神经网络

目 录
CONTENTS

01

Part One

神经网络相关概念

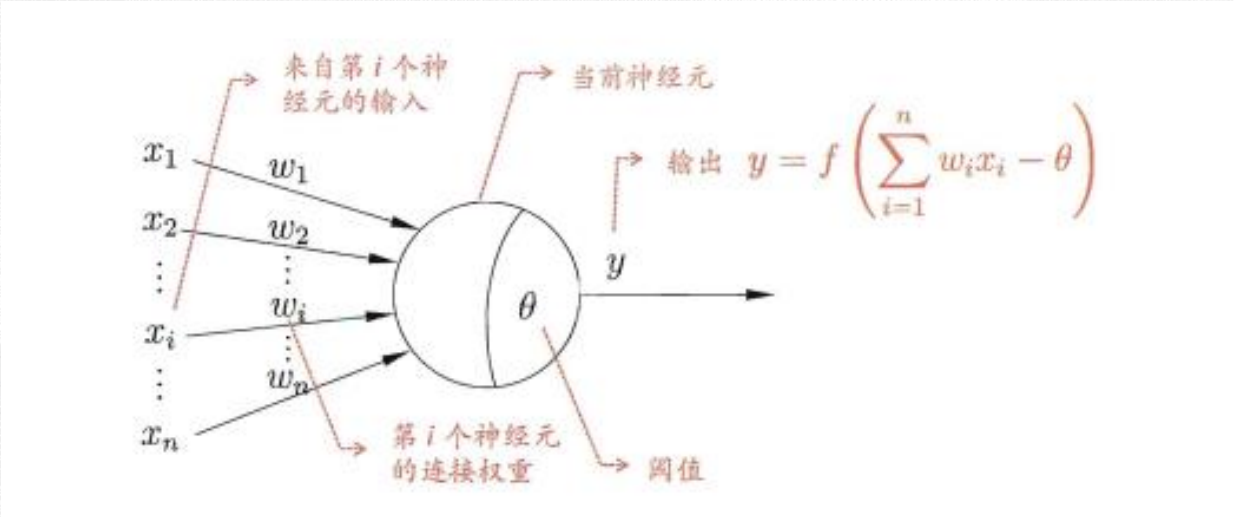
Neural Networks

神经网络相关概念

Neural Networks

神经元模型

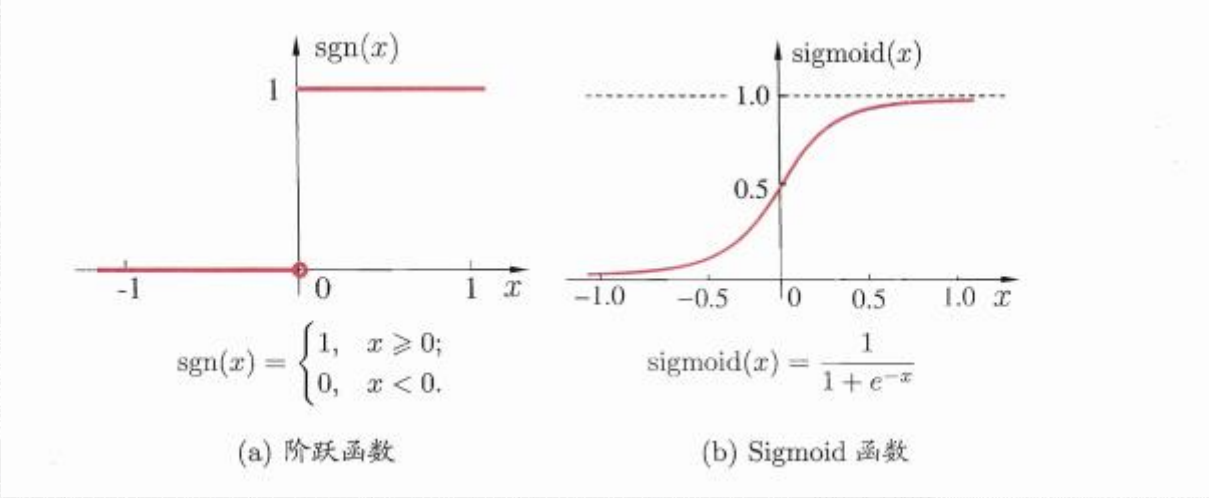
- 神经元是神经网络当中最基本的模型。在生物神经网络中每个神经元与其他神经元相连，当它“兴奋”时，就会向相连的神经元发送化学物质，从而改变这些神经元内的电位;如果某神经元的电位超过了 个“阈值”，那么它就会被激活 “兴奋起来，向其他神经元发送化学物质。
- 1943年，Warren McCulloch 和 Walter Pitts将这种模型抽象出来，建立了“M-P神经元模型” （如下图）

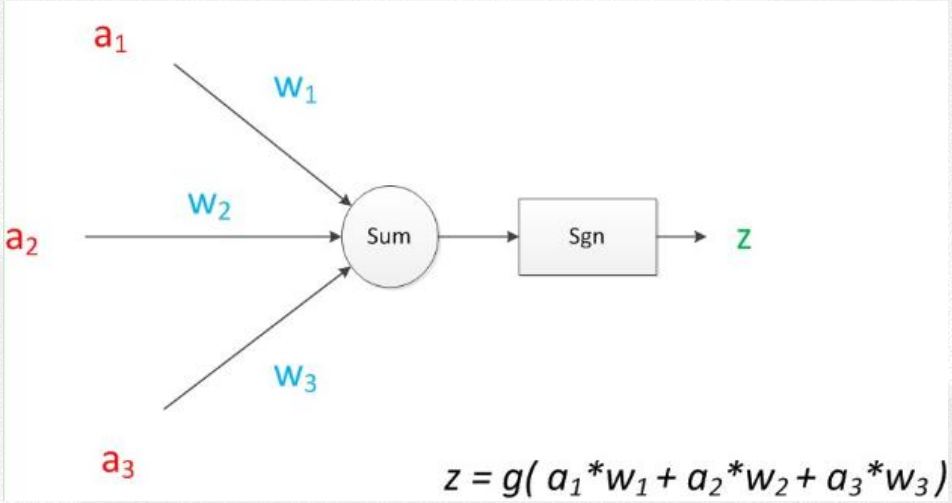


神经网络相关概念

Neural Networks

- 激活函数
- 神经元最终是通过激活函数的处理产生神经元的输出
- 常用的激活函数：tanh:f(x)=tanh(x)、ReLU：f(x)=max(x,0)、softmax:f(x)=log(1+exp(x))、Sigmoid:f(x)=1/(1+e^(-x))
-





神经网络相关概念

Neural Networks

感知机

- 1958年，计算科学家Rosenblatt提出了由两层神经元组成的神经网络,并取名为“感知机”（Perceptron）
- 感知机由两层神经元组成，输入层接收外界输入信息后传给输出层，输出层是M-P神经元。感知器是当时首个可以学习的人工神经网络。Rosenblatt现场演示了其学习识别简单图像的过程，在当时的社会引起了轰动。
- 1.“与”$(x1 \wedge x2)$:令$\omega 1 = \omega 2$=1，Θ =2，则y=f(1*x1+1*x2-2),仅在x1=x2=1时，y=1

  2.“或”$(x1 \vee x2)$:令$\omega 1 = \omega 2$ =1，$\Theta = 0.5$则y=f(1*x1+1*x2-0.5),当x1=或x2=1时，y=1

  3.“非”$(\backsim x1)$:令$\omega 1 = -0.6$，$\omega 2 = 0$，$\Theta = 2$，则y=f(-0.6*x1+0*x2+0.5),仅在x1时，y=0；当x1=0,y=1

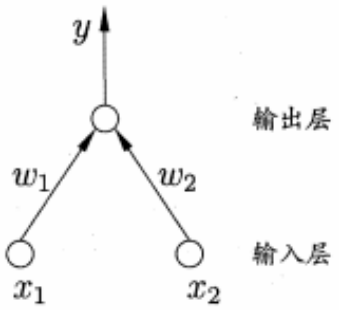
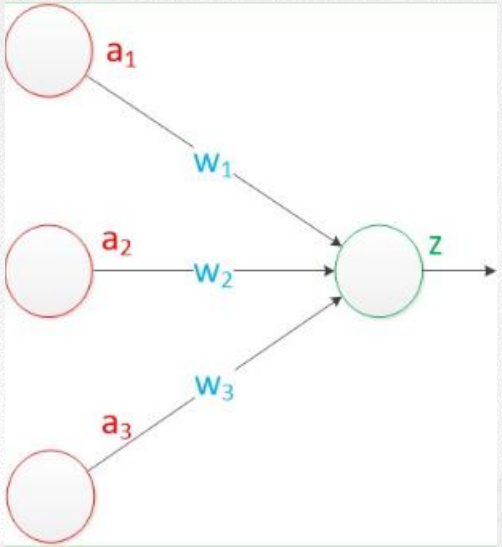
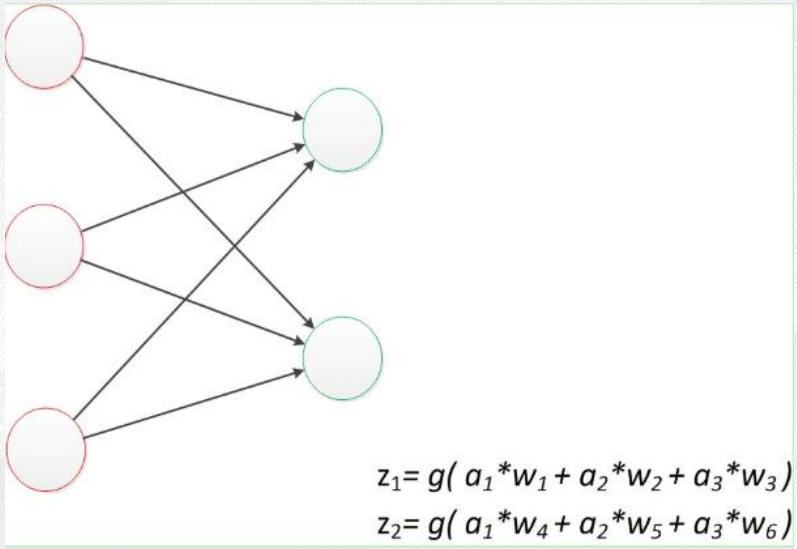


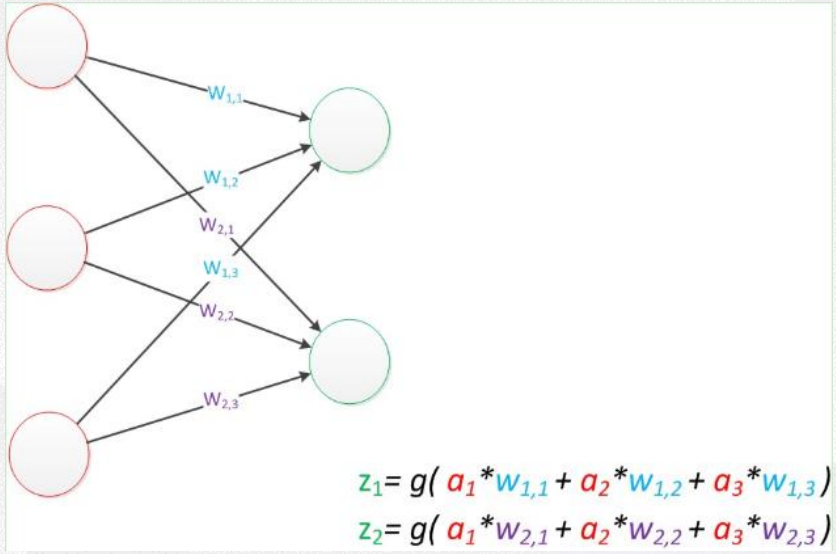
图 5.3 两个输入神经元的感知机网络结构示意图

神经网络相关概念

Neural Networks

- 感知机
  - 矩阵表达形式：Z = g(W*a)
  - 

  - 

  - 

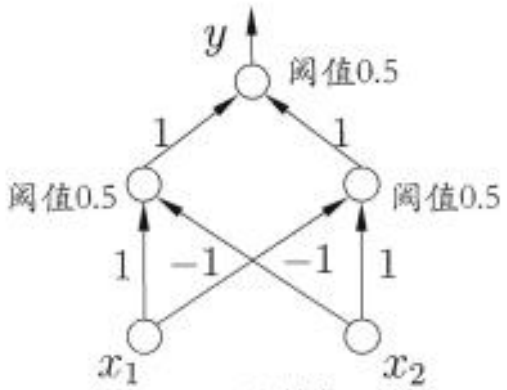
神经网络相关概念

Neural Networks

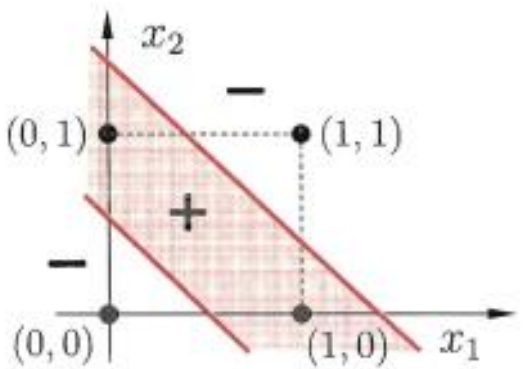
- 多层网络

- 为了解决非线性问题，需要使用多层功能神经元，如：使用两层感知机解决异或问题。

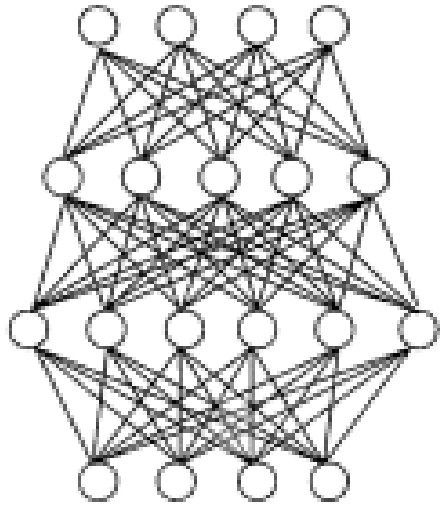
- 输入层和输出层之间加一层神经元（隐含层），隐含层和输出层神经元都拥有激活函数的功能神经元，并且有理论证明，两层神经网络可以无限逼近任意连续函数。



(a) 网络结构



(b) 分类区域



神经网络相关概念

Neural Networks

误差逆传播算法

- 误差传播算法（error BackPropagation,简称BP）算法，被成为迄今为止最成功的神经网络算法，在现实任务中使用神经网络时，大多数时在使用BP算法进行训练。

- 1986 年， Rumelhar 和 Hinton 等人提出了反向传播（Backpropagation，BP）算法，解决了两层神经网络所需要的复杂计算量问题，从而带动了业界使用两层神经网络研究的热潮

图19 David Rumelhart（左）以及 Geoffery Hinton（右）

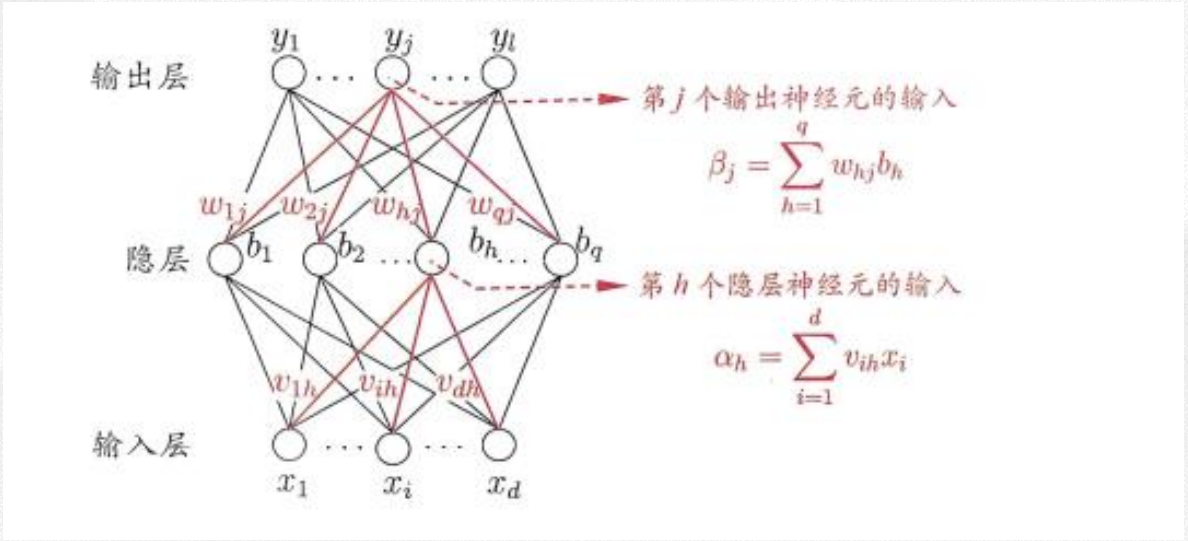
- 利用输出后的误差来估计输出层的直接前导层的误差，再用这个误差估计更前一层的误差，如此一层一层的反传下去，就获得了所有其他各层的误差估计。

神经网络相关概念

Neural Networks

误差逆传播算法

- 给定训练集D={(x1,y1),(x2,y2),…,(xm,ym)},$x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}^d$。
  假设一个神经网络拥有d个输入神经元、L个输出神经元、q个隐含层神经元。
  设输出层第j个神经元的阈值用θ_j表示，隐含层第h个神经元阈值用γ_h表示。
  输入层第i个神经元和隐层第h个神经元之间的权重为v_{ih}；隐含层第h个神经元和输出层第j个神经元之间权重为ω_{hj}
  记隐含层第h个神经元输入为$\alpha_h = \sum_{i=1}^{d} v_{ih} x_i$,输出层第j个神经元接收的输入为$\beta_j = \sum_{h=1}^{q} \omega_{hj} b_h$(其中$b_h$为隐含层第换个神经元的输出)
  假设隐藏层和输出层都使用sigmoid函数作为激活函数。
- Sigmoid函数有很好的性质$f(\acute{x}) = f(x)(1 - f(x))$



神经网络相关概念

Neural Networks

误差逆传播算法

对于训练例（xk,yk），假设神经网络的输出为$\hat{y}_k = (\hat{y}_1^k, \hat{y}_2^k, \ldots, \hat{y}_l^k)$,即$\hat{y}_l^k = f(\beta_j - \theta_j)$,

则有均方误差$E_k = \frac{1}{2}(\hat{y}_j^k - y_j^k)^2$,并且有（d+L+1）*q+L个参数需要确定，取其中权重参数ω_{hj}为例,遵从梯度下降

它的更新估计式：$\omega \leftarrow \omega + \Delta\omega$，其中$\Delta\omega_{hj} = -\eta \frac{\partial E_k}{\partial \omega_{hj}}$（$\eta$为学习率）

式子$\frac{\partial E_k}{\partial \omega_{hj}} = \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial \omega_{hj}}$,可知显然有$\frac{\partial \beta_j}{\partial \omega_{hj}} = b_h$

设$g_i = -\frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} = -(\hat{y}_j^k - y_j^k) \cdot f'(\beta_j - \theta_j) = \hat{y}_j^k(1 - \hat{y}_j^k)(y_j^k - \hat{y}_j^k)$

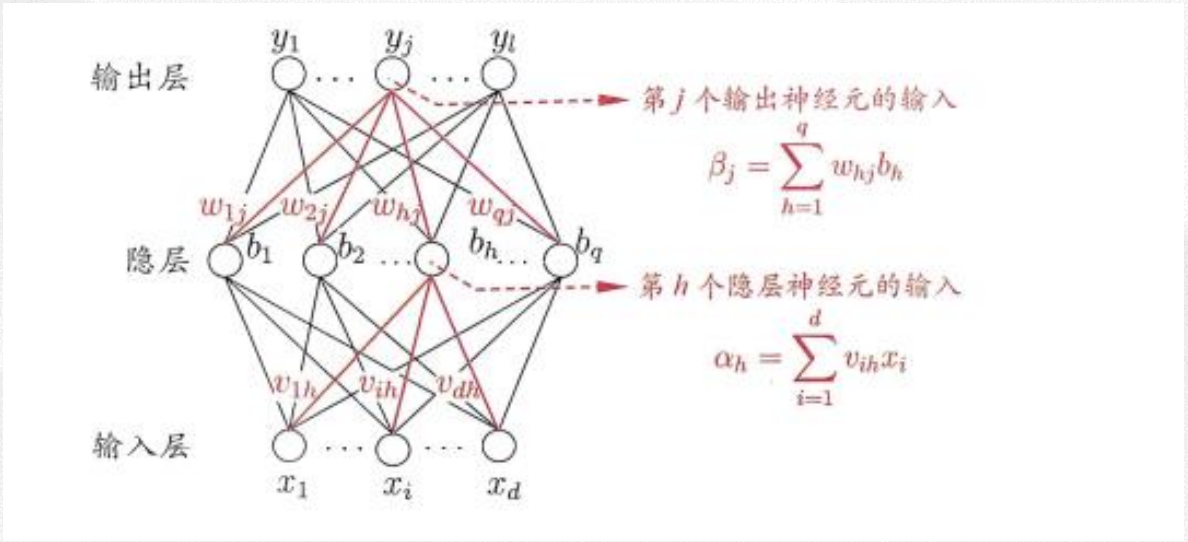
即$\Delta\omega_{hj} = \eta g_i b_h$

同理：$\Delta\theta_j = -\eta g_i$

$\Delta v_{ih} = \eta e_h x_i$

$\Delta\gamma_h = -\eta e_h$

$e_h = b_h(1 - b_h)\sum_{j=1}^{l} \omega_{hj} g_j$



神经网络相关概念

Neural Networks

过拟合问题

- 由于BP神经网络强大的表示能力，经常会发生过拟合问题
- 解决方法：

  1. “早停”：将数据分为训练集和验证集，训练集计算梯度、更新权值和阈值，验证集用来估计误差

  2. “正则化”：其思想是在误差目标函数中增加一个用于描述网路复炸程度的部分，例如连接权值和阈值的平方和。

$$E = \lambda \frac{1}{m} \sum_{k=1}^{m} E_k + (1-\lambda) \sum_{i} w_i^2 ,$$

- 其中$\lambda \epsilon$（0，1），用于对经验误差与网络复杂度进行折中

02

Part Two

卷积神经网络相关概念

Convolutional Neural Network

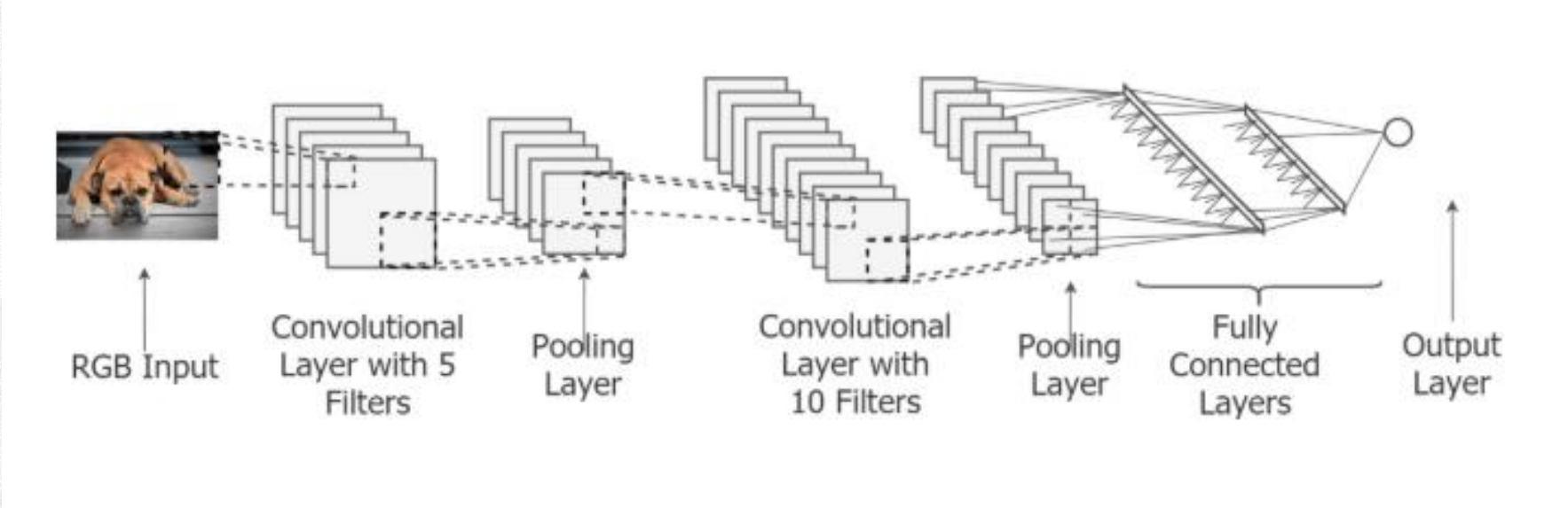
卷积神经网络相关概念

Convolutional Neural Network

卷积神经网络简介

- 全链接神经网络处理大尺寸图像具有很明显的缺点：
  1.图像展开为向量回丢失空间信息　2.参数过多，训练困难　　3.大量的参数容易导致过拟合

- 为了解决上述问题，提出了卷积神经网络



卷积神经网络相关概念

Convolutional Neural Network

卷积

- 卷积运算：对每个核单元值和与核单元重叠的对应图像像素值进行逐元素相乘，然后求和。精确值根据以下公式确定（m为内核宽度和高度，h为卷积输出，x为输入，w为卷积核）。

$$\mathrm{h}_{i,j} = \sum_{k=1}^{m} \sum_{l=1}^{m} w_{k,l} x_{i+k-1,j+l-1}$$

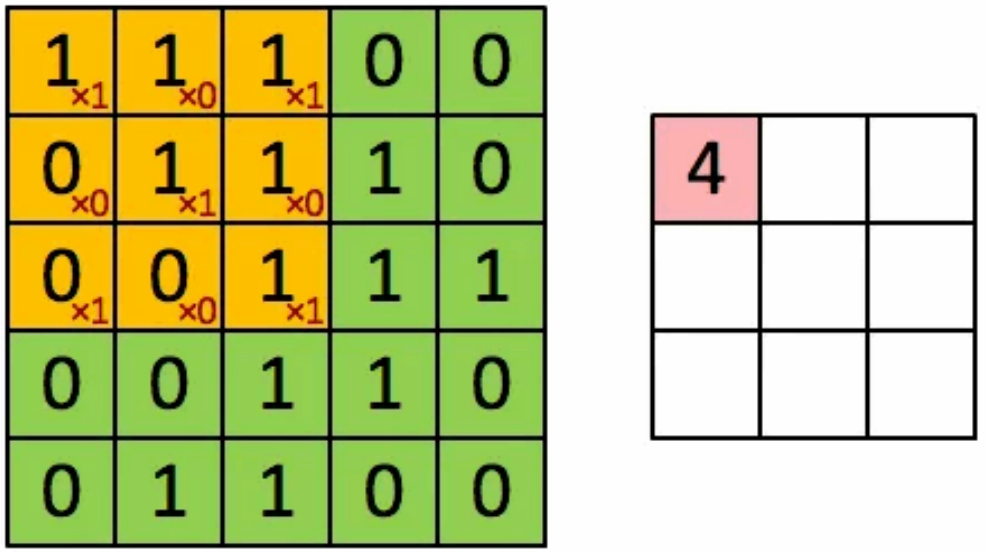
- 卷积作用：提取特征

卷积神经网络相关概念

Convolutional Neural Network

- 卷积

$$h_{i,j} = \sum_{k=1}^{m} \sum_{l=1}^{m} w_{k,l} x_{i+k-1,j+l-1}$$



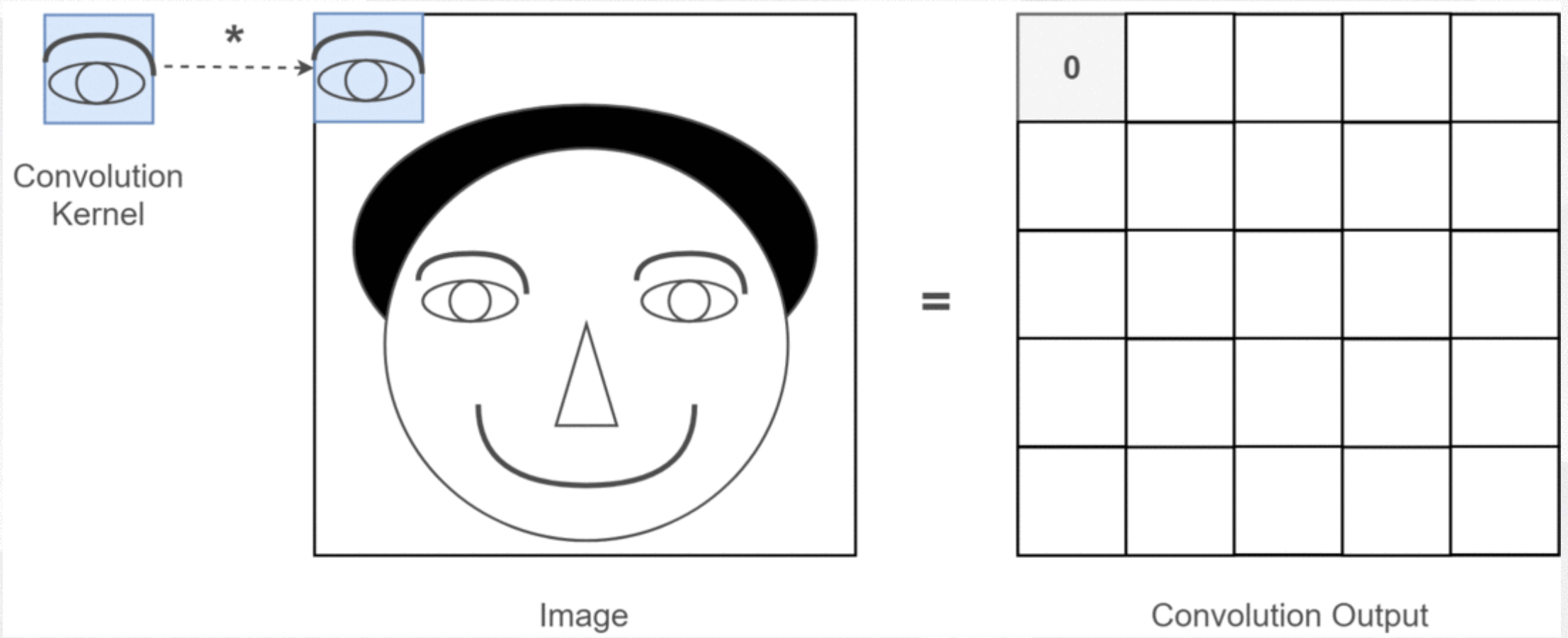
https://ezyang.github.io/convolution-visualizer/index.html

https://github.com/vdumoulin/conv_arithmetic

卷积神经网络相关概念

Convolutional Neural Network

- 卷积



卷积神经网络相关概念

Convolutional Neural Network

- feather map和卷积核的理解

- **feather map**：在cnn的每个卷积层，数据都是以三维形式存在的。你可以把它看成许多个二维图片叠在一起，其中每一个称为一个feature map。例如，灰度图像的feature map为1，rgb图像为3。

- **卷积核**：也被称为**过滤器**（filter），每个卷积核具有长、宽、深三个维度。在CNN的一个卷积层中：
  1.卷积核的长、宽都是人为指定的，长X宽也被称为卷积核的尺寸，常用的尺寸为3X3，5X5
  2.卷积核的深度与当前图像的深度（feather map的张数）相同，所以指定卷积核时，只需指定其长和宽 两个参数。
  3.卷积核中存储的是权重

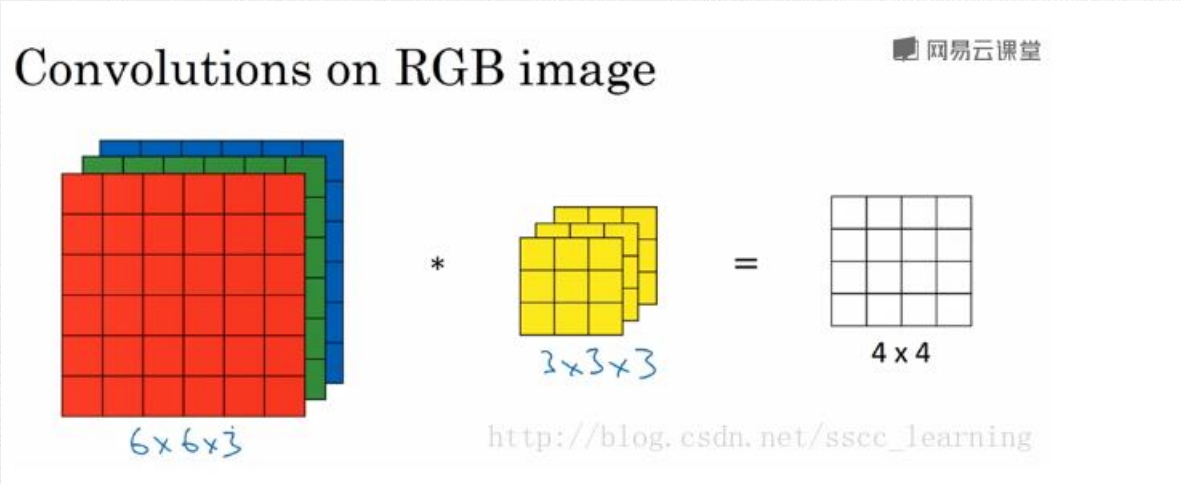
- **两者间的关系：一个卷积核对应一个featherMap**

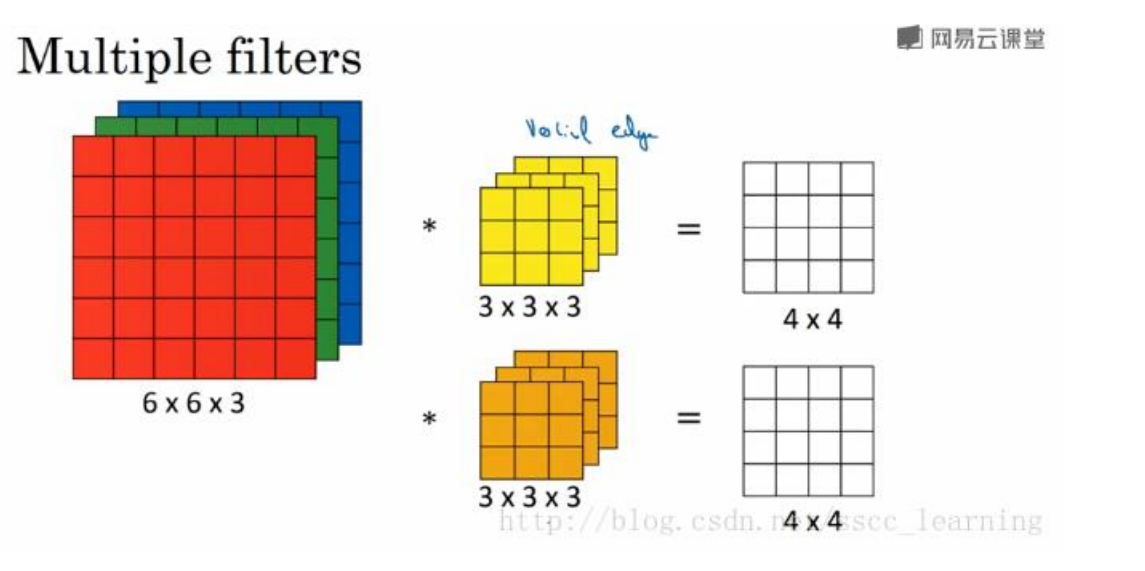
卷积神经网络相关概念

Convolutional Neural Network

卷积计算公式

$$a_{i,j} = f(\sum_{d=0}^{D-1}\sum_{m=0}^{F-1}\sum_{n=0}^{F-1} w_{d,m,n} x_{d,i+m,j+n} + w_b)$$



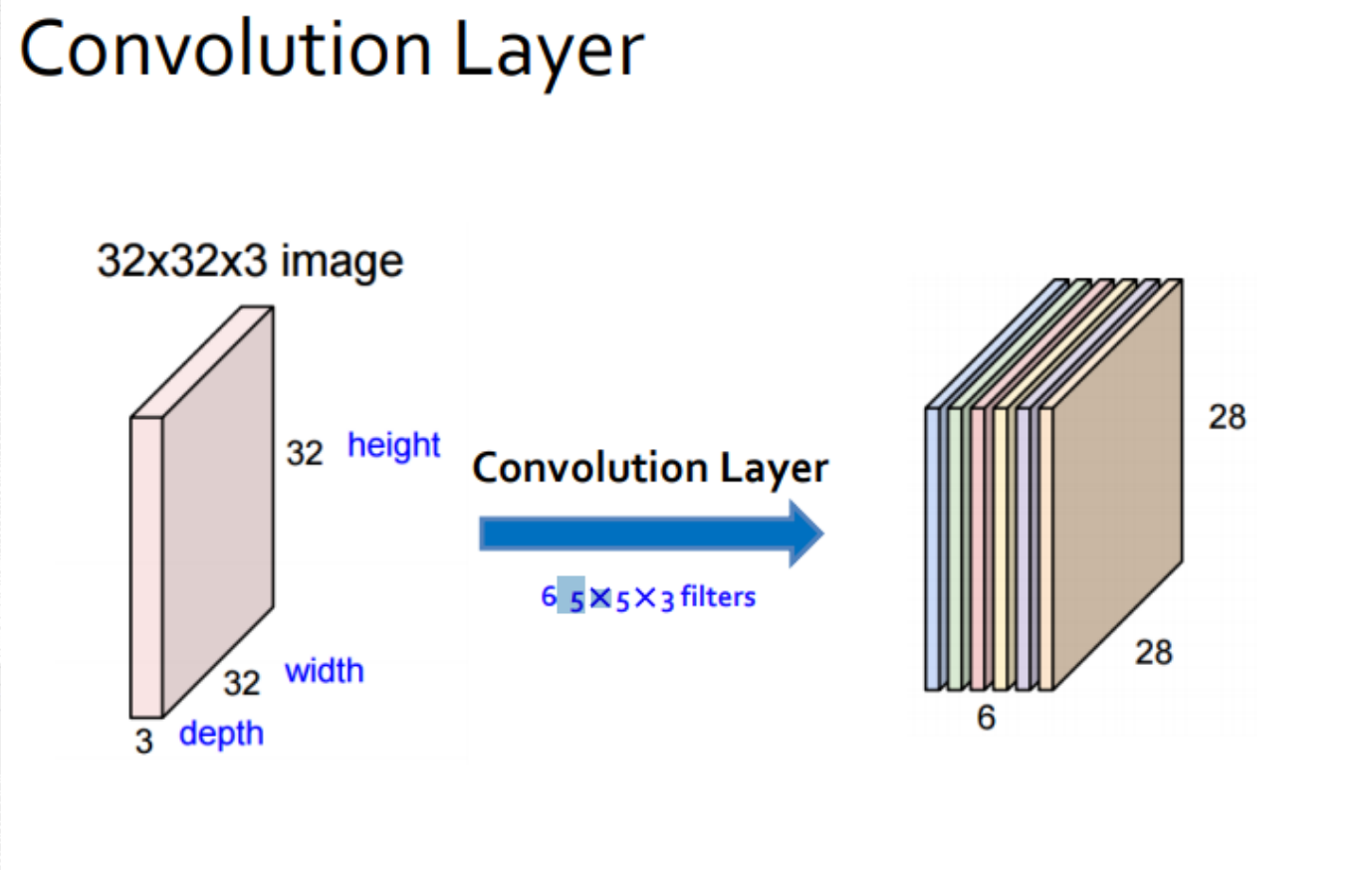


卷积神经网络相关概念

Convolutional Neural Network

卷积计算公式

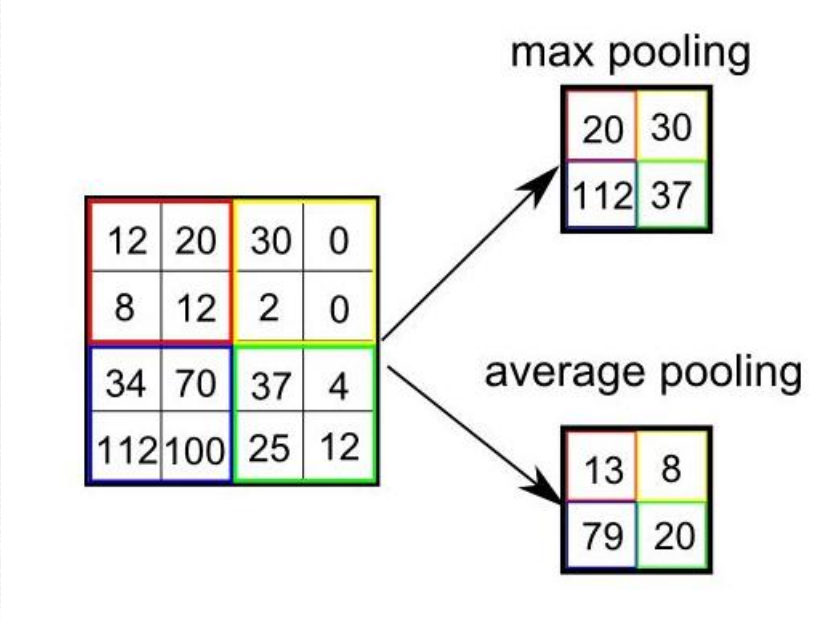
$$a_{i,j} = f(\sum_{d=0}^{D-1}\sum_{m=0}^{F-1}\sum_{n=0}^{F-1} w_{d,m,n} x_{d,i+m,j+n} + w_b)$$



卷积神经网络相关概念

Convolutional Neural Network

池化层

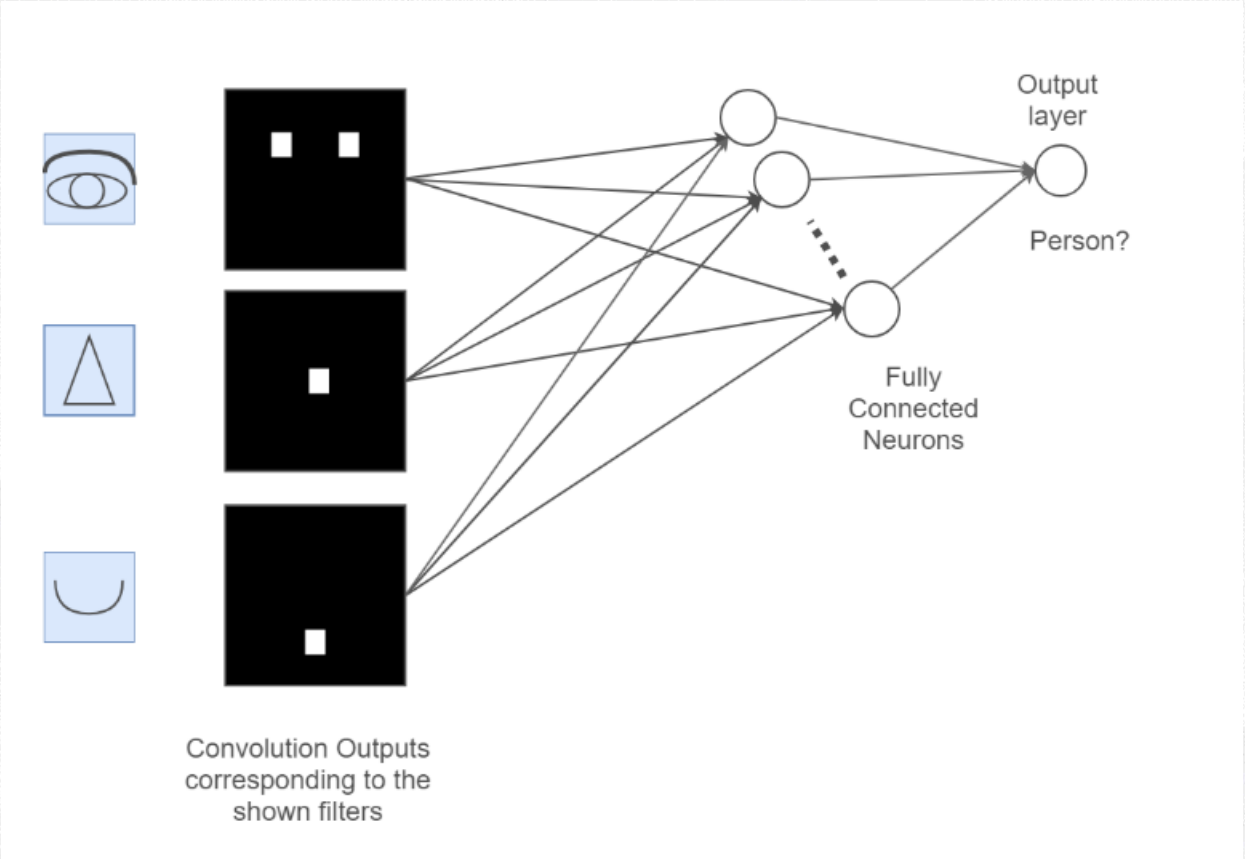
- 池化（Pooling）也叫做下采样（subsampling），用一个像素代替原图上邻近的若干像素，在保留feature map特征的同时压缩其大小。
- 池化的作用：1.防止数据爆炸，节省运算量和运算时间。 2.防止过拟合
- 

卷积神经网络相关概念

Convolutional Neural Network

全连接层

- 全连接的层将结合不同卷积核学习到的特征，以便网络可以构建有关整体图像的全局表示。简单的说就是将卷积层学习到的特征整合到一起

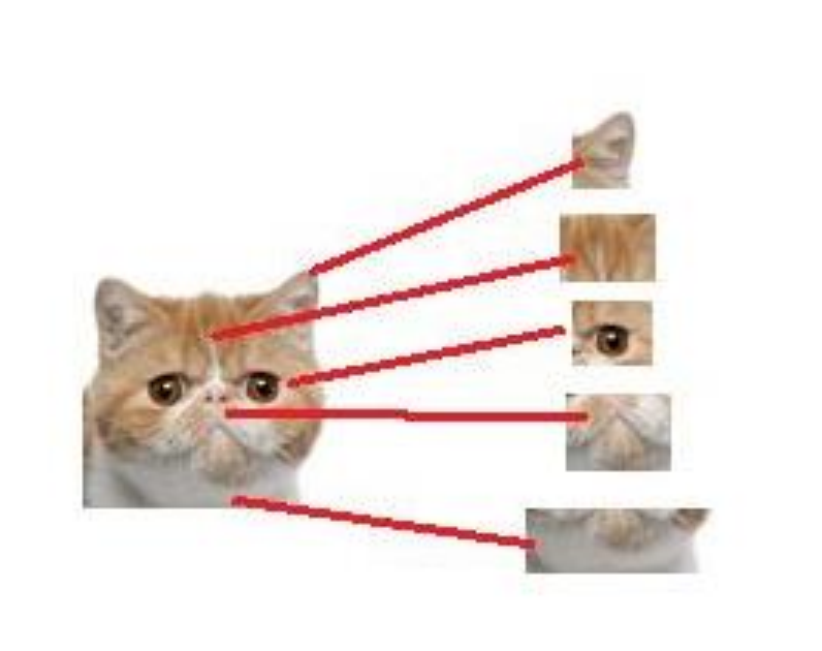


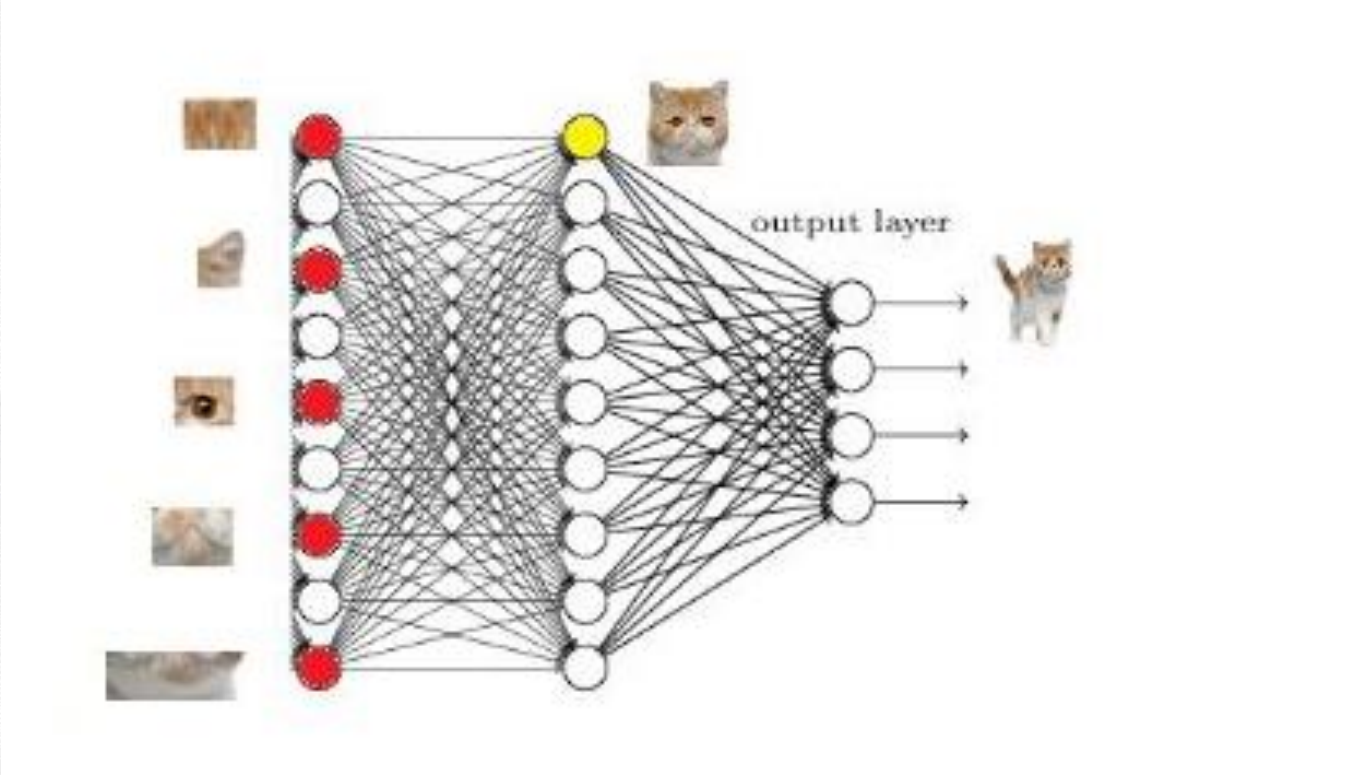
卷积神经网络相关概念

Convolutional Neural Network

全连接层

- 全连接的层将结合不同卷积核学习到的特征，以便网络可以构建有关整体图像的全局表示。





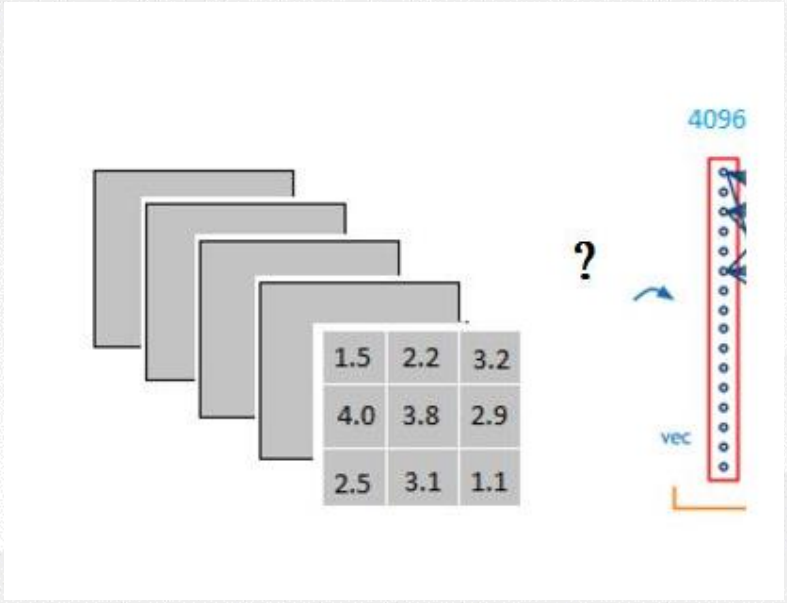
卷积神经网络相关概念

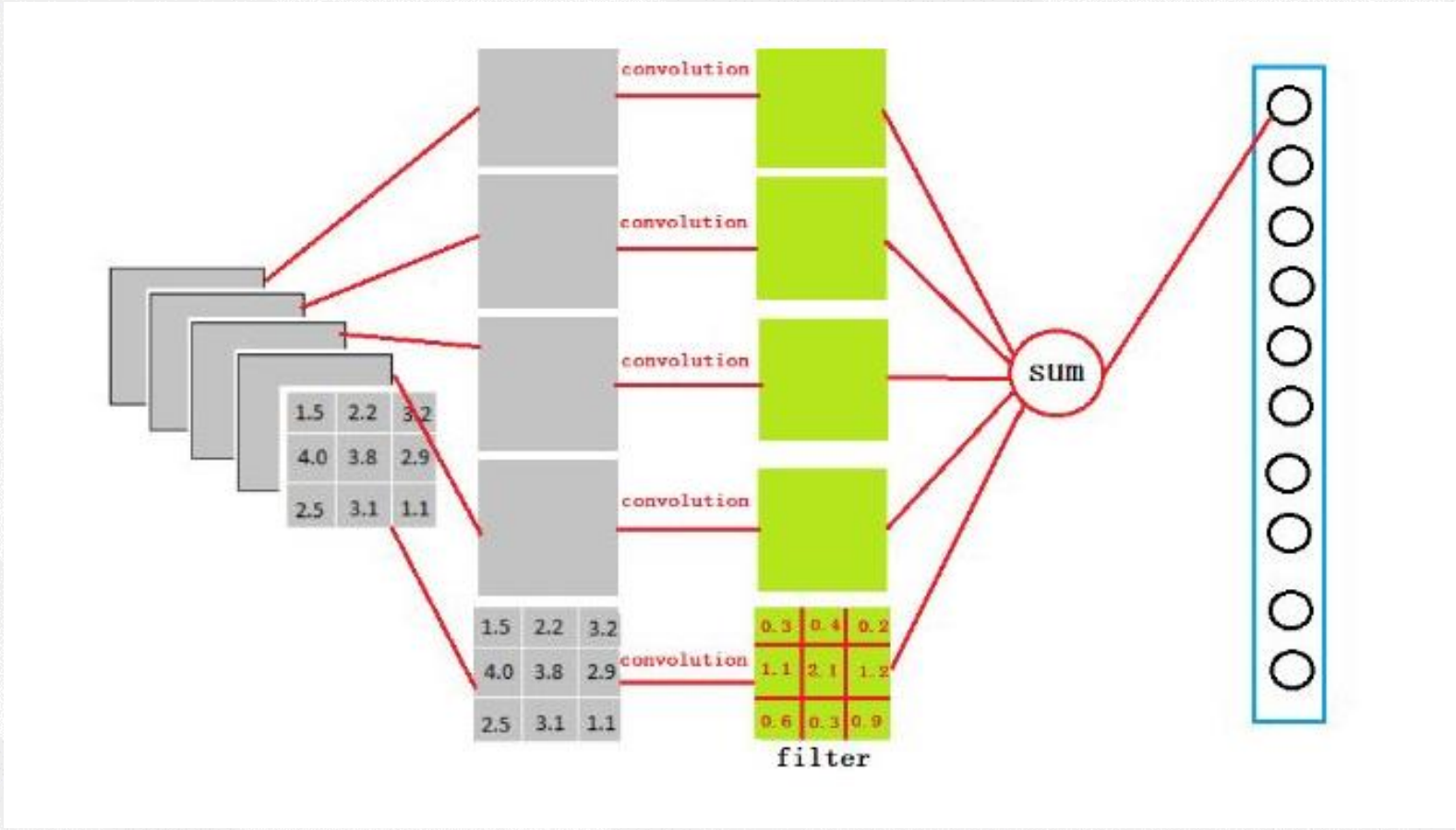
Convolutional Neural Network

- 全连接层

卷积：4096 * 3 * 3 * 5

- 理解为做了一次卷积

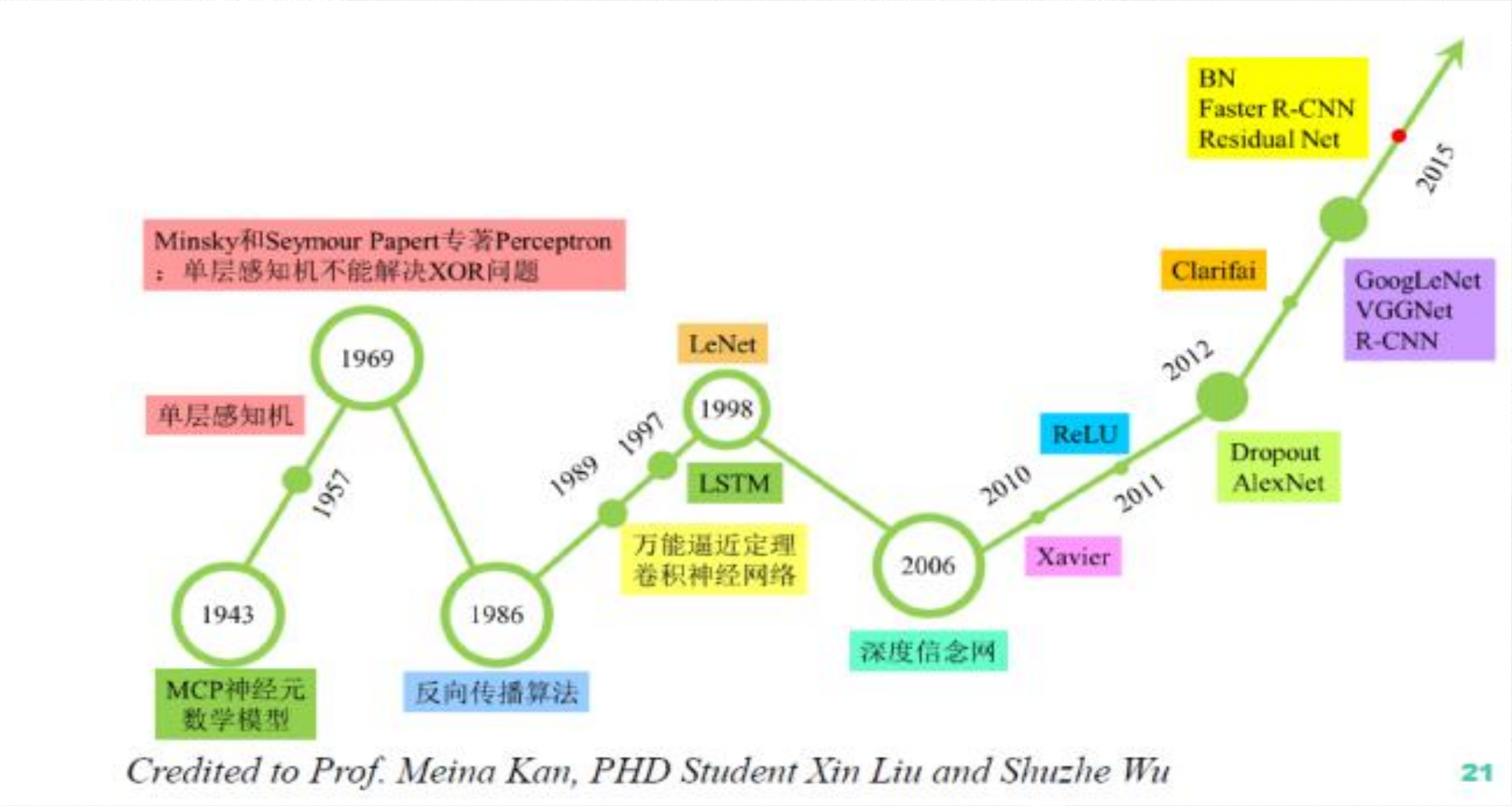




神经网络相关概念

Neural Networks

- 神经网络的历史



03

Part Three

编程实例

Specific examples of neural networks

编程实例

Specific examples of neural networks

- 参考资料

- 《机器学习》--周志华

- https://www.cnblogs.com/subconscious/p/5058741.html

- https://blog.csdn.net/xys430381_1/article/details/82529397

  https://zhuanlan.zhihu.com/p/33841176

感谢观看