**模型测试**

在已有的三个模型**CNNMalware\_Model1、My\_CNNMalware\_Model1、My\_CNNMalware\_lenet5**上进行参数调整。

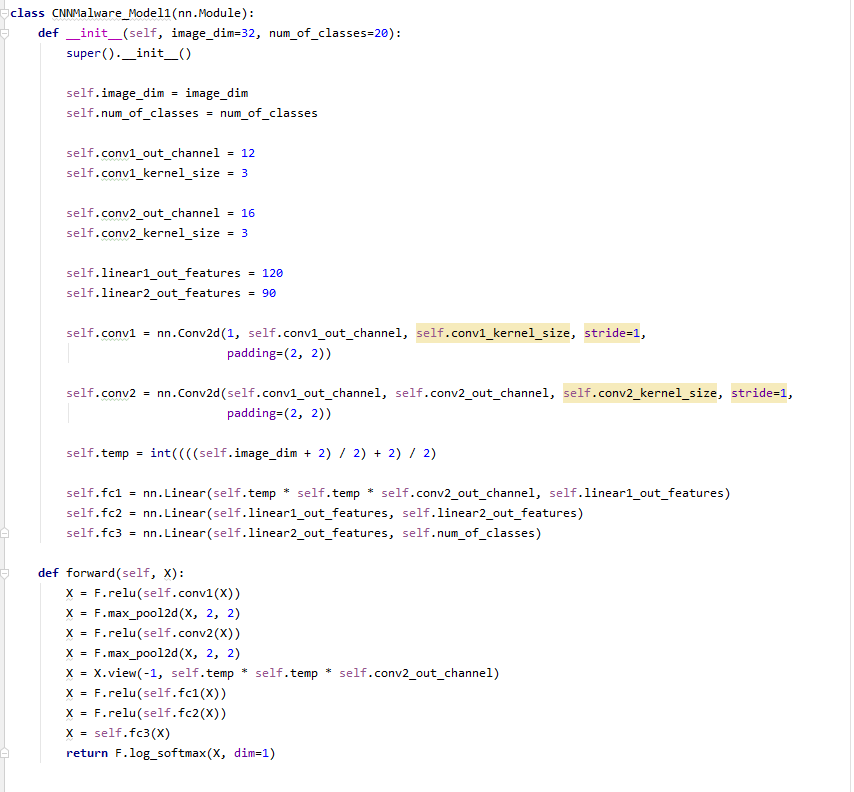
其中

**CNNMalware\_Model1** 为代码自带模型，

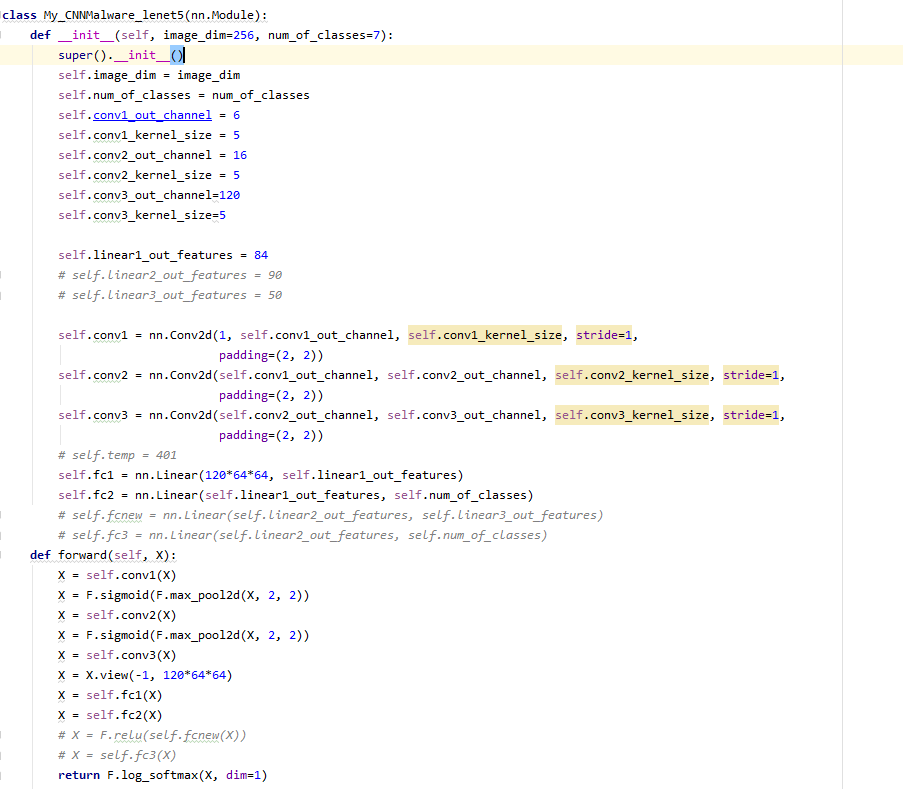
**My\_CNNMalware\_Model1** 为在自带模型上修改所得

**My\_CNNMalware\_lenet5** 为经典模型lenet5

它们的模型结构如下

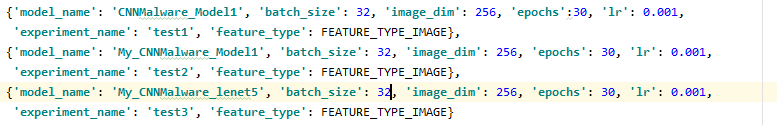






统一测试：

一次测试三个模型，测试参数如下：



结果如下

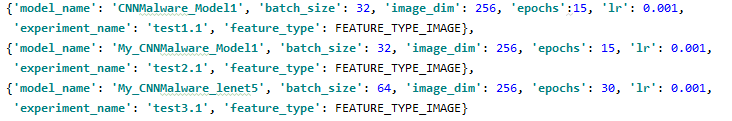
Test1：

Test2：

Test3：

通过观察混淆矩阵和损失函数，准确率曲线可以得出，模型1和模型2有较好的效果，但是30次的训练次数过多，出现了过拟合的情况，因此决定将模型1和模型2的训练次数改为15，而对于模型3，尝试增大batch\_size到64以观察效果

测试参数如下：



结果如下：

Test1.1：



Test2.1：



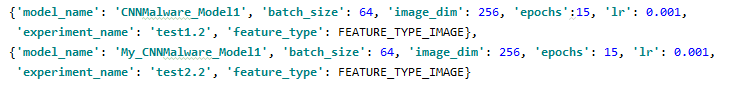
Test3.1：

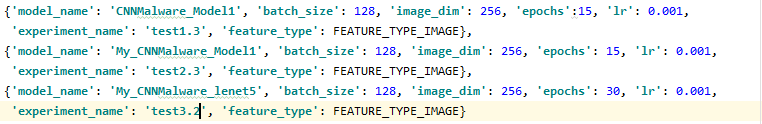


可以看出虽然模型1和2降低了一些准确率，但是这是训练次数减少了一半换来的，可以接受这样的代价，而模型3稍微增加了准确率，由此可以看出batch\_size的适当增加对准确率有一定影响。

决定对模型1和模型2和模型3将batch\_size增大到64和128进行测试

测试参数如下：





结果如下

Test1.2：



Test2.2：



Test1.3:



Test2.3:



Test3.2:

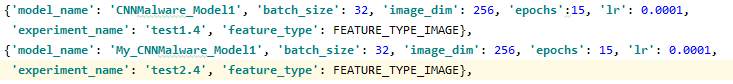


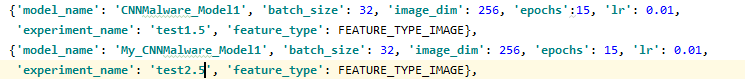
可以看到模型3在这两个参数下直接失去了有效性，因此决定将test3.1作为模型3的最终成绩，即稍高于要求的75%，是差强人意的。

而模型1与模型2都在batch\_size增大后出现了准确率的降低，因此决定将batch\_size就固定在32，转而更改其他参数，其中模型2相较模型1下降幅度更小，可以认为模型2在一定程度上更加优秀。

对模型2和模型1再进行lr的尝试更改。

测试参数如下：





结果如下：

Test1.4：



Test2.4：



Test1.5：



Test2.5:



可以看出，无论是增大还是减小lr值，都不能使结果更佳。

综上所述，可以认为模型2的test2.1中所呈现出的结果使最佳的，它的混淆矩阵（如下图）也同样再对角线上有清晰的结果。

