深度学习

深度学习（deep learning）是机器学习的一个子领域，它采用了一个特定的模型：一族通过某种方式连接起来的简单函数。由于这类模型的结构是受到人类大脑结构的启发而创造出来的，因此我们通常把它们称为神经网络（neural networks）。神经网络中的函数链条能够将复杂的概念分解为多个层次的更简单的概念，这就是深度学习的核心思想。

- 深度学习的起源可以追溯至20 世纪40 年代，其雏形出现在控制论中．近十年来深度学习又重新走入了人们的视野，深度学习问题和算法的研究也经历了一次新的浪潮．

- 虽然卷积网络的设计受到了生物学和神经科学的启发，但深度学习目前的发展早已超越了机器学习模型中的神经科学观点．它用相对简单的函数来表达复杂的表示，从低层特征概括到更加抽象的高层特征，让计算机从经验中挖掘隐含的信息和价值．

机器学习中典型问题形式

很多机器学习中的问题可以写为：

$$\min_{x\in\mathcal{W}} \quad \sum_{i=1}^{N} \frac{1}{2}\|a_i^\top x - b_i\|_2^2 + \mu\varphi(x) \quad \text{线性回归}$$

$$\min_{x\in\mathcal{W}} \quad \frac{1}{N}\sum_{i=1}^{N} \log(1+\exp(-b_i a_i^\top x)) + \mu\varphi(x) \quad \text{逻辑回归}$$

$$\min_{x\in\mathcal{W}} \quad \frac{1}{N}\sum_{i=1}^{N} \ell(f(a_i,x),b_i) + \mu\varphi(x) \quad \text{一般形式}$$

- (a_i, b_i) 是给定的数据对，b_i 是数据a_i对应的标签
- $\ell_i(\cdot)$: 度量模型拟合数据点i的程度(避免拟合不足)
- $\varphi(x)$: 避免过拟合的正则项: $\|x\|_2^2$ 或者$\|x\|_1$等等
- $f(a,x)$: 线性函数或者由深度神经网络构造的模型

多层感知机

多层感知机（multi-layer perceptron, MLP）也叫作深度前馈网络（deep feedforward network）或前馈神经网络（feedforward neural network），它通过已有的信息或者知识来对未知事物进行预测.

- 在神经网络中，已知的信息通常用数据集来表示. 数据集一般分为训练集和测试集：训练集是用来训练神经网络，从而使得神经网络能够掌握训练集上的信息;测试集是用来测试训练完的神经网络的预测准确性.
- 一个常见的任务是分类问题. 假设我们有一个猫和狗的图片集，将其划分成训练集和测试集（保证集合中猫和狗图片要有一定的比例）. 神经网络是想逼近一个从图片到$\{0,1\}$ 的函数，这里0 表示猫，1 表示狗.
- 因为神经网络本身的结构和大量的训练集信息，训练得到的函数与真实结果具有非常高的吻合性.

多层感知机模型介绍

给定训练集$D = \{\{a_1, b_1\}, \{a_2, b_2\}, \cdots, \{a_m, b_m\}\}$.

- 假设数据$a_i \in \mathbb{R}^p$, $b_i \in \mathbb{R}^q$. $a_{i1} = 1$. 图3 给出了一种由p个输入单元和q个输出单元构成的$(L+2)$层感知机，其含有一个输入层，一个输出层，和L个隐藏层. 该感知机的第l个隐藏层共有$m^{(l)}$个神经元，为了方便用$l = 0$表示输入层，$l = L+1$ 表示输出层，并定义$m^{(0)} = p$和$m^{(L+1)} = q$. 设$y^{(l)} \in \mathbb{R}^{m^{(l)}}$为第$l$层的所有神经元，令$y_1^{(l)} = 1, 0 \le l \le L$,
- 其余的元素则是通过上一层的神经元的值进行加权求和得到. 令参数$x = (x^{(1)}, x^{(2)}, \cdots, x^{(L+1)})$ 表示网络中所有层之间的权重，其中$x_{i,k}^{(l)}$ 是第$(l-1)$隐藏层的第k个单元连接到第l隐藏层的第i个单元对应的权重，则在第l隐藏层中，第i个单元（$i > 1$，当$l = L+1$时可取为$i \ge 1$）计算输出信息$y_i^{(l)}$为

$$y_i^{(l)} = t(z_i^{(l)}), \quad z_i^{(l)} = \sum_{k=1}^{m^{(l-1)}} x_{i,k}^{(l)} y_k^{(l-1)}. \tag{9}$$

这里函数$t(\cdot)$称为激活函数.

常见的激活函数

常见的激活函数类型有Sigmoid 函数

$$t(z) = \frac{1}{1+\exp(-z)},$$

Heaviside函数

$$t(z) = \begin{cases} 1, & z \geq 0, \\ 0, & z < 0, \end{cases}$$

以及ReLU函数

$$t(z) = \max\{0, z\}. \tag{10}$$

整个过程可以描述为

$$y^{(0)} \xrightarrow{x^{(1)}} z^{(1)} \xrightarrow{t} y^{(1)} \xrightarrow{x^{(2)}} \dots \xrightarrow{t} y^{(L+1)}.$$

上述过程可用下图表示:

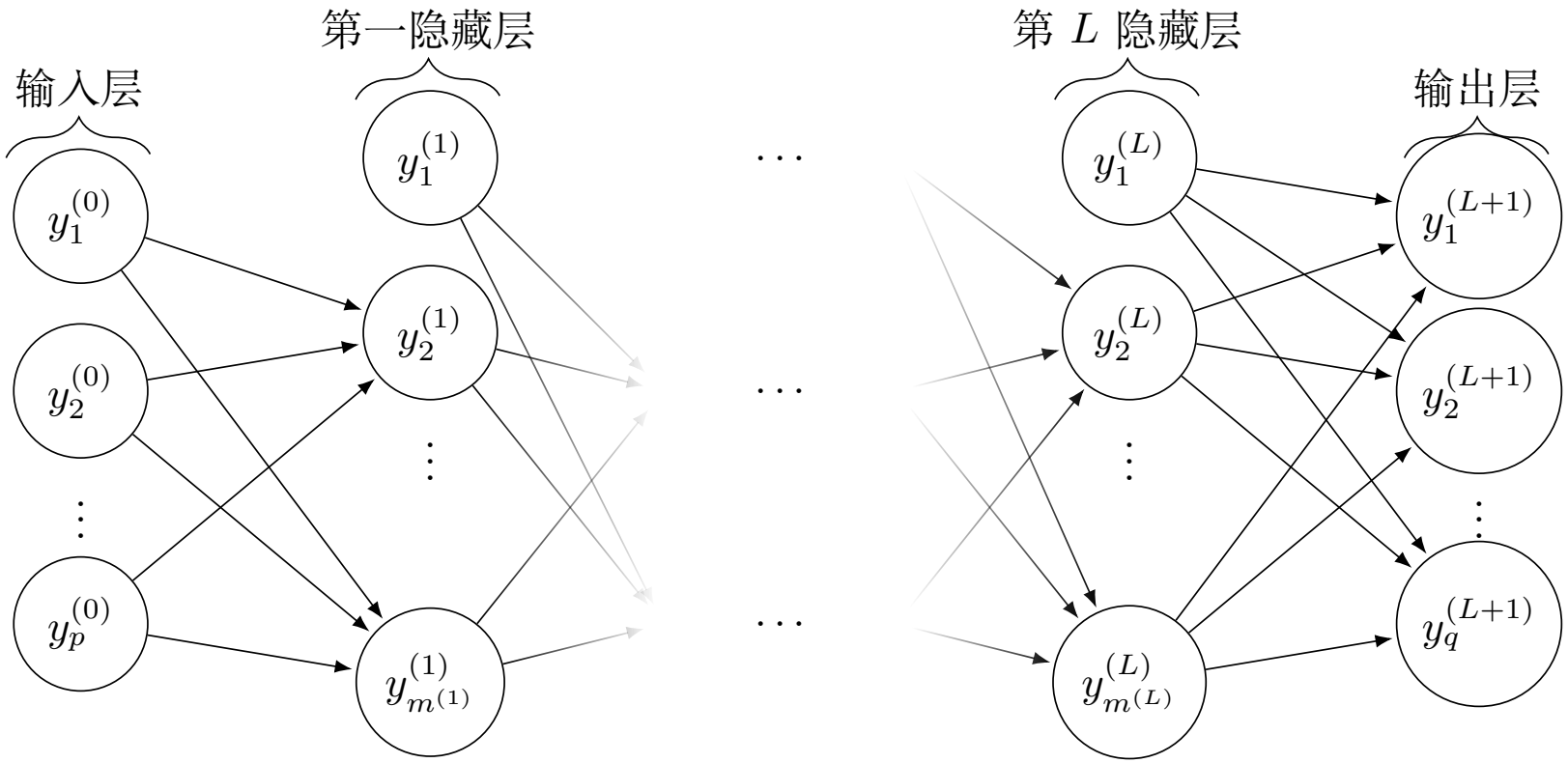


Figure: 带p个输入单元和q个输出单位的$(L+2)$层感知机的网络图，第l个隐藏层包含$m^{(l)}$个神经元.

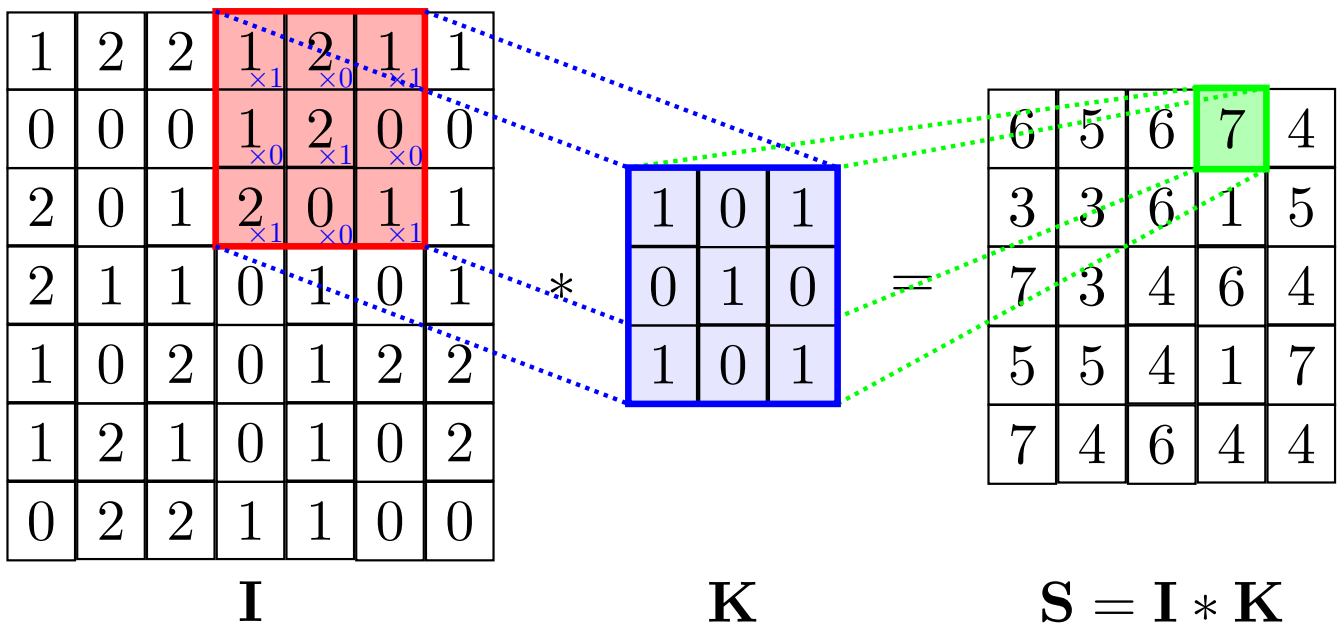
卷积神经网络Convolutional neural network (CNN)

- 给定二维图像$\boldsymbol{I} \in \mathbb{R}^{n\times n}$和卷积核$\boldsymbol{K} \in \mathbb{R}^{k\times k}$，定义卷积操作$\boldsymbol{S} = \boldsymbol{I} * \boldsymbol{K}$，它的元素是

$$\boldsymbol{S}_{i,j} = \langle \boldsymbol{I}(i : i+k-1, j : j+k-1), \boldsymbol{K} \rangle,$$

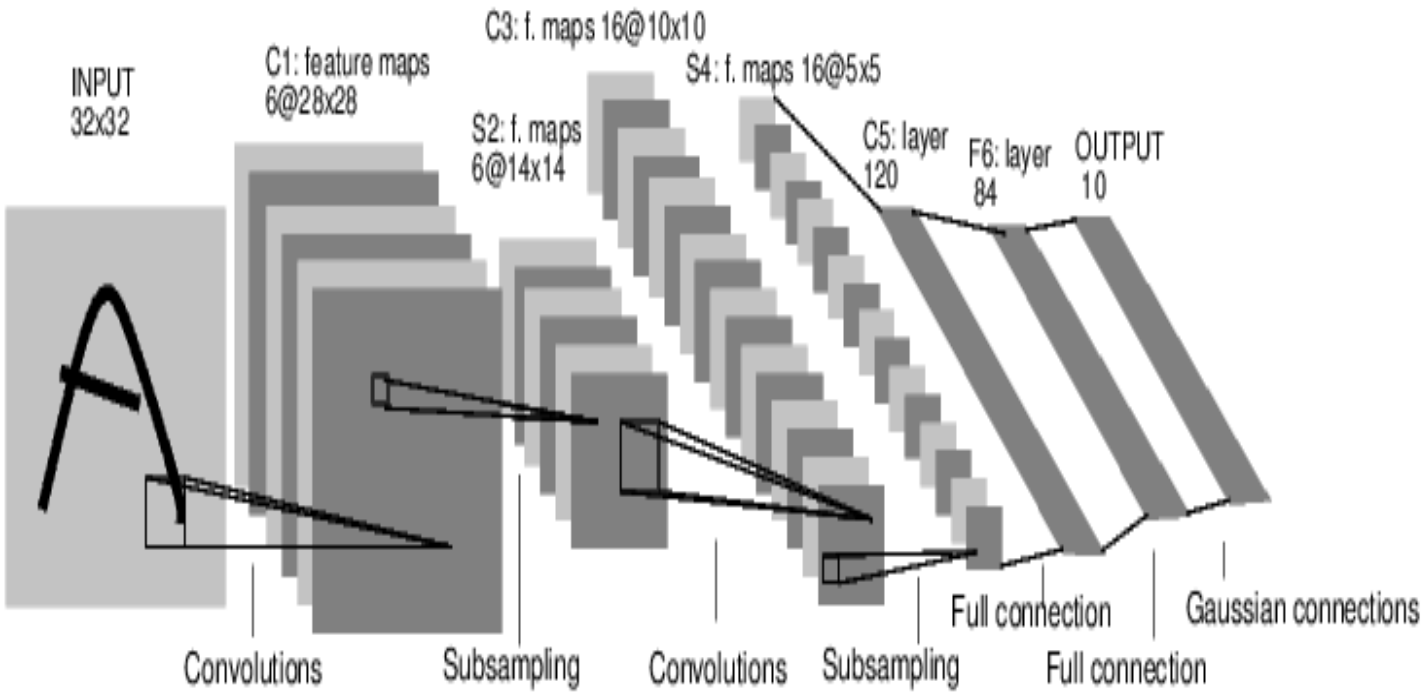
其中两个矩阵X, Y的内积是它们相应元素乘积之和

- 生成的结果$\boldsymbol{S}$可以根据卷积核的维数、$\boldsymbol{I}$的边界是否填充、卷积操作时滑动的大小等相应变化.



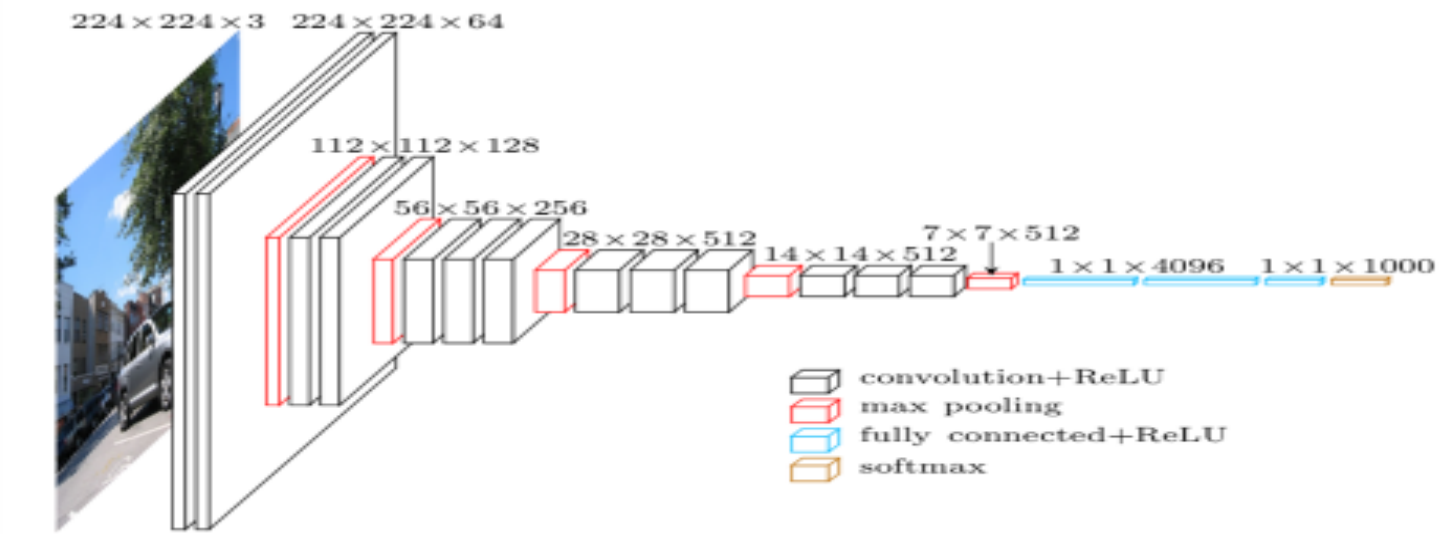
卷积神经网络Convolutional neural network (CNN)

LeCun等人开创性的建立了数字分类的神经网络。几家银行使用它来识别支票上的手写数字。



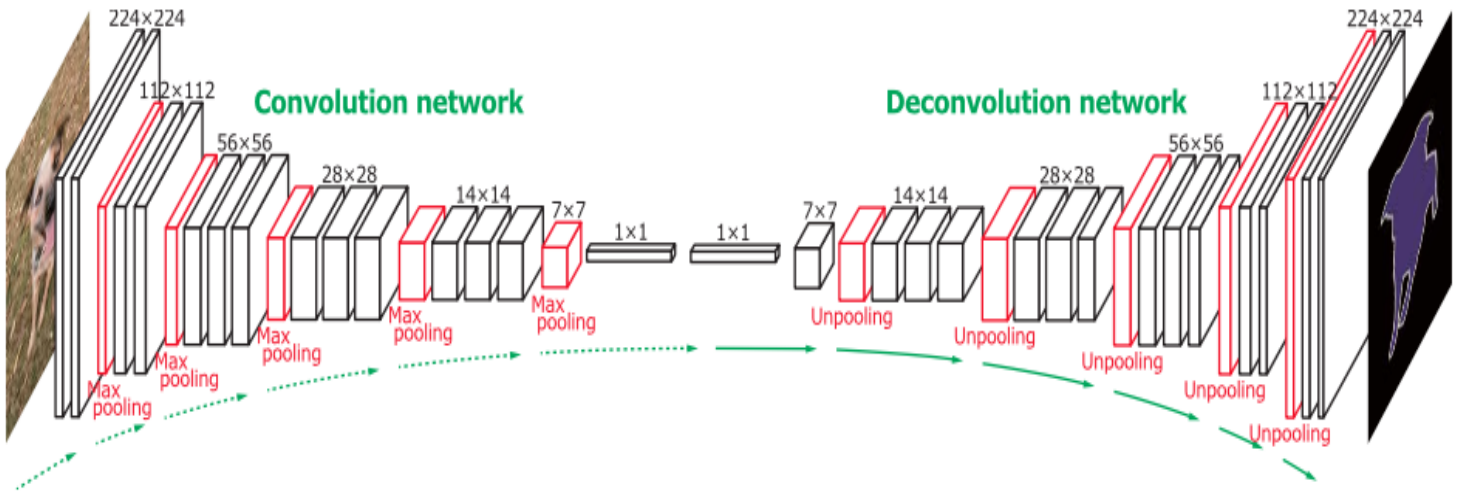
VGGNet

VGG小组（牛津）给出的19层卷积神经网络比AlexNet更简单层数更深。AlexNet中的所有大型过滤器都被3x3 过滤器的级联所取代（中间是非线性的）。在进行两次或三次卷积后放置最大合并，每次合并之后，过滤器的数量总是增加一倍。



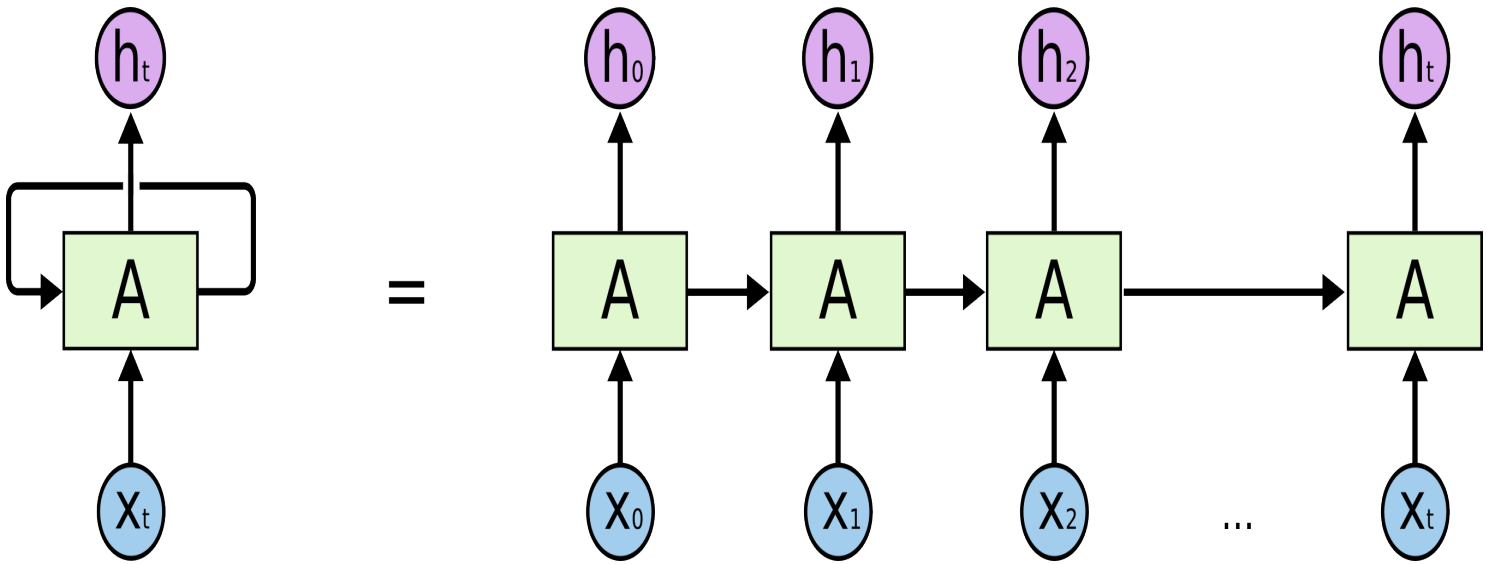
反卷积网络

生成网络是一种特殊的卷积网络，它使用转置卷积，也称为反卷积层。



递归神经网络Recurrent neural networks (RNN)

RNN建立在与前馈神经网络相同的计算单元上。RNN不必分层组织，并且允许定向循环。这样一来，他们就可以拥有内部存储器，从而可以处理顺序数据。如图所示，可以将RNN按顺序“展开”，从而将RNN转换为常规前馈神经网络。



深度学习中的优化算法

随机梯度类算法

- pytorch/caffe2 里实现的算法有adadelta, adagrad, adam, nesterov, rmsprop, YellowFin
  `https://github.com/pytorch/pytorch/tree/master/caffe2/sgd`
- pytorch/torch 里有：sgd, asgd, adagrad, rmsprop, adadelta, adam, adamax
  `https://github.com/pytorch/pytorch/tree/master/torch/optim`
- tensorflow 实现的算法有：Adadelta, AdagradDA, Adagrad, ProximalAdagrad, Ftrl, Momentum, adam, Momentum, CenteredRMSProp
  具体实现:
  `https://github.com/tensorflow/tensorflow/blob/master/tensorflow/core/kernels/training_ops.cc`

深度学习中的随机优化算法

考虑问题$\min_{x\in\mathbb{R}^d}\frac{1}{n}\sum_{i=1}^n f_i(x)$

参考: chapter 8 in

http://www.deeplearningbook.org/

- 梯度下降

$$x^{t+1}=x^t-\frac{\alpha^t}{n}\sum_{i=1}^n\nabla f_i(x^t)$$

- 随机梯度算法SGD

$$x^{k+1}=x^t-\alpha^t\nabla f_i(x^t)$$

- 带动量项的SGD

$$v^{t+1}=\mu^t v^t-\alpha^t\nabla f_i(x^t)$$
$$x^{t+1}=x^t+v^{t+1}$$