通过将误差反向传播来学习表示

左元翻译

日期：March 11, 2025

摘　　要

我们描述了一种新的学习过程——反向传播，用于由类神经元单元组成的网络。该过程不断调整网络中各连接的权重，以最小化网络实际输出向量与期望输出向量之间的差值。通过权重的调整，网络中非输入或输出的内部“隐藏”单元逐渐表示任务域的重要特征，任务的规律性则由这些单元的交互所捕获。这种生成有用新特征的能力，使反向传播有别于早期简单的方法（如感知机收敛过程）。

目录

1 正文

人们已经进行了许多设计自组织神经网络的尝试。其目标是找到一种强大的突触修改规则，使得任意连接的神经网络能够发展出适合特定任务域的内部结构。任务的定义是通过为输入单元的每个状态向量指定输出单元的期望状态向量来实现的。如果输入单元直接连接到输出单元，则相对容易找到学习规则，通过迭代调整连接的相对强度，逐步减小实际输出向量与期望输出向量之间的差异。然而，当我们引入隐藏单元时，学习变得更有趣但也更困难，因为这些隐藏单元的实际状态或期望状态并未由任务指定。(在感知机中，输入和输出之间存在 “特征分析器”，它们并非真正的隐藏单元，因为它们的输入连接是手动固定的，因此它们的状态完全由输入向量决定：它们并不学习表示。）学习过程必须决定在什么情况下隐藏单元应该被激活，以帮助实现期望的输入-输出行为。这相当于决定这些单元应该表示什么。我们证明，一种通用且相对简单的过程足以构建适当的内部表示。

该学习过程的最简单形式适用于分层网络，这些网络在底层有一层输入单元，任意数量的中间层，以及在顶层有一层输出单元。禁止在同一层内或从高层到低层的连接，但连接可以跳过中间层。通过设置输入单元的状态，将输入向量呈现给网络。然后，通过将方程（1）和（2）应用于来自较低层的连接，确定每一层中单元的状态。同一层内所有单元的状态是并行设置的，但不同层之间是顺序设置的，从底层开始向上处理，直到确定输出单元的状态。

单元 j 的总输入 x_j 是与 j 连接的单元的输出 y_i 以及这些连接上的权重 w_{ji} 的线性函数。

$$x_j = \sum_i y_i w_{ij} \tag{1}$$

可以通过为每个单元引入一个额外输入（其值恒为 1）来为单元设置偏置。这个额外输入上的权重称为偏置，其作用相当于一个符号相反的阈值。偏置可以像其他权重一样进行处理。

单元的输出 y_j 是一个实数值，且是其总输入的非线性函数。

$$y_j = \frac{1}{1 + e^{-x_j}} \tag{2}$$

不必严格使用方程（1）和（2）中给出的函数。任何具有有界导数的输入-输出函数都可以使用。然而，在应用非线性函数之前，使用线性函数来组合单元的输入可以极大地简化学习过程。

目标是找到一组权重，确保对于每个输入向量，网络生成的输出向量与期望输出向量相同（或足够接近）。如果存在一个固定的、有限的输入-输出案例集，则可以通过比较每个案例的实际输出向量和期望输出向量来计算网络在特定权重下的总误差。总误差 E 定义为：

$$E = \frac{1}{2} \sum_c \sum_j (y_{j,c} - d_{j,c})^2 \tag{3}$$

其中 c 是案例（输入-输出对）的索引，j 是输出单元的索引，y 是输出单元的实际状态，d 是其期望状态。为了通过梯度下降法最小化 E，需要计算 E 对网络中每个权重的偏导数。这只需对每个输入-输出案例的偏导数求和即可。对于给定案例，误差对每个权重的偏导数通过两次传递计算。我们已经在正向传递中描述了每一层单元的状态由它们从较低层单元接收到的输入通过方程（1）和（2）确定。反向传递将导数从顶层传播回底层，过程更为复杂。

反向传递从计算每个输出单元的 $\partial E/\partial y$ 开始。对特定案例 c 的方程（3）进行微分，并省略索引 c，得到：

$$\partial E/\partial y_j = y_j - d_j \tag{4}$$

然后我们使用链式法则来计算 $\partial E/\partial x_j$，

$$\partial E/\partial x_j = \partial E/\partial y_j \cdot y_j(1 - y_j)$$

这意味着我们知道输出单元的总输入 x 的变化将如何影响误差。但这个总输入只是较低层单元状态的线性函数，同时也是连接权重的线性函数，因此很容易计算改变这些状态和权重将如何影响误差。对于从单元 i 到单元 j 的权重 w_{ji}，其导数为：

$$\begin{aligned} \partial E/\partial w_{ji} &= \partial E/\partial x_j \cdot \partial x_j/\partial w_{jk} \\ &= \partial E/\partial x_j \cdot y_i \end{aligned} \tag{5}$$

而对于第 i 个单元的输出，由于 i 对 j 的影响，其对 $\partial E/\partial y_i$ 的贡献为：

$$\partial E/\partial x_j \cdot \partial x_j/\partial y_i = \partial E/\partial x_j \cdot w_{ji}$$

我们已经了解了如何在给定最后一层所有单元的 $\partial E/\partial y$ 的情况下，计算倒数第二层中任意单元的 $\partial E/\partial y$。因此，我们可以重复这一过程，依次为更早的层计算这一项，并在此过程中计算权重的 $\partial E/\partial w$。

使用 $\partial E/\partial w$ 的一种方法是在每个输入-输出案例后更新权重。这种方法的优点是不需要为导数单独分配内存。另一种方案（我们在本研究中使用的）是在更新权重之前，对所有输入-输出案例的 $\partial E/\partial w$ 进行累积。梯度下降法的最简单版本是将每个权重按照累积的 $\partial E/\partial w$ 的比例进行更新。

$$\Delta w = -\epsilon \partial E/\partial w \tag{6}$$

这种方法虽然不如利用二阶导数的方法收敛得快，但它更为简单，并且可以很容易地通过并行硬件中的局部计算实现。通过使用一种加速方法，可以在不牺牲简单性和局部性的前提下显著改进其性能。这种加速方法利用当前梯度来修改权重空间中点的速度，而不是直接修改其位置。

$$\Delta w(t) = -\epsilon \partial E/\partial w(t) + \alpha \Delta w(t-1) \tag{7}$$

其中，t 每遍历一次完整的输入-输出案例集就增加 1，而 α 是一个介于 0 和 1 之间的指数衰减因子，它决定了当前梯度和早期梯度对权重变化的相对贡献。

为了打破对称性，我们从小的随机权重开始。David Parker（个人交流）和 Yann Le Cun 分别独立发现了该学习过程的变体。

一个无法仅通过将输入单元直接连接到输出单元来完成的任务是对称性检测。为了检测一维输入单元阵列的二进制活动水平是否关于中心点对称，必须使用中间层，因为单独考虑单个输入单元的活动无法提供关于整个输入向量对称性或非对称性的证据，因此简单地累加来自单个输入单元的证据是不够的。（关于为什么需要中间单元的更正式证明见参考文献 2。）学习过程发现了一种仅使用两个中间单元的优雅解决方案，如图 1 所示。

另一个有趣的任务是存储两个家族树中的信息（图 2）。图 3 展示了我们使用的网络，图 4 展示了网络在 104 种可能的三元组中的 100 种上训练后，一些隐藏单元的“感受野”。

到目前为止，我们只处理了分层的前馈网络。图 5 展示了分层网络与迭代运行的递归网络之间的等价性。

该学习过程最明显的缺点是误差表面可能包含局部最小值，因此梯度下降法不能保证找到全局最小值。然而，在许多任务中的经验表明，网络很少陷入明显劣于全局最小值的较差的局部最小值中。我们只在那些连接数刚刚足够执行任务的网络中遇到过这种不良行为。增加一些额外的连接会在权重空间中创建额外的维度，这些维度提供了绕过在低维子空间中产生较差局部最小值的障碍的路径。

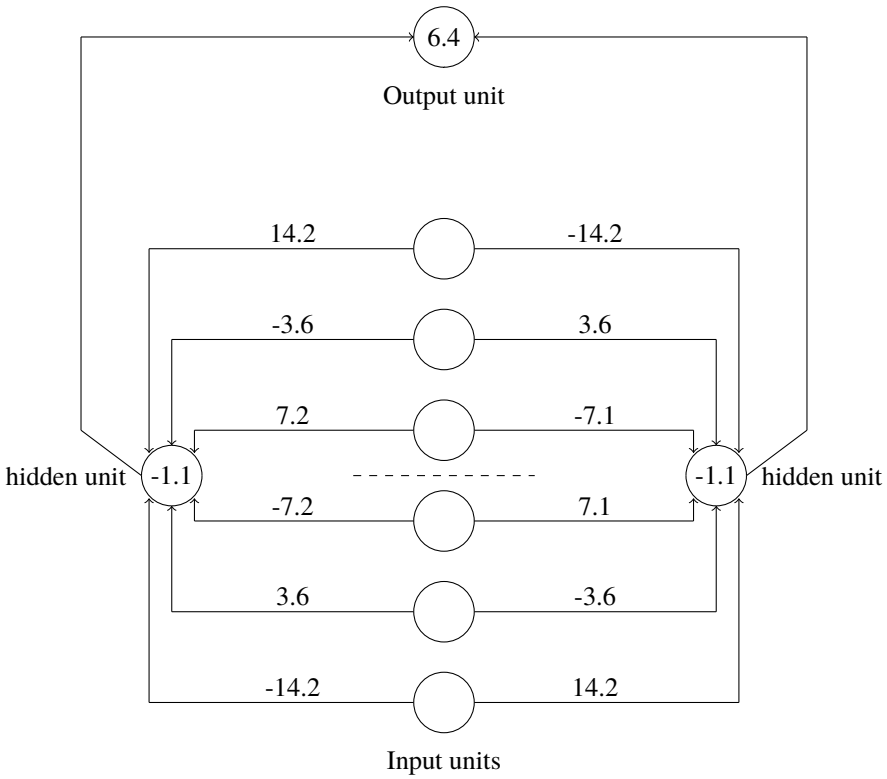

图 1: 一个已经学会检测输入向量镜像对称性的网络，弧上的数字表示权重，节点内的数字表示偏置。学习过程需要对 64 种可能的输入向量集进行 1,425 次遍历，每次遍历后根据累积的梯度调整权重。方程 (9) 中的参数值为 $\epsilon = 0.1$ 和 $\alpha = 0.9$。初始权重是随机的，并均匀分布在 -0.3 到 0.3 之间。该解决方案的关键特性是，对于给定的隐藏单元，关于输入向量中点对称的权重在大小上相等但符号相反。因此，如果呈现一个对称模式，两个隐藏单元将从输入单元接收到净输入为 0，并且由于隐藏单元具有负偏置，两者都将关闭。在这种情况下，具有正偏置的输出单元将处于开启状态。请注意，中点两侧的权重比例为 1:2:4。这确保了中点上方可能出现的八种模式中的每一种都会向每个隐藏单元发送唯一的激活总和，因此只有中点下方的对称模式才能完全平衡该总和。对于所有非对称模式，两个隐藏单元都将从输入单元接收到非零激活。两个隐藏单元具有相同的权重模式但符号相反，因此对于每一个非对称模式，一个隐藏单元将开启并抑制输出单元。

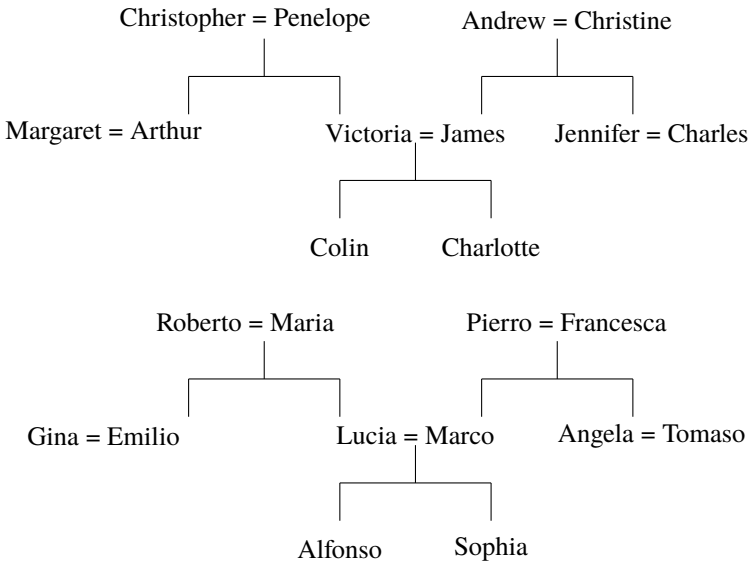

图 2: 两个同构的家谱树。信息可以表示为三元组的集合，形式为 < 人物 1>< 关系 >< 人物 2>，其中可能的关系包括 {父亲、母亲、丈夫、妻子、儿子、女儿、叔叔、阿姨、兄弟、姐妹、侄子、侄女}。一个分层网络可以被认为“知道”这些三元组，如果它能够在给定前两项时生成第三项。前两项通过激活两个输入单元来编码，然后网络必须通过激活代表第三项的输出单元来完成这个命题。

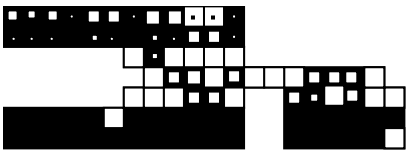

图 3: 一个五层网络在学习后的活动状态。底层左侧有 24 个输入单元用于表示 < 人物 1>，右侧有 12 个输入单元用于表示关系。这两组内的白色方块显示了单元的活动状态。第一组中有一个活动单元代表 Colin，第二组中有一个活动单元代表关系“有阿姨”。每组输入单元都完全连接到第二层中各自的 6 个单元组。这些组学习将人物和关系编码为分布式活动模式。第二层完全连接到中间的 12 个单元层，这些单元又连接到倒数第二层的 6 个单元。倒数第二层的活动必须激活正确的输出单元，每个输出单元代表一个特定的 < 人物 2>。在这种情况下，有两个正确答案（用黑点标记），因为 Colin 有两个阿姨。输入单元和输出单元在空间上排列，其中英国人在一行，同构的意大利人紧接在下方。

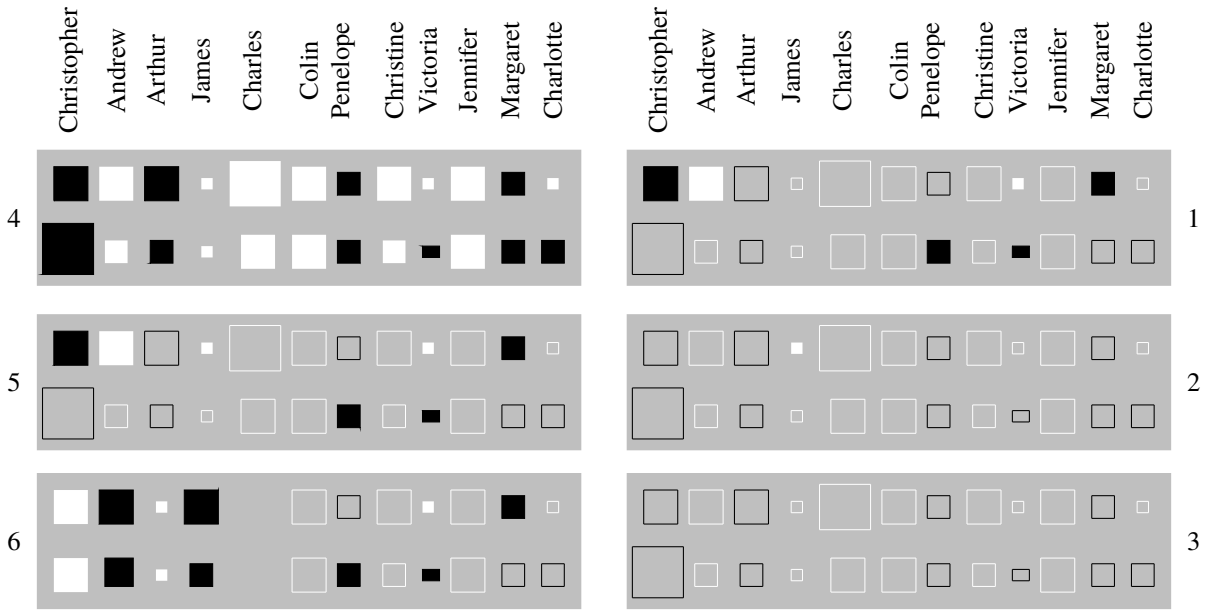

图 4: 从代表人物的 24 个输入单元到第二层中学习人物分布式表示的 6 个单元的权重。白色矩形表示兴奋性权重；黑色矩形表示抑制性权重；矩形的面积表示权重的大小。来自 12 个英国人单元的权重位于每个单元的顶部。单元 1 主要关注英国人和意大利人之间的区别，而大多数其他单元忽略了这一区别。这意味着英国人的表示与其对应的意大利人的表示非常相似。网络利用了两个家谱树之间的同构性来共享结构，因此它能够合理地从一个树推广到另一个树。单元 2 编码人物所属的世代，单元 6 编码人物来自家族的哪一支。隐藏单元捕获的特征在输入和输出编码中并不明显，因为这些编码为每个人使用单独的单元。由于隐藏特征捕获了任务领域的潜在结构，网络能够正确地推广到未训练的四个三元组上。我们训练了网络 1500 次遍历，前 20 次遍历使用 $\epsilon = 0.005$ 和 $\alpha = 0.5$，其余遍历使用 $\epsilon = 0.01$ 和 $\alpha = 0.9$。为了更容易解释权重，我们引入了 “权重衰减”，在每次权重变化后将每个权重减少 0.2%。经过长时间学习后，衰减与 $\partial E/\partial w$ 达到平衡，因此每个权重的最终大小表明了其在减少误差中的有用性。为了防止网络需要大权重来将输出驱动到 1 或 0，如果应该开启的输出单元活动值高于 0.8，而应该关闭的输出单元活动值低于 0.2，则认为误差为零。

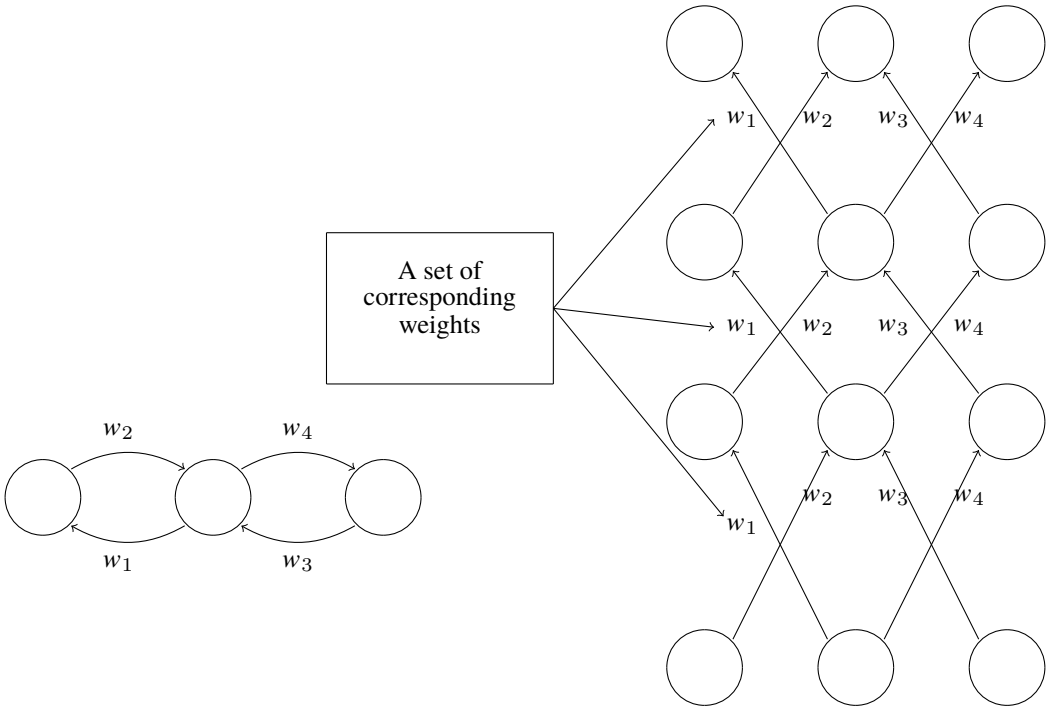

图 5: 一个同步迭代网络运行三次迭代及其等效的分层网络。迭代网络中的每个时间步对应于分层网络中的一层。分层网络的学习过程可以映射到迭代网络的学习过程中。在实现这一映射时，会出现两个复杂问题：首先，在分层网络中，前向传播过程中需要中间层单元的输出值来执行反向传播（见公式（5）和（6））。因此，在迭代网络中，需要存储每个单元的输出状态历史。其次，为了使分层网络与迭代网络等价，不同层之间的对应权重必须具有相同的值。为了保持这一特性，我们计算每组对应权重的平均梯度 $\partial E/\partial w$，然后按比例调整每组权重。在满足这两个条件的情况下，学习过程可以直接应用于迭代网络。这些网络可以学习执行迭代搜索或学习序列结构。

当前形式的学习过程并不是大脑学习的合理模型。然而，将该过程应用于各种任务表明，通过权重空间中的梯度下降可以构建有趣的内部表示，这表明值得寻找更符合生物学原理的神经网络梯度下降方法。

我们感谢系统发展基金会和海军研究办公室的财政支持。