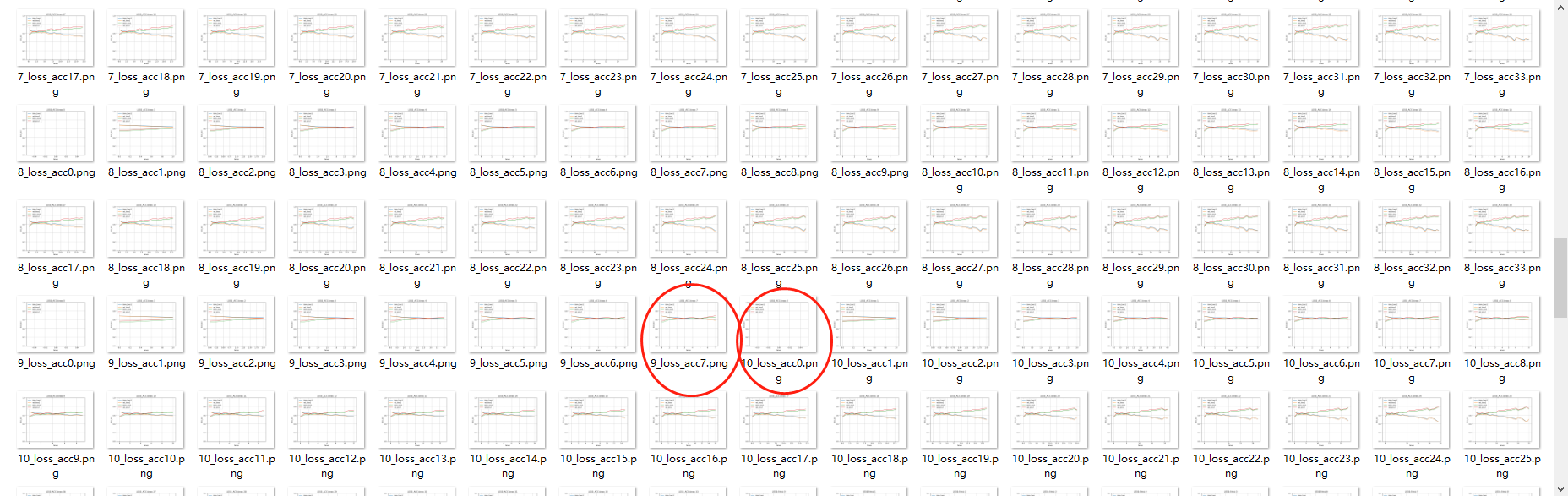
该程序在早上7：16运行结束，昨晚21：00开始运行。大概运行了10个小时（少于十个小时）。

结论：很显然没有训练到最好的情形，epoch在750-1250的时候，acc大概能训练到0.85左右。



（如图展示已经训练到第10折）

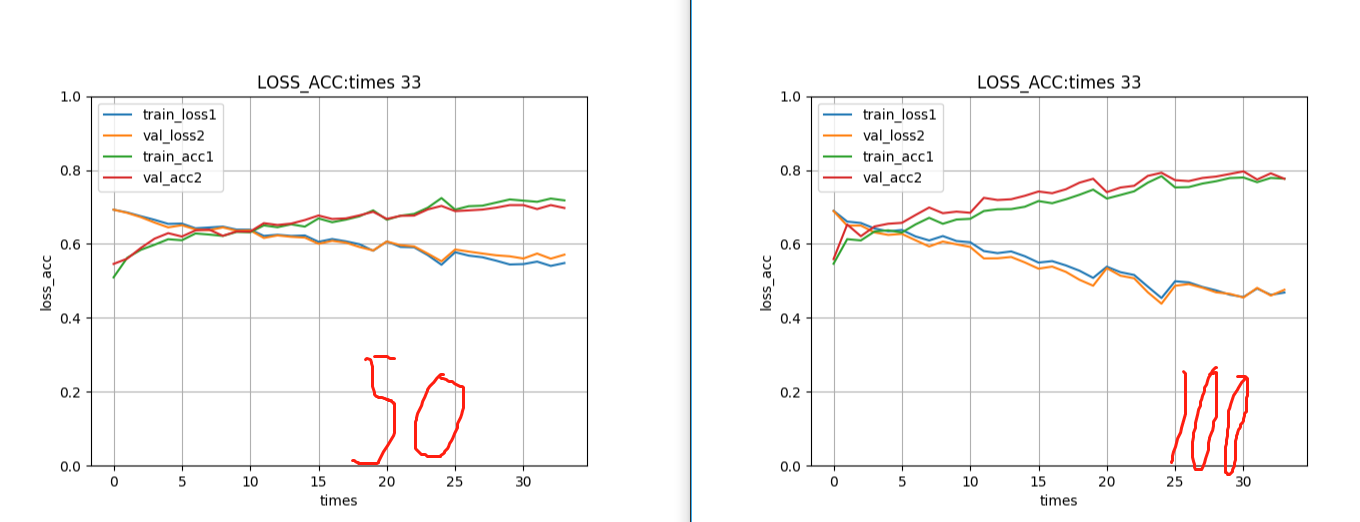
根据十折训练结果可得到：模型对数据有较好的预测效果，而不是仅对单一数据有好结果。



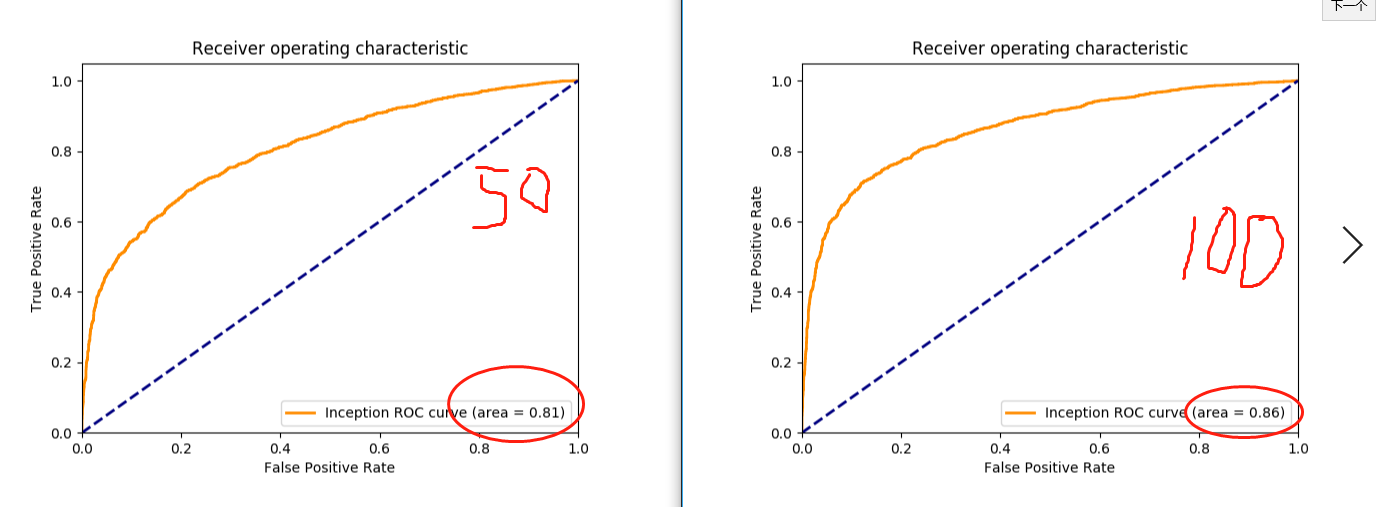
第二组实验中的程序大概在运行到13：00停止的。大概运行了16个小时。运行到第9折实验中的第七组数据。

可以看到epoch由50变成100后，运行时间增长了大概一倍以上.

【对比】选取第一折中的最终图像进行对比，我们可以看到很明显，第二组的结果比第一组好



符合实验预期。



Roc曲线也是训练了100epoch的好。。。

## 11总结

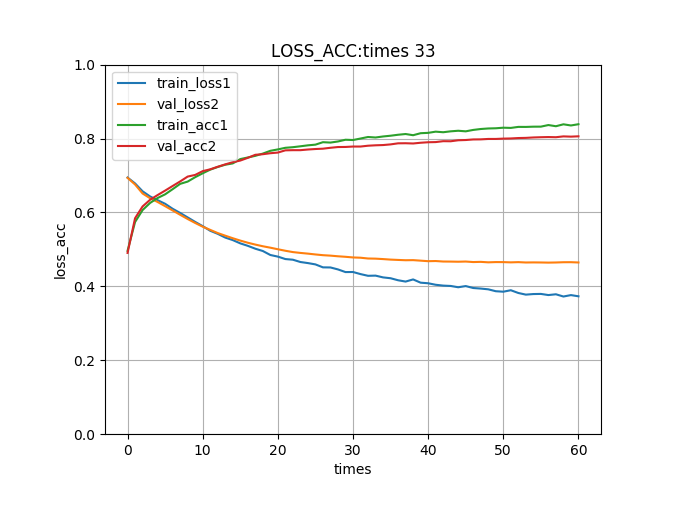
很显然，epoch越多越好。没有达到峰值的时候，模型还是不断变好的情况。很显然当前的模型只能说是较好，不能说是非常好。

（同一套模型有更好的结果，）

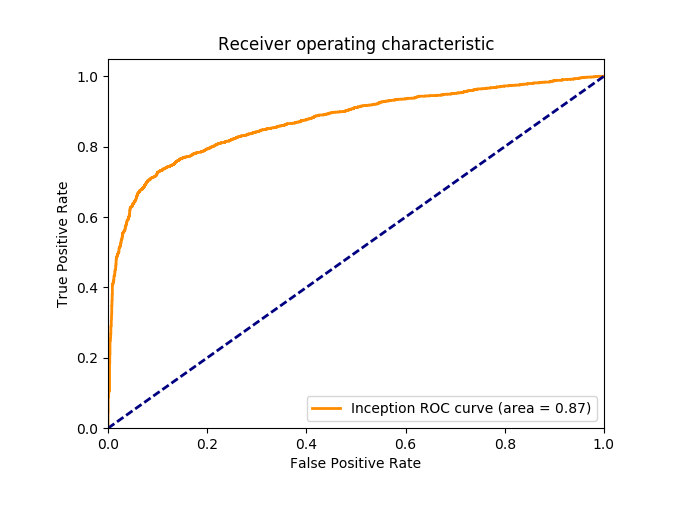
但是，该模型中，若训练好，仅仅使用了数据集中的靠前部分的负样本。这样的负样本选取是否合理？这样训练有没有可能有偏差？怎么能让全部数据都进行训练？

在0602的实验中，我对这个问题进行了探索，重新编辑实验代码，让其对所有负样本都学习到，同时，增加足够多的轮次（300轮，33组正负样本，epoch = 5）45000次训练。。。。【之前的是，1轮，33组，每组1250epoch。0000次训练。。。。】

结果记录如下。



16个小时，训练到60轮了。



MCC从0.1涨到了0.6

（需要等之后的输出看结果。）