第四讲 神经网络

2019年4月23日 9:02

1.反向传播

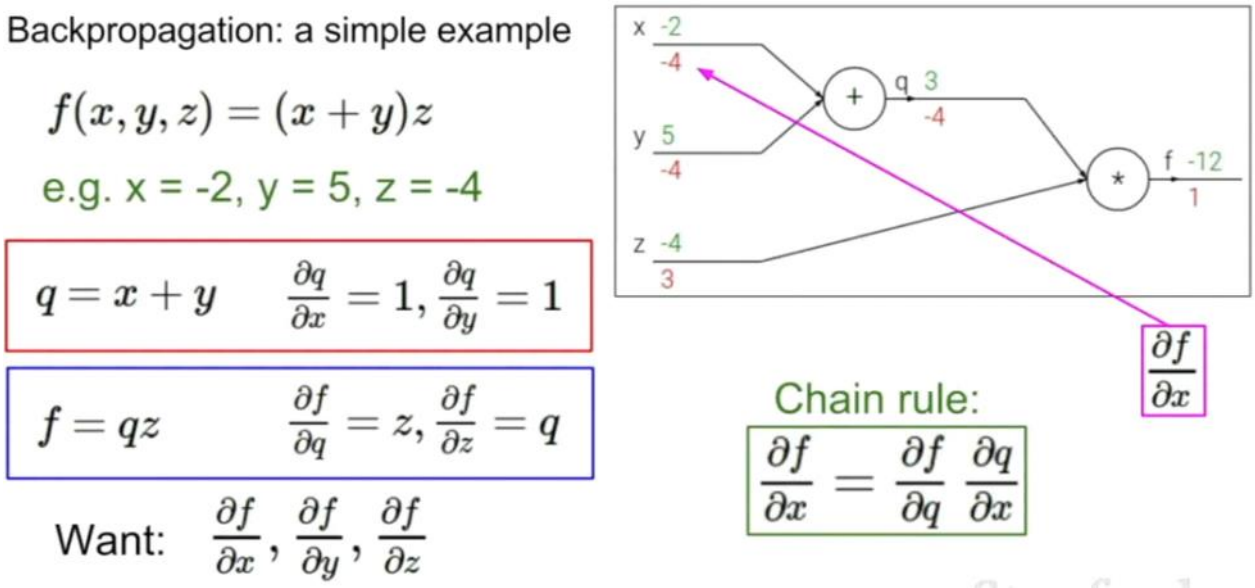


图4.1.1 求梯度

梯度：各个维度的偏导数组成的向量。其意义是函数在梯度方向上升最快。

当我们得到一个损失函数之后，实际相当于参数W和b作为函数的输入，loss值为输出，整个优化问题就变成了在损失函数上求最小值。而梯度方向是损失函数上升最快的方向，所以沿着反方向能是损失函数快速缩小。

为什么要用梯度？ 很多时候一个函数的最大最小值是无法直接得到的，使用梯度每次给w值增加沿反梯度方向一个值会使函数值变小，就这样一直尝试一直尝试，直到函数走到一个最小点(有可能是全局也有可能是局部)。**这一点也充分利用了计算机在计算上的优势。**
相关知识：凸优化，函数最值或极值。

这节课的小姐姐讲的有点乱，接着上次的课程，损失函数讲完了到优化的时候，反向传播是为了计算损失函数的最小值(极小值)，这是目的，而使用反向传播是方法。（小姐姐一来就开始讲求偏导，求梯度，就是不知道要干嘛，没有讲明为什么这么做，这也导致课上同学提出的问题很让人费解）所以接下来讲反向传播，在讲反向传播中求梯度中没有说明输入x和y其实对应的是(上节课中)损失函数中的参数和偏移量，我潜意识认为输入x和y对应的是数据集，所以导致很多问题无法理解。

反向传播：复杂函数拆分成链式函数（复合函数），用链式法则逐个求偏导数

2.神经网络

2.1前馈过程

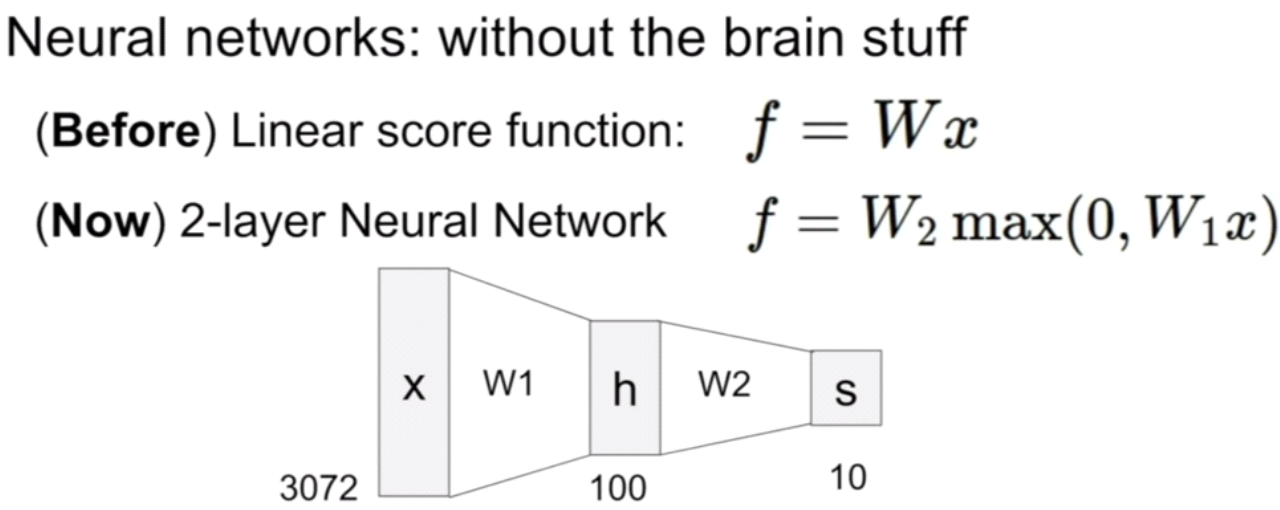


图4.2.1 线性模型与神经网络模型

线性模型与神经网络比较，其实神经网络输入层到第一隐层是线性的，W1x，经过第一隐层的激活函数之后在网后就变成了非线性了。

在后面会讲到，当神经元个数和网络层数足够多时，神经网络可以模拟任何函数，包括非线性函数。

模拟或拟合任意函数？？：

1.我们需要解决的问题(**目的**)，经过处理(**特征工程**)后，其**数据集**和对应的**结果集**是否符合某个函数的映射关系(**模型**)。

2.假如要解决的问题经过处理之后符合某个**函数**，那么我们就使用神经网络来模拟这个**函数**。

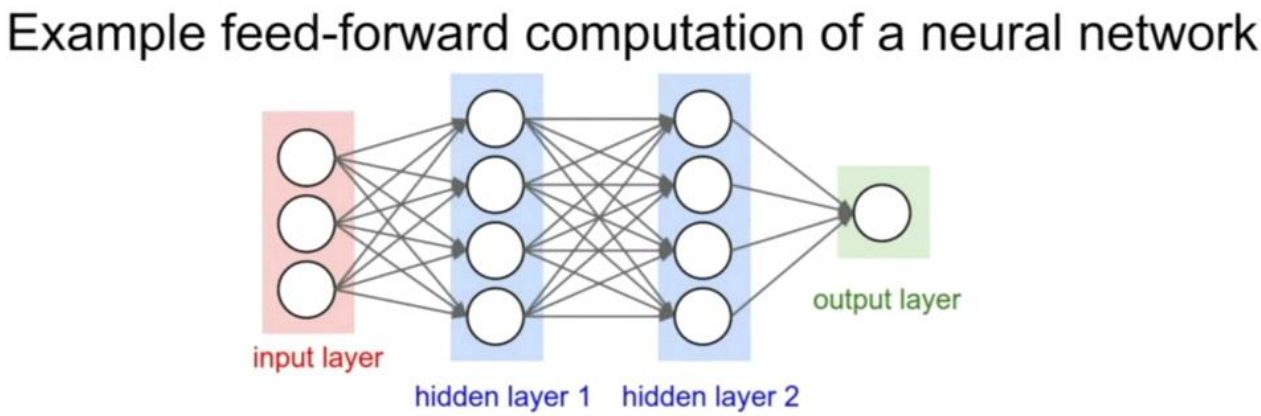
3.如何模拟？ 神经网络已经给了一套解决这个问题的方案，使用大量的数据集来训练，(假如数据集符合某个函数)，那么经过**优化**最后神经网络就会越来越逼近这个**函数**。

4.**优化过程**：通过定义损失函数得到神经网络权重与误差的关系，将问题转化为函数求最值(极值)。

小结：神经网络解决问题的思路是，假设某个问题是求从状态1到状态2的变化过程，并且变化过程是符合某个函数的，这时候就用神经网络来逼近这个函数。（监督学习）

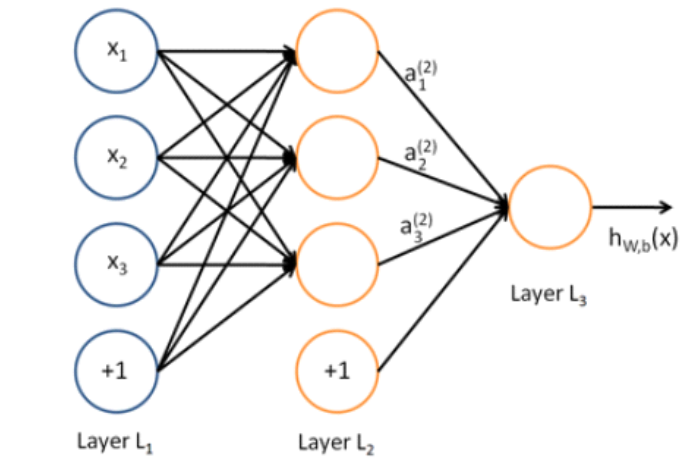
简述起来就是神经网络在尽量的拟合数据集。 ？人是这么学习的吗？

应用时要注意：1.变化过程是否真的符合某个函数（变量和因变量之间是否真的有映射关系）。




```
# forward-pass of a 3-layer neural network:
f = lambda x: 1.0/(1.0 + np.exp(-x)) # activation function (use sigmoid)
x = np.random.randn(3, 1) # random input vector of three numbers (3x1)
h1 = f(np.dot(W1, x) + b1) # calculate first hidden layer activations (4x1)
h2 = f(np.dot(W2, h1) + b2) # calculate second hidden layer activations (4x1)
out = np.dot(W3, h2) + b3 # output neuron (1x1)
```

图4.2.2 神经网络计算模型流程



$$a_1^{(2)} = f(W_{11}^{(1)}x_1 + W_{12}^{(1)}x_2 + W_{13}^{(1)}x_3 + b_1^{(1)})$$
$$a_2^{(2)} = f(W_{21}^{(1)}x_1 + W_{22}^{(1)}x_2 + W_{23}^{(1)}x_3 + b_2^{(1)})$$
$$a_3^{(2)} = f(W_{31}^{(1)}x_1 + W_{32}^{(1)}x_2 + W_{33}^{(1)}x_3 + b_3^{(1)})$$
$$h_{W,b}(x) = a_1^{(3)} = f(W_{11}^{(2)}a_1^{(2)} + W_{12}^{(2)}a_2^{(2)} + W_{13}^{(2)}a_3^{(2)} + b_1^{(2)})$$

图4.2.3 前馈计算过程

这讲中应该讲神经网络的前馈过程，但是这位小姐姐讲师讲的有点乱，听完没什么思绪。查了查资料总结一下前馈过程。

1.首先输入层每个输入乘第一个参数矩阵，其结果的值到达下一层第一个神经元节点。

2.经过神经元节点激活函数之后形成了一个新的数据源准备作为下一层的输入。

3.依次类推，完成第一隐藏层的计算。

4.第一隐藏层计算完毕之后，根据第一层的值计算第二隐藏层。

5.后续隐藏层的计算方式同此一样。

f()为激活函数，常见：Sigmoid函数，tanh函数，ReLU函数，Softmax函数后面会讲到。

疑问：为什么说神经网络能模拟任意函数?

Memery error 问题：
按照课程，在使用KNN给cifar-10做分类时出现了memery error，解决办法是，在与预测类比较时，一次读取一个batch的数据，做欧式距离保存其index和距离，这样由原来的矩阵由原来的(10000,32,32,3)变成(10000,1)，先保存，删除之前读取的batch数据，再在下一个batch做比较，然后追加到(10000,1)矩阵，依次读取5个batch得到一个(50000,1)的矩阵，这样sort一下然后取前k个值，得出KNN预测结果。源码已存放在个人的github仓库。