本次编程作业采用pytorch搭建模型并进行训练。

对于隐藏层数的研究，隐藏层神经元个数为16，学习率为0.001，优化器为Adam，激活函数是ReLU，而隐藏层数的变化为1-2-4-8-16-32。

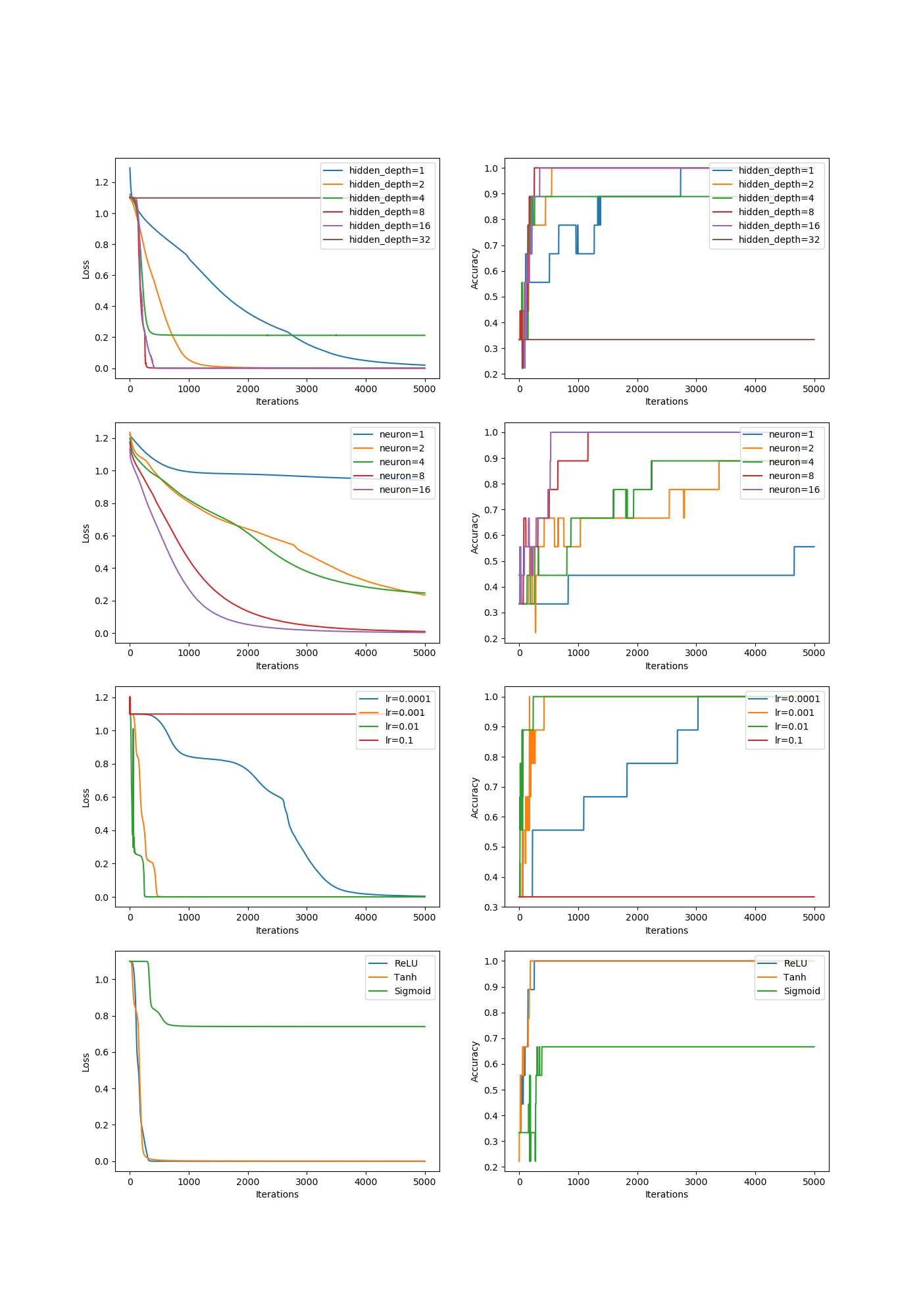
对于隐藏层神经元个数的研究，隐藏层数为1层，学习率为0.001，优化器为Adam，激活函数是ReLU，而隐藏层神经元个数的变化为2-4-8-16-32。

对于学习率的研究，隐藏层神经元个数为16，控制隐藏层数为8层，优化器为Adam，激活函数是ReLU，而学习率的变化为0.0001-0.001-0.01-0.1。

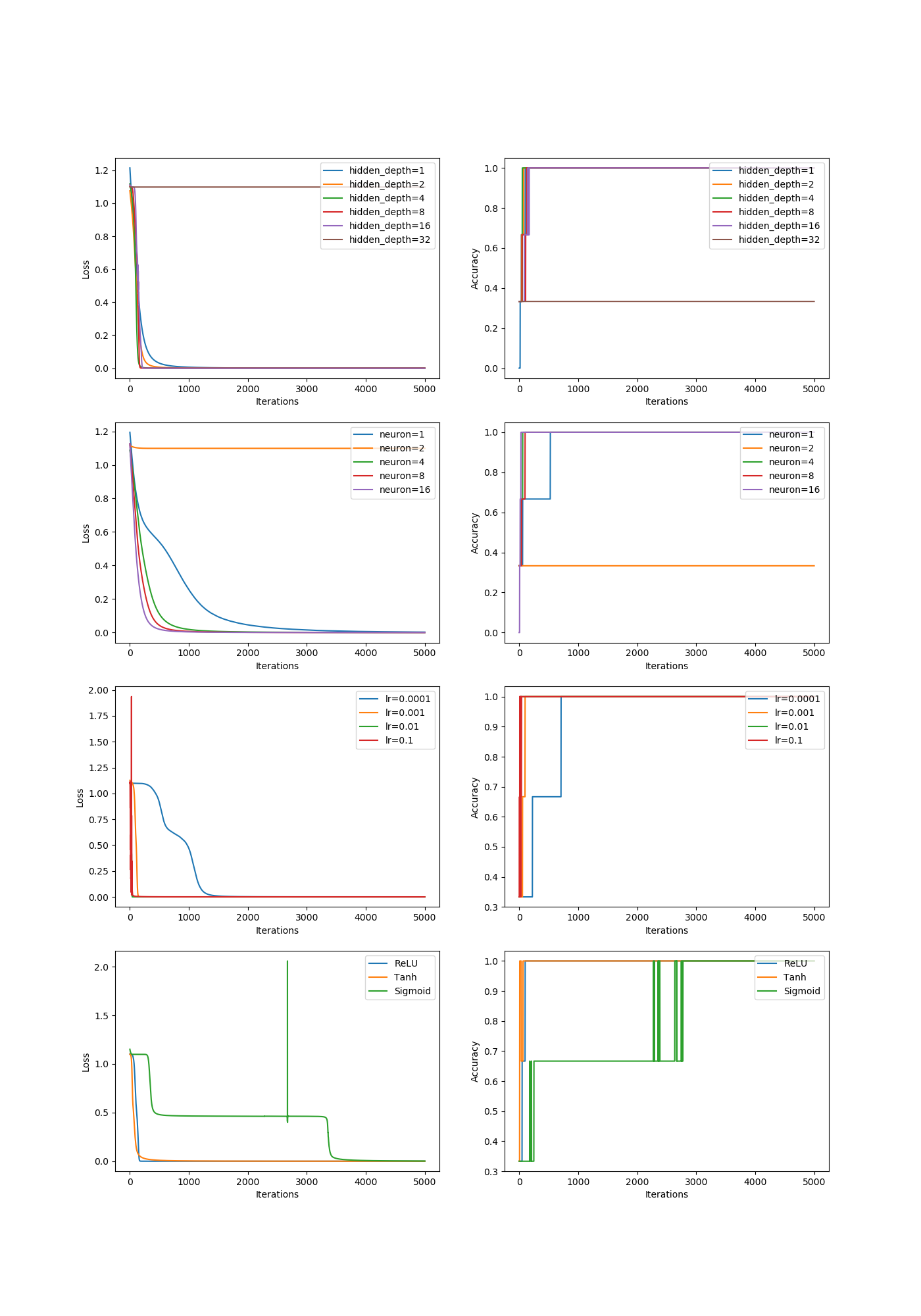
对于激活函数的研究，隐藏层神经元个数为16，控制隐藏层数为8层，学习率为0.001，优化器为Adam，而激活函数的变化为ReLU-tanh-Sigmoid。

训练结果如下。

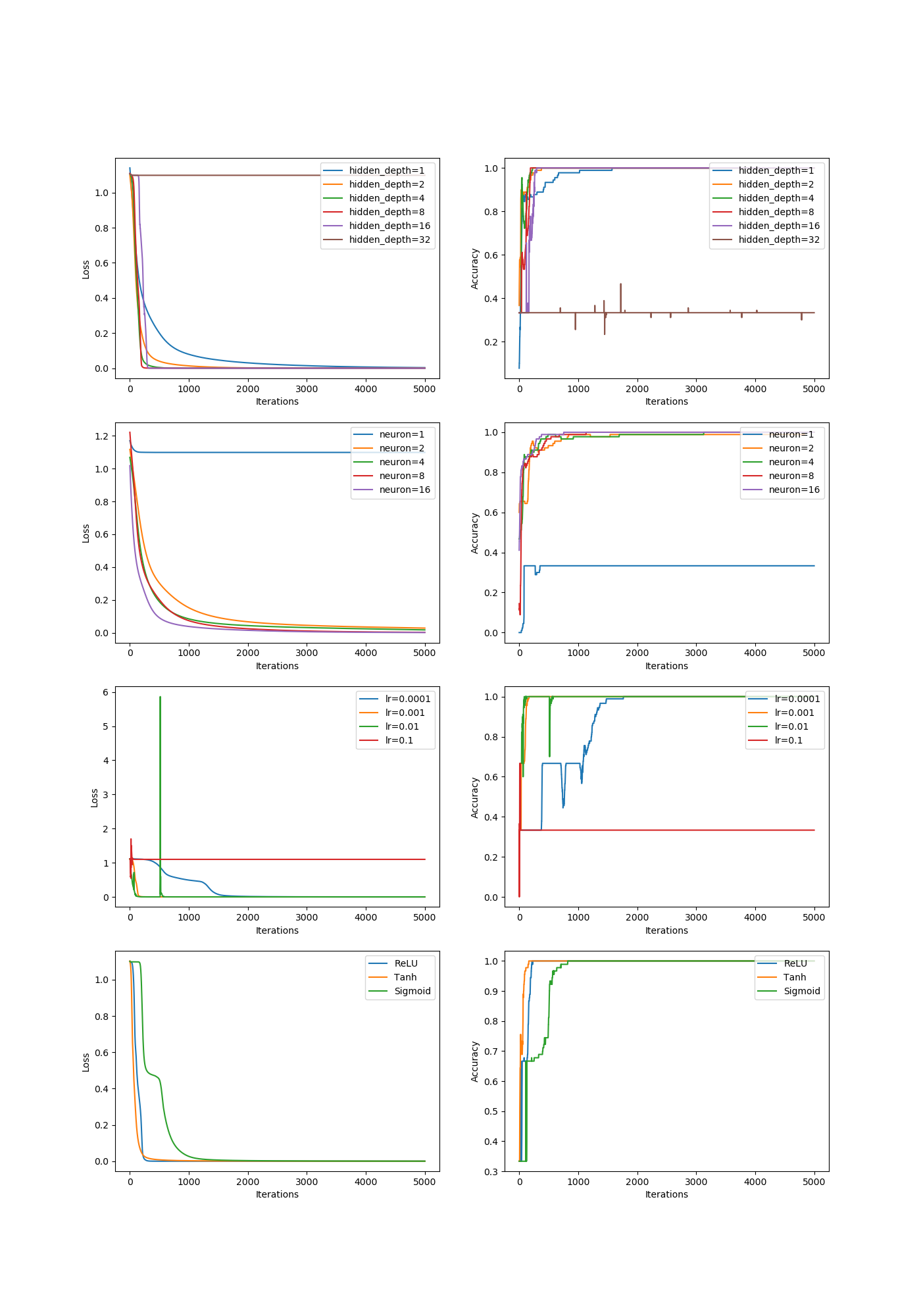
Problem1：



Problem2：



Problem3：



结果分析：  
对隐含层数分析：

随着网络层数的增加，在一定的范围内，模型的loss随着网络层数的增加而减少，这点符合常识，不再细讲。但是当到达了一定的程度，例如32层时，模型的loss在开始时略微下降之后模型也就未在优化。初步判断原因可能为1. 考虑到模型层数的增加，那么在进行反向传播运算时，因为层数过多，存在着梯度消失的问题，对应都可以采用残差网络ResNet进行相对应的解决。2. 此外，考虑到课堂上所提到的Vapnic Chervonenkis Bound的约束，当模型参数变化时，需要有更多的数据来进行训练以满足不等式的要求，但是在此次实验中训练数据有限，因此可能无法满足不等式要求，须增加训练集以满足要求。

对隐含层神经元分析：

神经元个数上升，训练结果也越好，但同时对于内存等资源占用越大。但是实际上并不需要如此多的神经元进行实现，通过更多次的迭代，较少的神经元数最终也能实现较好的结果，即“用时间换资源占用”。虽然此次实验为展示出来，但理论上，神经元各数较多，在训练数据集同样增加以满足训练的情况下，会产生过拟合的现象，此时需要通过正则化手段，例如dropout使神经元间歇性失活等方向来尽量减少过拟合的影响。

对学习率进行分析：

总的来说学习率的设置需要参考网络的模型，多次尝试，避免“多次优化但几乎仍在原地”即学习率过小的情况，同时也要避免“多次优化但反复横跳震荡“的学习率多大的情况。考虑到曾经比赛或研究时的经验，一些对学习率的设计可以作为trick来提升模型效果，例如模拟退火来改变学习利率以此使模型跳出局部最优解，在某次模型训练中确定此trick有效；将学习率与loss梯度进行捆绑等动态改变学习率等，都可以很好的提升模型最终的结果。

对激活函数的分析：

激活函数最重要的一点就是使模型实现非线性。在三次问题中，总的来说在比较loss降到最低时的时间时ReLU> tanh> Sigmoid。sigmoid和tanh在深层网络时很容易导致梯度消失，这是需要避免的一点。