A2 分析与简答

A2.1:

1. **Epoch（训练周期）**：一个训练周期是指完整地遍历整个训练数据集一次。在深度学习中，通常会进行多个训练周期，期间模型参数会不断更新以最小化选择的损失函数。每个训练周期包含多次迭代。
2. **Iteration（迭代）**：一次迭代是指使用一批训练数据来一次性更新模型的参数。在随机梯度下降（SGD）中，一次迭代涉及处理一批数据，并且模型参数会在每次迭代后更新。
3. **Batch（批次）**：一批数据是训练数据集的子集，用于一次迭代以更新模型的参数。批次的大小由批大小超参数决定。较大的批次大小可以加快训练速度，但需要更多内存，而较小的批次大小可能导致参数更新中的噪声增加，但能提供更准确的梯度。

选择 `batch_size` 的决策会显著影响训练过程：

- 较小的 `batch_size` ，例如1（也称为在线学习），可能导致参数更新带有噪声，但允许模型快速适应每个数据点。这对于非稳态数据或内存有限的情况可能有用。
- 较大的 `batch_size` 可以提供更稳定的更新，可能导致更快的收敛和更好的泛化，尤其是在训练数据大而多样化的情况下。然而，较大的批次需要更多内存，可能对非稳态数据不够有效。

`batch_size` 的选择是在训练速度和更新质量之间的权衡，通常需要通过实验来找到特定问题和架构的最适合的批次大小。

A2.2:

对于一个简单的1-1-1神经网络：

- **均方误差（MSE）损失函数：**

可有具体推导尝试：给定样本(1，1)

$$L(w_1,\ w_2) = (1-\sigma(w_2\sigma(w_1))$$

$$\frac{\partial\ L(w_1,w_2)}{\partial\ w_2} = -2[(1-\sigma(w_2\sigma(w_1))]^2\sigma(w_2\sigma(w_1))\sigma(w_1)$$

$$\frac{\partial^2\ L(w_1,w_2)}{\partial\ w_2^2} = 2[(1-\sigma(w_2\sigma(w_1))]^2\sigma(w_1)^2\sigma(w_2\sigma(w_1))[3\sigma(w_2\sigma(w_1))-1]$$

$$\frac{\partial\ L(w_1,w_2)}{\partial\ w_1} \ = \ 2[(1-\sigma(w_2\sigma(w_1))]^2\sigma(w_2\sigma(w_1))\sigma(w_1)[1-\sigma(w_1)]$$
$$\frac{\partial^2\ L(w_1,w_2)}{\partial\ w_1^2} \ = \ 2[(1-\sigma(w_2\sigma(w_1))]^2\sigma(w_1)^2\sigma(w_2\sigma(w_1))\sigma(w_1)^2[1-$$
$$\sigma(w_1)]^2\{[\sigma(w_2\sigma(w_1))-1][-2\sigma(w_2\sigma(w_1))+1]+\frac{1}{\sigma(w_1)}-2\}$$

由以上过程可知，二阶偏导都不恒大于0，因此非凸。

- **交叉熵损失函数**：交叉熵损失函数不是凸函数。通常用于分类问题。它导致具有许多局部最小值的损失曲面，使优化更具挑战性。参数空间可以具有复杂的非凸形状。

$$L(w_1,w_2)=log(\sigma(w_2\sigma(w_1)))$$
$$\frac{\partial L}{\partial w_2}=(1-\sigma(w_2\sigma(w_1)))*\sigma(w_1)$$
$$\frac{\partial^2 L}{\partial w_2^2}=\sigma(w_2\sigma(w_1)))(\sigma(w_2\sigma(w_1))-1)\sigma(w_1)^2\leq 0\text{，非凸}$$
$$\frac{\partial\ L}{\partial w_1}=(1-\sigma(w_2\sigma(w_1)))w_2\sigma(w_1)(1-\sigma(w_1))$$
$$\frac{\partial^2 L}{\partial w_1^2}=(1-\sigma(w_2\sigma(w_1)))w_2\sigma(w_1)(1-\sigma(w_1))[-\sigma(w_2\sigma(w_1))w_2\sigma(w_1)(1-\sigma(w_1))+1-$$
$$2\sigma(w_1)]\text{，不恒大于0，非凸}$$

综上可知，其非凸性得证。

A2.3：

在具有高斯噪音的回归问题中，我们将目标变量建模如下：

$$y_i=f(x_i)+\varepsilon_i$$
$$f(x_i) \ = \ w^Tx+b$$

其中，y_i是观测到的目标，$f(x_i)$是我们希望学习的真实函数，ε_i是均值为0、方差为σ^2的高斯噪音。

数据的似然性是高斯分布的乘积：

$$P(y_1,y_2,\ldots,y_n|f,x_1,x_2,\ldots,x_n,\sigma)=\prod_{i=1}^{n}\frac{1}{\sqrt{2\pi\sigma^2}}e^{-(y_i-f(x_i))^2/(2\sigma^2)}$$

最小化负对数似然可表示为：

$$-logP(Y|X) \ = \ \sum_{i=1}^{n}\frac{log(2\pi\sigma^2)}{2} \ + \ \frac{(y_i-f(x_i))}{2\sigma^2}$$

最大化似然性等价于最小化负对数似然，与均方误差（MSE）成正比。因此，最小化MSE等价于最大化观测数据的似然性，这是最大似然估计的原理。

A2.4：

在分类问题中，通常使用交叉熵损失。最大化分类的似然性可以表示为：

$$L(\theta)=\prod_{i=1}^{n}P(y_i|x_i,\theta)$$

最大化这个似然性等价于最小化负对数似然，从而得到交叉熵损失函数。因此，最小化交叉熵损失等价于最大化观测数据的似然性，这是分类问题的最大似然估计方法。

A2.5：

- **L1正则化**：L1正则化通过在损失函数中添加基于模型参数绝对值的惩罚项，鼓励模型参数中的一些值变为精确的零。这有助于稀疏解的产生，因为它迫使某些特征或参数对预测的影响降为零。L1正则化常用于特征选择和简化模型，因为稀疏解具有更少的活跃特征，使模型更易解释。
- **L2正则化**：L2正则化通过在损失函数中添加基于模型参数平方的惩罚项，鼓励模型参数变得小但不迫使它们变为零。这导致平滑和连续的解。L2正则化有助于防止过拟合，减小所有参数的大小而不消除任何参数。它对模型的泛化性能有益。
-

A2.6：

Batch normalization（批归一化，BatchNorm）是深度神经网络中的一种技术，用于提高训练的稳定性和速度。它通过对每个层的激活进行归一化来工作。以下是它对参数优化的作用：

💡
1. 加快收敛速度，若每层数据数量级相差较大，网络将难以收敛，难以训练，分布统一有利于训练和收敛
2. 防止过拟合，每次取样本都是随机取batch避免模型训练到样本顺序的影响，从一定程度上避免过拟合。
3. 防止梯度爆炸或消失，考虑sigmoid函数等，当输入过小或过大，其对应梯度将较小，不利于训练。