# 深度学习理论与实践练习题007

友情提示：

1. 编程题可用pycharm或vscode编辑、调试，也可用其他方式，形式不限；
2. 编程题附上代码及结果截图；
3. 深度学习理论部分题目，可手写拍照；

## Python

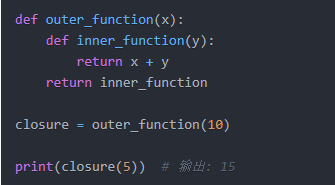
1. 解释下python中的继承并举例说明；

Python的继承：创建的新类在已有类（父类或基类）里继承属性和方法。创建的新的子类可以添加新的属性和新的方法以及重写或扩展父类方法。继承的目的是重用已有类的代码，减少代码的冗余，提高代码的结构和可维护性。



1. python中闭包是什么；

闭包是一种特殊的函数，可以在其内部捕获并记住外部函数的局部变量的状态，即使外部函数已经执行完毕，这些局部变量的值仍然能被内部函数访问。



在这个例子中，outer\_function 是外部函数，它包含一个内部函数 inner\_function。inner\_function 引用了外部函数 outer\_function 的局部变量 x。

当我们调用 outer\_function(10) 时，inner\_function 对象被创建并返回。这个时候 outer\_function 的执行已经结束。返回的 inner\_function 对象是一个闭包，因为它记住了 x 的值，即使在外部函数执行结束后。

接下来，我们在闭包 closure 上调用 \_\_call\_\_ 方法，传入参数 5，返回的结果是 15。这说明即使在 outer\_function 执行完毕之后，闭包仍然能够访问并使用变量 x 的值。

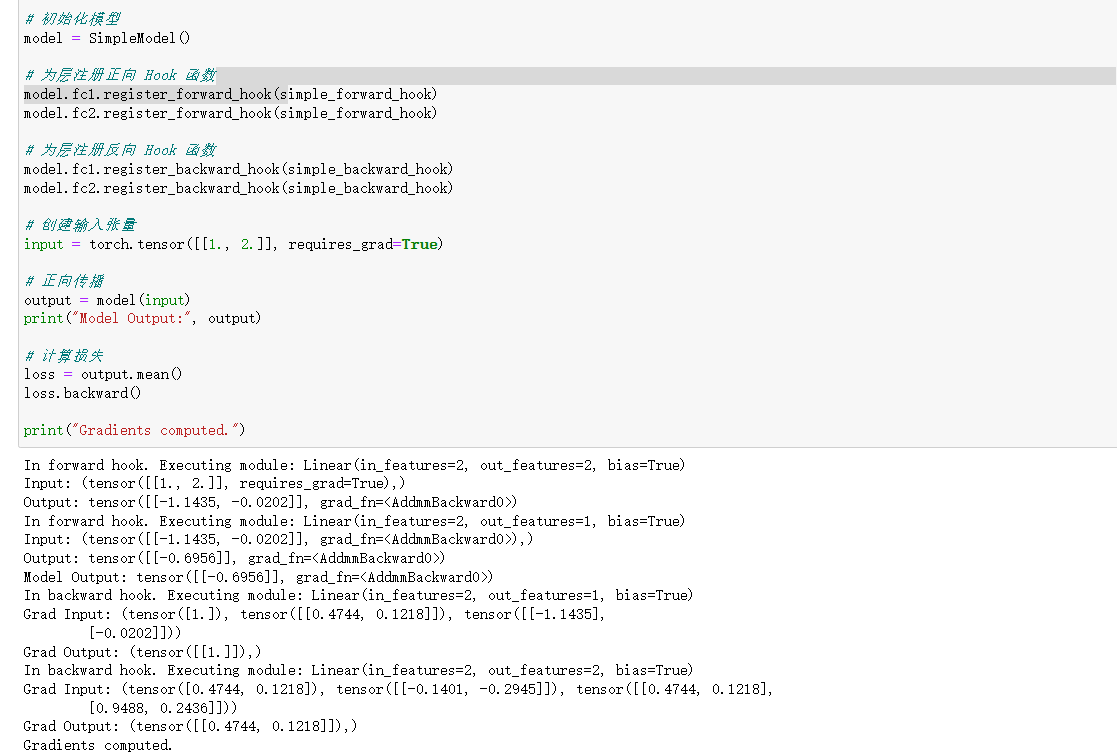
## 深度学习框架

1. Pytorch中hook使用并举例说明；

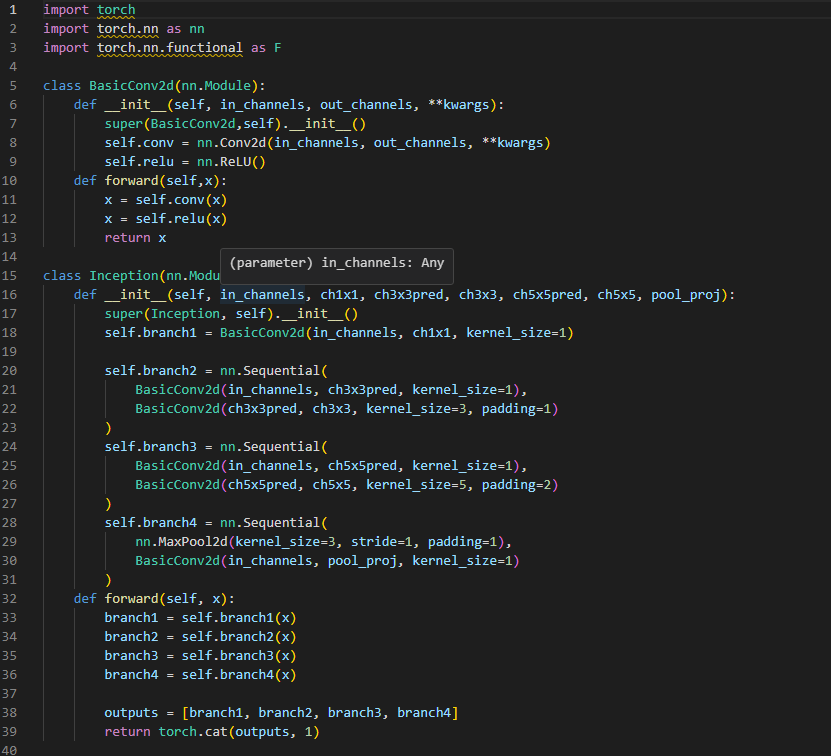
Hook的使用：用于在神经网络的计算过程中注入自定义操作，以实现自定义操作、检查激活值、计算梯度等目的。在Pytorch中有两种类型的Hook:正向Hook、反向Hook,正向Hook可通过注册至nn.Module上来实现，反向Hook可通过注册至torch.Tensor上实现。

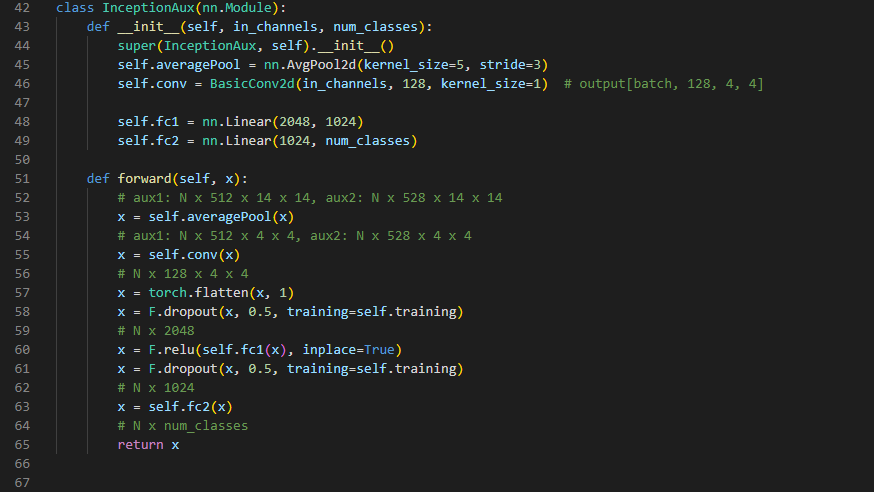
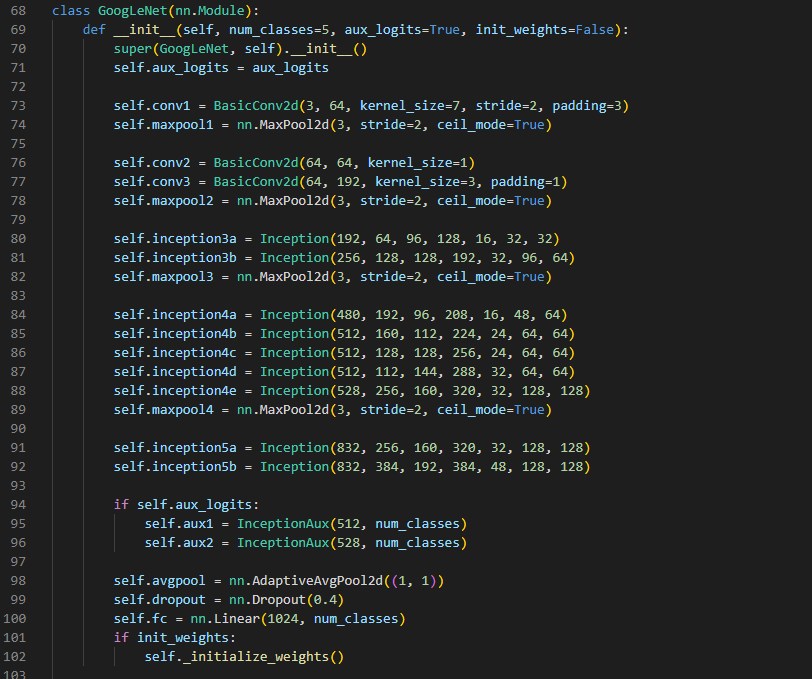
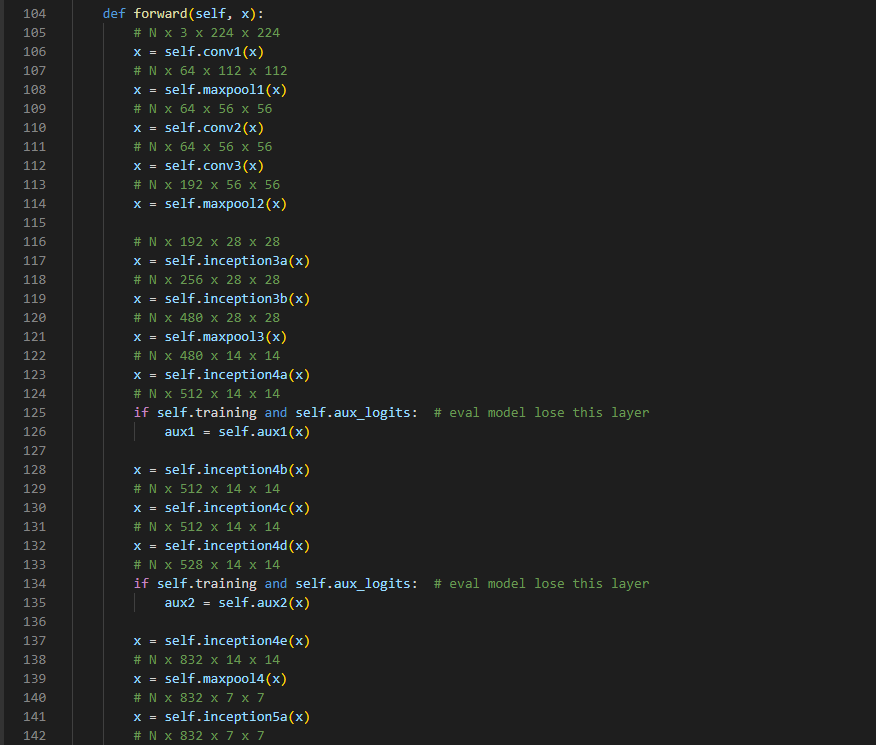
示例：

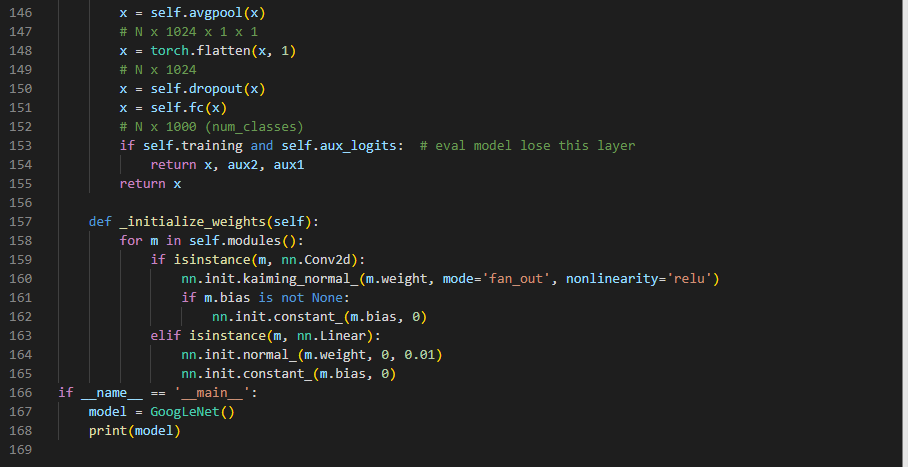




1. 基于flower数据集，使用pytorch实现googlenet网络；





## 深度学习理论知识点

1. 如何确定CNN的卷积核通道数和卷积输出层的通道数；

确定CNN的卷积核通道数和卷积输出层通道数的几个方法：

1. 网络结构和输入通道数：首先考虑网络的输入维度，卷积核的输入通道数必须和前一层的输出通道数相等；
2. 输出通道数展示特征：卷积层的输出通道数表示特征图数量，用于表示不同特征的组合。如果想增加模型学习更复杂特征的能力，可以考虑提高输出通道数。同时会导致参数数量增加，从而增加内存需求和计算成本，因此要权衡好正确的输出通道数。
3. 经验法则（2倍原则）：一个常见的原则是在网络的深度增加时，逐渐增加输出通道数，这种方法有助于提高特征空间的表达能力，同时相对较低的计算负担。
4. 参考现有的架构:参考现有成熟的网络架构，可以有效指引卷积核通道数和输出通道数的选择。
5. 任务和数据集：模型的适用任务和数据集自然会影响卷积核通道数和输出通道数。对于一些简单任务和小型数据集，仅需较少的通道数。而复杂数子问题和大型数据集可能需要更复杂的模型和更多的通道数。
6. 计算资源限制：卷积核通道数和输出通道数将影响模型的计算负担。因此，在设计CNN时，需要考虑可用计算资源和理想的计算速度。如果有严格的计算资源限制，可以降低通道数和模型复杂度。如果计算资源充足，可以尝试更复杂的模型。
7. Sigmoid、ReLu、Tanh这三个激活函数有什么优缺点；
8. sigmoid激活函数：将输入值映射到（0,1）区间内，是一种平滑且良好的非线性特性的激活函数，数学表达式如下：



优点：

1. 输出范围有界（0,1），有助于降低输出结果的波动范围
2. 平滑性：连续且光滑，对梯度下降算法等优化方法友好

缺点：

1. 梯度消失：当输入值的绝对值较大时，sigmoid函数的梯度接近于0。使得深度网络中出现梯度消失，网络难以学习
2. 输出非零中心：sigmoid输出非零中心，可能导致优化过程起伏波动，使训练变得复杂
3. 计算昂贵：涉及指数运算，计算成本较高
4. Relu激活函数

近年来较流行的激活函数，表达式如下：



优点：

1. 计算简单：计算复杂度低
2. 相比于Sigmoid和Tanh函数能够在一定程度上缓解梯度消失问题

缺点：

1. 非线性区域死亡：当输入值为负数时，ReLU函数输出为0，这可能导致神经元永久性失效。当网络学习率过高时，梯度可能变得非常大，从而导致梯度爆炸问题
2. 输出非零中心：可能导致优化过程起伏波动
3. Tanh激活函数

将输入值映射到（-1,1）区间。数学表达式如下：



优点：

（1）输出范围有界且零中心：Tanh 函数的输出在 (-1, 1) 之间，且输出均值接近 0。这使得训练过程更加稳定，优化过程较为平滑。

（2）平滑：Tanh 函数是连续且光滑的，适用于梯度下降等优化算法。

缺点：

（1）梯度消失：与 Sigmoid 函数一样，Tanh 函数在输入值绝对值较大时，梯度接近于 0。这可能导致深度网络中的梯度消失问题。

（2）计算昂贵：与 Sigmoid 函数相似，Tanh 函数涉及指数运算，计算成本相对较高。

年来，ReLU 及其变体（如 Leaky ReLU、Parametric ReLU 等）在许多任务中成为首选，因为它们计算简单且在一定程度上有助于缓解梯度消失问题

## 机器学习

1. SVM公式推导及原理理解；

参考链接：https://blog.csdn.net/qq\_36329233/article/details/82262415

1. 哪些机器学习模型不用做归一化并说明理由；

归一化：数据预处理的一种方法,他将特征值缩放到相同范围内，帮助提高模型的性能。并非所有模型都需要使用归一化，示例如下：

（1）决策树（Decision Trees）：决策树是基于数据集的特征值对数据进行递归划分，直到达到某种停止标准（如节点中的数据量达到一定阈值等）。因为决策树的划分仅基于单个特征的阈值，不涉及不同特征值之间的权重，所以归一化对于决策树模型没有影响。

（2）随机森林（Random Forests）：随机森林是决策树的集成模型，由多个决策树构成。由于每个决策树的构建和预测过程中都不涉及特征值的比例，所以不需要进行归一化。

（3）梯度提升树（Gradient Boosting Trees）：梯度提升树（如XGBoost、LightGBM 等）同样属于基于树的集成模型，其构建过程也不依赖于特征值范围。因此，不需要进行归一化。

（4）K-近邻（K-Nearest Neighbors）：虽然 K-近邻模型中计算数据点之间的距离，但它通过投票机制将基于距离的相似性转化为类别预测。这种投票机制使其对于特征值范围差异具有鲁棒性。然而，在许多情况下，执行归一化可以提高 K-近邻的性能，因为距离计算中的小范围特征可能会被大范围特征主导。在某些任务下，确定不同特征的相对权重很重要，则归一化很有帮助。

尽管上述模型不要求进行归一化，但是在某些应用场经理，采用归一化仍会对模型训练有所帮助。归一化有助于在训练过程中减少异常点对模型的影响，提高计算速度和收敛速度。

## C/C++/数据结构

1. C++中声明常量的关键字是（D）
2. extern；B. public；C.virtual；D.const
3. C++中如何理解引用传递、值传递和指针传递，并举例说明；