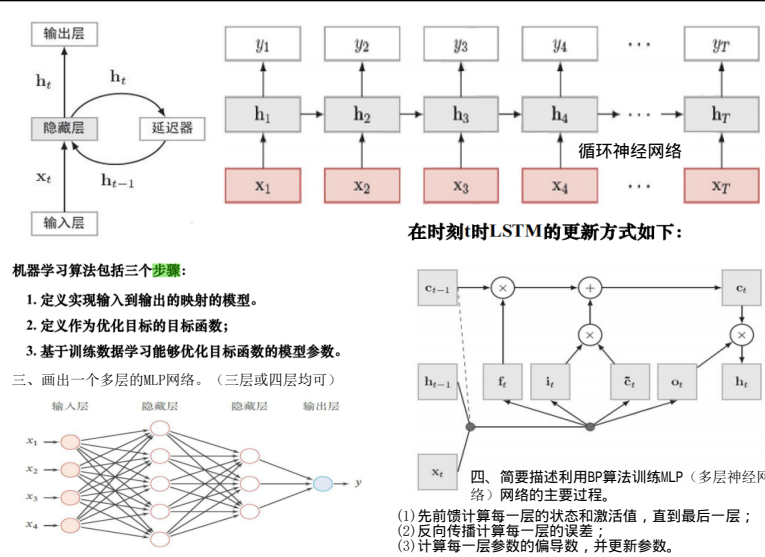


标量(Scalar): 单独的数量。**向量**(Vector): n 个实数的有序数组，一般为列向量 $n \times 1$ 。 **向量 a 的模**: $\|a\| = \sqrt{x_1^2 + a_2^2}$ 。**向量范数**(norm): 表示“长度”，向量空间内所有向量赋予非零的正长度或大小。**L1 范数**: $\|x\|_1 = \sum_{i=1}^n |x_i|$ ；**L2 范数**: $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2} = \sqrt{x^T x}$ ；**无穷范数**: $\|x\|_{\infty} = \max_i |x_i|$ ；**p 范数**: $\|x\|_p = (\sum_{i=1}^n |x_i|^p)^{1/p}$ 。 **映射**: 一个集合通过某种关系转为另一个集合。 一个集合中的元素(向量)，通过一个映射关系(矩阵)，得到另一个集合的元素(另一个向量)。 **矩阵范数**: 表示映射变化过程的大小一个度量。**对称矩阵A**: $diag(a)$, a 为一个 n 维向量，并满足: $A_{ij} = a_i$ 。 **1范数(列和)**: $\|A\|_1 = \max_j \sum_i |A_{ij}|$ ；**2范数(谱)**: $\|A\|_2 = \sqrt{\lambda_1}$, λ_1 为 $A^T A$ 的最大特征值；**∞范数(行和)**: $\|x\|_{\infty} = \max_i |x_i|$ ；**F 范数**: $\|A\|_F = (\sum_{i=1}^n \sum_{j=1}^n |a_{ij}|^2)^{1/2}$ 。 **张量**: 高阶张量。**导数**: 某点切线的斜率，函数在某点的变化率。**偏导数**: 多元函数沿坐标轴的变化率。**梯度**: 设函数 $f(x,y)$ 在平面区域 D 内具有一阶连续偏导数，则对其中一点 $p_0(x_0,y_0) \in D$ ，将向量 $f_x(x_0,y_0)$ 及 $f_y(x_0,y_0)$ 称为 f 在 p_0 的**梯度**。**实数函数**: 对 $-p$ 维向量 $\in R^p$, 函数 $y = f(x)$ 为 $f(x_1, \cdots, x_p) \in R$, 导数: $\nabla_x f(x) = [\partial f(x)/\partial x_1, \cdots, \partial f(x)/\partial x_p]^T$ 。**向量函数**: 对 $-p$ 维向量 $\in R^p$, 函数 $y = f(x)$ 为 $f(x_1, \cdots, x_p) \in R^q$, **导数**: $\nabla_x f(x) = [[\partial f_1(x)/\partial x_1, \cdots, \partial f_1(x)/\partial x_p] \cdots [\partial f_i(x)/\partial x_1, \cdots, \partial f_i(x)/\partial x_p]]^T$, 即 jacobian 矩阵。 **常见向量导数**: $dx^T/dx = 1$; $dx/dx^T = 1$; $dx^T A/dx = A$; $dAx/dx = A$; $dAx/dx = A^T$; $dxA/dx = A^T$; $dx^T x/dx = 2x$; $dx^T Ax/dx = (A + A^T)x$; $du^T Xv/dX = uv^T$; $\partial u^T X^T X u/dX = 2Xu^T$; $\partial (Xu - v)^T (Xu - v)/\partial X = 2(Xu - v)u^T$ 。 **乘法法则**: $y = f(x)$, $z = y(x)$, $\partial y^T z/dx = (\partial y/dx)^T z + (dz/dx)y$ 。 **Logistic 函数**: 将任意实数映射到(0,1)区间。 $\sigma(x) = 1/(1 + \exp(-x))$ 。输出为(0,1)，单调上升且连续。**导数**: $\sigma'(x) = \sigma(x)(1 - \sigma(x))$ 。 **Softmax 函数**: 多标量输出映射到(0,1)区间，用于多类分类，输出可理解为概率分布(归一性)。 $\text{softmax}(x_k) = \exp(x_k)/(\sum_{i=1}^n \exp(x_i))$ 。 输入为 K 维向量 $x = [x_1, \cdots, x_K]^T$, $\text{softmax}(x) = 1/(\sum_{i=1}^n \exp(x_i)) * [\exp(x_1) \cdots \exp(x_K)]^T = \exp(x)/(\sum_{i=1}^n \exp(x_i)) = \exp(x)/1_i \exp(x)$, $1_K = [1, \cdots, 1]_K$ 是 K 维的全 1 向量。**导数**: $\partial \text{softmax}(x)/\partial x = \text{diag}(\text{softmax}(x)) - \text{softmax}(x)\text{softmax}(x)^T$ 。 **机器学习中**: 研究使计算机从给定的数据学习规律，向学习到的规律对未知预测。 给定一些训练样本 (x_i, y_i) , $1 \leq i \leq N$ (x_i : 输入实例, y_i : 需要预测的目标)，让计算机自动寻找一个决策函数 $f(\cdot)$ 来建立 x 和 y 之间的映射关系。**模型**: $\hat{y} = f(\phi(x, \theta))$, x : 输入; $\phi(x)$: 表示特征提取过程。**目标**: 找到一组模型参数 θ ，使模型生成期望的预测输出。**损失函数**: $L(y, f(x, \theta))$ 。 基于损失函数，在所有训练样本上来评价决策函数的**风险函数**: $R(\theta) = (\sum_{i=1}^N L(y^{(i)}, f(x^{(i), \theta)}))/N$, 风险函数 $R(\theta)$ 是在已知的训练样本(经验集)上计算得的，称为**经验风险**。**经验风险最小化原则**(Empirical Risk Minimization): 对参数求解，使经验风险逼近理想期望风险的最小值。**存在问题**: 1. 用来训练的样本往往只是真实数据的一个很小的子集或者包含一定的噪声，不能完全反应全部数据的真实分布; 2. 经验风险最小化代表在真实数据的分布上表现最优。**3. 过拟合**或**经验风险最小化**很容易导致模型在训练集上错误率低，但在测试集上的错误率高的问题。**过拟合原因**: 特征数量过多，训练样本太少。**解决**: 增加样本数量或样本多样性、正则化。**Early-Stop**: 用一个检验集(validation Data)测试每次迭代的参数在验证集上是否最优。如果在验证集上的错误率不再下降就停止迭代; 无验证集时可在训练集上交叉验证。**泛化错误**(衡量机器学习模型是否可以很好地泛化到未知数据的能力): 一般表现为一个模型在训练集和验证集上错误率的差距。 **正则化**: 在目标函数引入模型参数的 L2 范数作为惩罚项，在优化目标函数时，参数值会被减小。 $\theta^* = \text{argmin}_{\theta} L(\theta) + \lambda \|\theta\|_2^2$, λ : 控制正则化项强度, $\|\theta\|_2^2$: 正则化项, 增加模型参数的复杂度，避免过拟合。正则化项也可用 L1 范数。 L1 范数(常用): 会使参数有一定稀疏性。**0-1 损失函数**: $L(y, f(x, \theta)) = I(y \neq f(x, \theta))$, 局限: 不考虑预测值与真实值的误差程度。**平方损失函数**(quadratic loss function): $L(y, \hat{y}) = (y - f(x, \theta))^2$ 。 对两类多类问题，假设 y 和 $f(x, \theta)$ 的取值为 $\{-1, +1\}$ 。 **Hinge 损失函数**(Hinge Loss Function): $L(y, f(x, \theta)) = \max(0, 1 - yf(x, \theta))$ ，交叉熵损失函数(Cross Entropy Loss function): $L(y, f(x, \theta)) = -\sum_{i=1}^C y_i \log f_i(x, \theta)$ 。 根据训练数据提供的信息以及反馈方式的的不同，机器学习算法分类: 1. **有监督学习**(Supervise Learning): 利用一组已知输入和输出 y 的数据来学习，使模型预测的输出标记和真实标记尽可能一致。根据输出类型分为回归和分类问题。 ①**回归**(Regression): 若输出 y 是连续值(实数或连续整数)， $f(x)$ 的输出也是连续值。**损失函数**: 通常为平方误差。**②. 分类**(Classification): 输出 y 是离散的分类标记(符号)。**损失函数**: 常用 0-1 损失或平方损失函数。学习得到的决策函数 $f(x)$ 叫分类器。**2. 无监督学习**(Unsupervised Learning): 用来学习的样本不包含输出目标，需学习算法自动学习到一些有价值的信息，如数据的分布和参数间关系。**应用**: 聚类和降维。**3. 增强学习**(Reinforcement Learning): 也叫强化学习，强调如何基于环境做出一系列的动作，以取得最大化的累积收益。每做出一个动作，并不一定立即得到收益。**2. 与 3 区别**: 增强学习不需要显式地输入输出对的方式给出训练样本，是一种在探索学习机制。**4. 弱监督和半监督学习**: 数据: 所有能计算机程序处理的形式的统称。**特征选择**: 很多算法的输入要是数学上可计算的。而现实中，原始数据常常是并不都以连续变量或离散变量形式存在。需抽取一些可表征数据的数值型特征。一般可表示为向量形式。**特征(表示)学习**: 自动地学习有效的特征(需抽取有效的、稳定的特征，传统的特征提取用人工方式)。**分类: 特征选择**(从多特征集里选有效子集)、**特征提取**(构造一个新的特征空间，并将原始特征投影新的空间中)、**样式**: 按照一定的抽样规则从全部数据中取出一部分数据，是实际观测得到的数据。**机器学习算法步骤**: 1. 选择一种目标函数形式; 2. 从训练数据中学习以目标函数参数。**模型选择学习算法**: 从训练集的样本中自动学习目标函数参数。**训练过程**: 训练数据集上，通过学习中学习，自动学习模型参数的过程。**不同机器学习算法区别**: 决策函数、学习算法。相同的决策函数可以有不同的学习算法。**优化方法**:

$\sum_{i=1}^C \exp(\theta_i^{(c,0)})$; **参数矩阵**: $W^T = [\theta_1, \theta_2, \cdots, \theta_C]^T$ 。 **Softmax 回归模型**: $z = W^T x$, $y = \text{softmax}(z) = [e^{z_1}, \cdots, e^{z_C}]^T / \sum_{c=1}^C e^{z_c}$, y 的第 c 维的值是第 c 类的预测后验概率。**损失函数**: 对于样本 (x, y) ，输出目标 $y \in \{1, \cdots, C\}$ 。用 C 维 y 表示输出: $y = [I(1 = c), I(2 = c), \cdots, I(C = c)]^T$ 。给定 N 个样本 $(x^{(i)}, y^{(i)})$ ，交叉熵损失函数: $L(y, f(x, \theta)) = -\sum_{i=1}^N y_i \log f_i(x, \theta)$ ；**风险函数**: $J(w) = -\sum_{i=1}^N \sum_{c=1}^C y_i^{(c)} \log y_c^{(i)}$ ， $-\sum_{i=1}^N \sum_{c=1}^C y_i^{(c)} \log y_c^{(i)}$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{(i)})\text{softmax}(z^{(i)T})$ ；**梯度下降法**: $\frac{\partial J(w)}{\partial w} = \frac{\partial \text{softmax}(z^{(i)})}{\partial z} \frac{\partial \text{softmax}(z^{(i)})}{\partial w} = \text{diag}(\text{softmax}(z^{(i)})) - \text{softmax}(z^{$

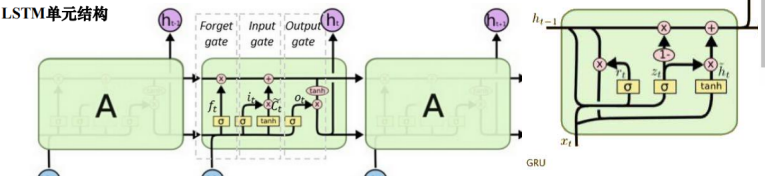
滤波器 f : 卷积输出: $y_{ij} = \sum_{m=1}^M \sum_{n=1}^N f_{m,n} x_{i-m,j-n+1}$, $j = 1 \sim n_2$ 。卷积核: 将图像中符合条件(激活值越大越符合条件)的部分筛选出来。**卷积代替全连接**: 全连接前馈神经网络, m 和 n 都很大时, 权重矩阵的参数非常多, 训练的效率会非常低。用卷积来代替全连接, 第1层的每个神经元都和第1-1层的的一个局部窗口内的神经元相连, 构成一个局部连接网络。第1层的第 i 个神经元的输入: $a_i^{(1)} = f(\sum_{m=1}^M w_{i-m+1}^{(1)} a_0^{(0)} + b^{(1)}) = f(w_{i-m+1+1}^{(1)} a_1 + b_i)$, $w^{(1)}$ 为 M 维的滤波器。**卷积形式**: $a^{(1)} = f(w^{(1)} \otimes a^{(0)} + b^{(1)})$ 。在卷积神经网络中, $m+1, n+1$ 层的神经元: $X_i^{(1)} = f(\sum_{j=1}^n w_{i-j+1}^{(1)} X_j^{(0)} + b^{(1)})$, $W^{(1)}$ 为 $u \times v$ 滤波器。卷积形式: $X^{(1)} = f(W^{(1)} \otimes X^{(0)} + b^{(1)})$ 。**多卷积核**: 一个滤波器只能产生一种输出, 为了增强卷积层的表示能力, 可以使用不同的滤波器来得到多组输出。每个卷积核都会将图像生成成为另一幅图像。比如两个卷积核就可以将生成两幅图像, 这两幅图像可以看成是一张图像的不同通道。如果我们把滤波器看成是一个特征提取器, 每组输出都可以看成是输入图像经过一个特征抽取后得到的特征。第1层的每一组特征极都依赖于前一层的所有元素: $X_i^{(1)} = f(\sum_{j=1}^n w_{i-j+1}^{(1)} X_j^{(0)} + b^{(1)})$ 。若第1层-1层的第 k 组特征映射依赖于前一层的所有元素, 则 $T_{p,k} = 1$, 否则为 0。**输出**: $X^{(1)} = f(\sum_{p,T} T_{p,k} W^{(1,p)} \otimes X^{(0,T-1,p)} + b^{(1,k)})$, 连接表 T 的非零数为 K , 每个滤波器的大小为 uv , 参数量: $kuv + n_1$ 。**子采样层**: 卷积层虽然可以显著减少连接的个数, 但每个特征映射的神经元个数未显著减少。若后面接一个分类器, 则分类器的输入维数依然很高, 易出现过拟合。**池化(子采样)**: 由于自然图像具有统计特性空间不变性, 意味着在一个图像区域有高的特征极可能在一个区域周围适用, 因此对不同位置的特征进行聚合统计, 计算图像中1个区域上的某个特定特征的平均值(或最大值)。这些概要统计特征不仅具有低得多的维度, 同时还会改善结果(不容易过拟合)。**池化目的**: 解决卷积后特征维度过高的问题, 大大降低特征的维数, 避免过拟合。**池化操作**: 对于卷积层得到的一个特征映射 $X(i)$, 可以将其划分为很多区域(可重叠), 取区域中所有神经元的最大值或者平均值。 $pool_{max}(R_k) = \max_{i \in R_k} x_i$, $pool_{avg}(R_k) = \frac{1}{|R_k|} \sum_{i \in R_k} x_i$; (1)平均池化: 卷积核中每个地权都是 0.25, 步长为 2。**效果**: 把原图模糊缩减至原来的 1/4。(2)最大值池化: 卷积核中每个权重中只有 1 为(最强位置), 其余均为 0, 步长为 2。**效果**: 把原图缩减至原来的 1/4, 保留每个区域的最强输入。**完整子采样层运算**: $X^{(1+1)} = f(Z_i^{(1+1)}) = f(w^{(1+1)} down(R_k) + b^{(1+1)})$ 池化后做一次线性与非线性运算, $w^{(1+1)}$ 和 $b^{(1+1)}$ 分别是可训练的权重和偏置参数。**子采样作用**: 可以使得下一层的神经元对一些小形态改变保持不变性, 并拥有更大的感受野。**LeNet-5**: 不计输入层, LeNet-5 共有 7 层。输入层: $32 \times 32 \times 1$ 。C1 层: 卷积层, 滤波器大小 5*5, 有 6 个, 得到 16 组 28×28 的特征映射, 神经元个数为 $6 \times 28 \times 28$, 可训练参数个数(含偏置) $6 \times 25 \times 6$, 连接数 $6 \times 28 \times 28 \times (5 \times 5 + 1)$; S2 层: 子采样, C1 层每组特征映射中 2×2 领域点次采样为 1 个点, 神经元个数 $14 \times 14 \times 6$, 可训练参数 $6 \times (1+1)$, 连接数: $6 \times 14 \times 14 \times (4+1)$; C3 层: 卷积层, 连接表显示与上一层总共 有 60 个连接, 所以有 60 个滤波器, 大小 5*5, 得到 16 组大小为 10×10 的特征映射, 神经元个数为 $16 \times 10 \times 10$, 可训练参数 $60 \times 5 \times 5 + 16$, 连接数 $60 \times 100 \times 25 + 100 \times 16 \times 1$; S4 层: 子采样, 2×2 领域点次采样为 1 个点, 得 16 组 5×5 特征映射, 可训练参数 $16 \times (1+1)$, 连接数 $16 \times 5 \times 5 \times (4+1)$; C5 层: 卷积层, 得 120 组 1×1 得特征映射, 每个特征映射与 S4 层得全部特征映射相连, 有 $120 \times 16 \times 5 \times 5$ 的滤波器, 神经元个数 120 个, 可训练参数 $120 \times 16 \times 5 \times 5 + 120$, 连接数 $120 \times 16 \times (16 \times 25 + 1)$; F6 层: 全连接层, 有 84 个神经元, 可训练参数 $84 \times (120+1)$, 连接数与可训练参数相同; 输出层由 10 个欧式径向基函数(Radial Basis Function,RBF)组成。参数个数=滤波器个数*卷积核大小+输出特征组数; 连接数=神经元个数*(每个神经元连接层数*卷积核大小+1)。

【循环神经网络(RNN)】前馈神经网络(全连接神经网络/卷积神经网络): 元素之间是相互独立的(每层之间的节点是无连接的), 输入与输出也是独立的(多个输入之间是完全没有关系的, 只能单独的去处理独立的输入)。**序列输入**: 某些任务需要能够更好的处理序列的信息, 即前面的输入和后面的输入是有关的。当处理序列数据时, 前馈神经网络无法包含输入数据之间的关联关系。为了能够更好的处理大量的序列数据, 一种方法是循环神经网络。**循环神经网络**: 通过使用带自反馈的神经元, 对前面的信息进行记忆并使其参与到当前输出的计算, 理论上能处理任意长度的序列数据。一个序列当前的输出与当前的输入以及前面的输出有关。给定一个输入序列 $\{x_t\}$, **隐藏层的活性值 h_t 更新方式**: $h_t = 0, t = 0; f(h_{t-1}, x_t)$, otherwise。**信息记忆表现**: 隐藏层的输入不仅包括输入层的输出还包括上一时刻隐藏层的输出。**特点**: 1.相比于前馈神经网络, 循环神经网络更符合生物神经网络的结构。2.循环神经网络已经被广泛应用于语音识别、语言模型以及自然语言处理等任务上。3.参数训练可通过随时间进行**反向传播算法**。**最大问题**是训练时梯度需要随着时间进行反向传播。4.当输入序列比较长时, 会存在梯度爆炸和消失问题。5.长短期记忆神经网络(LSTM)是循环神经网络的一个拓展。**简单循环网络(输入层、一个隐藏层和一个输出层)**。假设时刻 t 时, 输入为 x_t , 隐层状态(隐层神经元活性)为 h_t , h_t 可表示为: $h_t = f(Uh_{t-1} + Wx_t + b)$, f 是非线性函数(通常为 logistic 或 tanh)。 $o_t = g(Vh_t + b)$ 。**展开**: 将循环神经网络展开后可得到链式结构的神经网络, 链式结构的节点之间通过隐状态连接。**参数训练**: **随时间进行反向传播算法**: 假设循环神经网络在每个时刻 t 都有一个监督信息, 损失为 J_t 。则整个序列的损失: $J = \sum_{t=1}^T J_t$ 。



五、卷积神经网络由哪些层组成, 各层的主要功能是什么?
卷积神经网络由输入层、多组卷积层、降采样层(池化层)、全连接层和输出层组成。(特征提取器+分类器)
1. 输入层的功能是输入特征数据, 进行训练或者测试。
2. 卷积层的功能是从输入数据中提取特征, 特征提取过程就是对输入数据(通常是图像)进行卷积运算, 提取出图像中的特征。卷积层具有局部感受野和权重共享的特点。
3. 池化层的功能是进行特征选择, 降低特征数量, 从而减少参数数量, 防止过拟合。在保留有用信息的基础上减少数据量, 即提取特征在不同位置和规模上的变化, 同时聚合不同特征映射的响应。
4. 全连接层的功能主要是将图像特征图的“分布式特征表示”映射到样本标记空间。在整个卷积神经网络中起到“分类器”的作用。
5. 输出层的作用是输出分类结果。

二、简述什么是过拟合现象, 并列举三种常用的防止过拟合方法。
原因: 1.特征数量过多(模型过于复杂, 参数过多); 在训练集上错误率最低, 但是在测试集上错误率高, 模型的泛化性能差。2.训练样本太少
防止过拟合方法: 1.增加训练样本数(样本增强) 2.正则化(结构风险最小化) ①权重衰减: 通过添加衰减参数来惩罚权重较大的参数实现(L2范数) ②Dropout: 按概率随机消除神经节点, 降低神经元间的关联性和模型的复杂度。3. Early stop: 如果在验证集上的错误率不再下降, 就停止迭代。使用验证集测试参数是否存在验证集最优。没验证集可用交叉验证法。



损失 J 关于 U 的梯度为: $\frac{\partial J}{\partial U} = \sum_{t=1}^T \sum_{k=1}^K \frac{\partial J}{\partial a_t^{(k)}} \frac{\partial a_t^{(k)}}{\partial U} = \sum_{t=1}^T \sum_{k=1}^K \frac{\partial J}{\partial a_t^{(k)}} \frac{\partial a_t^{(k)}}{\partial U} \frac{\partial a_t^{(k)}}{\partial U}$, h_{t-1} 是关于 U 和 h_{t-2} 的函数, 而 h_{t-1} 又是关于 U 和 h_{t-2} 的函数。因此, 由链式法则可得 $\frac{\partial J}{\partial U} = \sum_{t=1}^T \sum_{k=1}^K \frac{\partial J}{\partial a_t^{(k)}} \frac{\partial a_t^{(k)}}{\partial U} \frac{\partial a_t^{(k)}}{\partial U}$, 其中, $\frac{\partial a_t^{(k)}}{\partial U} = \Pi_{i=k+1}^K \frac{\partial a_t^{(k)}}{\partial h_{t-1}} = \Pi_{i=k+1}^K U^T \text{diag}[f'(h_{t-1})]$ 。因此, $\frac{\partial J}{\partial U} = \sum_{t=1}^T \sum_{k=1}^K \sum_{i=k+1}^K \frac{\partial J}{\partial a_t^{(k)}} \left(\Pi_{i=k+1}^K \frac{\partial a_t^{(k)}}{\partial h_{t-1}} \right) \frac{\partial a_t^{(k)}}{\partial U}$ 。定义: $v = \left\| U^T \text{diag}[f'(h_{t-1})] \right\|$, 则公式中的括号里面为 v^{t-k} 。若 $\gamma > 1$, 当 $t-k \rightarrow \infty$ 时, $v^{t-k} \rightarrow \infty$, 会造成系统不稳定(梯度爆炸问题)。若 $\gamma < 1$, 当 $t-k \rightarrow \infty$ 时, $v^{t-k} \rightarrow 0$, 会出现深度和浅度前馈神经网络类似的梯度消失问题(常出现: 一般情况下使用的非线性激活函数为 logistic 函数或 tanh 函数, 其导数值都小于 1, 而权重矩阵 $\|U\|$ 也不会太大)。定义: $\|U\| \leq \gamma_k \leq 1, \|\text{diag}[f'(h_{t-1})]\| \leq \gamma_f \leq 1$, 则 $\left\| \frac{\partial a_t^{(k)}}{\partial U} \right\| = \Pi_{i=k+1}^K \frac{\partial a_t^{(k)}}{\partial h_{t-1}} = \Pi_{i=k+1}^K U^T \text{diag}[f'(h_{t-1})]$, 得 $\left\| \frac{\partial a_t^{(k)}}{\partial U} \right\| \leq \left\| U^T \right\| \|\text{diag}[f'(h_{t-1})]\| \leq \gamma_k \gamma_f \leq 1$ 。经过 $t-k$ 次传播之后, $\left\| \frac{\partial a_t^{(k)}}{\partial U} \right\| \leq (\gamma_k \gamma_f)^{t-k}$, 若时间间隔 $t-k$ 过大, $\left\| \frac{\partial a_t^{(k)}}{\partial U} \right\|$ 会趋于 0。**长期依赖问题**: 虽然简单循环神经网络从理论上可以建立长时间间隔的状态之间的依赖关系, 但是由于梯度爆炸或消失问题, 实际上只能学习到短期周期的依赖关系。**改进方案**: 为了避免梯度爆炸或消失问题, 关键是使得: $U^T \text{diag}[f'(h_{t-1})] = 1$ 。**方法1**: 选取合适的参数, 同时使用非饱和的激活函数。**局限**: 需要很多人工经验, 同时限制了模型的广泛应用。**方法2**: 改变模型, 比如让 $U=1$, 同时使用 $f'(h_{t-1}) = 1$, $h_t = h_{t-1} + Wg(x_t)$, g 是非线性激活函数。**局限**: 丢失了神经元在反馈边上的非线性激活的性质。**方法3**: 引入一个新的状态 c_t 专门来进行信息的反馈传递, 同时把 c_t 的信息非线性地传递给 h_t : $c_t = c_{t-1} + Wg(x_t)$, $h_t = \tanh(c_t)$ 。**局限**: 因为 c_t 和 c_{t-1} 是线性关系, 同时不断累积 c_t 的信息, 会使 c_t 变得越来越大。**引入门机制**: 控制信息的累积速度, 只有选择遗忘之前累积的信息。**长短期记忆神经网络(LSTM)**: 循环神经网络的一个变体, 可以有效地解决简单循环神经网络的梯度爆炸或消失问题。模型的关键: 引入了一组记忆单元, 允许网络可以学习何时遗忘历史信息, 何时用新信息更新记忆单元。**LSTM与RNN比较**: 1.所有 RNN 都具有一种重复神经网络模块的链式的形式。在标准的 RNN 中, 这个重复的模块只有一个非常简单的结构, 例如一个 \tanh 层: $h_t = f(Uh_{t-1} + Wx_t + b)$ 。2.LSTM 同样具有一种重复神经网络模块的链式的形式, 但是重复的模块拥有与标准 RNN 不同的结构。不同于单一神经网络层, LSTM 模型引入了一组记忆单元, 允许网络学习何时遗忘历史信息, 何时用新信息更新记忆单元, 而信息的流动是通过门单元控制的。在时刻 t 时, 记忆单元状态 C_t 记录到当前时刻为止的所有历史信息, 并受三个“门”控制: 输入门 i_t , 遗忘门 f_t 和输出门 o_t , 三个门的元素的值在 [0,1] 之间。**LSTM的关键**是引入记忆单元的状态 C_t , 记忆单元的状态 C_t , 表示成 h_t 在图上贯穿运行。由于记忆单元的状态 C_t 直接在整个网络链上运行, 只有一些少量的线性交互。通过状态单元 C_t 的信息传递很稳定。在 LSTM 中, 信息的交互是通过门结构实现的。LSTM 用门结构来去除或者增加信息到记忆单元的状态。每个门结构包含一个 sigmoid 神经网络层和一个 point-wise 乘法操作。Sigmoid 层输出 0 到 1 之间的数值, 描述每个部分有多少量可以通过。0 代表“不许任何通过”, 1 就指“允许任意通过”。LSTM 拥有三个门, 来控制记忆单元的状态。1.遗忘门: 决定从单元状态 C_t 通过 \tanh 进行简化信息。丢弃的信息是由遗忘门根据上一时刻的隐状态 h_{t-1} 和当前输入 x_t 完成。遗忘门会读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值 f_t 给状态向量 C_{t-1} , 1 表示“完全保留”, 0 表示“完全舍弃”。 $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ 。2.输入门: 确定被存入到记忆单元的状态中的新信息。①.sigmoid 层称“输入门层”, 决定要更新新的值。输入门读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值给候选值向量 i_t : $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ 。②.tanh 层创建一个新的候选值向量 i_t , 用以加入到记忆单元的状态向量中。 \tilde{C}_t 根据 h_{t-1} 和 x_t 生成: $\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ 。③.更新旧的记忆单元状态, 将 C_{t-1} 更新为 C_t 。1).把旧状态与遗忘门输出 f_t 相乘, 丢弃掉需要丢弃的信息; 2).输入门的输出与候选值向量相乘决定更新每个状态向量的程度; 3).用遗忘门的旧状态加上新的候选值, 得到新的记忆单元状态向量 $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$ 。3.输出门: 基于记忆单元的状态确定输出值。①.输出门基于 h_{t-1} 和 x_t 运行确定记忆单元状态的哪个部分将输出出去, 产生输出 o_t : $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$ 。②.把记忆单元状态 C_t 通过 \tanh 进行简化信息。丢弃的信息是由遗忘门根据上一时刻的隐状态 h_{t-1} 和当前输入 x_t 完成。遗忘门会读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值 f_t 给状态向量 C_{t-1} , 1 表示“完全保留”, 0 表示“完全舍弃”。 $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ 。③.输入门: 确定被存入到记忆单元的状态中的新信息。④.sigmoid 层称“输入门层”, 决定要更新新的值。输入门读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值给候选值向量 i_t : $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ 。⑤.tanh 层创建一个新的候选值向量 i_t , 用以加入到记忆单元的状态向量中。 \tilde{C}_t 根据 h_{t-1} 和 x_t 生成: $\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ 。⑥.更新旧的记忆单元状态, 将 C_{t-1} 更新为 C_t 。1).把旧状态与遗忘门输出 f_t 相乘, 丢弃掉需要丢弃的信息; 2).输入门的输出与候选值向量相乘决定更新每个状态向量的程度; 3).用遗忘门的旧状态加上新的候选值, 得到新的记忆单元状态向量 $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$ 。3.输出门: 基于记忆单元的状态确定输出值。①.输出门基于 h_{t-1} 和 x_t 运行确定记忆单元状态的哪个部分将输出出去, 产生输出 o_t : $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$ 。②.把记忆单元状态 C_t 通过 \tanh 进行简化信息。丢弃的信息是由遗忘门根据上一时刻的隐状态 h_{t-1} 和当前输入 x_t 完成。遗忘门会读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值 f_t 给状态向量 C_{t-1} , 1 表示“完全保留”, 0 表示“完全舍弃”。 $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ 。③.输入门: 确定被存入到记忆单元的状态中的新信息。④.sigmoid 层称“输入门层”, 决定要更新新的值。输入门读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值给候选值向量 i_t : $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ 。⑤.tanh 层创建一个新的候选值向量 i_t , 用以加入到记忆单元的状态向量中。 \tilde{C}_t 根据 h_{t-1} 和 x_t 生成: $\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ 。⑥.更新旧的记忆单元状态, 将 C_{t-1} 更新为 C_t 。1).把旧状态与遗忘门输出 f_t 相乘, 丢弃掉需要丢弃的信息; 2).输入门的输出与候选值向量相乘决定更新每个状态向量的程度; 3).用遗忘门的旧状态加上新的候选值, 得到新的记忆单元状态向量 $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$ 。3.输出门: 基于记忆单元的状态确定输出值。①.输出门基于 h_{t-1} 和 x_t 运行确定记忆单元状态的哪个部分将输出出去, 产生输出 o_t : $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$ 。②.把记忆单元状态 C_t 通过 \tanh 进行简化信息。丢弃的信息是由遗忘门根据上一时刻的隐状态 h_{t-1} 和当前输入 x_t 完成。遗忘门会读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值 f_t 给状态向量 C_{t-1} , 1 表示“完全保留”, 0 表示“完全舍弃”。 $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ 。③.输入门: 确定被存入到记忆单元的状态中的新信息。④.sigmoid 层称“输入门层”, 决定要更新新的值。输入门读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值给候选值向量 i_t : $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ 。⑤.tanh 层创建一个新的候选值向量 i_t , 用以加入到记忆单元的状态向量中。 \tilde{C}_t 根据 h_{t-1} 和 x_t 生成: $\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ 。⑥.更新旧的记忆单元状态, 将 C_{t-1} 更新为 C_t 。1).把旧状态与遗忘门输出 f_t 相乘, 丢弃掉需要丢弃的信息; 2).输入门的输出与候选值向量相乘决定更新每个状态向量的程度; 3).用遗忘门的旧状态加上新的候选值, 得到新的记忆单元状态向量 $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$ 。3.输出门: 基于记忆单元的状态确定输出值。①.输出门基于 h_{t-1} 和 x_t 运行确定记忆单元状态的哪个部分将输出出去, 产生输出 o_t : $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$ 。②.把记忆单元状态 C_t 通过 \tanh 进行简化信息。丢弃的信息是由遗忘门根据上一时刻的隐状态 h_{t-1} 和当前输入 x_t 完成。遗忘门会读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值 f_t 给状态向量 C_{t-1} , 1 表示“完全保留”, 0 表示“完全舍弃”。 $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ 。③.输入门: 确定被存入到记忆单元的状态中的新信息。④.sigmoid 层称“输入门层”, 决定要更新新的值。输入门读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值给候选值向量 i_t : $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ 。⑤.tanh 层创建一个新的候选值向量 i_t , 用以加入到记忆单元的状态向量中。 \tilde{C}_t 根据 h_{t-1} 和 x_t 生成: $\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ 。⑥.更新旧的记忆单元状态, 将 C_{t-1} 更新为 C_t 。1).把旧状态与遗忘门输出 f_t 相乘, 丢弃掉需要丢弃的信息; 2).输入门的输出与候选值向量相乘决定更新每个状态向量的程度; 3).用遗忘门的旧状态加上新的候选值, 得到新的记忆单元状态向量 $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$ 。3.输出门: 基于记忆单元的状态确定输出值。①.输出门基于 h_{t-1} 和 x_t 运行确定记忆单元状态的哪个部分将输出出去, 产生输出 o_t : $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$ 。②.把记忆单元状态 C_t 通过 \tanh 进行简化信息。丢弃的信息是由遗忘门根据上一时刻的隐状态 h_{t-1} 和当前输入 x_t 完成。遗忘门会读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值 f_t 给状态向量 C_{t-1} , 1 表示“完全保留”, 0 表示“完全舍弃”。 $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ 。③.输入门: 确定被存入到记忆单元的状态中的新信息。④.sigmoid 层称“输入门层”, 决定要更新新的值。输入门读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值给候选值向量 i_t : $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ 。⑤.tanh 层创建一个新的候选值向量 i_t , 用以加入到记忆单元的状态向量中。 \tilde{C}_t 根据 h_{t-1} 和 x_t 生成: $\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ 。⑥.更新旧的记忆单元状态, 将 C_{t-1} 更新为 C_t 。1).把旧状态与遗忘门输出 f_t 相乘, 丢弃掉需要丢弃的信息; 2).输入门的输出与候选值向量相乘决定更新每个状态向量的程度; 3).用遗忘门的旧状态加上新的候选值, 得到新的记忆单元状态向量 $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$ 。3.输出门: 基于记忆单元的状态确定输出值。①.输出门基于 h_{t-1} 和 x_t 运行确定记忆单元状态的哪个部分将输出出去, 产生输出 o_t : $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$ 。②.把记忆单元状态 C_t 通过 \tanh 进行简化信息。丢弃的信息是由遗忘门根据上一时刻的隐状态 h_{t-1} 和当前输入 x_t 完成。遗忘门会读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值 f_t 给状态向量 C_{t-1} , 1 表示“完全保留”, 0 表示“完全舍弃”。 $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ 。③.输入门: 确定被存入到记忆单元的状态中的新信息。④.sigmoid 层称“输入门层”, 决定要更新新的值。输入门读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值给候选值向量 i_t : $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ 。⑤.tanh 层创建一个新的候选值向量 i_t , 用以加入到记忆单元的状态向量中。 \tilde{C}_t 根据 h_{t-1} 和 x_t 生成: $\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ 。⑥.更新旧的记忆单元状态, 将 C_{t-1} 更新为 C_t 。1).把旧状态与遗忘门输出 f_t 相乘, 丢弃掉需要丢弃的信息; 2).输入门的输出与候选值向量相乘决定更新每个状态向量的程度; 3).用遗忘门的旧状态加上新的候选值, 得到新的记忆单元状态向量 $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$ 。3.输出门: 基于记忆单元的状态确定输出值。①.输出门基于 h_{t-1} 和 x_t 运行确定记忆单元状态的哪个部分将输出出去, 产生输出 o_t : $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$ 。②.把记忆单元状态 C_t 通过 \tanh 进行简化信息。丢弃的信息是由遗忘门根据上一时刻的隐状态 h_{t-1} 和当前输入 x_t 完成。遗忘门会读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值 f_t 给状态向量 C_{t-1} , 1 表示“完全保留”, 0 表示“完全舍弃”。 $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ 。③.输入门: 确定被存入到记忆单元的状态中的新信息。④.sigmoid 层称“输入门层”, 决定要更新新的值。输入门读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值给候选值向量 i_t : $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ 。⑤.tanh 层创建一个新的候选值向量 i_t , 用以加入到记忆单元的状态向量中。 \tilde{C}_t 根据 h_{t-1} 和 x_t 生成: $\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ 。⑥.更新旧的记忆单元状态, 将 C_{t-1} 更新为 C_t 。1).把旧状态与遗忘门输出 f_t 相乘, 丢弃掉需要丢弃的信息; 2).输入门的输出与候选值向量相乘决定更新每个状态向量的程度; 3).用遗忘门的旧状态加上新的候选值, 得到新的记忆单元状态向量 $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$ 。3.输出门: 基于记忆单元的状态确定输出值。①.输出门基于 h_{t-1} 和 x_t 运行确定记忆单元状态的哪个部分将输出出去, 产生输出 o_t : $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$ 。②.把记忆单元状态 C_t 通过 \tanh 进行简化信息。丢弃的信息是由遗忘门根据上一时刻的隐状态 h_{t-1} 和当前输入 x_t 完成。遗忘门会读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值 f_t 给状态向量 C_{t-1} , 1 表示“完全保留”, 0 表示“完全舍弃”。 $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ 。③.输入门: 确定被存入到记忆单元的状态中的新信息。④.sigmoid 层称“输入门层”, 决定要更新新的值。输入门读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值给候选值向量 i_t : $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ 。⑤.tanh 层创建一个新的候选值向量 i_t , 用以加入到记忆单元的状态向量中。 \tilde{C}_t 根据 h_{t-1} 和 x_t 生成: $\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ 。⑥.更新旧的记忆单元状态, 将 C_{t-1} 更新为 C_t 。1).把旧状态与遗忘门输出 f_t 相乘, 丢弃掉需要丢弃的信息; 2).输入门的输出与候选值向量相乘决定更新每个状态向量的程度; 3).用遗忘门的旧状态加上新的候选值, 得到新的记忆单元状态向量 $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$ 。3.输出门: 基于记忆单元的状态确定输出值。①.输出门基于 h_{t-1} 和 x_t 运行确定记忆单元状态的哪个部分将输出出去, 产生输出 o_t : $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$ 。②.把记忆单元状态 C_t 通过 \tanh 进行简化信息。丢弃的信息是由遗忘门根据上一时刻的隐状态 h_{t-1} 和当前输入 x_t 完成。遗忘门会读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值 f_t 给状态向量 C_{t-1} , 1 表示“完全保留”, 0 表示“完全舍弃”。 $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ 。③.输入门: 确定被存入到记忆单元的状态中的新信息。④.sigmoid 层称“输入门层”, 决定要更新新的值。输入门读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值给候选值向量 i_t : $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ 。⑤.tanh 层创建一个新的候选值向量 i_t , 用以加入到记忆单元的状态向量中。 \tilde{C}_t 根据 h_{t-1} 和 x_t 生成: $\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ 。⑥.更新旧的记忆单元状态, 将 C_{t-1} 更新为 C_t 。1).把旧状态与遗忘门输出 f_t 相乘, 丢弃掉需要丢弃的信息; 2).输入门的输出与候选值向量相乘决定更新每个状态向量的程度; 3).用遗忘门的旧状态加上新的候选值, 得到新的记忆单元状态向量 $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$ 。3.输出门: 基于记忆单元的状态确定输出值。①.输出门基于 h_{t-1} 和 x_t 运行确定记忆单元状态的哪个部分将输出出去, 产生输出 o_t : $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$ 。②.把记忆单元状态 C_t 通过 \tanh 进行简化信息。丢弃的信息是由遗忘门根据上一时刻的隐状态 h_{t-1} 和当前输入 x_t 完成。遗忘门会读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值 f_t 给状态向量 C_{t-1} , 1 表示“完全保留”, 0 表示“完全舍弃”。 $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ 。③.输入门: 确定被存入到记忆单元的状态中的新信息。④.sigmoid 层称“输入门层”, 决定要更新新的值。输入门读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值给候选值向量 i_t : $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ 。⑤.tanh 层创建一个新的候选值向量 i_t , 用以加入到记忆单元的状态向量中。 \tilde{C}_t 根据 h_{t-1} 和 x_t 生成: $\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ 。⑥.更新旧的记忆单元状态, 将 C_{t-1} 更新为 C_t 。1).把旧状态与遗忘门输出 f_t 相乘, 丢弃掉需要丢弃的信息; 2).输入门的输出与候选值向量相乘决定更新每个状态向量的程度; 3).用遗忘门的旧状态加上新的候选值, 得到新的记忆单元状态向量 $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$ 。3.输出门: 基于记忆单元的状态确定输出值。①.输出门基于 h_{t-1} 和 x_t 运行确定记忆单元状态的哪个部分将输出出去, 产生输出 o_t : $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$ 。②.把记忆单元状态 C_t 通过 \tanh 进行简化信息。丢弃的信息是由遗忘门根据上一时刻的隐状态 h_{t-1} 和当前输入 x_t 完成。遗忘门会读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值 f_t 给状态向量 C_{t-1} , 1 表示“完全保留”, 0 表示“完全舍弃”。 $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$ 。③.输入门: 确定被存入到记忆单元的状态中的新信息。④.sigmoid 层称“输入门层”, 决定要更新新的值。输入门读取 h_{t-1} 和 x_t , 输出一个在 0 到 1 之间的数值给候选值向量 i_t : $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$ 。⑤.tanh 层创建一个新的候选值向量 i_t , 用以加入到记忆单元的状态向量中。 \tilde{C}_t 根据 h_{t-1} 和 x_t 生成: $\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$ 。⑥.更新旧的记忆单元状态, 将 C_{t-1} 更新为 C_t 。1).把旧状态与遗忘门输出