人工智能面试问题总结

1、机器学习

- 机器学习算法
- sklearn库的使用
- 前向传播
- 反向传播
- 求导
- 梯度下降算法
- 最小二值法
- 线性回归
- 曲线拟合
- 激活函数
- 损失函数
- 过拟合与欠拟合
- 极大似然估计与交叉熵损失函数
- 逻辑回归
- 常见的数据增强方法
- 标准化，归一化
- pytorch框架
- tensorflow框架

1.1、什么是数据标准化，归一化

1.1.1、数据标准化概念

- **概念：**

  数据标准化（Data Normalization）是一种数据预处理方法，主要用于将数据转换到相同的尺度，以消除量纲的影响，使不同特征的数据具有相似的分布，从而提高机器学习模型的稳定性和收敛速度。

- **特点，优势：**

  标准化后的数据**具有相同尺度，减少特征之间量纲不一致的影响**，有助于提高某些机器学习算法的性能。

- **常见的数据标准化的方法：**

  - Z-score 标准化（标准化得分，零均值标准化）

    公式：

    其中，μ 是特征的均值，σ是特征的标准差。

    假设X={X1,X2,...,Xn}，其均值和标准差分别为：

归一化后的数据均值为 **0**，标准差为 **1**，适用于正态分布数据

这里，经过Z-score 标准化的数据一般会在[-3, 3]之间，有的也可能会落在[-1, 1]或者[-2, 2]之间

1.1.2、数据归一化

- **概念：**

  数据归一化是一种数据预处理方法，主要用于将数据转换到特定的数值范围（通常是 [0,1] 或[-1,1] [][−1,1]），以消除特征量纲（Scale）差异，提高机器学习模型的训练效果和收敛速度。

  归一化后的数据每个特征的取值范围相同，有助于提高某些机器学习算法的性能。

- **均值归一化(Mean Normalization)**

  归一化后的数据范围一般在 **[−1,1]** 之间

  类似于 Min-Max 归一化，但中心点调整为 0

- **Min-Max 归一化（最小-最大缩放）：**

  x 是原始数据

1.1.3、面试问题

问题：有了解过数据预处理中的标准化，归一化方法不？或：什么是数据的标准化与归一化

答：嗯...，数据的标准化，归一化其实都是用来对数据进行预处理的，像有时候有些训练数据，他们的特征值可能会相差比较大，为了提高模型的泛华能力，提高模型的性能，所以我们需要对这些训练数据进行标准化，归一化处理；其实就是通过一些方法让数据转换到相同的尺度，一般就是[0-1]或者[-1, 1]之间。这样可以消除量纲影响，减少模型对某些特征的偏倚， 例外也可以提高机器学习模型的训练效果，稳定性，和收敛速度；增强模型的鲁棒和泛化能力。

1.1.4、标准化和归一化对比

1.2、数据增强手段/方法

- **概念：**

  数据增强是一种用于**扩展训练数据集**的技术，主要用于提高机器学习和深度学习模型的**泛化能力**，减少过拟合，特别适用于计算机视觉、自然语言处理（NLP）和语音识别等领域。

- 常见的数据增强的方法：
  - 对图像做几何变化
    - **随机翻转（Flip）**：水平翻转、垂直翻转
    - **旋转（Rotation）**：随机旋转一定角度（如 ±30°）

        - **缩放（Scaling）**：放大或缩小图像
        - **平移（Translation）**：沿 X/Y 轴随机移动图像
        - **剪裁（Cropping）**：随机裁剪或中心裁剪
    - 图像颜色变换
        - **亮度调整（Brightness Adjustment）**：增加或减少亮度
        - **对比度调整（Contrast Adjustment）**：提高或降低对比度
        - **色调变化（Hue Shift）**：改变色彩平衡
        - **饱和度调整（Saturation Adjustment）**：改变颜色饱和度
        - **Gamma 变换**：调整 Gamma 值以改变亮度
    - 噪声与模糊
        - **高斯噪声（Gaussian Noise）**：添加随机噪声
        - **椒盐噪声（Salt & Pepper Noise）**：随机像素变黑或变白
        - **模糊（Blur）**：高斯模糊、运动模糊
        - **JPEG 压缩（JPEG Compression）**：模拟低质量图像
    - 高级增强
        - **Cutout**：随机遮挡部分图像
        - **Mixup**：将两张图像混合（用于分类任务）
        - **CutMix**：将一张图像的部分区域替换为另一张图像
        - **Style Transfer**：风格迁移

1.2.1、面试问题

问题：对于数据增强这个快，大概有哪些手段或者方法，有了解过吗？

答：数据增强这块，我一般用得比较多的，像图像几何变化，比如图像翻转啊，90,180等，图像缩放，平移，拆解；有时候也会根据情况适当做一些图像颜色变化，比如，亮度的调节，对比度，饱和度，色调这些；例外，对数据进行归一化，标准化处理，这些可能用得比较多吧。主要还是考虑提供模型的泛化能力，因为有时候模型在训练数据上表现的很好，准确率都比较高，但是推理预测很多时候都不准，做在训练或者推理的时候，对数据进行一些数据增加方面的处理，可能会达到更好的效果。

1.3、激活函数

==什么是激活函数呢？激活函数有什么作用呢？常见的激活函数都有哪些？以及如何选择合适的激活函数？==

1.3.1、什么是激活函数及其作用

==问题：知道激活函数不？什么是激活函数？激活函数的作用是什么？==

激活函数主要就是给神经网络增加一些非线性化因素的，因为，在生活中某些实际数据或者很多问题并不是纯线性关系，有的可能是非线性的，或者说存在一些非线性化的特征，这个时候就需要引入激活函数；说真的啊，没有激活函数，网络将仅能执行线性变换，这样即使堆叠多层（就算你网络层数搭得再深），也无法拟合复杂的非线性关系。所以就需要引入激活函数，才能引入一些非线性化的因素。

它可以是得神经网络能够解决更加复杂的问题，逼近复杂的函数关系，同时也增强了神经网络的表达能力和学习能力，

激活函数其实一般是用在输入与输出层之间的，

1.3.2、常见的激活函数有哪些?

==问题：你一般用得比较多的激活函数都有哪些？==

激活函数其实表多，一般我这边用得比较多有，像sigmoid，tanh， ReLu，例外像这个.....，Leaky ReLU，ELU，Softmax，**Mish** ，**Swish**也会偶尔用到。

1.3.3、常见激活函数的特点及对比

==问题：像你刚才提到的几个激活函数，都有什么不一样的特点吗？==

答：特点这块.....

像sigmoid激活函数，它的值是在0-1之间的，它一般用在二分类网络比较多见一点（因为它输出的是一个概率值嘛），所以比较适合二分类任务。不过它也有一些缺点，一个就是存在梯度消失的问题，当输入值很大或者很小的话，梯度接近0， 就会导致反向传播时难以更新参数，还有一点就是它的输出不是以0为中心嘛，可能会导致梯度下降的效率会低一点，表现就是收敛得慢 一点。

对于，tanh，它的值是在-1,1之间的，因为它是过零点以零为中心的，所以相对优化过程会更高效，例外一点，它在非线性变换上比Sigmoid函数更加陡峭，有助于提供更强的非线性特性

而，Relu激活函数，它的值一般是输入值小于0的时候为0， 大于0的时候y=x的，它的计算会比较简单一点，收敛速度会更快一点，相对sigmoid, tanh来说，缓解了梯度消失问题；不过当输入值过大的时候，可能会存在梯度爆炸的问题；它有一个最致命的就是可能存在死亡ReLu，就是当输入值为负时，输出为零，某些神经元可能无法更新。不过这个死亡RuLu问题，在后面提出的LeakyReLU, ELU得到一定的缓解。

差不多大概就这样些特点吧.....

- **Sigmoid**

  公式： ，**输出范围**：（0，1）

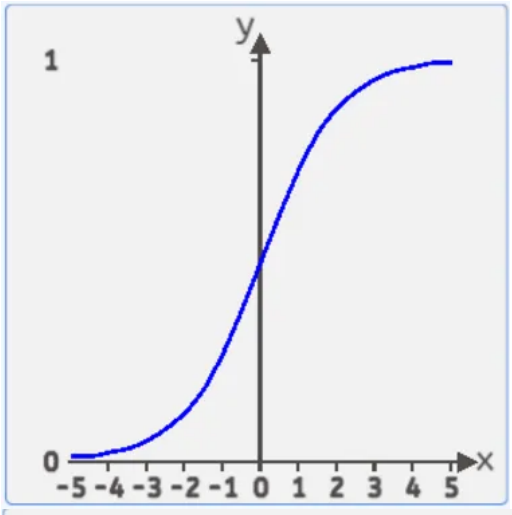
  导数： ， 导数范围：（0，0.5）

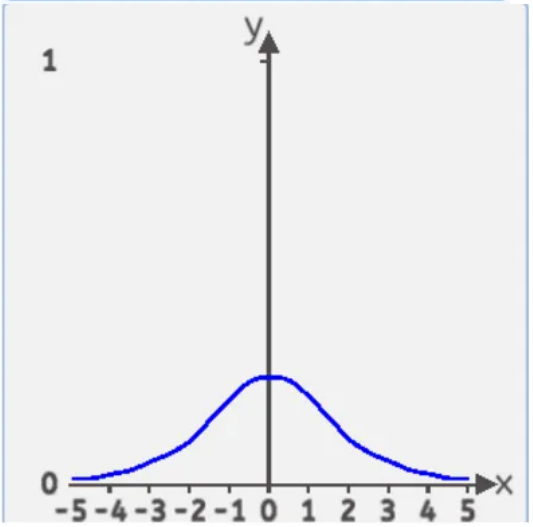
  优点：输出为概率值，适合二分类任务。

  缺点：

  - 梯度消失问题：当输入值很大或很小时，梯度接近 0，导致反向传播时难以更新参数。
  - 输出不是零中心的，可能导致梯度下降时优化效率较低。

  图像：





- **tanh**

  公式： ，输出范围：(-1, 1)

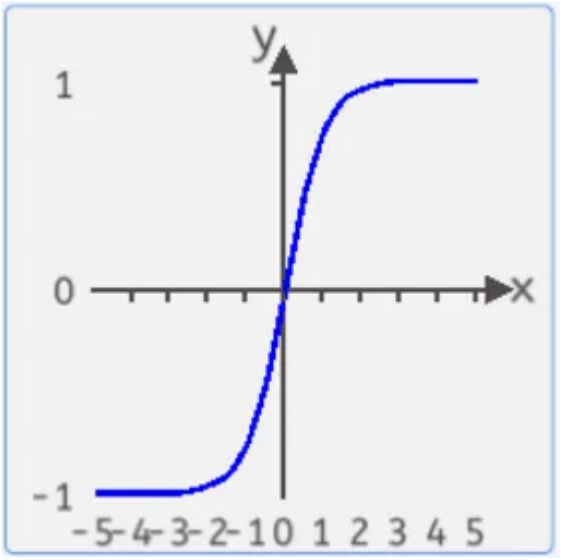
  导数： ，输出范围:（0，1）

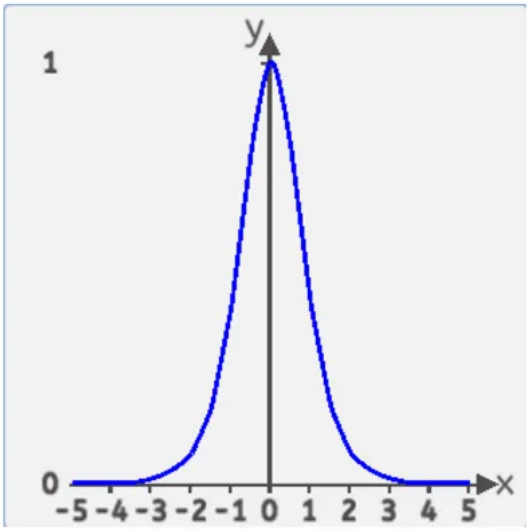
  优点：

  - 输出是零中心的，优化过程更高效。
  - 它在非线性变换上比Sigmoid函数更加陡峭，有助于提供更强的非线性特性

  缺点：仍然存在梯度消失问题，尤其是输入值很大或很小时。

  图像：





- **ReLU**

  公式： ，输出范围：x<0时为0， x大于0时，y=x -------->[0,∞)

  导数： ，输出范围：x<0时，导数为0， x大于0时，导数为1

  优点：

  - 计算简单，收敛速度快，所以，大规模深度神经网络中，ReLU函数的计算速度远快于Sigmoid和tanh函数，加快了模型训练的速度。
  - 在正数区域梯度为常数，相对sigmoid, tanh来说，缓解了梯度消失问题。
  - 稀疏激活性，在输入小于零的情况下，ReLU函数的输出是零，这表现为稀疏激活性。这意味着在激活后，一部分神经元将被激活，而其他神经元则保持不活跃。这有助于减少神经网络中的冗余计算和参数数量，提高了网络的泛化能力。

  缺点:

  - 当输入值很大时，可能导致梯度爆炸。
  - **死神经元问题**：当输入值为负时，输出为零，某些神经元可能无法更新。

- **Leaky ReLU（带泄漏的 ReLU）**

  公式： ，**输出范围**：(−∞,∞)

  优点：

  - 解决了 ReLU 的“死神经元”问题，通过引入负区间的斜率，避免神经元永远输出零。

缺点：

- 可能会存在梯度爆炸问题。
- 需要手动调节 α 值。

- **ELU（Exponential Linear Unit）**

  公式： ，**输出范围**：

  优点：

  - 输出为零中心，避免了 ReLU 和 Leaky ReLU 的优化问题。
  - 处理负值部分时更为平滑。

  缺点：

  - 计算相对复杂。
  - 当 α\alphaα 参数较大时，可能会出现梯度爆炸问题。

- **Softmax**

  公式：

  输出范围：(0,1)，且所有输出的和为 1。

  优点：常用于多分类问题的输出层，能够将网络的输出转换为概率值，表示每个类别的预测概率。

  缺点：不适用于回归问题，只适合分类问题。

- **Swish**

  公式：

  输出范围：(−∞,∞)(-\infty, \infty)(−∞,∞)

  优点：

  - 在深度神经网络中表现优越，特别是对梯度消失问题有一定的缓解作用。
  - 相比 ReLU，Swish 可以更好地保留负区间的信息，提供更好的训练性能。

  缺点：相比 ReLU，计算稍微复杂

- **Mish**

  公式：

  输出范围：

  优点：被认为是 Swish 的改进版，在多个任务中表现出更强的学习能力，尤其是在深层神经网络中。

  缺点：计算比 ReLU 更复杂，训练速度可能稍慢。

1.4、损失函数

- 什么是损失函数？它在训练神经网络时的作用是什么？
- 损失函数和目标函数有什么区别？
- 在机器学习中，为什么要最小化损失函数？最小化的目标是什么？

==问题1：什么是损失函数？它在训练神经网络时的作用是什么？==

答：**损失函数（Loss Function）**，也称为 **代价函数（Cost Function）**，是衡量神经网络输出与真实标签之间差异的函数。它的作用是在模型训练过程中提供一个量化的标准，用于评估模型的预测能力，并指导优化过程，以便使得模型的预测更接近真实值。

损失函数作用可归纳几点：1）衡量误差 2）反向传播 3）优化目标 4）模型评估标准

==问题2：常见的损失函数有哪些？==

答：均方误差（MSE，Mean Squared Error）， 它一般用于回归问题，衡量预测值与真实值之间的平方差，**交叉熵损失（Cross-Entropy Loss）**：常用于分类问题，特别是二分类和多分类问题。它衡量的是模型预测的类别概率分布与真实标签之间的差异。这个两个是我这边用得最多的.....

==问题3：损失函数和目标函数有什么区别？==

答：其实在很多情况下损失函数就是目标函数，机器学习和神经网络训练中有些相似；不同点在于，目标函数不仅仅是损失函数，它可以包含**多个部分**，如损失函数和正则化项的组合；目标函数可以是损失函数的加权组合，也可以是包含额外约束的函数。

公式：

其中，J(θ)是目标函数，L(y,y^) 是损失函数，θ是模型的参数。

==问题4：在机器学习中，为什么要最小化损失函数？最小化的目标是什么？==

答：为什么要最小化损失函数？这个.....，因为它主要是用来衡量预测误差的啊，我们训练模型的目的就是让预测值与真实值更接近，也就是误差最小嘛；比如均方误差MSE损失函数， ，他是预测值-去真实值的平方和取平均，那如果损失函数越少，表明预测值越接近真实值，那误差越小，模型的准确度就越高；例外，最小化损失函数也可以为我们指导优化训练过程，通过优化算法（如梯度下降）来最小化损失函数。在每次迭代中，优化算法通过计算损失函数对模型参数的梯度，然后调整参数，使得损失函数值逐步减小。最小化的目标就是让它尽量接近于0了，通过最小化训练误差，可以提高模型的预测能力，泛化能力，从来优化模型的参数。

==问题5：常见的损失函数都有哪些，一般适用的场景？==

答：我这边用得比较多的就是2个，均方误差（MSE，Mean Squared Error），**交叉熵损失（Cross-Entropy Loss）**，均方误差一般都是用在回归问题中，交叉熵损失一般都是用在分类问题中比较多。

追问：还有用过其他的吗？

其他的？，没怎么用过....，我一样用得多的就是这样多一点.....

==提醒：不会就不会，要不不好意思.....，自信回答，没用过就没用过.....==

- 如何判断模型是否已经收敛，损失函数是否达到了最优值？
- 在计算损失函数时，如何避免出现梯度爆炸或梯度消失的问题？
- 如果损失函数值过小或过大，如何调整网络训练过程中的数值稳定性问题？

- 在使用交叉熵损失函数时，如果遇到数值下溢或上溢，应该如何处理？
- 在某些情况下，损失函数可能会收敛得很慢，你如何调整损失函数或者训练过程来加速收敛？
- 你如何判断一个损失函数的选择是否合适，特别是在模型效果不好时？

1.6、优化器

- 什么是优化器（Optimizer）？它的作用是什么？
- 什么是梯度下降法（Gradient Descent）？它的工作原理是什么？
- 梯度下降有哪几种常见的变体？请解释它们的优缺点。
- 常见的优化器有哪些？请简要说明它们的原理。
- Adam优化器的原理是什么？它相比传统的SGD有什么优势？
- 在实际应用中，如何选择合适的优化器？
- 学习率（Learning Rate）在优化中的作用是什么？如何选择合适的学习率？
- 在训练深度神经网络时，为什么Adam优化器常常优于SGD？
- 如何处理深度学习中梯度消失或梯度爆炸的问题？
- RMSprop和AdaGrad的区别是什么？它们适合哪些类型的任务？
- 在训练大型神经网络时，如何调节优化器以避免过拟合？
- 您曾经遇到过优化器收敛不好的问题吗？您是如何处理的？
- 如果一个任务收敛较慢或者无法收敛，您会从哪些方面入手来调整优化器？

1.7、过拟合与嵌拟合

- 什么是过拟合（Overfitting）和欠拟合（Underfitting）？它们分别表现为什么样的情况？
- 如何判断一个模型是过拟合还是欠拟合？
- 为什么过拟合会发生？为什么欠拟合会发生？
- 如何避免或减轻过拟合？
- 如何避免或减轻欠拟合？
- 你如何调整模型的复杂度来避免过拟合和欠拟合？
- 假设你有一个深度神经网络模型，在训练过程中发现它出现了过拟合，你会采取哪些具体措施来解决？
- 在实际项目中，你如何平衡训练数据的大小与模型的复杂度，以避免过拟合或欠拟合？
- 过拟合的模型一定是复杂的模型吗？
- 欠拟合的模型一定是简单的模型吗？
- 你在之前的项目中遇到过过拟合或欠拟合的情况吗？你是如何解决的？
- 你如何在一个较小的数据集上训练模型，同时避免过拟合？