Only-label 成员推理复现

目标数据集：CIFAR10

代码理解

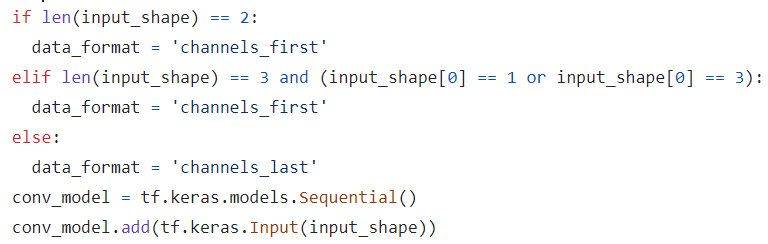
主要有模型代码model.py、训练代码training.py、攻击代码attacks.py组成。

1. 模型代码model.py

（1）CNN make-conv()函数

创建一个卷积神经网络模型CNN。

a.



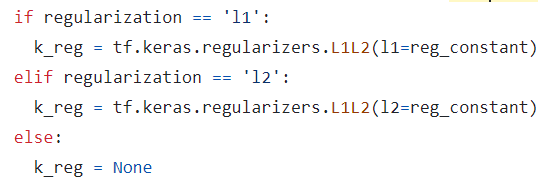
根据输入的张量类型，选择图像数据格式。channel\_first表示将通道数放在了第一个位置，channel\_last就表示将通道数放在了最后位置。

b.



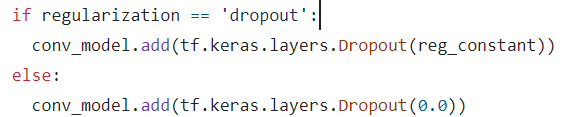
创建模型，共8个卷积层、2个池化层，分为两部分。

c.



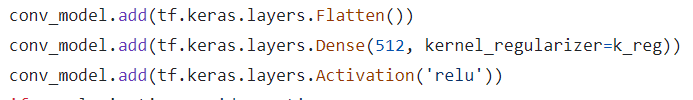
正则化处理，有l1、l2或无范数。

d.



Dropout层，是用来防止过拟合的。通过随机丢弃一些神经元（将输出设置为0）来减少对模型过分的依赖。Else部分表示丢弃率为0，即无该层。

e



全连接层。Flatten（）将卷积层的多维输出转换成一维。后添加一个有512个神经元的全连接层，并应用正则化，类型就是k\_reg，激活函数还是常见的relu。

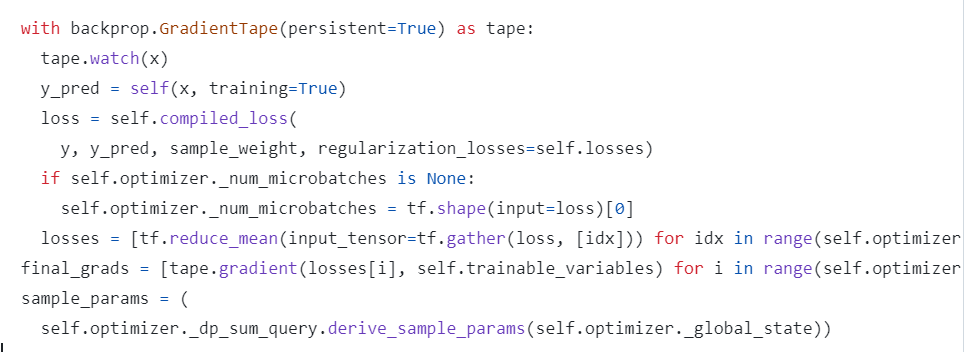
1. make\_fc.()

该部分和上面函数的功能差不多，只是神经元数量和激活函数（用的tanh）略微不同。

1. 训练代码training.py

（1）train\_step() 每一步的训练过程。

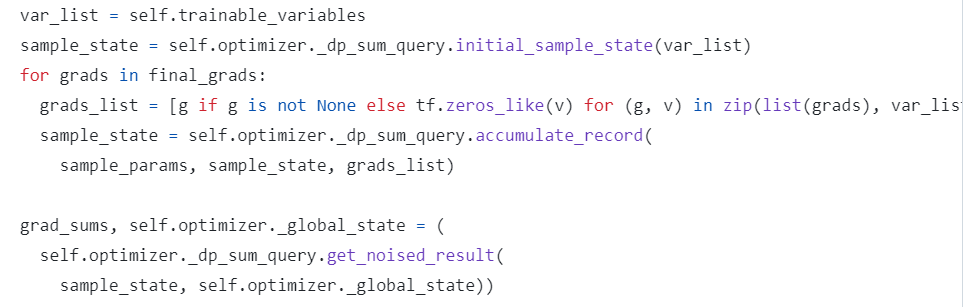
a.



计算训练过程中的loss。使用GradientTape 来计算梯度，结合了微批次（microbatch）的处理方式。y\_pred这一部分应该是在前向传播处理，得到预测值。后面就可以根据这个预测值和实际值进行loss的计算。

采用微批次的处理方式。将整体的loss分解成微批次的losses，由其计算微批次梯度，再来优化。这样可以有利于梯度稳定性和优化。

b.



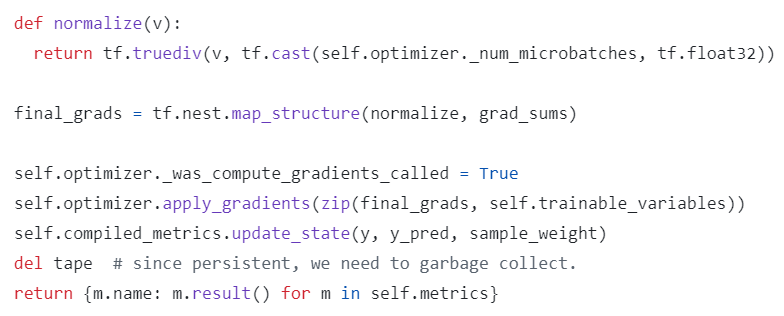
引入差分隐私，用于在梯度更新前对梯度进行噪声注入，起到数据保护的作用。

Var\_list包含模型中所有可训练的变量，即需要在反向传播中更新的权重和偏置等参数。根据这个，可以初始化模型的样本状态。

在for循环里，处理每个微批次的梯度，并将相关数据、信息保存。

最后，根据刚刚算出来的梯度，添加噪声。这样就可以保护训练时的相关数据。

C.



标准化操作。return部分是将输入v除以微批次数量，得到一个平均梯度。

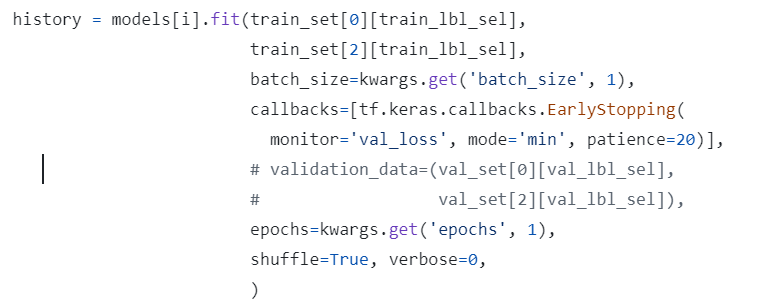
tf.nest.map\_structure 用于对 grad\_sums 中的每个元素应用 normalize 函数。所以final\_grads就是每个变量对应的平均梯度。

后面就是trian\_step()的关键点，根据梯度来启用梯度，更新模型权重。

（2）trian\_model (对一组模型训练的过程)

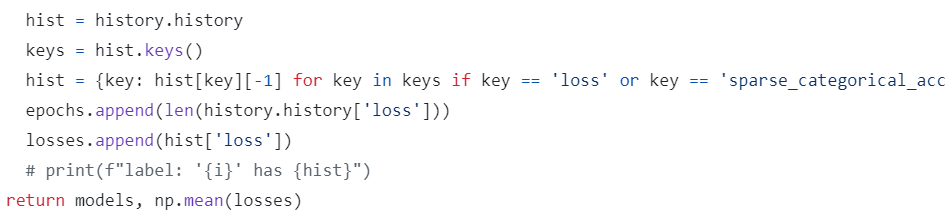
训练models列表里面的所有模型，但如果没有对应的训练样本就跳过。

A.



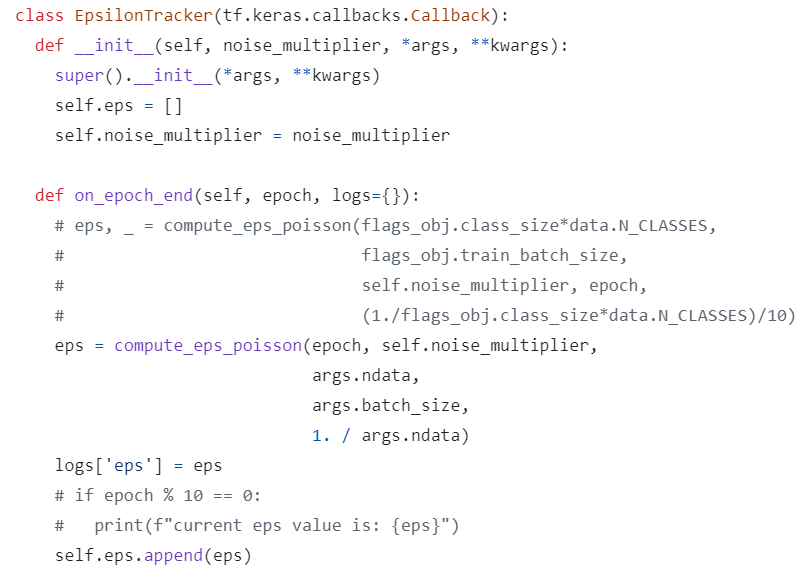
用fit（）进行训练。Trian\_lbl\_\_sel是用来提供样本的特征label的，EarlyStopping是用来监控模型loss，当loss不再减少（有预期减少值），就直接停止训练。

B.



这部分就是保存相关模型数据了，如loss、epoch等。

1. EpsilonTracker（）

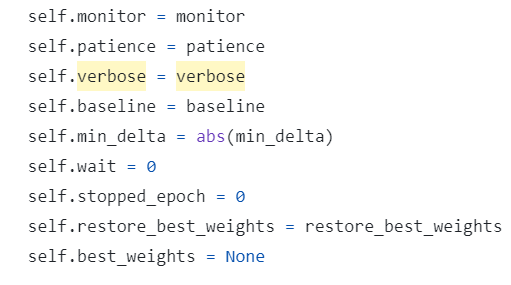


跟踪并记录每个训练轮次结束时的隐私损失参数，并更新相关数据。

1. EarlyStopping（）

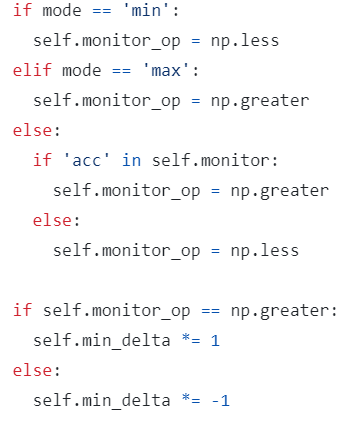
提前停止训练的监控函数，当满足loss的降低已经到了一定标准后，就停止训练。

A



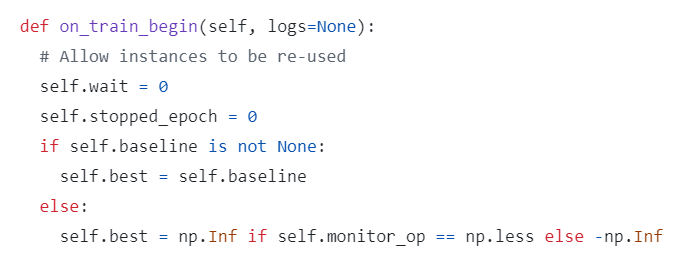
定义这个监控过程的系列参数。Monitor表示要监控的指标，本项目在默认下的监控量就是loss，如果想引进差分隐私保护的话，通过更改命令可以改变监控量为eps；patience表示耐心值，即如果在patience轮数下，监控量的改变值一直小于<min\_delta（规定的最小改变值），就会停止训练；baseline表示基线值，monitor的值大于这个值，符合其他条件就可以停止训练；wait作为变量，用于记录已经有多少轮monitor变化<min\_delta；restore\_best\_weights表示是否更新该轮的权重为最好权重，这得根据best\_weights和loss来看。

B



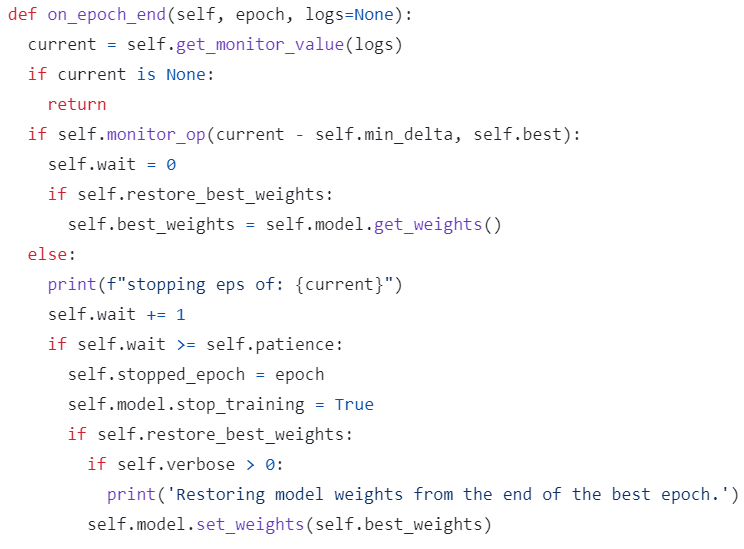
根据不同的任务需要，选择不同min\_delta。如果涉及到差分隐私，那么其eps在训练过程中肯定是组件增大的，所以min\_delta就要是正值。Loss同理。

C



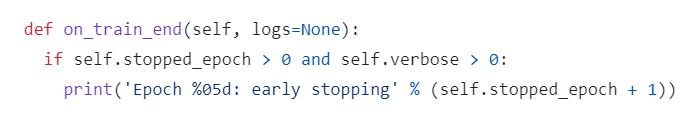
初始化。在训练开始前，将各种变量的值进行初始化，选择相应的min\_delta等。

D



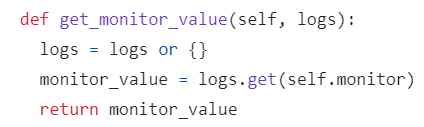
在每一epoch结束后，都要进行检查，判断十分需要立即停止训练，判断思路就是A中所说的。这个就是根据思路的代码思想，计算出各种变量的值，然后进行比较。如果符合，就把stop\_training定义为True，停止训练。最后就是是否需要保存本轮的结果为最好结果以及对应的权重。

E



训练结束时，进行打印。但是在默认代码进行下，verbose的值是不会大于0的，本行不会输出。

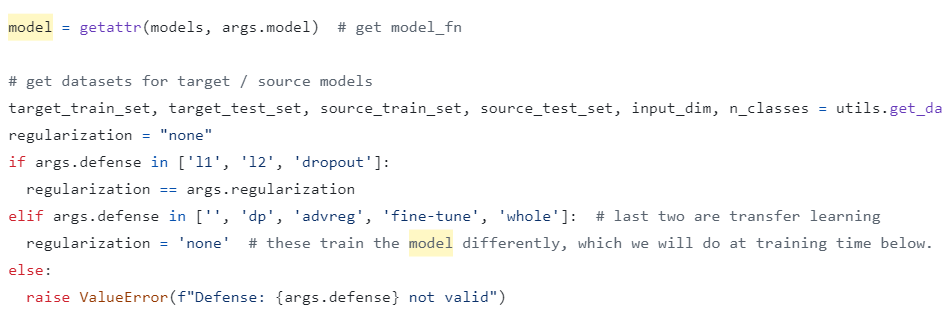
F



从日志中获取monitor的值。

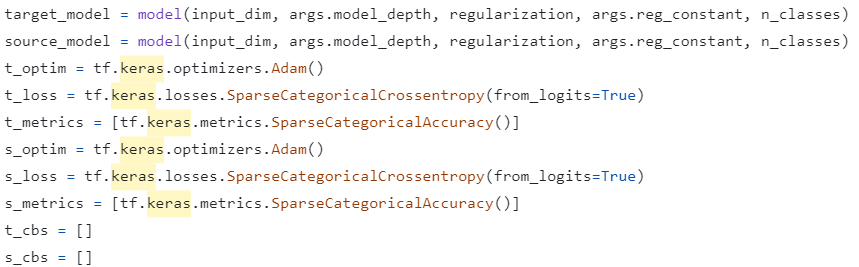
1. main（）函数

A



根据预先定义的参数，获取模型、数据集、测试集、正则化形式。

B



定义模型的各种变量和参数，包括loss、范数、优化等，后续要进行处理。

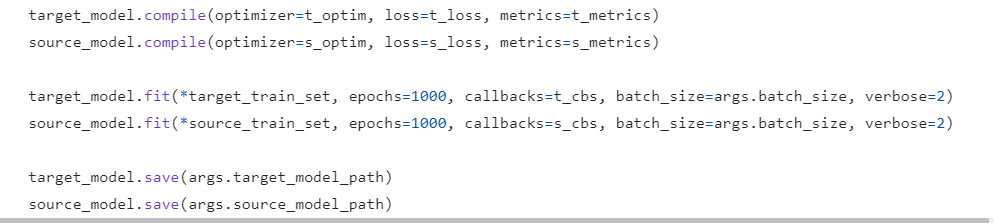
C



如果防御机制是dp（事先已经规定好的），就根据dp来设置特定的学习率、微批次大小、噪声和范数。

该部分剩下的两个if也一样，都是根据不同的防御机制来选择不同的指标数值和相应的EarlyStopping机制。

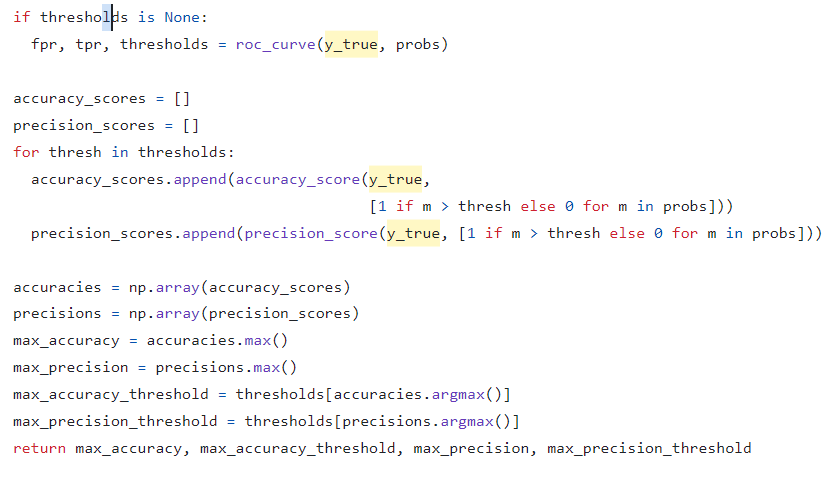
D



编译模型、训练模型、保存模型。有一个源模型和目标模型，现在源模型上进行相关推断分析，再到目标模型上。

3.攻击代码 attack.py

（1）get\_max\_accuracy()函数

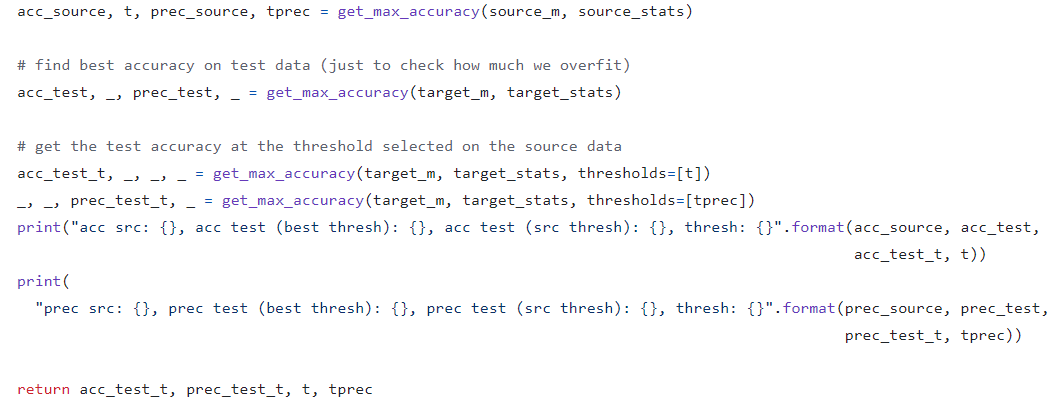


如果没有提前设定阈值thresholds（即用来进行判定的临界值），就通过计算roc\_curve曲线来收拢阈值。

精准度precision\_scores是根据accuracy和阈值之间的比较得出的。

最后返回最大准确率、精准度。

1. get\_thresholds()函数



使用源模型的输出确定最佳决策阈值，然后在目标模型上评估该阈值的有效性。利用get\_max\_accuracy()函数完成下述三个任务：

**在源数据上找到最佳阈值。**

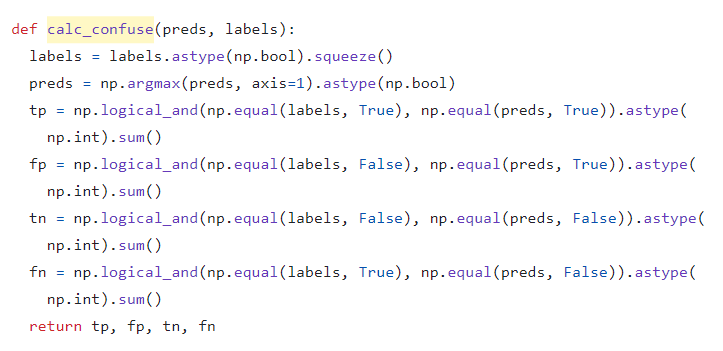
在目标数据上找到最佳准确率。

**在源数据上选择的阈值下评估目标模型。**

1. **trian\_model（）函数**

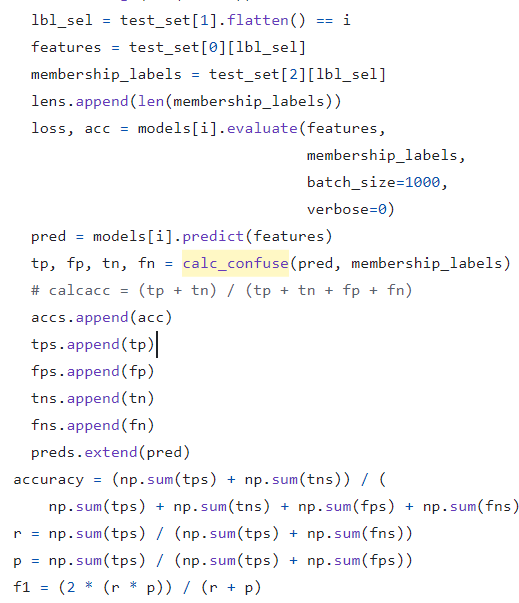
**这个和training.py里面的train\_model（）是一样的。**

1. **calc\_confuse（）函数**



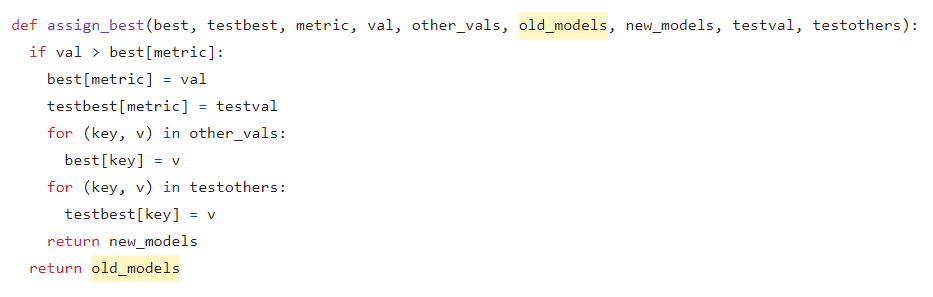
计算分类模型的混淆矩阵（Confusion Matrix）中的四个核心指标：真阳数（TP）、假阳数（FP）、真阴数（TN）和假阴例（FN）。可以帮助揭示模型是否倾向于某一类别。（限于二分类任务，或者把n分类等效转化成二分类再来用）。

1. test\_model()函数



核心部分如上，就是进一步计算并保存一些列评估指标，这些指标数值的初始值在其他函数里算出来了，这里只是用公式算比率之类的。

（6）assign\_best()函数



根据给定的性能指标 metric 来决定是否更新模型和对应的性能记录。比较当前模型在某个指标上的表现（val）与之前记录的最佳表现值（best[metric]），如果当前模型的表现更好，就会更新相关的记录，并返回新的模型。否则，保留旧模型不变。相关值也是在其他部分已经算出来了。

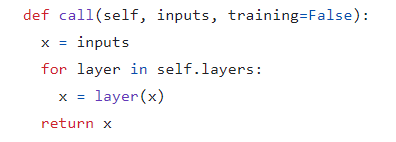
1. AttackModel()类

A



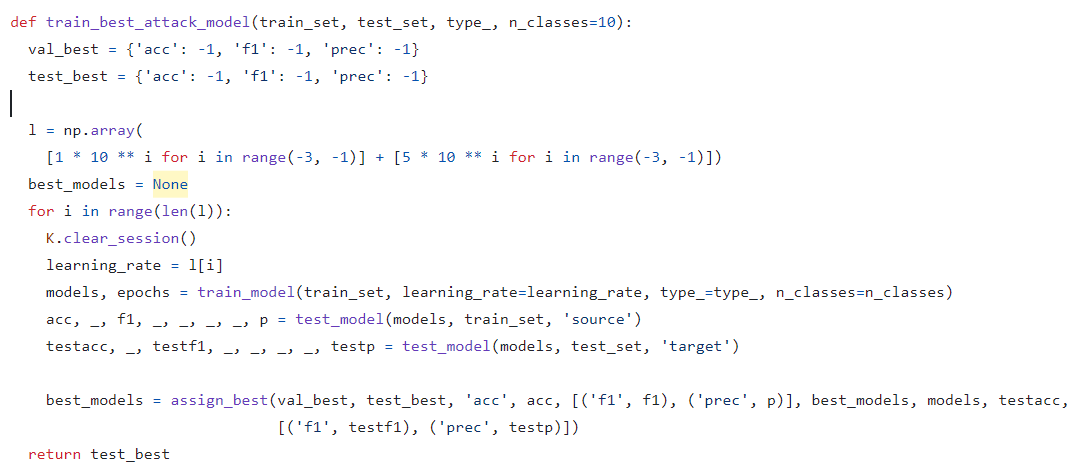
根据预先的设定（可在命令中修改，否则就是默认值）确定攻击模型的网络结构。定义两个全连接层：self.out用于最后分类，self.x1根据relu函数对输出进行相应的处理。

B



Inputs就是一个批次的样本，放入模型后， 以此进入每一层对输入样本进行相应的处理。

（7）train\_best\_attack\_model（）函数



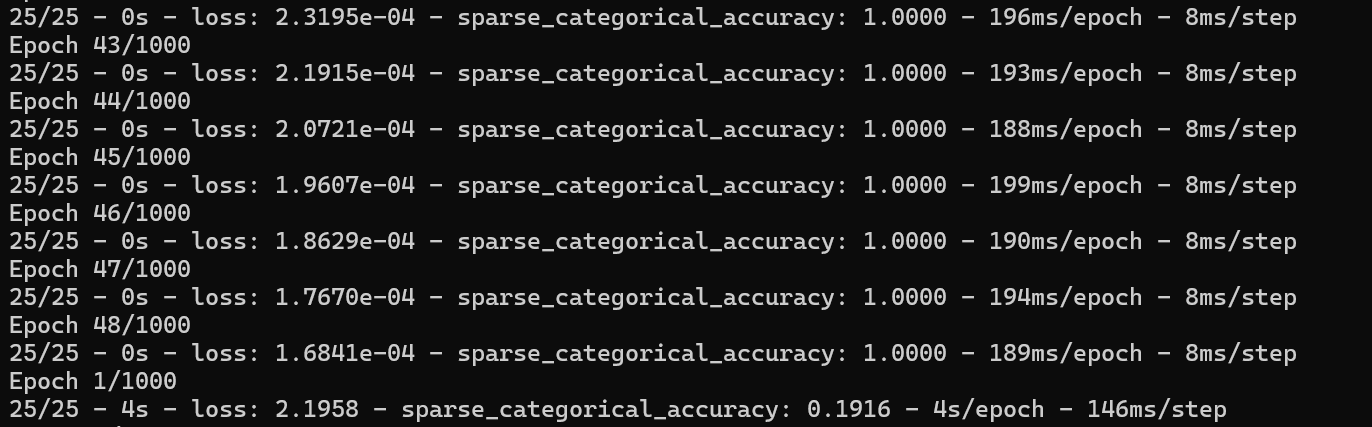
这个函数就是用来寻找最好的模型，在指定训练集、测试集下对一组模型进行训练、评估，最后返回测试结果最好的模型的acc、f1、precision。其中涉及到的函数都是前面的，该函数本身没有什么处理重点。

复现过程：

运行如下命令

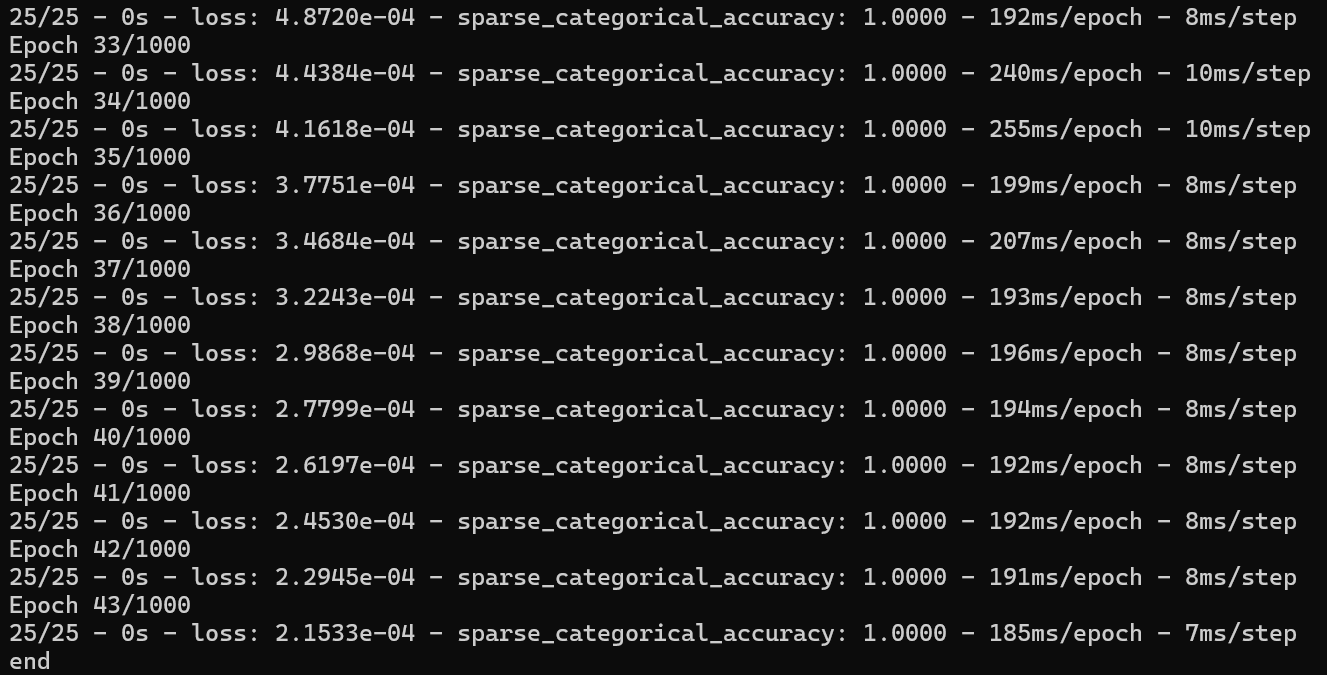


发现：



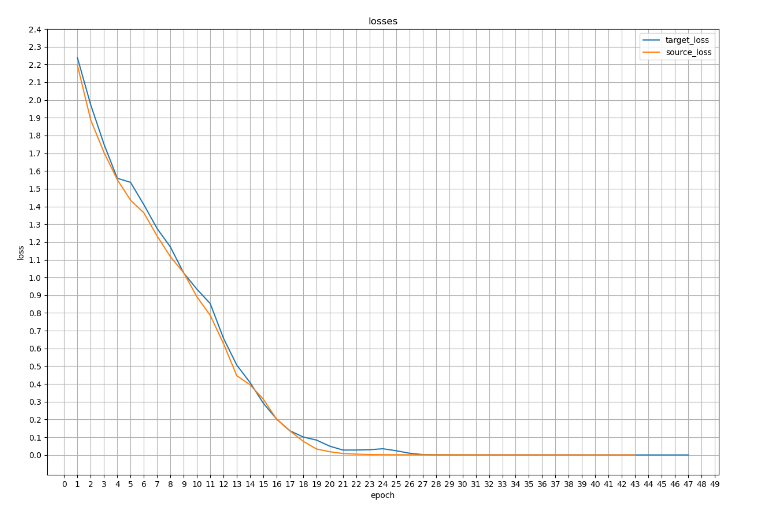
这里从epoch=48之间到epoch=1，表明前面的target\_model训练结束，并EarlyStoppoing。

后面的就是source\_model了。



这个“end”是我自己加上的，用于表示训练阶段的完成。

两个模型的训练loss变化曲线如下：



准确率变化如下：

