卷积神经算子开发

# 卷积神经网络

卷积神经网络（ConvolutionalNeuralNetworks,CNN）是一类包含[卷积](https://baike.baidu.com/item/%E5%8D%B7%E7%A7%AF/9411006" \t "_blank)计算且具有深度结构的[前馈神经网络](https://baike.baidu.com/item/%E5%89%8D%E9%A6%88%E7%A5%9E%E7%BB%8F%E7%BD%91%E7%BB%9C/7580523)（FeedforwardNeuralNetworks），是[深度学习](https://baike.baidu.com/item/%E6%B7%B1%E5%BA%A6%E5%AD%A6%E4%B9%A0/3729729" \t "_blank)（deeplearning）的代表算法之一[1-2]。卷积神经网络具有[表征学习](https://baike.baidu.com/item/%E8%A1%A8%E5%BE%81%E5%AD%A6%E4%B9%A0/2140515" \t "_blank)（representationlearning）能力，能够按其阶层结构对输入信息进行平移不变分类（shift-invariantclassification），因此也被称为“平移不变人工神经网络（Shift-InvariantArtificialNeuralNetworks,SIANN）

## 历史

对卷积神经网络的研究可追溯至日本学者福岛邦彦（KunihikoFukushima）提出的neocognitron模型。在其1979[7-8]和1980年[9]发表的论文中，福岛仿造生物的[视觉皮层](https://baike.baidu.com/item/%E8%A7%86%E8%A7%89%E7%9A%AE%E5%B1%82/10986729" \t "_blank)（visualcortex）设计了以“neocognitron”命名的神经网络。neocognitron是一个具有深度结构的神经网络，并且是最早被提出的深度学习算法之一[10]，其隐含层由S层（Simple-layer）和C层（Complex-layer）交替构成。其中S层单元在[感受野](https://baike.baidu.com/item/%E6%84%9F%E5%8F%97%E9%87%8E/8989338" \t "_blank)（receptivefield）内对图像特征进行提取，C层单元接收和响应不同感受野返回的相同特征[9]。neocognitron的S层-C层组合能够进行特征提取和筛选，部分实现了卷积神经网络中[卷积层](https://baike.baidu.com/item/%E5%8D%B7%E7%A7%AF%E5%B1%82/22701737" \t "_blank)（convolutionlayer）和池化层（poolinglayer）的功能，被认为是启发了卷积神经网络的开创性研究[11]。

第一个卷积神经网络是1987年由AlexanderWaibel等提出的时间延迟网络（TimeDelayNeuralNetwork,TDNN）[12]。TDNN是一个应用于语音识别问题的卷积神经网络，使用[FFT](https://baike.baidu.com/item/FFT/4766072" \t "_blank)预处理的语音信号作为输入，其隐含层由2个一维卷积核组成，以提取频率域上的平移不变特征[13]。由于在TDNN出现之前，人工智能领域在[反向传播算法](https://baike.baidu.com/item/%E5%8F%8D%E5%90%91%E4%BC%A0%E6%92%AD%E7%AE%97%E6%B3%95/522948" \t "_blank)（Back-Propagation,BP）的研究中取得了突破性进展[14]，因此TDNN得以使用BP框架内进行学习。在原作者的比较试验中，TDNN的表现超过了同等条件下的[隐马尔可夫模型](https://baike.baidu.com/item/%E9%9A%90%E9%A9%AC%E5%B0%94%E5%8F%AF%E5%A4%AB%E6%A8%A1%E5%9E%8B/7932524" \t "_blank)（HiddenMarkovModel,HMM），而后者是二十世纪80年代语音识别的主流算法[13]。

1988年，WeiZhang提出了第一个二维卷积神经网络：平移不变人工神经网络（SIANN），并将其应用于检测[医学影像](https://baike.baidu.com/item/%E5%8C%BB%E5%AD%A6%E5%BD%B1%E5%83%8F/4954291" \t "_blank)[3]。独立于Zhang(1988)，YannLeCun在1989年同样构建了应用于计算机视觉问题的卷积神经网络，即LeNet的最初版本[5]。LeNet包含两个卷积层，2个全连接层，共计6万个学习参数，规模远超TDNN和SIANN，且在结构上与现代的卷积神经网络十分接近[11]。LeCun(1989)[5]对权重进行随机初始化后使用了随机梯度下降（StochasticGradientDescent,SGD）进行学习，这一策略被其后的深度学习研究所保留。此外，LeCun(1989)在论述其网络结构时首次使用了“卷积”一词[5]，“卷积神经网络”也因此得名。

LeCun(1989)[5]的工作在1993年由[贝尔实验室](https://baike.baidu.com/item/%E8%B4%9D%E5%B0%94%E5%AE%9E%E9%AA%8C%E5%AE%A4/686816" \t "_blank)（AT&TBellLaboratories）完成代码开发并被部署于NCR（NationalCashRegisterCoporation）的支票读取系统[11]。但总体而言，由于数值计算能力有限、学习样本不足，加上同一时期以[支持向量机](https://baike.baidu.com/item/%E6%94%AF%E6%8C%81%E5%90%91%E9%87%8F%E6%9C%BA/9683835" \t "_blank)（SupportVectorMachine,SVM）为代表的核学习（kernellearning）方法的兴起，这一时期为各类图像处理问题设计的卷积神经网络停留在了研究阶段，应用端的推广较少[2]。

在LeNet的基础上，1998年YannLeCun及其合作者构建了更加完备的卷积神经网络LeNet-5并在手写数字的识别问题中取得成功[15]。LeNet-5沿用了LeCun(1989)的学习策略并在原有设计中加入了池化层对输入特征进行筛选[15]。LeNet-5及其后产生的变体定义了现代卷积神经网络的基本结构，其构筑中交替出现的卷积层-池化层被认为能够提取输入图像的平移不变特征[16]。LeNet-5的成功使卷积神经网络的应用得到关注，微软在2003年使用卷积神经网络开发了光学字符读取（OpticalCharacterRecognition,OCR）系统[17]。其它基于卷积神经网络的应用研究也得到展开，包括[人像识别](https://baike.baidu.com/item/%E4%BA%BA%E5%83%8F%E8%AF%86%E5%88%AB18/1705333" \t "_blank)[[18]](https://baike.baidu.com/item/%E4%BA%BA%E5%83%8F%E8%AF%86%E5%88%AB18/1705333" \t "_blank)、[手势识别](https://baike.baidu.com/item/%E6%89%8B%E5%8A%BF%E8%AF%86%E5%88%AB19/12629655" \t "_blank)[[19]](https://baike.baidu.com/item/%E6%89%8B%E5%8A%BF%E8%AF%86%E5%88%AB19/12629655" \t "_blank)等。

在2006年[深度学习](https://baike.baidu.com/item/%E6%B7%B1%E5%BA%A6%E5%AD%A6%E4%B9%A0/3729729" \t "_blank)理论被提出后[20]，卷积神经网络的表征学习能力得到了关注，并随着数值计算设备的更新得到发展[2]。自2012年的AlexNet[21]开始，得到GPU计算集群支持的复杂卷积神经网络多次成为[ImageNet](https://baike.baidu.com/item/ImageNet/17752829" \t "_blank)大规模视觉识别竞赛（ImageNetLargeScaleVisualRecognitionChallenge,ILSVRC）[22]的优胜算法，包括2013年的ZFNet[23]、2014年的VGGNet、GoogLeNet[24]和2015年的ResNet[25]。

## 结构

### 输入层

卷积神经网络的输入层可以处理多维数据，常见地，一维卷积神经网络的输入层接收一维或二维数组，其中一维数组通常为时间或频谱采样；二维数组可能包含多个通道；二维卷积神经网络的输入层接收二维或三维数组；三维卷积神经网络的输入层接收四维数组[16]。由于卷积神经网络在[计算机视觉](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E6%9C%BA%E8%A7%86%E8%A7%89/2803351)领域应用较广，因此许多研究在介绍其结构时预先假设了三维输入数据，即平面上的二维像素点和[RGB](https://baike.baidu.com/item/RGB/342517)通道。

与其它神经网络算法类似，由于使用[梯度下降算法](https://baike.baidu.com/item/%E6%A2%AF%E5%BA%A6%E4%B8%8B%E9%99%8D%E7%AE%97%E6%B3%95/1095627" \t "_blank)进行学习，卷积神经网络的输入特征需要进行标准化处理。具体地，在将学习数据输入卷积神经网络前，需在通道或时间/频率维对输入数据进行归一化，若输入数据为像素，也可将分布于[0~255]的原始像素值归一化至[0~1]区间[16]。输入特征的标准化有利于提升卷积神经网络的学习效率和表现[16]。

### 隐含层

卷积神经网络的隐含层包含[卷积层](https://baike.baidu.com/item/%E5%8D%B7%E7%A7%AF%E5%B1%82/22701737)、池化层和全连接层3类常见构筑，在一些更为现代的算法中可能有Inception模块、残差块（residual block）等复杂构筑。在常见构筑中，卷积层和池化层为卷积神经网络特有。卷积层中的卷积核包含权重系数，而池化层不包含权重系数，因此在文献中，池化层可能不被认为是独立的层。以LeNet-5为例，3类常见构筑在隐含层中的顺序通常为：输入-卷积层-池化层-全连接层-输出。

### 输出层

卷积神经网络中输出层的上游通常是全连接层，因此其结构和工作原理与传统前馈神经网络中的输出层相同。对于图像分类问题，输出层使用逻辑函数或归一化指数函数（softmax function）输出分类标签[16]  。在物体识别（object detection）问题中，输出层可设计为输出物体的中心坐标、大小和分类[16]  。在图像语义分割中，输出层直接输出每个像素的分类结果[16]  。

# 算子

算子是一个函数空间到函数空间上的映射O：X→X。广义上的算子可以推广到任何空间，如内积空间等。

## 批归一化层

### 输入

卷积层的输入一般为三维张量 ，其中为分别表征张量的通道长度、行长度、列长度。

### batchNorm处理

每个池化层通过自身的数据进行计算无需额外数据参与，平均池化层计算所需的参数如下：

kernel\_size ：分别为池化操作的滑动窗行与列方向大小；

eps ： 常数确保除法计算结果稳定性；

stride ：分别为池化滑动窗行与列方向步近长度，通常为池化核大小；

padding ：分别为输入张量各维度各边界要补齐0的层数；

通过池化处理得到的输出张量大小为, 其中为分别表征张量的通道长度、行长度、列长度，的具体计算公式如下，表示向下取整处理。

最大batch-norm运算可以形式化的表示为

## 卷积层（convolutional layer）

卷积层的功能是对输入数据进行特征提取，比如图像中可能有很多种边缘特征，包括对角线特征、三角形特征、圈特征，甚至麻花形的特征等。为了提取出图像中不同的特征，神经网络需要有多个不同的卷积核。提取出不同的特征之后，每一个神经网络层会输出多个特征图（或者说有多个输出通道），每一个特征图代表一种特征。

卷积层内部包含多个卷积核，组成卷积核的每个元素都对应一个权重系数和一个偏差量（bias vector），类似于一个前馈神经网络的神经元（neuron）。卷积层内每个神经元都与前一层中位置接近的区域的多个神经元相连，区域的大小取决于卷积核的大小，在文献中被称为“[感受野](https://baike.baidu.com/item/%E6%84%9F%E5%8F%97%E9%87%8E/8989338)（receptive field）”，其含义可类比[视觉皮层](https://baike.baidu.com/item/%E8%A7%86%E8%A7%89%E7%9A%AE%E5%B1%82/10986729)细胞的感受野[2]  。卷积核在工作时，会有规律地扫过输入特征，在感受野内对输入特征做矩阵元素乘法求和并叠加偏差量[1]

### 数学定义

定义任意两个信号的卷积为这里的代表卷积的运算符号，是中间变量，两个信号的卷积仍是以t为变量的信号。离散的信号的卷积和：

### 输入

卷积层的输入一般为三维张量 ，其中为分别表征张量的通道长度、行长度、列长度。

### 普通Conv处理

普通卷积层包含weights权重和biases偏置两种类型参数，权重参数，其中分别表征卷积核个数、卷积核内部的通道长度、行长度、列长度，参数偏执参数，卷积层计算所需的参数如下：

kernel\_size ：分别为卷积操作的滑动窗行与列方向大小；

stride ：分别为卷积滑动窗行与列方向步近长度，通常为卷积核大小；

padding ：分别为输入张量各维度各边界要补齐0的层数；

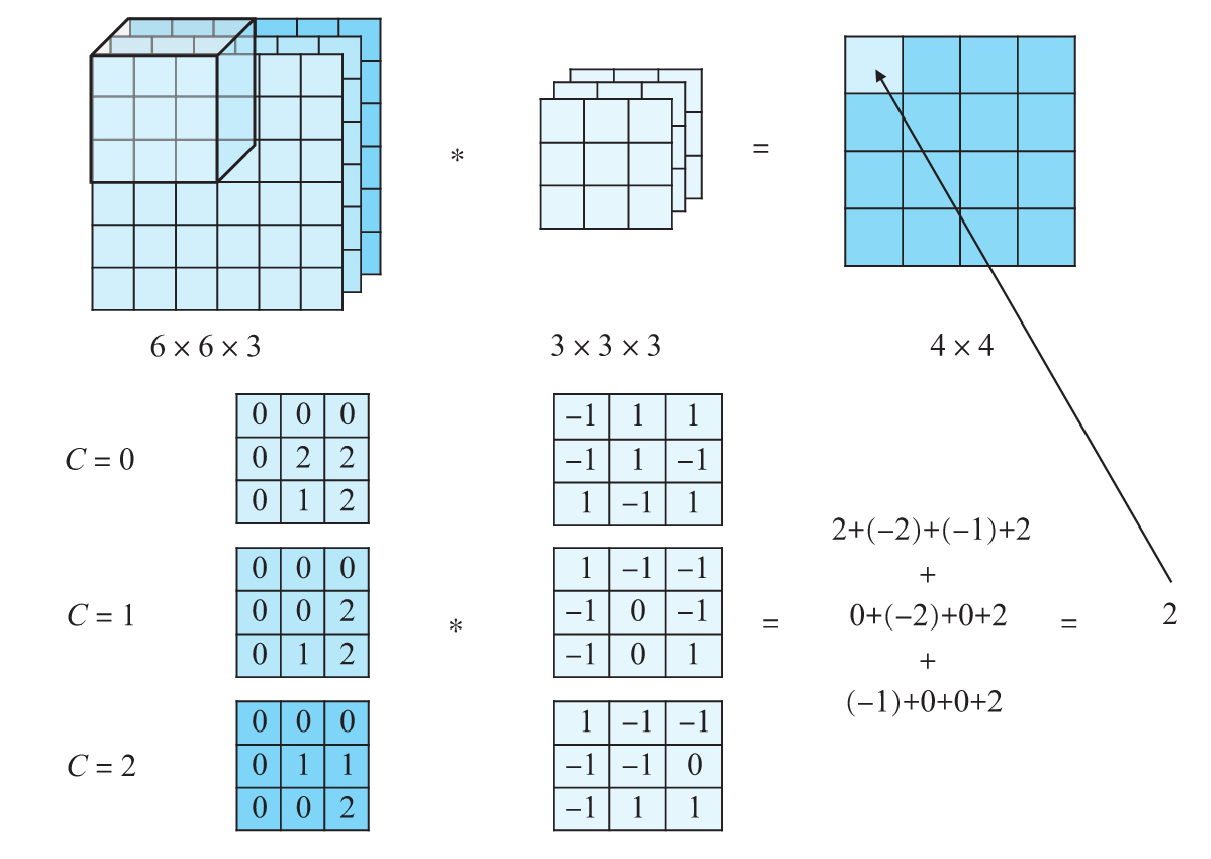
通过卷积处理得到的输出张量大小为, 其中为分别表征张量的通道长度、行长度、列长度，根据卷积输入的张量与池化层参数，的具体计算公式如下，表示向下取整处理。

表征行方向的padding补零个数，表征行方向的步进长度，表征列方向的padding补零个数，表征列方向的步进长度。

普通卷积计算运算可以形式化的表示为：

其中公式中的为输入张量的padding处理后的结果，与所表征的数据范围将发生变化,表征卷积的输出结果，值需表示最后卷积层的输出包含激活层的处理。考虑到计算结果采用向上取整的方式，当中索引值超过其数组范围时，返回数值为0。

以图3.5中的卷积为例：该神经网络卷积层的输入是3个6×6大小的特征图（即6×6×3的三维矩阵），分别表示对角线特征、圈特征、三角形特征；卷积核是3个不同的3×3卷积核（即3×3×3的三维矩阵），对应输入的3个不同的特征图；二者卷积输出一个4×4大小的特征图。在卷积运算中，输入特征图的通道数和卷积核的通道数必须一致。如果输入有3个特征图（也称为3个特征通道），就需要有3个卷积核来共同计算出最终的输出特征图。



### Dilation Conv处理

dilation卷积层包含weights权重和biases偏置两种类型参数，权重参数，其中分别表征卷积核个数、卷积核内部的通道长度、行长度、列长度，参数偏执参数，卷积层计算所需的参数如下：

kernel\_size ：分别为卷积操作的滑动窗行与列方向大小；

dilation ：分别为卷积滑动窗中各有效元素间的距离；

stride ：分别为卷积滑动窗行与列方向步近长度，通常为卷积核大小；

padding ：分别为输入张量各维度各边界要补齐0的层数；

通过卷积处理得到的输出张量大小为, 其中为分别表征张量的通道长度、行长度、列长度，根据Dilation卷积输入的张量与池化层参数，的具体计算公式如下，表示向下取整处理。

dilation卷积计算运算可以形式化的表示为：

其中公式中的为输入张量的padding处理后的结果，与所表征的数据范围将发生变化,表征卷积的输出结果，值需表示最后卷积层的输出包含激活层的处理。考虑到计算结果采用向上取整的方式，当中索引值超过其数组范围时，返回数值为0。

## 激活层（activation）

神经网络中的每个神经元节点接收上一层神经元的输出值作为本神经元的输入值，并将输入值传递给下一层，在多层神经网络中上层神经节点的输出和下层节点的输入之间具有一个函数关系，这个函数称为激活函数（又称激励函数）

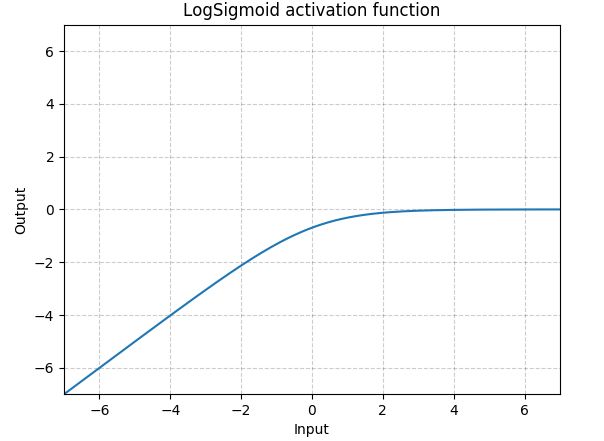
如果不用激励函数，每层节点输入与上层节点输出为线性关系，网络的逼近能力有限。无法实现更加强大的网络表达能力。（不再是输入的线性组合，而是几乎可以逼近任意函数）。

早期研究神经网络主要采用sigmoid函数或者tanh函数，输出有界，很容易充当下一层的输入。近些年Relu函数及其改进型（如Leaky-ReLU、P-ReLU、R-ReLU等）在多层神经网络中应用比较多。下面我们来总结下这些激活函数：

### LogSigmoid函数

LogSigmoid是Sigmoid中参数为1的情况，Sigmoid函数解析表达与求导公式如下所示：

LogSigmoid函数解析表达、求导公式和对应的曲线如下所示：

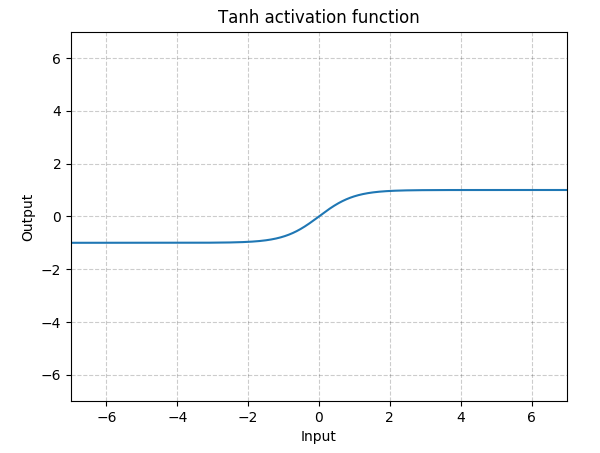


Sigmoid是[BP神经网络](https://baike.baidu.com/item/BP%E7%A5%9E%E7%BB%8F%E7%BD%91%E7%BB%9C" \t "_blank)中常用的非线性作用函数，能够把输入的连续实值变换为0和1之间的输出，即对于输入非常大的负数，输出为0；对于输入非常大的正数，输出为1。函数都是连续的、单调递增的数值函数，常被应用于基于BP（误差反向传播）算法的神经网络中。

sigmoid函数曾经被经常使用，不过近年来，越来越少了。主要是因为它固有的一些缺点。1）在深度神经网络中梯度反向传递容易存在梯度消失现象，神经网络的权重参数数值大多在-1和1之间，由反向传播算法的数学推导和Sigmoid函数的导数可以获取，梯度从后向前传播时每传递一层梯度值都会减小为原来的0.25倍，针对神经网络模型较深的情况，梯度在经过多层后将变得很小接近于零值，无法更新权重参数；2）：Sigmoid输出数值不是0均值。导致对求局部梯度则都为正，这样在反向传播的过程中要么都往正方向更新，要么都往负方向更新。可以通过设置batch-size可以缓解这个问题；3）Sigmoid函数中存在幂运算，计算求解耗时加长，对于规模比较大的深度网络，将会较大地增加训练时间。

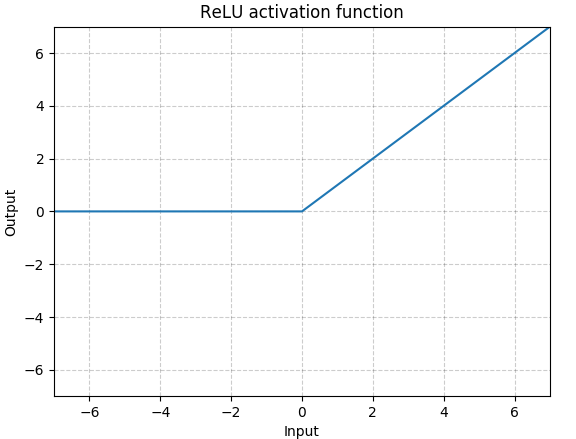
### Tanh函数

Tanh为双曲正切函数，其解析表达、求导公式和对应的曲线如下所示：



### ReLU函数

RELU线性整流函数（Rectified Linear Unit）又称修正线性单元，其解析表达、求导公式和对应的曲线如下所示：

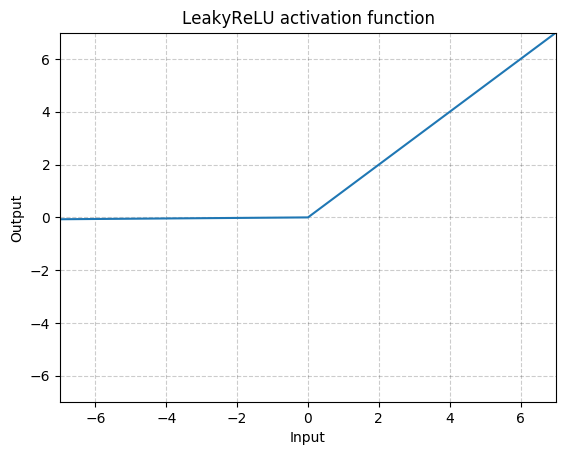


RELU是一种[人工神经网络](https://baike.baidu.com/item/%E4%BA%BA%E5%B7%A5%E7%A5%9E%E7%BB%8F%E7%BD%91%E7%BB%9C" \t "_blank)中常用的激活函数，相比于传统的神经网络激活函数，诸如[逻辑函数](https://baike.baidu.com/item/%E9%80%BB%E8%BE%91%E5%87%BD%E6%95%B0)（Logistic sigmoid）和tanh等[双曲函数](https://baike.baidu.com/item/%E5%8F%8C%E6%9B%B2%E5%87%BD%E6%95%B0" \t "_blank)，线性整流函数有着以下几方面的优势：1）参考仿生物学原理，相关大脑方面的研究表明生物神经元的信息编码通常是比较分散及稀疏的，通常情况下，大脑中在同一时间大概只有1%-4%的神经元处于活跃状态。使用线性修正以及正则化可以对机器神经网络中神经元的活跃度（即输出为正值）进行调试；而逻辑函数Sigmoid在输入为0时达到0.5，即已经是半饱和的稳定状态，不够符合实际生物学对模拟神经网络的期望。2）更加有效率的[梯度下降](https://baike.baidu.com/item/%E6%A2%AF%E5%BA%A6%E4%B8%8B%E9%99%8D" \t "_blank)以及反向传播：避免了梯度爆炸和梯度消失问题，3）简化计算过程：没有了其他复杂激活函数中诸如指数函数的影响；同时活跃度的分散性使得神经网络整体计算成本下降。

### Leaky-ReLU函数

Leaky ReLU带泄露线性整流函数（Rectified Linear Unit）又称修正线性单元，其解析表达、求导公式和对应的曲线如下所示：

,



## 池化层

CNN卷积神经网络使用池化层来缩减模型的大小，提高计算速度，同时提高所提取特征的鲁棒性，存在两种类型的池化方式，平均池化（Average Pooling）和最大值池化（Max Pooling）。

### 输入

卷积层的输入一般为三维张量 ，其中为分别表征张量的通道长度、行长度、列长度。

### 平均池化处理

每个池化层通过自身的数据进行计算无需额外数据参与，平均池化层计算所需的参数如下：

kernel\_size ：分别为池化操作的滑动窗行与列方向大小；

dilation ：分别为池化滑动窗中各有效元素间的距离；

stride ：分别为池化滑动窗行与列方向步近长度，通常为池化核大小；

padding ：分别为输入张量各维度各边界要补齐0的层数；

通过池化处理得到的输出张量大小为, 其中为分别表征张量的通道长度、行长度、列长度，根据平均池化输入的张量与池化层参数，的具体计算公式如下，表示向下取整处理。

最大池化运算可以形式化的表示为：

### 最大池化处理

每个池化层通过自身的数据进行计算无需额外数据参与，平均池化层计算所需的参数如下：

kernel\_size ：分别为池化操作的滑动窗行与列方向大小；

stride ：分别为池化滑动窗行与列方向步近长度，通常为池化核大小；

padding ：分别为输入张量各维度各边界要补齐0的层数；

通过池化处理得到的输出张量大小为, 其中为分别表征张量的通道长度、行长度、列长度，根据最大池化输入的张量与池化层参数，的具体计算公式如下，表示向下取整处理。

平均池化运算可以形式化的表示为：

## 全连接层

### FC函数

## 概率函数

### Softmax函数

Softmax归一化指数函数其解析表达、求导公式和对应的曲线如下所示：

，

将有限项离散概率分布的梯度对数归一化。因此，Softmax函数在包括多项逻辑回归，多项线性判别分析，朴素贝叶斯分类器和人工神经网络等的多种基于概率的多分类问题方法中都有着广泛应用。特别地，在多项逻辑回归和线性判别分析中，函数的输入是从个不同的线性函数得到的结果，而样本向量x属于第j个分类的概率为为改善数值性能，会对进行修正：各神经元同时减去它们中的最大值，以防止数值计算可能出现的上溢和下溢。由指数函数的性质可知，这相当于上式分子、分母同时乘以常数，不影响计算结果。