

1.7 绪论人工智能就是让机器具有人类的智能,即让机器的行为看起来就像是人所表现出的智能行为一样。**图灵测试**“一个人在不接触对方的情况下,通过一种特殊的方式,和对方进行一系列的问答。如果在相当长时间内,他无法根据这些问题判断对方是人还是计算机,那么就可以认为这个计算机是智能的”。人工智能这个学科的诞生有着明确的标志性的事件,就是1956年的达特茅斯会议。在这次会议上,“人工智能”被提出并作为本研究领域的名称。人工智能使一部机器像人一样进行感知、认知、决策、执行的人工智能程序或系统,具体应用机器感知(计算机视觉、语音信息处理)、学习(模式识别、机器学习、强化学习)、语言(自然语言处理)、记忆(知识表示)、决策(规划、数据挖掘)。**知识工程**基于手工设计规则建立专家系统(80年代末期);结果容易解释;系统构建费时费力;依赖专家主观经验,难以保证一致性和准确性。**机器学习**基于数据自动学习(90年代中期);减少人工繁杂工作,但结果可能不易解释;提高信息处理的效率,且准确率较高;来源于真实数据,减少人工规则主观性,可信度高。**机器学习解决一些模式识别任务流程包含以下几个步骤**。1.浅层学习不涉及特征学习,其特征主要靠人工经验或特征转换方法来抽取;实际应用中心,特征往往比分类器更重要。2.预处理经过数据预处理,如去除噪声等;如在文本分类中,去除停用词等。3.特征提取从原始数据中提取有效特征,如在图像分类中,提取边缘、尺度不变特征变换特征等。4.特征转换对特征进行一定的加工,比如降维和升维,降维包括:特征抽取→PCA、LDA;特征选择→互信息、TF-IDF



表示学习(Representation Learning)是机器学习的一个子问题。特征工程：需要借助人类智能。表示学习：如何自动从数据中学习好的表示。**难点**：没有明确的目标。“好的表示”是一个非常主观的概念，没有一个明确的标准。一般而言，好的表示应该具有以下几个**优点**：1.具有很强的表示能力，使后续的学习任务变得简单；2.具有一般性，是任务或领域独立性的。**深度学习**通过构建具有一定“深度”的模型，可以使模型来自动学习的特征表示(从底层特征，到中层特征，再到高层特征)，从而最终提升预测或识别的准确性。**深度学习**通过学习构建多隐层的模型和海量训练数据(可为无标签数据)，来学习更有用的特征，从而最终提升分类或预测的准确性。可通过学习一种深层非线性网络结构，实现复杂函数逼近，表征输入数据分布式表示。**与浅层学习区别**：1.强调模型结构的深度，通常有更多的隐层节点。2.明确指出特征学习的重要性，通过逐层特征变换，将样本在原空间的特征表示变换到一个新特征空间，从而使分类或预测更加容易。与人工规则构造特征的方法相比，利用大数据来学习特征，更能够刻画数据的丰富内在信息。**深度学习应用**：1.**计算机视觉**：图像分类、目标检测(目标检测不仅需要确定图像中包含什么，还要对图像中的物体进行定位)、语义分割(语义分割为图像中的每个像素都赋予相应的语义类别标签)、实例分割(为图像中的每个像素都赋予相应的语义类别标签同时区分不同实例对象)。2.**语音技术**：语音识别、语音合成、音高识别。3.**自然语言处理**

分类: 1. 语音识别; 2. 语音合成; 3. 语音识别; 4. 自然语言处理; 5. 文本分类; 6. 机器翻译; 7. 自动问答; 8. 信息检索。 **常见网络结构: 1. 前馈神经网络**在网络的前端接受输入, 然后将信息向网络后端逐层传播; 前馈神经网络中, 相邻层的节点通常是全连接的, 即每层中的每个节点与相邻层中的所有节点有连接; 前馈神经网络通过反向传递算法进行训练, 训练由输入和对应于输入的期望输出构成的数据集实现。对于某一输入, 期望输出与实际输出之间的差被作为网络训练的驱动; 理论上, 如果前馈神经网络有足够的隐藏节点, 它将对任意输入到输出之间的映射关系进行建模; 实践上, 由于庞大的前馈神经网络将难以优化。目前前馈神经网络通常和其他的网络结构组合使用, 实现相应的任务。 **2. 自编码器**的基本思想是使用神经网络实现自动信息编码(这里的编码指压缩而非加密); 自编码器的特点有: 相对于输入层和输出层来说非常小的隐藏层, 实现对信息在隐藏层进行压缩; 中间层之前的网络部分被称为编码网络, 中间层之后的部分被称为解码网络, 中间层的输出则被称为输入的表达; 使用反向传递算法对自编码器进行训练, 而训练的驱动则是输入数据与解码输出的差; 自编码器的应用主要有两个方面: 第一是数据去噪, 第二是为进行可视化而降噪。 **3. CNN**主要用于图像识别, 也可用于其他任务如语音识别和自然语言处理; CNNs 主要包括卷积层, 卷积层中的每个节点仅与相邻层中邻近的节点相连接; 卷积神经网络中的特征的尺寸通常会随着网络的加深而逐渐变小。这样既可以减少整体的计算量也可以减轻深层节点的感受野; 除卷积层外, CNNs 还包含池化层, 它可以加速特征图中的细节, 主要包含最大池化和平均池化; CNNs 可以和 FNN 组合应用, 完成分类任务。 **4. GAN** 包含两个网络部分, 一个网络用于数据生成, 另外一个网络用于对生成的数据进行判别; 训练过程中, 判别网络将对收到的真实数据或者生成网络产生的数据作出相应判别, 判别网络在判别过程中产生的误差被用于作为网络训练的驱动; 以上过程在生成数据和判别网络间造成了对抗, 使判别网络区分真实数据和生成数据的能力不断提升, 同时使生成网络生成的数据内容更加难以与真实数据区分; GANs 的应用包括图像合成, 语义图像编辑, 风格变换, 图像超分辨率等。 **5. RNN** 的基本思想是利用序列输入的信息之间的关系; RNNs 对序列输入中的每个元素进行相同的运算处理, 并在处理每个元素时将之前的计算信息作为依赖信息; 另一种理解 RNNs 的方式是认为 RNNs 有记忆, 能够捕捉过去的信息用于当前输入的处理; 理论上, RNNs 可以处理任意长的序列输入, 但实际上, 由于梯度爆炸问题, RNNs 仅能处理有限长的序列输入。 **其他挑战:** 从无标注的数据中学习; 模型大不便于移动设备及嵌入式系统使用; 数据+知识, 深度学习与知识图谱、逻辑推理、符号学习相结合; 从认知性的任务扩展到决策性任务; 多任务学习/迁移学习(泛化)可扩展性和效率; 人工智能的体系结构。 **2. 机器学习基础** **机器学习**主要是研究如何使计算机从给定的数据中学习规律, 即从观测数据(样本)中寻找规律, 并利用学习到的规律(模型)对未知或无法观测的数据进行预测。机器学习 \rightarrow 构建一个映射函数。一般地, 机器学习是给定一些训练样本 (x_i, y_i) , $1 \leq i \leq N$ (其中 x_i 是输入实例, y_i 是需要预测的标), 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。 **损失函数**在所有的样本分布上的期望训练成本(经验数据)上计算得来的, 因此被称之为**经验风险**。通过对参数求解, 使经验风险逐渐逼近理想的最小值, 称为**经验风险最小化原则**, 由此机器学习问题转化成了最优化问题。 **凸优化**与**非凸优化**: 非凸函数无法获得全局最优解, **判断是不是凸优化方法**: 1 目标函数 f 如果不是凸函数, 则不是凸优化问题 2 决策变量 x 中包含离散变量 ($0-1$ 变量或整数变量), 则不是凸优化问题 3 约束条件写成 $g(x) \leq 0$, g 如果不是凸函数, 则不是凸优化问题 **非凸优化转换为凸优化**: 修改目标函数, 使之转化为凸函数; 抛弃一些约束条件, 使新的可行域为凸集并且包含原可行域。 **机器学习 \rightarrow 凸优化**, 经验风险最小化原则**存在问题**: 用来训练的样本往往只是真实数据分布的一小

小的子集或者包含一定的噪声,不能完全反映全部数据的真实分布;经验风险最小的模型不代表能够在真实数据的分布上表现最优;经验风险最小化可能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,即过拟合。期望风险 \neq 经验风险。**泛化错误**表现为一个模型在训练集和测试集上错误率的差距;是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;泛拟合对应着高泛化错误,可以通过避免学习算法的过拟合来减小泛化错误。**最小化期望泛化错误等价于最小化偏差和方差之和**。**过拟合**表现为高方差如果习得的假设能非常好地拟合训练集,但却不能泛化到新的样本(为新鲜样本进行正确的预测),将训练样本本身的特特点当做所有样本的一般性质,导致泛化性能下降。**欠拟合**表现为**高偏差**如果限制模型复杂度,降低拟合能力,可能会欠拟合,即模型尚未学好训练样本的一般性质。**什么情况下会出现过拟合?**1.模型过于复杂(对应例子中多项式项数过多)2.训练样本过少。**解决过拟合的问题?**1.限制模型复杂度2.增加样本数量(对于特定任务往往样本数量有限,可通过对样本增加扰动进行扩充)。**正则化(所有损害优化的方法都是正则化)**通过限制模型复杂度,从而提高模型泛化性能。**1.增加约束约束L1/L2约束**2.**干预优化过程**权重衰减、随机梯度下降、早停法。**结构风险最小化(SRM)准则**为解决过拟合问题,在经验风险最小化的基础上引入参数的正则化来限制模型能力,使其不要过度最小化经验风险。结构风险最小化的基本思路是如果模型的参数被限制为较小的值的话(参数值比较小),那么会得到一个在假设空间中形式更简单的假设。在目标函数中引入模型参数的L2范数作为惩罚项,这样在优化目标函数的过程中,参数值会被减小。正则化项也可以使用其它函数,比如L1范数。L1范数的引入通常会使得参数不一定在测试集上最优,将其作为特征选择的手段。**Early-Stop**训练的过程中,由于拟合的原因,在训练样本上收敛的函数,并不一定在测试集上最优。**解决方法**使用一个验证集来测试每一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。如果没有验证集,可以在训练集上进行交叉验证。**损失函数**是衡量决策函数(模型)好坏的参数,给定一个机器学习模型,对于输入实例 x ,模型的预测为 $f(x, \theta)$ 。而实例的真实标记为 y ,如果预测 $f(x, \theta) \neq y$,需要定义一个度量函数来定量地计算预测结果错误的程度。**1.0-1 损失函数**当前预测错误时,损失函数值为1,预测正确时,损失函数值为0。0-1损失函数不考虑预测值和真实值的误差程度,即预测错误差一点和差很多损失是一样的;0-1损失是非凸的函数,在求解的过程中,存在很多不稳定,通常在实际的使用中将0-1损失函数作为一个标准,选择0-1损失函数的代理函数作为损失函数**2.平方损失函数**:直接测量机器学习模型的输出与实际结果之间的距离的平方。通常作为回归问题中的损失函数使用**3.Hinge损失函数**:对于两类分类问题,假设 y 和 $f(x, \theta)$ 的取值为 $\{-1, +1\}$,如果预测结果 $f(x, \theta)$ 与标记 y 一致,则损失函数为零,否则损失函数为2,在机器学习中心,Hinge可以用来解间距最大化的问题,最有代表性的就是SVM问题。**4.交叉熵损失函数**: $L(y, f(x, \theta)) = -\sum_{i=1}^n y_i \log f_i(x, \theta)$ 对于多类分类问题,预测目标 $y \in \{1, \dots, C\}$ 为函数的类别,模型输出 $f(x, \theta) \in \mathbb{R}^C$ 为每个类的条件概率向量,令 $f(x, \theta) = [P(y=1|x), \dots, P(y=C|x)]$,模型预测的第 i 个类的条件概率 $P(y=i|x) = f_i(x, \theta)$, $f_y(x, \theta)$ 可以看作真实实例 y 的似然函数,多类分类机器学习模型的参数可以直接用最大似然估计来优化,考虑到计算方面的问题,通常使用负对数似然作为损失函数,即交叉...**数据**指计算机程序能处理的对象的不同,可以是数字、字母和符号等,在不同的任务中,表现形式的多样,比如图像、声音、文字、传感器数据等。**样本**按照一定的抽样规则从全部数据中取出的部分数据,是实际观测得到的数据。**训练集、验证集和测试集**样本集合就称为数据集,为对机器学习算法进行调优和检验,可将数据集分成:训练集、验证集和测试集,训练集用于模型学习,验证集用于调参,测试集用于模型测试。**机器学习算法步骤**1.定义实现输入到输出的映射的模型(模型)2.定义作为优化目标的目标函数(损失函数)3.基于训练数据集学习能够优化目标函数的模型参数(风险函数)**4.线性回归**通过样本特征的线性组合来进行连续值的预测。**重要性**很多真实的问题能够用线性模型近似;线性回归经常被作为更大系统的模块使用;线性回归问题能够用解析的方式解决;线性预测为学习深度学习中的很多核心概念提供了引导**常见的机器学习问题**回归、分类、聚类。**回归问题**一个问题的输入是变量 x ,目标 y 是连续值(实数或连续整数),预测函数 $f(x)$ 的输出也是连续值。如果预测函数 $f(x)$ 是通过样本特征的线性组合来进行连续值的预测的,则为**线性回归问题**; f 是线性函数 $(x; w, b) = w^T x + b$ **训练网络的顺序**读取数据-数据送入网络-得到网络输出-输出与标签计算损失-最小化损失-更新梯度**求期望** $\langle \theta \rangle = \sum_{i=1}^n (y_i - \hat{y}_i)^2 = \sum_{i=1}^n (y_i - y_i \theta_1 - x_i \theta_2)^2$, $J(\theta) = (y - X\theta)^T (y - X\theta)$, 利用公式 $\partial a/b \partial \theta = a^T \cdot \partial a^T \partial \theta / \partial \theta = 2A^T \theta$, $\partial \langle y - \hat{y} \rangle / \partial \theta = \partial (y^T - 2y^T \theta + \theta^T X^T \theta) / \partial \theta$, 等价于 $\theta = (X^T X)^{-1} X^T y$ 。**5.逻辑回归**线性分类器中,判别函数的输入向量与输出向量呈线性关系。一般的,如果一个线性函数能够将样本完全正确的分开,就称这些数据是**线性可分**的,否则称为**非线性可分**。分类问题中,由于输出目标 y 是一些离散的标签,而判别函数输出 f 值域为实数,因此无法直接进行预测。线性判别函数 + 非线性决策函数。**Logistic回归**1. f 是线性函数, $f(x; w, b) = w^T x + b$ 2. 引入非线性函数 g 来计算类别标签的条件概率 $p(y=c | x)$ 函数 g 应具备什么特性? 能够把线性函数的值域从实数区间“挤压”到了 $(0, 1)$ 之间,以表示概率。**Logistic函数** $\sigma(x) = \frac{1}{1 + \exp(-x)}$ **Logistic回归** 原始线性分类函数的形式: $\hat{y} = \begin{cases} 1 & \text{if } w^T x > 0 \\ 0 & \text{if } w^T x \leq 0 \end{cases}$ **模型**使用Logistic函数将属于类别 $y=1$ 的得分转换为后验概率 $p(y=1|x) = \sigma(w^T x) = \frac{1}{1 + \exp(-w^T x)}$, x 和 w 为增广的输入向量和权重向量。**学习准则(交叉熵损失)** $R(w) = -\frac{1}{n} \sum_{n=1}^n (y^{(n)} \log \hat{y}^{(n)} + (1 - y^{(n)}) \log(1 - \hat{y}^{(n)}))$ **优化算法(梯度下降)** $\frac{\partial R(w)}{\partial w} = -\frac{1}{n} \sum_{n=1}^n x^{(n)} (y^{(n)} - \hat{y}^{(n)})$ **解决多类问题的基本思路**1. 把多类问题分成多个二类问题,①通过两个二类分类器来对多类分类-通过构建C-1对多类的分类器,把多类分类问题转换为C个二类分类问题。其中,每个二类分类问题都是把某一类和其他类用一个超平面分开;②通过构建C(C-1)/2个两两分类器,把多类分类问题转换为C(C-1)/2个二类分类问题。其中,每个二类分类问题都是把C类中某两类用一个超平面分开。决策规则:当对一个样本进行分类时,首先,用每个二类分类器对其进行分类,相应地得到其所属的类别,所有C(C-1)/2个分类器得到相应数量的分类结果。然后,进行投票,选择一个得分最高的类别,作为预测结果。**2. 直接设计多类分类器 SoftMax回归多类分类器**作为Logistic回归在多类分类问题上的推广,softmax回归用于解决多类分类问题(不同于logistic回归解决的二类问题),因此分类 y 可以取C个不同的值(而不是二类分类问题中的2个),Softmax回归也称为多项或多类的Logistic回归。**交叉熵**是按照概率分布 q 对真实分布 p 的信息进行编码的长度,交叉熵主要用于度量两个概率分布 p, q 间的差异性。**KL散度**度量两个分布

之间的差异性;是用概率分布 q 来近似 p 时所造成的信息损失量。对于输入样本 x , **Softmax 回归模型**能够输出一个 C 维向量(向量元素的和为 1), 以表示输入样本 x 属于这 C 个类的概率分布 **SoftMax 回归**

模型 $P(y = c | x) = \text{softmax}(w_c^T x) = \frac{\exp(w_c^T x)}{\sum_{i=1}^C \exp(w_i^T x)}$ **学习准则(交叉熵损失)** $R(W) = -\frac{1}{N} \sum_{n=1}^N (y_n)^T \log \hat{y}_n$ **优化算法(梯度下降)** $\frac{\partial R(W)}{\partial W} = -\frac{1}{N} \sum_{n=1}^N x^{(n)} (y^{(n)} - \hat{y}^{(n)})^T$ **常见的评价指标②** 给定测试集 $T = \{(x_1, y_1), \dots, (x_N, y_N)\}$, 对于所有的 $y_i \in \{1, \dots, C\}$ 。假设分类结果为 $Y = \{\hat{y}_1, \dots, \hat{y}_N$, 下述公式中 $| \cdot |$ 为指示函数

- 正确率 Acc** $= \frac{\sum_{i=1}^N |y_i = \hat{y}_i|}{N}$ = 测试集中被正确识别的样本数/测试集样本总数
- 错误率 Err** $= \frac{\sum_{i=1}^N |y_i \neq \hat{y}_i|}{N}$ = 测试集中被错误识别的样本数/测试集样本总数
- 准确率 (查准率、精确率、精度)** $P_c = \frac{\sum_{i=1}^N |y_i = \hat{y}_i|}{\sum_{i=1}^N |y_i = c|}$ = 1 类 c 中被正确识别为类 c 的样本的个数/被识别为类 c 的样本的个数 (正确识别/错误识别)
- 召回率** $R_c = \frac{\sum_{i=1}^N |y_i = \hat{y}_i|}{\sum_{i=1}^N |y_i = c|}$ = 1 类 c 中被正确识别为类 c 的样本的个数/类 c 中样本的总数。 **正确率是对算法的整体性能的反响, 准确率和召回率对每个类都进行性能估计。**
- F1 值** 是根据正确率和召回率二者给出的一个综合的评价指标 $F1_c = P_c * R_c * 2 / (P_c + R_c)$

线性模型	激活函数	损失函数	优化方法
线性回归	-	$(y - \mathbf{w}^T \mathbf{x})^2$	最小二乘、梯度下降
Logistic 回归	$\sigma(\mathbf{w}^T \mathbf{x})$	$\mathbf{y} \log \sigma(\mathbf{w}^T \mathbf{x})$	梯度下降
Softmax 回归	$\text{softmax}(\mathbf{W}^T \mathbf{x})$	$\mathbf{y} \log \text{softmax}(\mathbf{W}^T \mathbf{x})$	梯度下降

6.人工神经网络
人工神经元 (M-P 神经元模型) 神经元接收来自其他 d 个神经元传递过来的输入信号, 这些输入信号通过带有权重的连接进行传递, 神经元接收到的总输入值将与神经元的阈值 (bias) 进行比较, 然后通过“激活函数”处理产生神经元的输出。**最简单的神经网络** 感知器具有一个神经元、采用阈值激活函数的前向网络。通过对网络权值的训练, 可以使感知器对一组输入矢量的响应达到元素为 0 (-1) 或 1 的目标输出, 从而实现对输入矢量分类的目的。为使模型形式更简洁, 常使用增广的输入和权重向量表示感知器模型。**多类感知器** 为使感知器能处理多类问题以及更复杂的结构化学习任务, 引入一个特征向量, 将输入输出映射到一个向量空间中。Gen(x) 表示输入 x 的所有的输出目标集合。当处理 C 类分类问题时, $Gen(x) = \{1, \dots, C\}$ 。这样, 得到一个更为泛化的感知器: $y = \arg \max_w w^T \varphi(x, y)$ 。训练过程中, 调整 w 的值, 使得 y 值为样本的实际标称值, 乘积最大。测试过程中, 依次验证 C 个特征函数 Gen(x, y), 令乘积最大的 y 作为预测结果。当类别 y 为离散变量时 ($y \in \{1, \dots, C\}$), 通过 one-hot 编码, 可将类别表示为向量形式。**感知器的局限性: 异或问题、双层感知器解决异或问题: 隐藏层包含 2 个神经元, ReLU ($g(z) = \max\{0, z\}$)**
人工神经网络的三要素
①. 节点 (采用什么激活函数?)
②. 连边 (权重 (参数) 是多少?)
③. 连接方式 (如何设计层次结构?)
④. 连续的非线性激活函数 (对神经元的输入加权并进行非线性变换, 增强网络的表达能力)
原因 1. 在神经网络中, 加入一些非线性因素, 能够使得神经网络可以更好地解决较为复杂的问题。2. 连续非线性激活函数是可行的, 所以可以用最优化的方法来求解。**激活函数性质要求** 1. 连续并可导 (允许少数点上不可导) 的非线性函数。可导的激活函数可以直接利用数值优化的方法来学习网络参数。2. 激活函数及其导函数尽可能简单, 提高网络计算效率。3. 激活函数的导函数的值域在一个合适的区间内, 不能太大也不能太小, 否则会影响训练的效率和稳定性。**常用激活函数**
1. Sigmoid 型 (为两端饱和函数) Logistic 函数具有“挤压”功能, 输出可看作概率分布; Tanh 函数: 零中心化, 可提升收敛速度; Hard-Logistic = $\max(\min(0.25x + 0.5, 1), 0)$; Hard-Logistic = $\max(\min(x, 1), -1)$ 这两个函数类似对 Logistic 和 Tanh ($\tanh(x) = 2\sigma(2x) - 1$) 函数的分段近似, 与 Logistic 和 Tanh 函数相比, 降低了计算开销)。**2. 整流线性单元 (ReLU)** 最常用, 具有单侧抑制、宽兴奋边界的生物学合理性, 可缓解梯度消失问题, 稀疏性激活, 缺点: 有可能导致神经元的死亡; Leaky ReLU 在 $x < 0$ 时也保持一个很小的梯度, 避免永远不能被激活的情况, γ 为超参。3. 其他: 指数线性单元 ELU (近似零中心化, γ 为超参), Softplus (可看成 ReLU 的平滑版本, 其导数刚好为 Logistic 函数, 具有单侧抑制、宽兴奋边界的特性, 但不具有稀疏性)
隐藏单元激活函数
输出单元
1. 线性输出单元
2. Sigmoid 单元
3. Softmax 单元
为什么要深度? 1. 单隐层可以近似任何函数, 但是其规模可能巨大。2. 随着深度的增加, 网络的表示能力呈指数增加, 意味着描述能力为深度的指数级。3. 更深层的网络具有更好的泛化能力。**参数数量的增加不一定会带来模型性能的提升, 更深的模型往往表现更好, 不仅仅是因为模型更大。想要学得好的函数应该由许多更简单的函数复合在一起而得到。**
神经网络的结构设计考虑
①. 1. 增加网络深度
2. 改变层与层之间的连接方式: 前一层的一个单元仅与后一层的一个子集相连接; 可以极大地减少参数的数量; 仅通过连接方式高度依赖于具体的问题。**3. 增加跳跃连接:** 从第 i 层与第 i+2 层甚至更高层之间建立连接; 使得梯度更容易从输出层流向更接近输入层的层, 利于模型优化
神经网络结构
①. 1. 前向网络: 每个神经元按照接收信息的先后分成不同的组, 每一组可以看一个神经网络; 每一层中的神经元接收来自前一层神经元的输出, 并输出给下一层神经元; 整个网络中信息朝一个方向传递, 没有反向的信息传递, 可以用一个有向无环图表示; 前馈网络包括全连接前馈神经网络和卷积神经网络。**2. 反馈网络:** 神经元不但可以接收其他神经元的信息, 也可以接收自己的历史信息; 神经元具有记忆功能, 在不同的时刻具有不同的状态; 信息传递可以是单向或者双向传递, 可用一个有向循环图或有向图来表示; 反馈网络包括循环神经网络、Hopfield 网络、玻尔兹曼机、受限玻尔兹曼机等

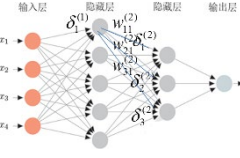
神经网络层运算 (前馈) $z^{(l)} = W^{(l)} \cdot a^{(l-1)} + b^{(l)}, a^{(l)} = f(z^{(l)})$

根据以上递归函数, 前馈神经网络可以通过逐层的信息传递, 得到网络最后的输出: $a^1: x \rightarrow z^{(1)} \rightarrow \dots \rightarrow a^{(l-1)} \rightarrow z^{(l)} \rightarrow a^{(l)} \rightarrow$ **反向传播算法** 通过链式法则可以逐一计算每个参数对偏导, 但是效率低下; 在神经网络的训练中经常使用反向传播算法来高

效地计算梯度 **权重向量偏导数** $\frac{\partial J(W, b, x, y)}{\partial w_{ij}^{(l)}} = \frac{\partial J(W, b, x, y)}{\partial z^{(l)}} \cdot \frac{\partial z^{(l)}}{\partial w_{ij}^{(l)}}$ **误差项**

$\delta^{(l)} = \frac{\partial J(W, b, x, y)}{\partial z^{(l)}}$ 为目标函数关于第 l 层的神经元 $z^{(l)}$ 的偏导数, 表示第 l

层的神经元对最终误差的影响。δ(i)也反映了最终的输出误差对第i层的神经元的敏感程度。 $\frac{\partial W(b,w,x,y)}{\partial w_i^{(j)}} = \delta_i^{(j)} \cdot a_i^{(j-1)}$ **误差的反向传播**第1层的误差项可以通过第1+1层的误差项计算得到。第1层的一个神经元的误差项是所有与该神经元相连的第1+1层的神经元的误差项的权重和,再乘上该神经元激活函数的梯度。图中: $\delta_1^{(1)} = (w_{11}^{(2)}\delta_1^{(2)} + w_{21}^{(2)}\delta_2^{(2)} + w_{31}^{(2)}\delta_3^{(2)}) \cdot f'(z_1^{(1)})$



利用反向传播算法,只要计算出最后一层的误差项,每层参数的误差项: $\delta_i^{(j)} = \frac{\partial W(b,w,x,y)}{\partial w_i^{(j)}} = \frac{\partial a_i^{(j)}}{\partial z_i^{(j)}} \cdot \frac{\partial z_i^{(j)}}{\partial a_i^{(j-1)}}$

$\frac{\partial W(b,w,x,y)}{\partial w_i^{(j)}} = \text{diag}(f_i'(z_i^{(j)})) \cdot ((w^{(j+1)})^T \delta_i^{(j+1)})$,即 $\delta_i^{(j)} \leftarrow f_i'(z_i^{(j)}) \odot ((w^{(j+1)})^T \delta_i^{(j+1)})$,得到每层的误差项后,就可以进一步计算出每层参数的导数 $\frac{\partial W(b,w,x,y)}{\partial w_i^{(j)}} = \delta_i^{(j)} \cdot (a_i^{(j-1)})^T$,

使用每层参数的导数对参数进行更新: $W^{(l)} = W^{(l)} - \alpha \frac{\partial W(b)}{\partial w^{(l)}} = W^{(l)} - \alpha \sum_{i=1}^N \frac{\partial W(b,w,x,y)}{\partial w^{(l)}} \cdot \lambda W$, (λ是惩罚系数, α是学习率)

前馈神经网络的训练过程可以分为以下三步 1.先前馈计算每一层的状态和激活值,直到最后一层 2.反向传播计算每一层的误差 3.计算每一层参数的偏导数,并更新参数**使用计算机自动梯度计算** 1.数值微分 2.符号微分 3.自动微分(图3)

神经网络参数优化的主要问题 (非凸优化问题) 梯度消失问题 (sigmoid函数导数值的域都小于1,误差经过每一层传递都会不断衰减,但网络层数很深时,梯度就不停的衰减,甚至ReLU使得整个网络很难训练 **解决** 1.使用线性整流激活函数(比如ReLU)或近似线性函数比如(softplus),误差可以很好的传播,训练速度得到很大的提高 2.采用残差结构。误差在反向传播过程中不断衰减甚至消失。 **模型优化问题** 1.结构差异大没有通用的优化算法、超参数多 2.非凸优化问题参数初始化、逃离局部最优 3.梯度消失(爆炸)问题 **改善方法** ⑤优化策略 1.更有效的优化算法来提高优化方法的效率和稳定性 动态学习率调整、梯度估计修正 2.更高的参数初始化方法、数据预处理方法来提高优化效率 3.修改网络结构优化地形指在高维空间中损失函数的曲面形状,好的优化地形通常比较平滑,使用ReLU激活函数、残差连接、逐层归一化等 3.使用更好的超参数优化方法 **常见优化方法** 梯度下降法(沿着梯度向量相反的方向,梯度减少最快,更容易找到函数的最小值。)、牛顿法、拟牛顿法和共轭梯度法。 **梯度下降法**:作为一种迭代算法,梯度下降法实现简单,常用于求解无约束优化的极值问题。 **梯度下降法及其变体**在深度学习中被广泛采用以实现对深度神经网络模型的优化。 1.批量梯度下降法:有利于寻找全局最优,梯度方差小,易于并行实现; 缺点:样本数目多时,训练过程很慢 2.随机梯度下降法(SGD):训练速度很快 梯度方差大;不易并行实现 3.小批量梯度下降法:折中方法 mini-batch 随机梯度下降法。迭代时,采用一小部分训练样本,兼顾批量梯度下降和随机梯度下降的优点,即算法训练过程较快,也能保证最终参数训练的准确率。 **批量大小**不影响随机梯度的期望,但是会影响随机梯度的方差,批量越大,随机梯度的方差越小,引入的噪声也会越小,训练也越稳定,因此可以设置较大的学习率;而批量较小时,需要设置较小的学习率,否则模型会不收敛, **学习率设置**过大不会收敛,过小的收敛太慢。在机器学习中,常使用自适应调整学习率的方法,提高学习效率。 **学习率衰减**一开始要保持大些保证收敛速度,收敛到最优点附近时要小些以避免来回震荡(分段衰减、逆时衰减、指数衰减、余弦衰减 **学习率预热** 批量大较小时,需要较大的学习率。但开始参数随机初始化,梯度往往较大,加上较大的初始学习率,训练不稳定;为提高训练稳定性,可在最初几轮迭代时采用较小学习率,等梯度下降到一定程度后再恢复到初始学习率 **周期性学习率调整**为逃离局部最小值或鞍点,训练过程中周期性增大学习率。短期内有损失收敛性,长期来看有助于找到更好的局部最优解。 **循环学习率**让学习率在一个区间内周期性地增大和缩小。 三角循环学习率:使用线性缩放来调整学习率 带热重启的随机梯度下降:周期性地重启并采用余弦衰减 **自适应学习率学习率衰减的局限性** 1.非自适应,不能够根据当前梯度情况做出调整 2.每个参数的维度上收敛速度都不相同,应该根据不同参数的收敛情况分别设置学习率。 **动量法**用之前累积动量来替代真正的梯度。每次迭代的梯度可以看作是加速度。仍然不能保证收敛到全局最优,但有一定可能跳出局部极值点 **AdaGrad** 在训练中自动的对学习率进行调整,对于变化频率较低参数采用较大的α更新; 对于变化频率较高的采用较小的α更新。 **优点**:在参数空间更为平缓的方向,会取得更大的进步(因为平缓,所以历史梯度平方和较小,对应学习下降的幅度较小)。 **缺点**:在训练的中后期,分母上梯度平方的累加将越来越大,从而从梯度趋近于0 **RMSProp** Adagrad 累加之前所有的梯度平方,而RMSProp 仅仅是计算对应的加权平均值,因此可缓解 Adagrad 算法学习率下降较快的问题 **Adam 优化器**结合 AdaGrad 和 RMSProp 两种优化算法的优点。对梯度的二阶矩估计(即梯度的均值)和二阶矩估计(即梯度的未中心化的方差)进行综合考虑,计算出更新步长, **优点**:1.实现简单,计算高效,对内存需求少;2.参数的更新不受梯度的伸缩变换影响;3.超参数具有很好的解释性;且通常无需调整或仅需很少的调整;4.更新步长能够被限制在大致的范围内(初始学习率);5.能自然地实现步长退火过程(自动调整学习率);6.很合适应用于大规模的数据及参数的场景;7.适用于不稳定目标函数;8.适用于梯度稀疏或梯度存在很大噪声的问题。在很多机器学习和深度学习的应用中,Adam 优化器被广泛地使用,在很多情况下被作为默认的优化器。

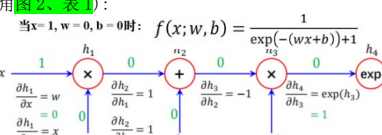
参数初始化与数据预处理 梯度下降法需要在开始训练时给每一个参

类别	优化算法
固定衰减学习率	分段常数衰减、逆时衰减、(自然)指数衰减、余弦衰减
周期性学习率	循环学习率、SGDR
自适应学习率	AdaGrad、RMSprop、AdaDelta
梯度估计修正	动量法、Nesterov 加速梯度、梯度截断
综合方法	Adam≈动量法+RMSprop

数赋一个初始值。初始化为 0(对称权重问题)初始化范围太小(会导致信号慢慢消失,还会使得 Sigmoid 型激活函数丢失非线性能力)初始化范围太大(会导致信号变得过大,还会使得 Sigmoid 型激活函数变得饱和,导致梯度消失) **初始化方法** 1.预训练初始化:Pre-Training→Fine-Tuning;2.固定值初始化;偏置(bias)通常用 0 初始化;3.随机初始化方法:基于固定方差的参数初始化,基于方差缩放的参数初始化,正交初始化方法。 **数据预处理**简单缩放(x-min(max-min)、逐样本均值缩放、特征标准化(0 均值 1 方差)、白化(降低输入的冗余性,降低特征之间的相关性) **超参数优化**神经网络中的超参数:层数、每层神经元个数、激活函数、学习率(各优化算法中包含的参数)、正则化系

数、mini-batch 大小 **优化难点** 1.超参数优化是一个组合优化问题 无法用 GD 来优化 2.评估一组超参数配置的时间代价非常高 **优化方法**: 网格搜索、随机搜索、贝叶斯优化(根据经验)、动态资源分配、神经网络搜索。 **网格搜索**是一种通过尝试超参数的组合来寻址合适一组超参数配置的方法; **贝叶斯优化**是一种自适应的超参数优化方法,根据当前已经试验的超参数组合,来预测下一个可能带来最大收益的组合; **动态资源分配**超参数优化中,每组超参数配置的评估代价比较高。如果可以在较早的阶段就可以估计出一组配置的效果会比较差,那么就可以中止这组配置的评估,将更多的资源留给其它配置。 **⑤遇到的问题 过拟合与正则化、如何提高神经网络的泛化能力** L1 和 L2 正则化、提前停止、dropout(训练时,每次参数更新前,p%的概率丢弃每个神经元;测试时,不使用 dropout,每个参数乘以 1-p%;简化了网络、防止过拟合,可看成一种集成学习,可看作一种贝叶斯学习的近似)、数据增强(改变图像、引入噪声等增加数据多样性及训练数据量) **8.CNN 全连接前馈网络处理图像问题** 1.参数太多:随隐藏层神经元数量增多,参数规模增加,导致神经网络训练效率低,容易过拟合 2.局部不变性特征难以提取:自然图像中物体具有局部不变性,如尺度缩放、平移、旋转等操作不影响语义信息。 FCN 前馈网络难提取局部不变性特征,需要数据增强提高性能 **CNN 与普通神经网络区别** 1.局部感受野:相邻层的神经元间的连接非全连接 2.权重共享:同一层中某些神经元之间近邻的权重共享 3.权值共享:根据距离较近的像素联系较为紧密,距离较远的像素相关性较弱的特性,每个神经元只对局部图像进行感知,在更高层将局部的信息综合起来得到全局信息,而没有必要对每个图像进行感知 **权值共享前提**:数据的统计特性空间不变性,即在数据的一个区域神经元的特征也能用在其他区域 **以图像为例**:对于图像上的所有位置,都能使用相同的特征提取算子提取特征。更直观的解释,当从一个大小 x 图像中随机选取一小块,如 n×n 作为样本,且从这个小块样本中学习到了一些特征,这时可以把从这个 n×n 样本中学习的特征作为探测器,应用到该图像的任意地方去提取特征 **优点**:进一步减少参数个数 **滑动步长** S:卷积核在滑动时的时间间隔 **填充** 在输入向量两端进行补零 **卷积核分类** **按输出长度不同** 1.窄卷积:步长 T=1,两端不补零 P=0,卷积后输出长度为 M-K+1 2.宽卷积:步长 T=1,两端补零 P-K-1,卷积后输出长度 M-K-1 3.等宽卷积:步长 T=1,两端补零 P=(K-1)/2,卷积后输出长度 M **二维卷积**在图像处理中,卷积经常作为特征提取的方法,图像在经过卷积操作后得到结果称为特征映射;高通滤波器可以用来对图像进行平滑操作 **卷积公式** H2=(H1-K+2P)/S+1 **若一种卷积核能提出图像的一种特征,如需要提取不同的特征怎么办?** 采用多种卷积核。 **权重共享**作为参数的卷积核 w(1)对于第1层的所有神经元都是相同的。 **权重共享**可以理解为一个卷积核只捕捉输入数据中的一种特定的局部特征。因此,如果要提取多种特征就需要使用多个不同的卷积核 **卷积神经网络特性** 平移不变性是由卷积+池化共同实现的特性:图像经过平移,相应的特征图上的表达也是平移的池化:比如最大化池化,它返回感受野中的最大值,如果最大值被移动了,但仍然是在这个感受野中,那么池化层也仍然会输出相同的最大值。这两种操作使得即使图像被平移,卷积保证仍然能检测到它的特征,池化则尽可能地保持保持一致的表达。 **卷积层**图像中被卷积核覆盖区域的像素分布模式与卷积核中的数值分布模式越相似,卷积运算结果越大。通过这种方式,符合特定模式的图像特征被卷积核筛选出来。 **池化层**卷积层虽然可以显著减少网络中连接的数量,但特征映射组中的神经元个数并没有显著减少。如果后面接一个分类器,分类器的输入维数极高,很容易出现过拟合。为了解决这个问题,可以在卷积层之后加一个池化层,降低特征维数,避免过拟合。池化层也叫子采样层,其作用是进行特征选择,降低特征数量,从而减少参数数量。池化是指对每个区域进行下采样得到一个值,作为这个区域的概括。 **最大化池化**:取区域内所有神经元的最大活性值作为这个区域的表示,能够抑制网络参数误差造成的估计值偏移的现象。 **平均池化**:取区域内所有神经元活性值的平均值,主要用来抑制相邻值之间的差别过大,造成的方差过大。 **池化的优点** 1.更好的获取平移不变性 2.更高的计算效率(减少神经元) **③CNN 流程及各层特点** **卷积层提取特征、池化层降维防止过拟合、全连接层输出结果卷积核**也叫滤波器,代表某种图像特征。比如垂直边缘、水平边缘、颜色、纹理等等,这些所有神经元加起来就好比就是整张图像的特征提取器集合。卷积核越深越能检测图像更高级别、更高层次、更复杂、更抽象、更泛化的特征。 **例题**输入图片大小 32×32×3,10 个卷积核,大小为 5×5×3,步长为 1, pad=2;输出大小为 32×32×10 (**卷积核通道数=前层特征图数**) 卷积层每个卷积核具有 5×5×3+1=76 个参数(1 for bias)=76×10=760 **全连接层**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。 **卷积神经网络参数学习**训练输入的数据和分类对象是已定的,网络的深度(隐藏层的层数)和卷积核(神经元)的数量、卷积核的大小,都是训练前设定好的一起参。如果训练参数设置不合理会导致过拟合或者欠拟合; 在全连接前馈神经网络中,梯度主要通过每一层的误差项进行反向传播,并进一步计算每层参数的度,和全连接前馈网络类似,卷积网络也可以通过误差反向传播算法来进行参数学习,参数为卷积核中的权重以及偏置。卷积方式在卷积运算中,通过增加操作的步长 S>1 实现对输入特征的下采样,降低特征维数;通过减少转置卷积的步长 S<1 实现上采样,提高特征维数。 **转置卷积(反卷积)**将低维特征映射到高维特征的卷积操作 **微步卷积**步长 S<1 的转置卷积(插零实现) **空洞卷积**给卷积核插入“空洞”来变相地增加其大小(不增加参数数量,同时增加输出单元感受野) **如何增加输出单元的感受野** 1.增加卷积核的大小 2.增加层数,如两层 3×3 的卷积单元可近似一层 5×5 卷积的效果 3.在卷积之前进行池化操作 **Lenet 例题** C3 连接表(右下图 2、表 1):

当 x=1, w=0, b=0 时: $f(x;w,b) = \frac{1}{\exp(-((w+x)b))+1}$



前向模式: $\frac{\partial f(x;w,b)}{\partial w} |_{x=1,w=0,b=0} = \frac{\partial f(x;w,b)}{\partial w} \frac{\partial h_5}{\partial h_4} \frac{\partial h_5}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial h_2} \frac{\partial h_2}{\partial h_1} \frac{\partial h_1}{\partial w} = 1 \times -0.25 \times 1 \times 1 \times 1 \times 1 \times 1 \times 1 = 0.25$

反向模式: $\frac{\partial f(x;w,b)}{\partial b} |_{x=1,w=0,b=0} = \frac{\partial f(x;w,b)}{\partial b} \frac{\partial h_5}{\partial h_4} \frac{\partial h_5}{\partial h_4} \frac{\partial h_4}{\partial h_3} \frac{\partial h_3}{\partial h_2} \frac{\partial h_2}{\partial h_1} \frac{\partial h_1}{\partial b} = 1 \times -0.25 \times 1 \times 1 \times 1 \times 1 \times 1 \times 1 = 0.25$

9.RNN 前馈神经网络的前提假设都是:元素之间独立.输入输出独立。只能单独的去处理独立的输入,前馈神经网络中每层之间的节点是无连接的。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系的,相比于前馈神经网络, **④RNN 特点** **能够更好地处理变长的序列数据;更加符合生物神经网络的结构。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任务上。 RNN 原理**通过带自反馈的神经元,会对历史信息进行记忆并使其参与到当前输出的计算,因此,理论上 RNN 能处理任意长度的序列数据。 **RNN 的信息记忆作用具体表现**为:隐藏层的输入不仅包括当前的输入还包括上一时刻隐藏层的输出。一个序列当前的输出与当前的输入以及过去的输入都有关。 **理论上 RNN 能处理任意长度**的序列数据。 一个简单的循环神经网络由输入层、一个隐藏层和一个输出层组成。

- 去掉有 W 的带箭头的圈,它就变成普通的全连接神经网络
- x 是一个向量,它表示输入层的值
- s 是一个向量,它表示隐藏层的值, s 不仅取决于当前这次的输入 x,还取决于上一次隐藏层的值
- o 也是一个向量,它表示输出层的值
- U 是输入层到隐藏层的权重矩阵
- V 是隐藏层到输出层的权重矩阵
- W 是隐藏层上一次的值作为这一次的输入的权重矩阵

基础的神经网络只在层与层之间建立权连接,RNN 最大的不同之处在于层内部的神经元也建立了权连接。 **Many-to-One** 输入为一串文字,输出为分类类别 **One-to-One** 输入为一张图片,输出为图片的文字描述 **Many-to-Many 模型(同步)** 输入输出相同,输入为视频序列,输出帧对应的标签 **Many-to-Many 模型(异步)** 输入输出个数不同,机器翻译中,源语言和目标语言的句子往往并没有相同的长度。 **随时间反向传播(BPTT)**,沿着需要优化的参数的负梯度方向不断寻找更优的点直至收敛(本质上还是 BP 算法)。 **三个步骤**:1.前向计算每个神经元的输出值 2.反向计算每个神经元的误差项 δ j 值,它是误差函数 E 对神经元 j 的加权输入 net j 的偏导数 3.计算每个权重的梯度;最后再用随机梯度下降算法更新权重。 **最终的梯度是各个时刻的梯度之和。最大问题**是训练时梯度需要随着时间进行反向传播,输入序列较长时,会存在梯度爆炸和消失问题,实际上只能学习到短期内的依赖关系,这就是所谓的长期依赖问题。 **改进**一种方式是选取合适参数,使用非饱和的激活函数。但这样需要很人工经验,限制了广泛应用。另一种方式是改变模型但丢失了神经元在反馈边上的非线性激活的性质。 **解决方案**引入一个新的状态 c t,专门来进行信息的反馈传递,同时把信息非线性地传递 ht。但 c t 变得越来越大,因此引入门机制来控制信息的累积速度,选择遗忘之前累积的信息。这就是 **长短期记忆神经网络(LSTM)** **关键**引入了记忆单元,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。而信息流动的是通过门单元控制的。 **三个门** **输入门**,遗忘门,输出门,控制神经元的流动状态,每个门结构包含一个 sigmoid 神经网络层和一个 point-wise 乘法操作。 **门限循环单元(GRU)**是一种比 LSTM 更加简化的版本。在 LSTM 中,输入门和遗忘门是互补关系,因为同时用两个门比较冗余。GRU 将输入门和与遗忘门合并成一个门:更新门,同时还合并了记忆单元和神经元活性,两个门:重置门和更新门。 **重置门 z**用来控制候选状态中有多少信息是从历史信息中得到; **更新门 u**用来控制当前的状态需要遗忘多少历史信息并接受多少新信息。

④RNN 原理概念, ②分类问题模型评估③神经网络训练方法④神经网络关键结构、结构特点、设计时需要注意的事项; **除文中标出部分外,预计答案还有:结构为输入层、隐藏层、输出层**, (特点:输入层不做运算,每个节点对应一个输入向量的元素,实现数据的透明传递;中间层和输出层是计算层;输出单元:1.线性输出单元 2.Sigmoid 单元 3.Softmax 单元。) **⑤深度神经网络优化策略及遇到问题** **③CNN 网络原理流程** 卷积计算的原理:一维卷积经常用在信号处理中,用于计算信号的延迟累积,即一个信号发生器每个时刻 t 产生一个信号 xt,其信息的衰减率为 wk, 设当 k=1 个时间步长后,信息为原来的 w 倍,假设 w1=1, w2=1/2, w3=1/4, 则一维时延: $y_t = 1 \times x_t + 1/2 \times w_{x-1} + 1/4 \times w_{x-2} = w_1 \times x_t + w_2 \times x_{t-1} + w_3 \times x_{t-2} = \sum_{i=1}^3 w_i x_{t-i}$;二维卷积:给定一个图像 X∈R^M×N 和一个滤波器 W∈R^U×V, 一般 U<M, V<N: $y_{ij} = \sum_{u=1}^U \sum_{v=1}^V w_{uv} x_{i-u,j-v+1}$, 给定一个输入信息 X 和滤波器 W 的二维卷积积位为: **Y = W * X**。

输入图片大小: 32×32×3
10个卷积核, 大小为 5×5×3, 步长为 1, pad=2
输出图片大小: (32-5+2×2)/1+1=32
因此输出大小为 32×32×10

例:
输入图片大小: 32×32×3
10个卷积核, 大小为 5×5×3, 步长为 1, pad=2

这一层中的参数数量?
一般结构给定: 卷积层每个卷积核具有 5×5×3+1=76 个参数(1 for bias)=76×10=760
总结: 给定一个卷积核
• 需要四个参数
✓ 滤波器数目 F
✓ 滤波器大小 K
✓ 步长 S
✓ 零填充个数 P
• 输入图片大小: W1×H1×D1
• 经过卷积后输出大小为 W2×H2×D2
➢ $W_2 = (W_1 - K + 2P) / S + 1$
➢ $H_2 = (H_1 - K + 2P) / S + 1$
➢ $D_2 = F$
• 通过参数共享, 每个滤波器引入 K×K×D1+1 个参数, 因此一共有 (F×(K×K×D1)+F) 个参数
• 输出结果中, 第 4 个切片 (大小为 W2×H2) 是对输入通过第 4 个滤波器以步长 S 做卷积然后加上 bias 的结果

