深度学习入门者必看：25个你一定要知道的概念

原标题：深度学习入门者必看：25个你一定要知道的概念

王瀚森 编译自 Analytics Vidhya

量子位 出品 | 公众号 QbitAI

> 人工智能，深度学习，机器学习……不管你在从事什么工作，都需要了解这些概念。否则的话，三年之内你就会变成一只恐龙。 —— 马克·库班

库班的这句话，乍听起来有些偏激，但是“话糙理不糙”，我们现在正处于一场由大数据和超算引发的改革洪流之中。

首先，我们设想一下，如果一个人生活在20世纪早期却不知电为何物，是怎样一种体验。在过去的岁月里，他已经习惯于用特定的方法来解决相应的问题，霎时间周围所有的事物都发生了剧变。以前需要耗费大量人力物力的工作，现在只需要一个人和电就能完成了。

而在现在的背景下，机器学习、深度学习就是新的“电力”。

所以呢，如果你还不了解深度学习有多么强大，不妨就从这篇文章开始。在这篇文章中，作者Dishashree Gupta为想了解深度学习的人，罗列并解释了25个这一领域最常用的术语。

这25个术语被分成三组：

神经网络中的基础概念(包含常用的一些激活函数)

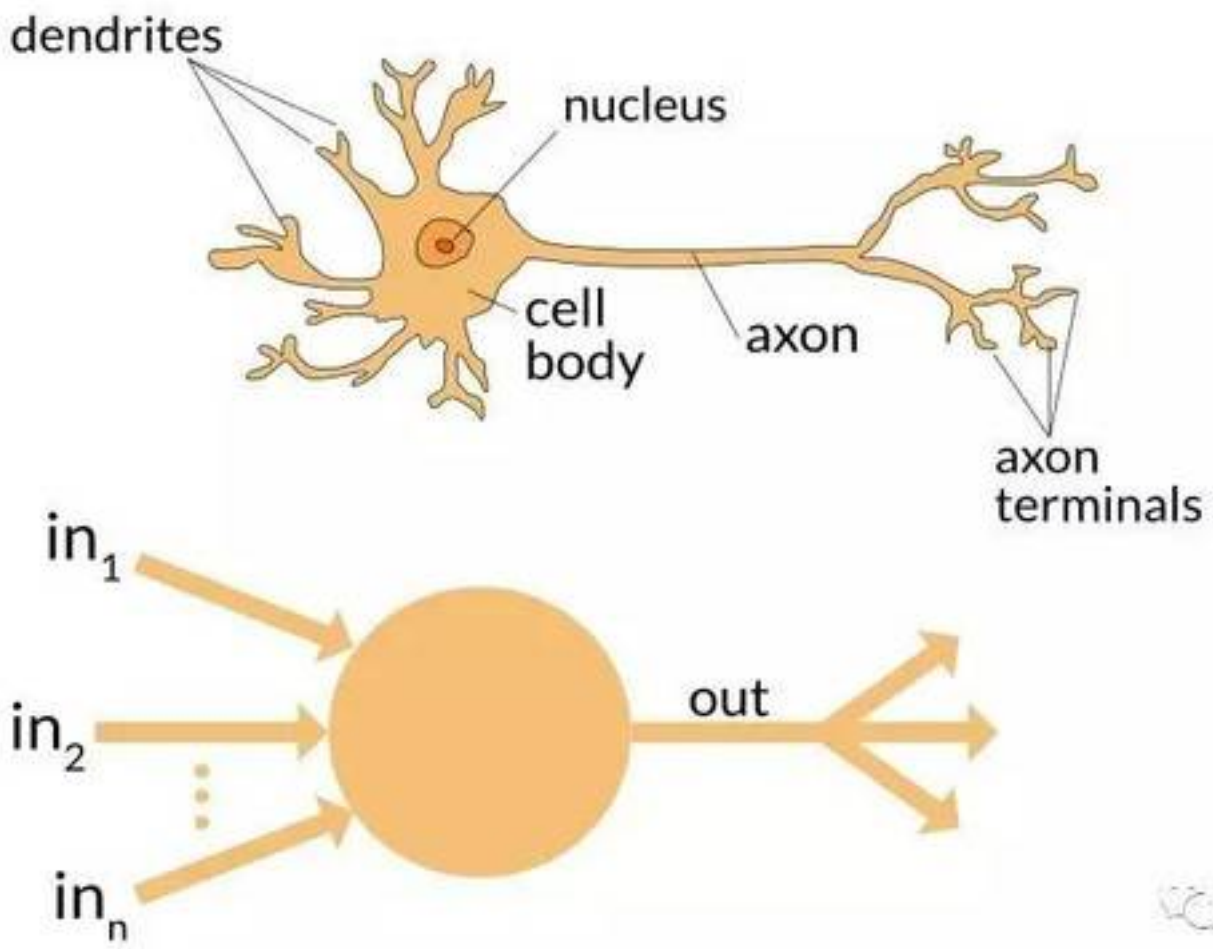
卷积神经网络

递归神经网络

基础概念：（1）神经元(Neuron)

正如我们大脑中的基本组成单元，神经元是组成神经网络的基础结构。设想一下当接触到新的信息时，我们的身体会对其进行处理，最后产生一些特定的反应。

相似地，在神经网络中，在收到输入的信号之后，神经元通过处理，然后把结果输出给其它的神经元或者直接作为最终的输出。

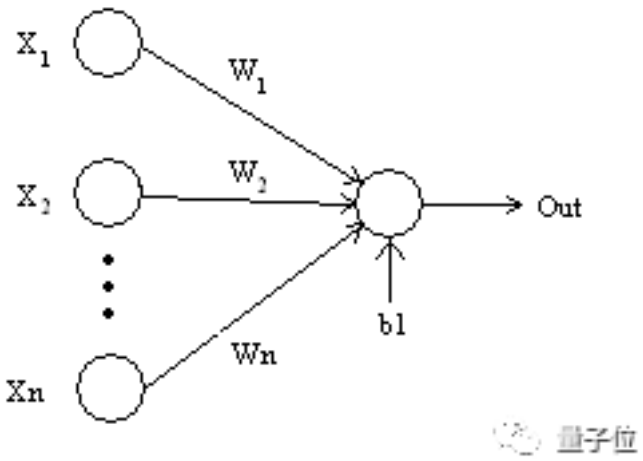


(2) 加权/权重(Weights)

当输入信号进入到神经元后，会被乘以相应的权重因子。举例来说，假设一个神经元有两个输入信号，那么每个输入将会存在着一个与之相应的权重因子。在初始化网络的时候，这些权重会被随机设置，然后在训练模型的过程中再不断地发生更改。

在经过训练后的神经网络中，一个输入具有的权重因子越高，往往意味着它的重要性更高，对输出的影响越大。另一方面，当权重因子为0时意味着这个输入是无价值的。

如下图所示，假设输入为a，相应的权重为W1。那么通过赋权节点后相应的输入应变为a*W1。



(3) 偏置/偏倚(Bias)

除了权重之外，输入还需要经过另外一种线性处理，叫做偏置。通过把偏置b与加权后的输入信号a*W1直接相加，以此作为激活函数的输入信号。

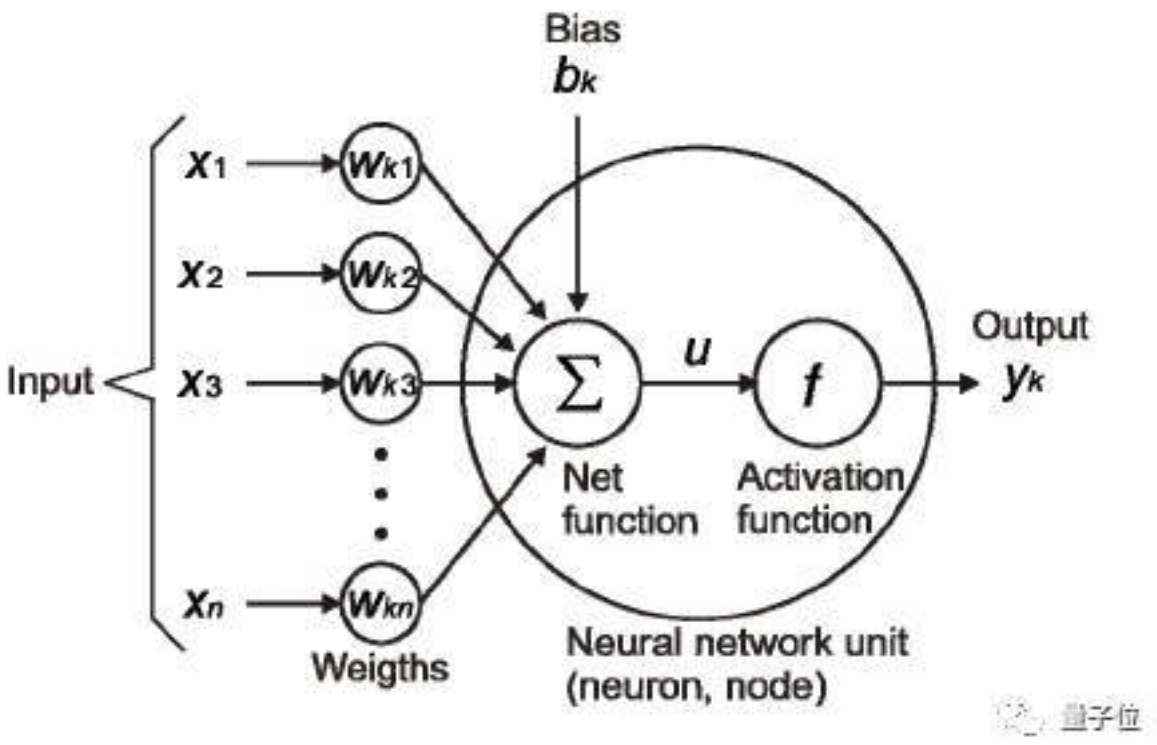
(4) 激活函数

之前线性处理之后的输入信号通过激活函数进行非线性变换，从而得到输出信号。即最后输出的信号具有f(a*W1+b)的形式，其中f()为激活函数。

在下面的示意图中， 设X1…Xn等n个输入分别对应着权重因子Wk1…Wkn以及相应的偏置b1…bn。我们把输入Xi乘以对应的权重因子Wki再加上bi的结果称为u。

u=∑w*x+b

这个激活函数f是作用在u上的，也就是说这个神经元最终的输出结果为yk = f(u)

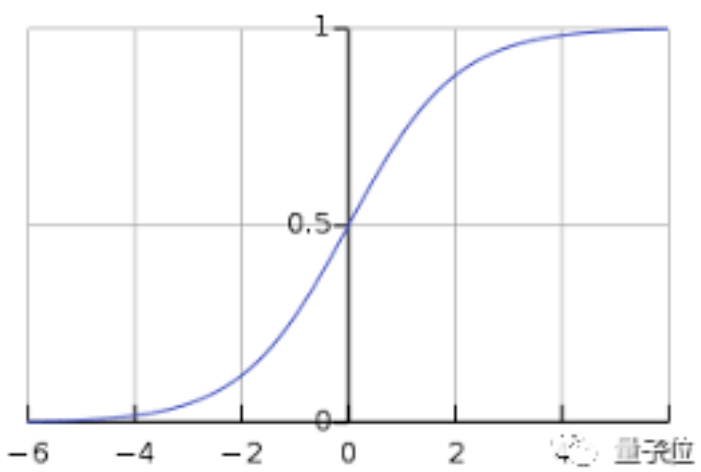


接下来我们讲一讲常用的一些激活函数：Sigmoid函数， 线性整流函数(ReLU) 和 softmax函数

(a) Sigmoid函数

作为最常用的激活函数之一，它的定义如下：

$$sigmoid(x) = 1/(1+e^{-x})$$



△来源：维基百科

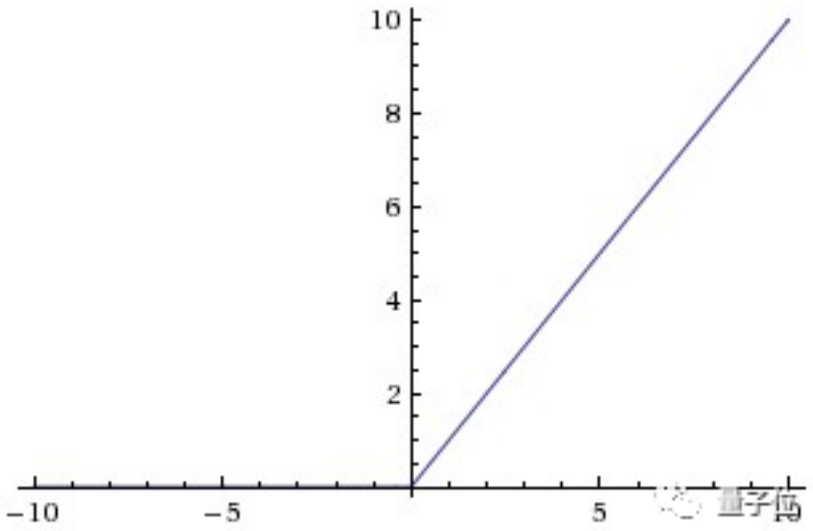
sigmoid函数为值域在0到1之间的光滑函数，当需要观察输入信号数值上微小的变化时，与阶梯函数相比，平滑函数(比如Sigmoid函数)的表现更好。

(b) 线性整流函数(ReLU-Rectified Linear Units)

近来的神经网络倾向于使用ReLU替代掉sigmoid函数作为隐层的激活函数，它的定义如下：

f(x) = max(x,0).

当x大于0时，函数输出x，其余的情况输出为0。函数的图像是：



△来源：cs231n

使用ReLU函数的好处是，对于所有大于0的输入，导数是恒定的，这能够加快训练网络的速度。

(c) softmax函数

softmax激活函数通常应用在分类问题的输出层上。

它与Sigmoid函数相似，唯一的不同是softmax函数输出结果是归一化的。sigmoid函数能够在双输出的时候奏效，但当面对多种类分类问题的时候，softmax函数可以方便地直接将各个分类出现的概率算出。

(5) 神经网络

神经网络是构成深度学习系统的框架。神经网络的任务是找到一个未知函数的近似表达方式，它是由彼此相连的神经元所组成，这些神经元会在训练网络的过程中根据误差来更改它们的权重和偏置。激活函数将非线性变化用线性变化的组合来表示，最终产生输出。

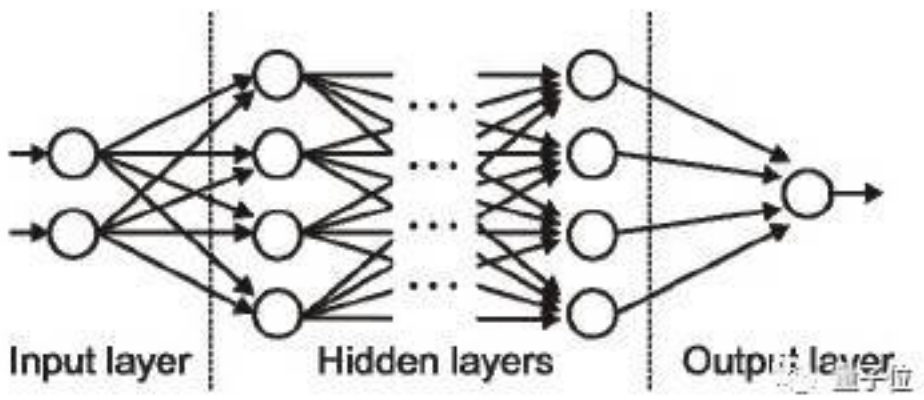
关于神经网络最好的定义是由Matthew Mayo给出的：

神经网络是由大量彼此相连、概念化的人造神经元组成的，这些神经元彼此之间传递着数据，相应的权重也会随着神经网络的经历而进行调整。神经元们有着激活的阈值，当它们遇到相应的数据以及权重时会被激活，这些被激活的神经元组合起来导致了“学习”行为的产生。

(6) 输入层/输出层/隐藏层

从名字中就能看出，输入层代表接受输入数据的一层，基本上是网络的第一层；输出层是产生输出的一层，或者是网络的最后一层，而网络中间的处理层叫做隐藏层。

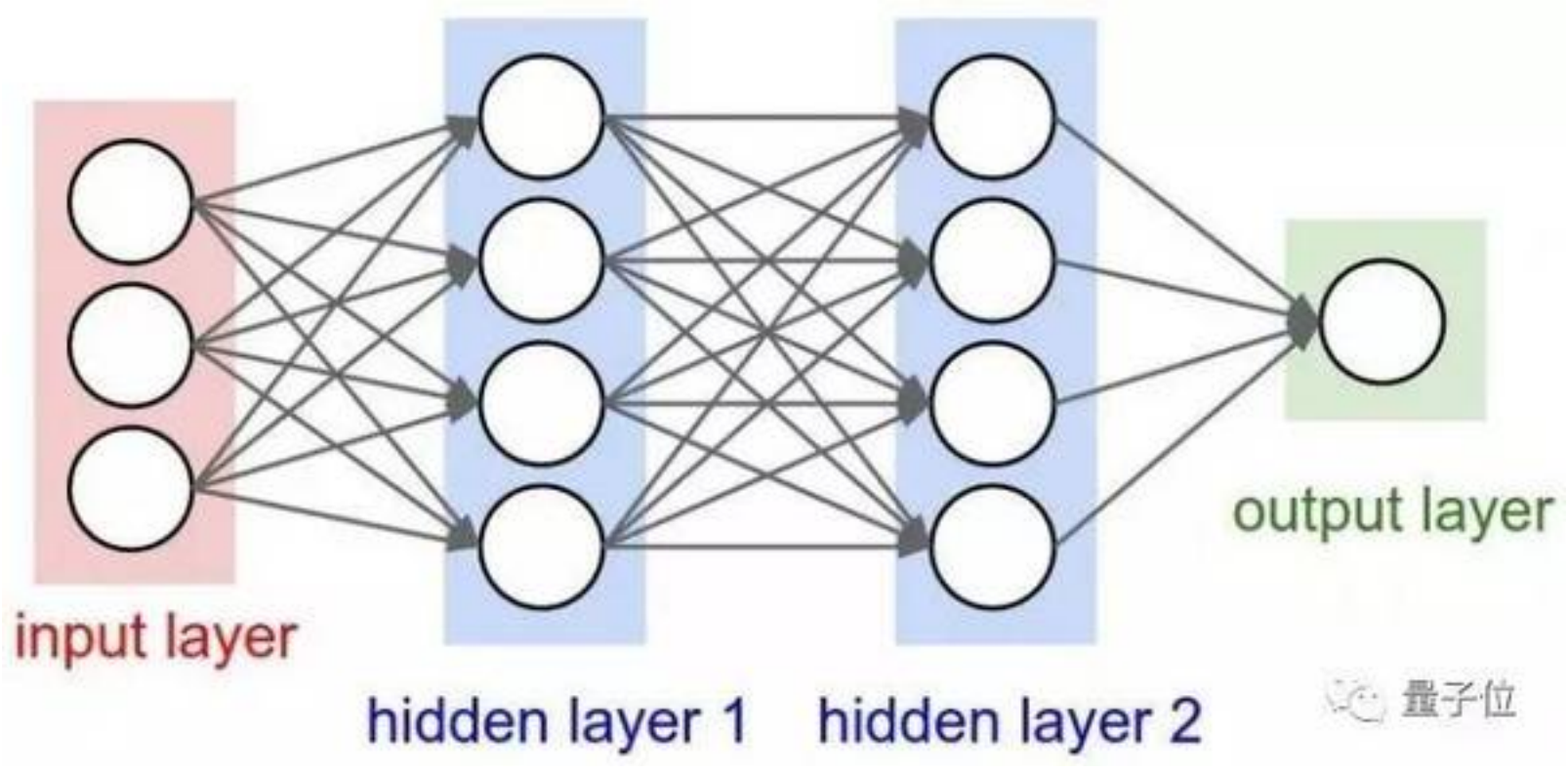
这些隐藏层对输入的数据进行特定的处理，再将其输入到下一层。输入层和输出层是可见的，而中间层通常是被隐藏起来的。



△来源：cs231n (7) 多层感知器(MLP-Multi Layer Perceptron)

一个单一的神经元不能够完成复杂的任务，因此需要将它们堆叠起来工作进而产生有用的输出。

最简单的神经网络包括一个输入层、一个隐藏层和一个输出层。每一层都由多个神经元组成，每一层的每个神经元都与下一层中的所有神经元相连。这样的网络可以被称为是全连接网络。



(8) 正向传播(forward propagation)

正向传播是指信号从输入层经过隐藏层到输出层的传输过程。在正向传播

中，信号是沿着单一方向进行传播，即输入层给隐藏层提供输入，进而最终产生相应的输出。

(9) 成本函数(cost function)

在神经网络的建造过程中，建造者们希望输出的结果能够尽可能地接近实际值，因此使用成本函数来描述网络的这种准确性。

神经网络的目标是增加预测的准确性从而减少误差，即最小化成本函数。通常情况下，最优化的输出结果往往对应着成本函数的最小值。

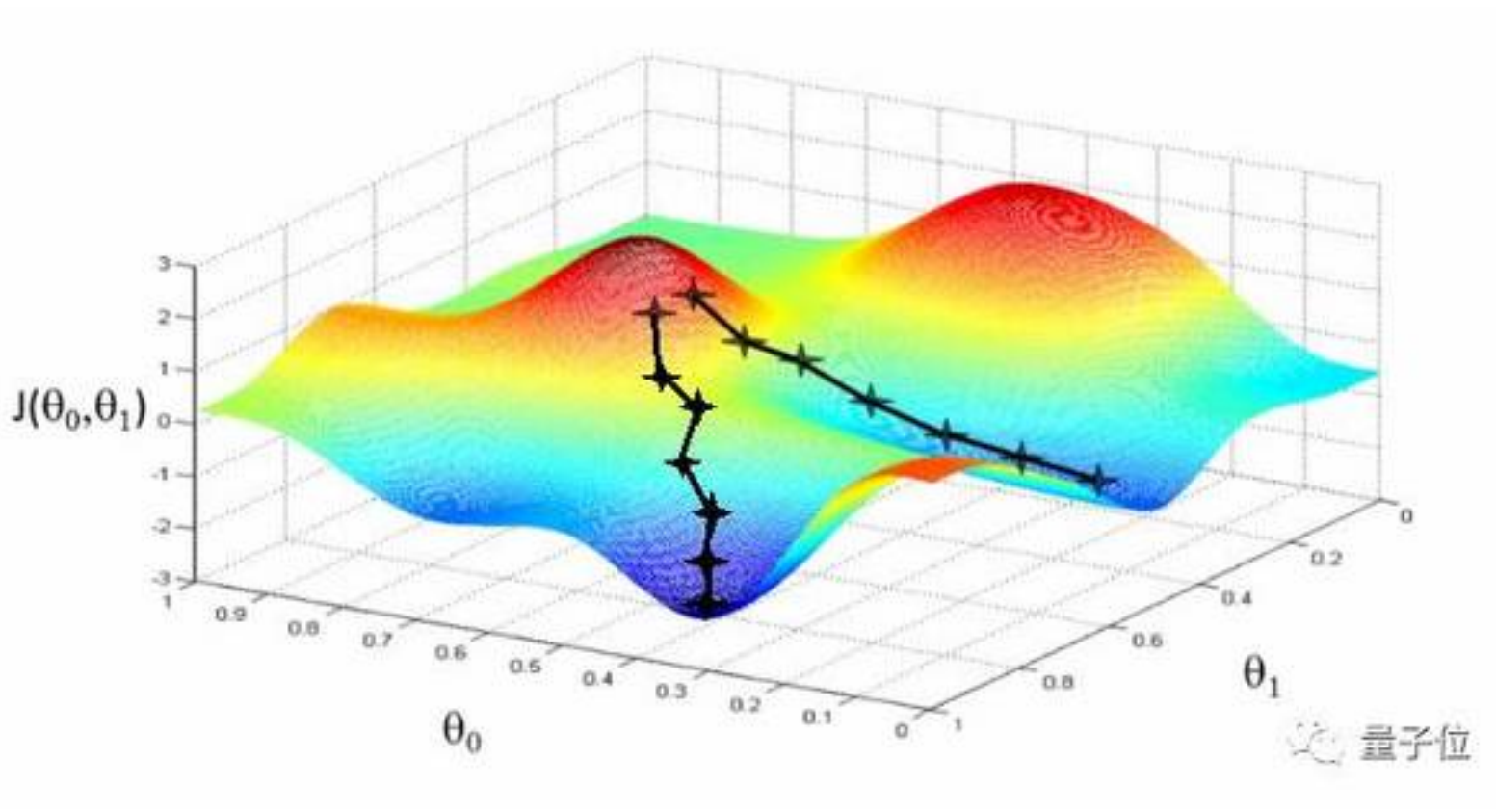
如果采用均方误差作为成本误差，即表示为$C= 1/m \sum(y - a)2$，其中m是训练过程中输入数据的个数，a是相应的预测值，y代表实际值。

模型学习的过程就是围绕着最小化成本而进行的。

(10) 梯度下降(gradient descent)

梯度下降是一种最小化成本函数的优化算法。

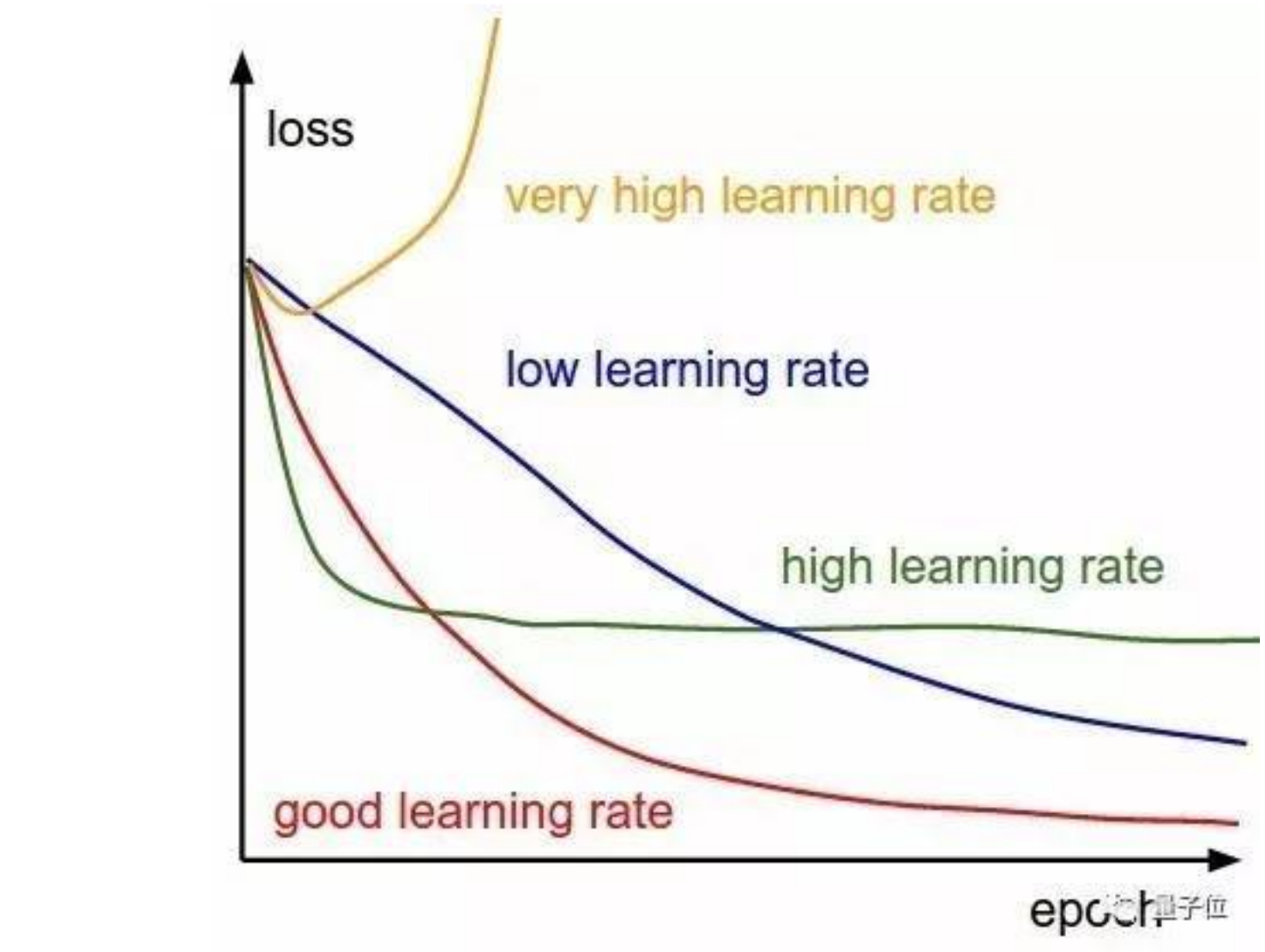
在梯度下降中，从起始点x开始，一次移动一点，比如说移动delta h，然后将位置信息更换为x-delta h，如此重复下去，直到达到局部的极小值，此时认为极小值就是成本最小的地方。



数学上说，沿着函数的负梯度运动就能得到函数的局域最小值。

(11) 学习速率(learning rate)

学习速率定义为在每次迭代过程中对成本函数的最小化次数。简单来说，学习速率就是指朝着成本函数最小值的下降速率。选择学习速率需要很谨慎，过大会导致可能越过最优解，过小会导致收敛花费太多的时间。



(12) 反向传播(back propagation)

在定义一个神经网络的过程中， 每个节点会被随机地分配权重和偏置。一次迭代后，我们可以根据产生的结果计算出整个网络的偏差，然后用偏差结合成本函数的梯度，对权重因子进行相应的调整，使得下次迭代的过程中偏差变小。这样一个结合成本函数的梯度来调整权重因子的过程就叫做反向传播。

在反向传播中，信号的传递方向是朝后的，误差连同成本函数的梯度从输出层沿着隐藏层传播，同时伴随着对权重因子的调整。

(13) 分批(Batches)

当在训练一个神经网络的时候，相对于一次性将所有的数据全输入进去，有一个更好的方法：先将数据随机地分为几个大小一致的数据块，再分批次输

入。跟一次性训练出来的模型相比，分批训练能够使模型的适用性更好 。

(14) 周期(epochs)

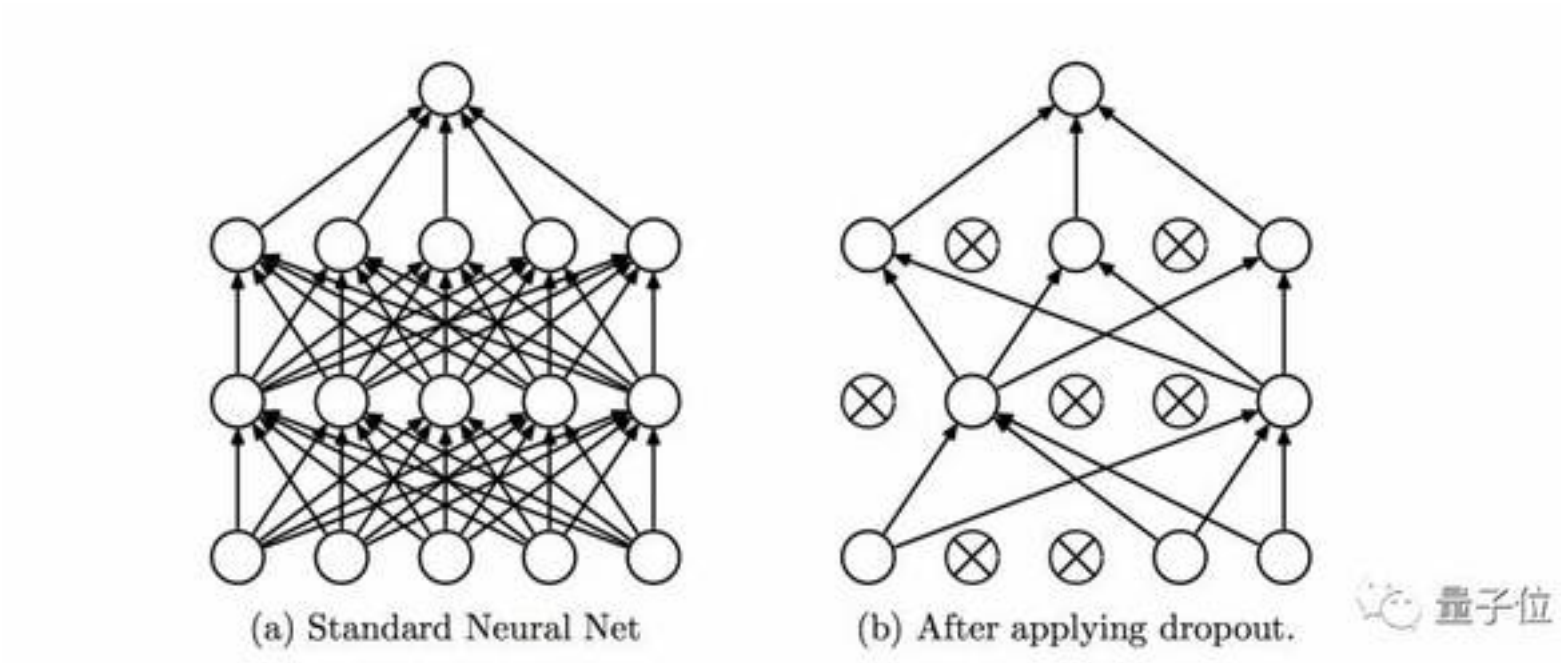
一个周期表示对所有的数据批次都进行了一次迭代，包括一次正向传播和一次反向传播，所以一个周期/纪元就意味着对所有的输入数据分别进行一次正向传播和反向传播。

训练网络周期的次数是可以选择的，往往周期数越高，模型的准确性就越高，但是，耗时往往就越长。同样你还需要考虑如果周期/纪元的次数过高，那么可能会出现过拟合的情况。

(15) Dropout方法

Dropout是一个可以阻止网络过拟合的规则化方法。就像它的名字那样，在训练过程中隐藏的某些特定神经元会被忽略掉(drop)。这意味着网络的训练是在几个不同的结构上完成的。

这种dropout的方式就像是一场合奏，多个不同结构网络的输出组合产生最终的输出结果。

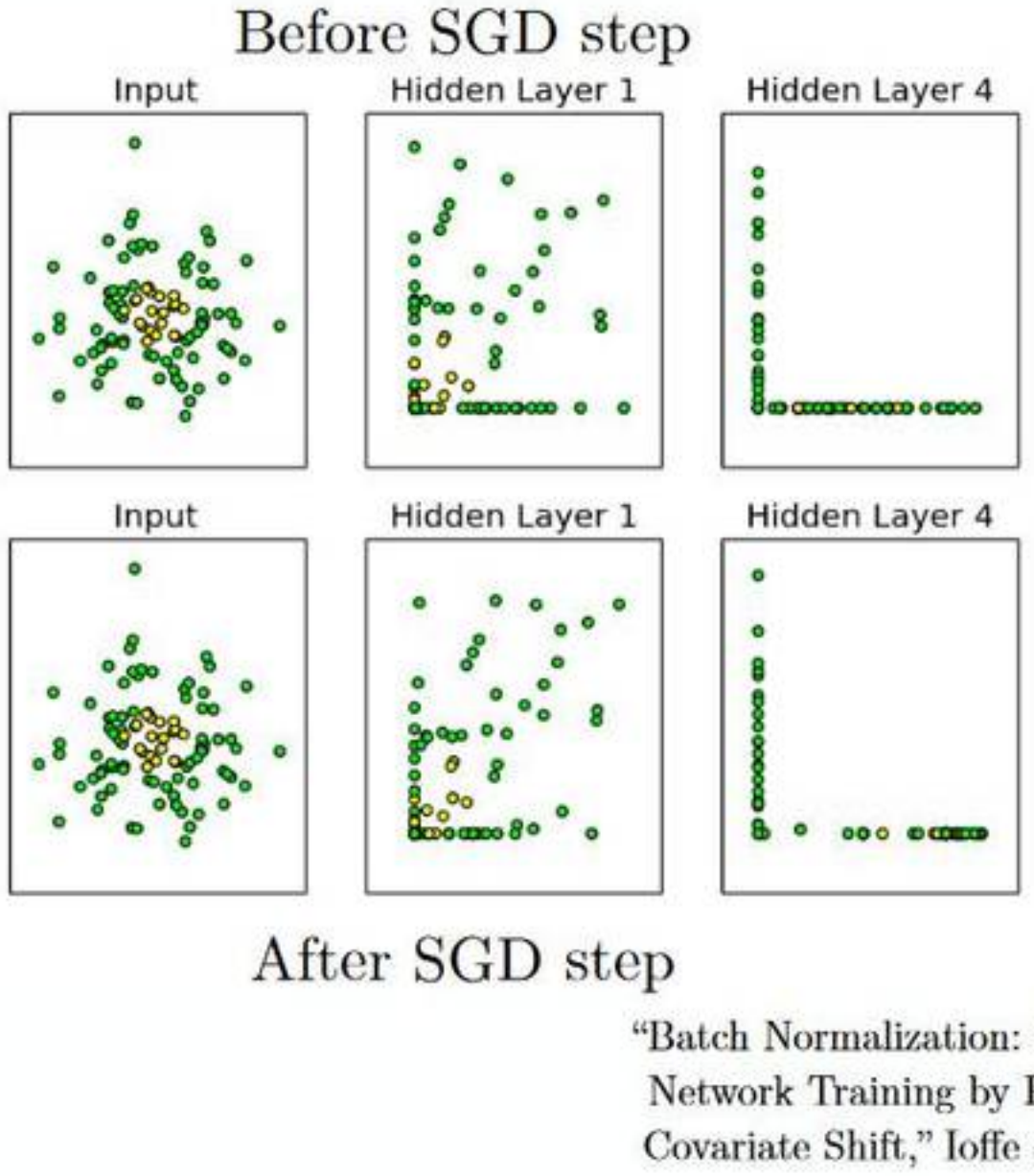


△来源：Hinton论文《Improving neural networks by preventing co-adaptation of feature detectors》

地址： https://arxiv.org/pdf/1207.0580.pdf(16) 分批标准化(Batch Normalization)

分批标准化就像是人们在河流中用以监测水位的监察站一样。这是为了保证

下一层网络得到的数据拥有合适的分布。在训练神经网络的过程中，每一次梯度下降后权重因子都会得到改变，从而会改变相应的数据结构。



但是下一层网络希望能够得到与之前分布相似的数据，因此在每一次数据传递前都需要对数据进行一次正则化处理。

$$\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{W}$$

$$\tilde{\boldsymbol{Z}} = \boldsymbol{Z} - \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{Z}_{i,:}$$

$$\hat{\boldsymbol{Z}} = \frac{\tilde{\boldsymbol{Z}}}{\sqrt{\epsilon + \frac{1}{m}\sum_{i=1}^{m} \tilde{\boldsymbol{Z}}_{i,:}^2}}$$

$$\boldsymbol{H} = \max\{0, \boldsymbol{\gamma}\hat{Z} + \boldsymbol{\beta}\}$$

"Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," Ioffe and Szegedy 2015

卷积神经网络 (17) 过滤器/滤波器(Filter)

CNN中的滤波器，具体是指将一个权重矩阵，我们用它乘以输入图像的一部分，产生相应的卷积输出。

比方说，对于一个28×28的图片，将一个3×3的滤波器与图片中3×3的矩阵依次相乘，从而得到相应的卷积输出。滤波器的尺寸通常比原始图片要小，与权重相似，在最小化成本的反向传播中，滤波器也会被更新。

就像下面这张图片一样，通过一个过滤器，依次乘以图片中每个3×3的分块，从而产生卷积的结果。

INPUT

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

FILTER

1	0	1
0	1	0
1	0	1

CONVOLVED FEATURE

4		

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

1	0	1
0	1	0
1	0	1

4		
2		

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

1	0	1
0	1	0
1	0	1

4	3	
2		

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

1	0	1
0	1	0
1	0	1

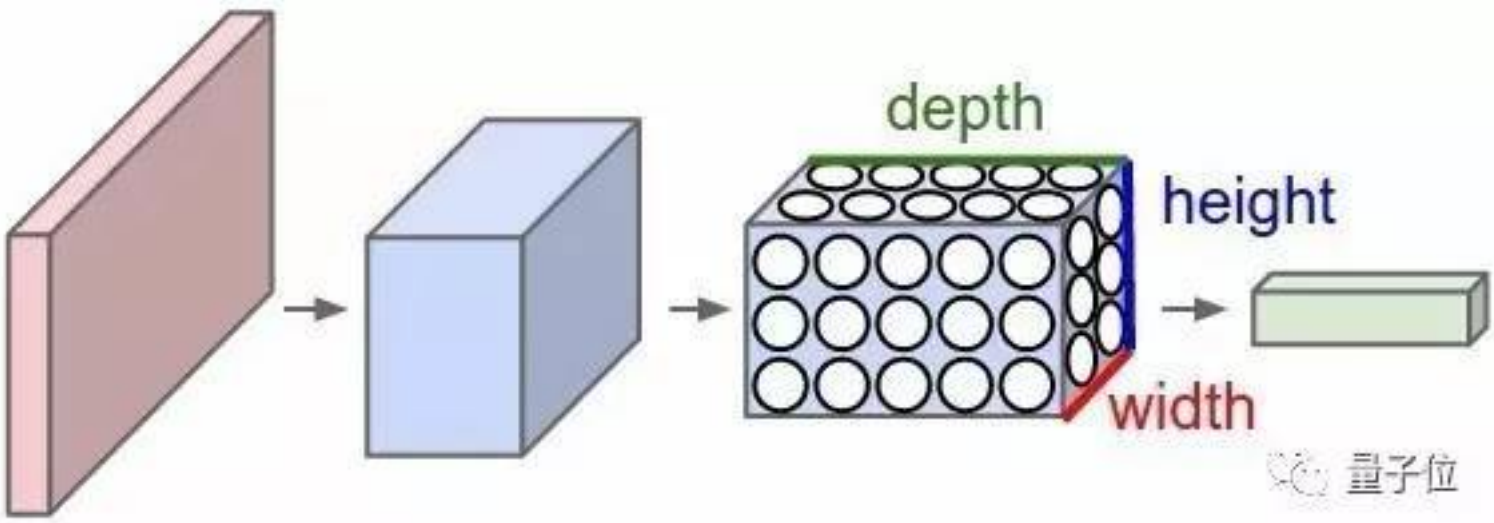
4	3	4
2	4	3
2	3	4

(18) CNN(卷积神经网络)

卷积神经网络通常用来处理图像数据，假设输入数据的形状为28×28×3(28pixels×28pixels×RGB Value)，那么对于传统的神经网络来说就会有2352(28×28×3)个变量。随着图像尺寸的增加，那么变量的数量就会急剧增加。

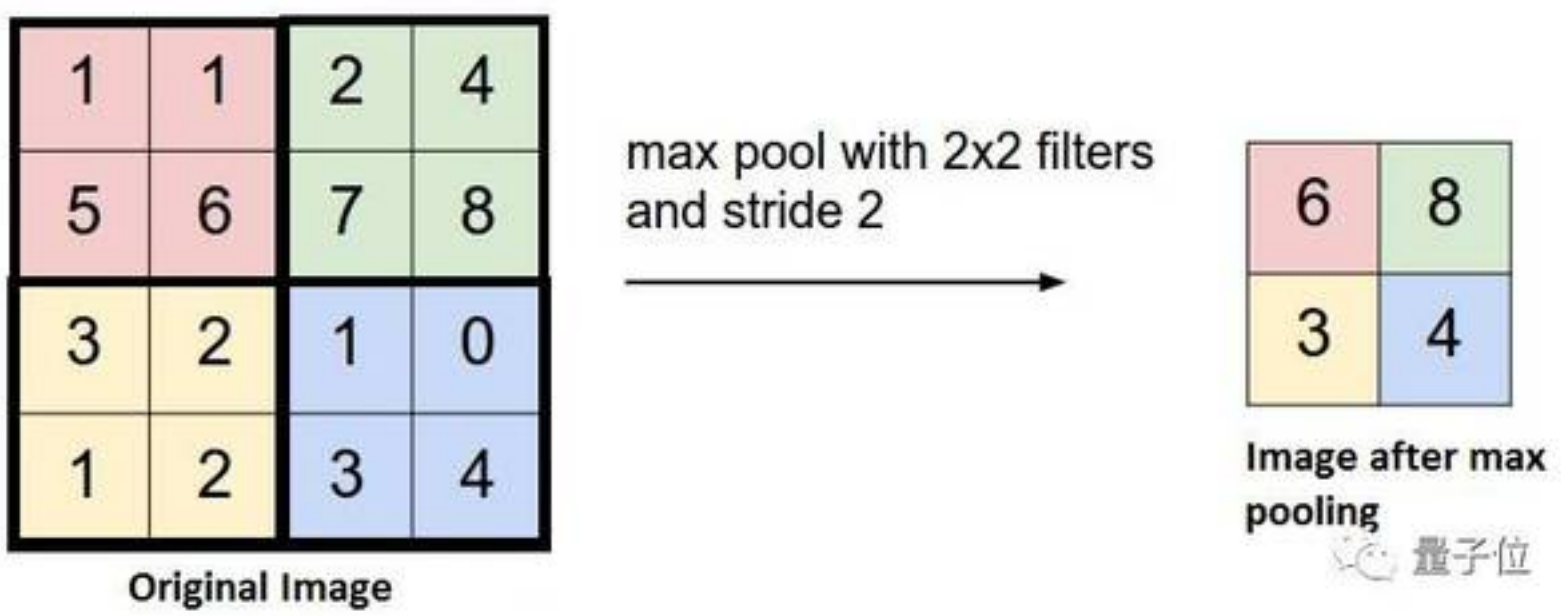
因此，通过对图片进行卷积，可以减少变量的数目。随着过滤器沿着图像上宽和高的两个方向滑动，就会产生一个相应的2维激活映射，最后再沿纵向将所有的激活映射堆叠在一起，就产生了最后的输出。

可以参照下面这个示意图。



△来源：cs231n (19) 池化(pooling)

为了进一步减少变量的数目同时防止过拟合，一种常见的做法是在卷积层中引入池化层(pooling layer)。如下图所示，最常用的池化层的操作是将原始图片中每个4×4分块取最大值形成一个新的矩阵，这叫做最大值池化(max pooling)。



△来源：cs231n

当然也有人尝试诸如平均池化(average pooling)之类的方式，但在实际情况中最大化池化拥有更好的效果。

(20) 补白(padding)

如下图所示，补白(padding)通常是指给图像的边缘增加额外的空白，从而使得卷积后输出的图像跟输入图像在尺寸上一致，这也被称作相同补白(Same Padding)。

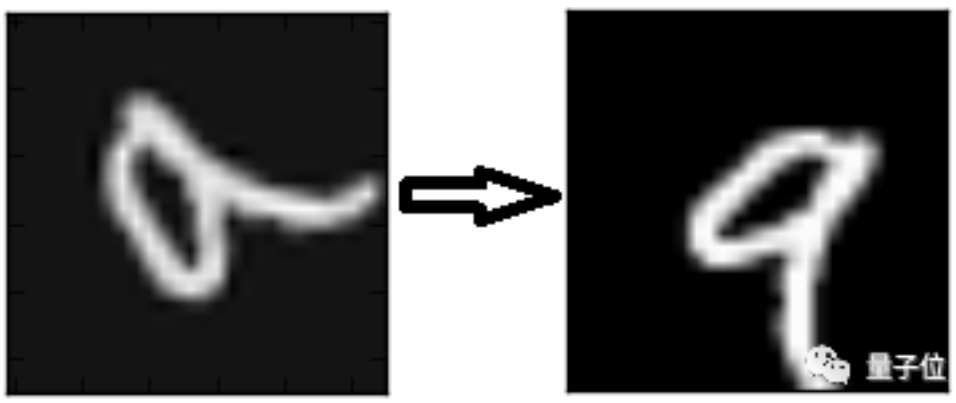
0	0	0	0	0	0
0	35	19	25	6	0
0	13	22	16	53	0
0	4	3	7	10	0
0	9	8	1	3	0
0	0	0	0	0	0

有效补白(Valid Padding)指的是保持图片上每个真实的像素点，不增加空白，因此在经历卷积后数据的尺寸会不断变小。(译者注：具体是指有效补白每次会丢弃最后不满足于一次卷积的像素点，比如说filter是3*3的，那么对于一行有32个pixel的数据，经历一次卷积后就会丢掉最后2个pixel；而通过相同补白，增加一个空白位，使每行有33个pixel，卷积后数据的尺寸不会变小。

(21) 数据增强(Data Augmentation)

数据增强(Data Augmentation)指的是从已有数据中创造出新的数据，通过增加训练量以期望能够提高预测的准确率。

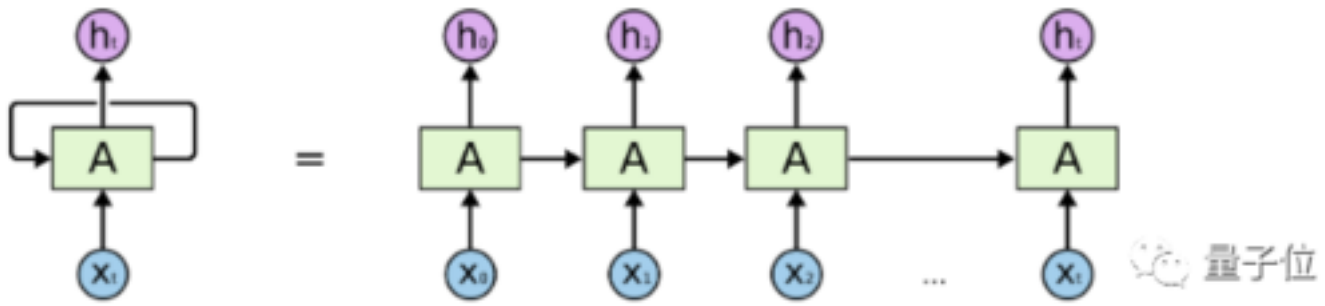
比如在说数字识别中，我们遇到的数字可能是倾斜或旋转的，因此如果将训练的图片进行适度的旋转，增大训练量，那么模型的准确性就可能会得到提高。通过“旋转”的操作，训练数据的品质得到了提升，这种过程被称作数据增强。



递归神经网络 (22) 递归神经元(Recurrent Neural Network)

对于递归神经元来说，经由它自己处理过的数据会变成自身下一次的输入，这个过程总共会进行t次。如下图所示，将递归神经元展开就相当于t个不同

的神经元串联起来，这种神经元的长处是能够产生一个更全面的输出结果。



△来源：cs231n

(23) 递归神经网络(RNN-Recurrent Neural Network)

递归神经网络通常被用于处理序列化的数据，即前一项的输出是用来预测下一项的 。

递归神经网络中存在环的结构，这些神经元上的环状结构使得它们能够存储之前的数据一段时间，从而使得能够预测输出。

与递归神经元相似，在RNN中隐藏层的输出会作为下一次的输入，如此往复经历t次，再将输出的结果传递到下一层网络中。这样，最终输出的结果会更全面，而且之前训练的信息被保持的时间会更久。

(24) 梯度消失问题

当激活函数的梯度很小时就会发生梯度消失问题。在反向传播的过程中，权重因子会被多次乘以这些小的梯度，因此会越变越小，随着递归的深入趋于“消失”， 使得神经网络失去了长程可靠性。这在递归神经网络中是一个较普遍的问题。

(25) 梯度爆炸问题

与梯度消失问题对应，当激活函数的梯度较大时，就会发生梯度爆炸问题。在反向传播的过程中，部分节点的大梯度使得他们的权重变得非常大，从而削弱了其他节点对于结果的影响，这个问题可以通过截断(即设置一个梯度允许的最大值)的方式来有效避免。

【完】

招聘

我们正在招募编辑记者、运营等岗位，工作地点在北京中关村，期待你的到来，一起体验人工智能的风起云涌。

相关细节，请在公众号对话界面，回复：“招聘”两个字。

One More Thing…

今天AI界还有哪些事值得关注？在量子位（QbitAI）公众号会话界面回复“今天”，看我们全网搜罗的AI行业和研究动态。笔芯~

另外，欢迎加量子位小助手的微信：qbitbot，如果你研究或者从事AI领域，小助手会把你带入量子位的交流群里。

追踪人工智能领域最劲内容

责任编辑：