HW2

一．判断题

1. 神经网络随着网络层数的增加表达能力变得更强，因此网络中的激活函数可以去掉，只需要加深网络就能够训练足够强的神经网络。（x）

   如果去掉了激活函数，不管网络的层数有多少，整个网络就相当于只有一次线性变换（根据矩阵乘法的结合律），也就起不到增加神经网络表达能力的效果了。

2. Long short-term Memory (LSTM)网络具有记忆和遗忘功能，适用于序列建模。（√）

3. 注意力机制(attention)相比 LSTM 更高效，因为在每个 time step 计算时都可以读到全局信息，而不需要像 LSTM 那样串行计算。（√）

4. 训练卷积神经网络（CNN）时，如果对训练样本通过平移、旋转和缩放等操作额外生成一些补充样本，会从整体上降低训练样本的质量，影响网络提取特征，从而导致预测准确率下降。（x）

   对训练样本进行平移、旋转和缩放等操作可以生成补充样本，这样可以增大训练的数据集并降低过拟合的风险，因此预测准确率会上升。

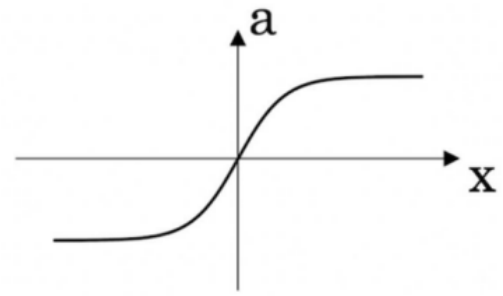
5. 在网络中加入 Dropout 和 Batch Normalization 都是深度学习中常见的防止过拟合的手段，在所有类型的深度神经网络中都适用。（x）

   Batch Normalization在batch_size较小的时候效果比较差，因为BN是使用batch中样本的均值和方差来模拟全部数据的均值和方差。另外将Dropout和Batch Normalization应用在RNN等动态网络上的时候效果也不太好。

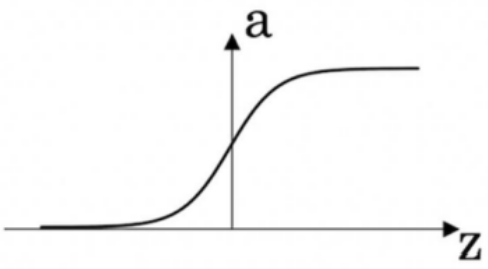
二．选择题

1. 下图中哪一个表述 ReLU 激活函数？（C）

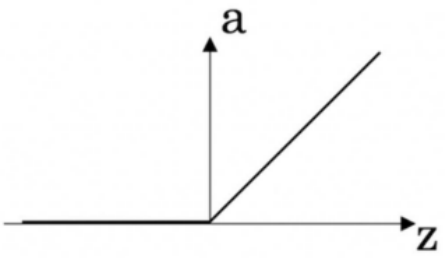
A .



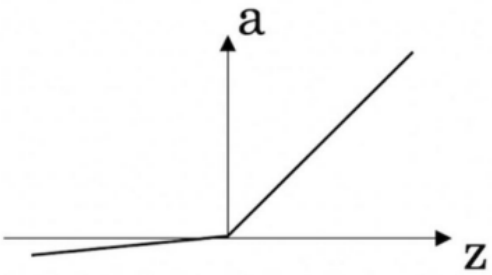
B.



C .



D.



A: tanh, B: sigmoid, D: Leaky ReLU

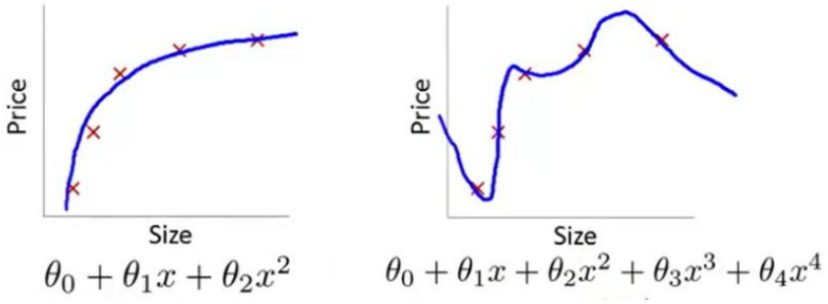
2. 输入为 64*64 的 **RGB** 图片，使用 32 个 3*3 的卷积核提取特征，步长为 1，不使用 padding 填充，则输出的大小为：（c）
   （a）64*64*64 (b) 64*32*32 (c) 32*62*62 (d) 32*64*64

3. 上一题中，该卷积层的参数有多少个？ （d）
   （a）64 （b）32 （c）32*3*3 （d）32*3*3*3

三．简答题

1. 深度学习模型为何在训练中容易出现过拟合？试分析原因并给出如何在深度学习模型训练中缓解过拟合？

   原因：过度拟合是指对训练数据集学习得太好，以至于把数据的一些局部特征或者噪声带来的特征都给学到了，导致在进行测试的时候误差较大，模型无法正确地对数据进行分类。



$\theta_0 + \theta_1 x + \theta_2 x^2$ $\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

比如上图中，左边的图是适当拟合的结果，右图是过拟合的结果

解决办法：

（1）在损失函数后面添加正则项，可以让训练模型最大限度拟合训练数据集，但又不会完全拟合训练数据，使模型有更好的泛化能力（如下面的loss function）。

$$min_\theta \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000 \cdot \theta_3^2 + 1000 \cdot \theta_4^2$$

（2）使用 Dropout ：在训练的每一次迭代过程中随机地丢弃神经网络中的神经元。当我们丢弃不同神经元集合的时候，就等同于训练不同的神经网络。不同的神经网络会以不同的方式发生过拟合，所以丢弃的净效应将会减少过拟合的发生。

（3）降低模型复杂度：可以通过简单地移除层或者减少神经元的数量使得网络规模变小

（4）把明显异常的数据剔除

2. 深度学习模型训练过程中为何会出现梯度消失和梯度爆炸问题？有哪些方法可以解决梯度消失或梯度爆炸？

   原因：当使用基于梯度下降和反向传播训练深度神经网络时，我们通过从最后一层到初始层遍历网络来寻找偏导数。在由n个隐藏层组成的网络中，n个导数将相乘。由于网络层数的加深，梯度的膨胀或缩小效应不断累积，最终很容易造成模型无法收敛。如果导数很大，那么梯度会随着我们沿着模型向下传播而指数增加，直到它们最终爆炸，这就是我们所说的爆炸梯度问题，在深层网络中权值初始化值太大的情况下容易出现。如果导数很小，当我们在模型中传播时，梯度将以指数形式减小，直到它最终消失，这就是消失梯度问题，容易在深层网络中采用了不合适的激活函数的情况下出现。

   解决方法：

- **预训练、微调：**每次训练一层隐节点，训练时将上一层隐节点的输出作为输入，而本层隐节点的输出作为下一层隐节点的输入，此过程就是逐层“预训练”；在预训练完成后，再对整个网络进行“微调”
- **梯度剪切、正则：**主要是针对梯度爆炸提出的，其思想是设置一个梯度剪切阈值，然后更新梯度的时候，如果梯度超过这个阈值，那么就将其强制限制在这个范围之内；正则化是通过对网络权重做正则限制过拟合，如下面损失函数，如果发生梯度爆炸，权值的范数就会变的非常大，所以通过正则化项，可以部分限制梯度爆炸的发生。

$$Loss = (y - W^T x)^2 + \alpha ||W||^2$$

- **使用 relu、leakrelu、elu 等激活函数：**如果激活函数的导数为1，那么就不存在梯度消失爆炸的问题了，每层的网络都可以得到相同的更新速度
- **使用batch normalization：**BN通过对每一层的输出规范为均值和方差一致的方法，即严重偏离的分布强制拉回比较标准的分布，这样使得激活输入值落在非线性函数对输入比较敏感的区域，这样输入的小变化就会导致损失函数较大的变化，使得让梯度变大，避免梯度消失问题产生
- **LSTM：**在实际参数更新过程中，LSTM内部复杂的“门”结构控制连乘部分的值σ接近于1，则经过多次连乘(训练)后，梯度也不会消失；而σ的值不会大于1，故不会出现梯度爆炸