

大牛的《深度学习》笔记，60 分钟带你学完 Deep Learning (下)

2016-08-04 Zouxy AI 科技评论

CCF-GAIR

今年夏天，雷锋网将在深圳举办一场盛况空前的“全球人工智能与机器人峰会”(简称 CCF-GAIR)。大会现场，谷歌，DeepMind，Uber，微软等巨头的人工智能实验室负责人将莅临深圳，向我们零距离展示国外人工智能震撼人心、撬动地球的核心所在。如果你不想错过这个大会的盛世狂欢，请点击 GAIR.leiphone.com 以 6.5 折的超低折扣购票！

导读：昨天我们为大家带来了大牛 Zouxy 学习深度学习的笔记的上篇。今天我们继续为大家带来教程的下篇，让我们看看这位大牛在深度学习领域还有什么独到的理解~

六、浅层学习 (Shallow Learning) 和深度学习 (Deep Learning)

浅层学习是机器学习的第一次浪潮。

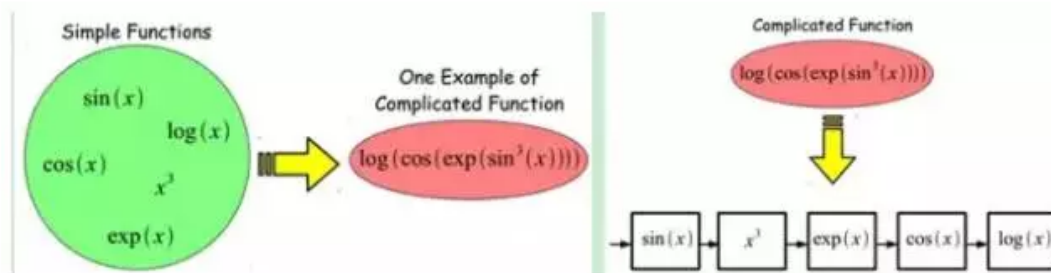
20 世纪 80 年代末期，用于人工神经网络的反向传播算法（也叫 Back Propagation 算法或者 BP 算法）的发明，给机器学习带来了希望，掀起了基于统计模型的机器学习热潮。这个热潮一直持续到今天。人们发现，利用 BP 算法可以让一个神经网络模型从大量训练样本中学习统计规律，从而对未知事件做预测。这种基于统计的机器学习方法比起过去基于人工规则的系统，在很多方面显出优越性。这个时候的人工神经网络，虽也被称作多层感知机 (Multi-layer Perceptron)，但实际是种只含有一层隐层节点的浅层模型。

20 世纪 90 年代，各种各样的浅层机器学习模型相继被提出，例如支撑向量机 (SVM, Support Vector Machines)、Boosting、最大熵方法 (如 LR, Logistic Regression) 等。这些模型的结构基本上可以看成带有一层隐层节点 (如 SVM、Boosting)，或没有隐层节点 (如 LR)。这些模型无论是在理论分析还是应用中都获得了巨大的成功。相比之下，由于理论分析的难度大，训练方法又需要很多经验和技巧，这个时期浅层人工神经网络反而相对沉寂。

深度学习是机器学习的第二次浪潮。

2006 年，加拿大多伦多大学教授、机器学习领域的泰斗 **Geoffrey Hinton** 和他的学生 **Ruslan Salakhutdinov** 在《科学》上发表了一篇文章，开启了深度学习在学术界和工业界的浪潮。这篇文章有两个主要观点：1）多隐层的人工神经网络具有优异的特征学习能力，学习得到的特征对数据有更本质的刻画，从而有利于可视化或分类；2）深度神经网络在训练上的难度，可以通过“逐层初始化”（**layer-wise pre-training**）来有效克服，在这篇文章中，逐层初始化是通过无监督学习实现的。

当前多数分类、回归等学习方法为浅层结构算法，其局限性在于有限样本和计算单元情况下对复杂函数的表示能力有限，针对复杂分类问题其泛化能力受到一定制约。深度学习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，并展现了强大的从少数样本集中学习数据集本质特征的能力。（多层的好处是可以用较少的参数表示复杂的函数）



深度学习的实质，是通过构建具有很多隐层的机器学习模型和海量的训练数据，来学习更有用的特征，从而最终提升分类或预测的准确性。因此，“深度模型”是手段，“特征学习”是目的。区别于传统的浅层学习，深度学习的不同在于：1）强调了模型结构的深度，通常有 5 层、6 层，甚至 10 多层的隐层节点；2）明确突出了特征学习的重要性，也就是说，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。

I 七、Deep learning 与 Neural Network

深度学习是机器学习研究中的一个新的领域，其动机在于建立、模拟人脑进行分析学习的神经网络，它模仿人脑的机制来解释数据，例如图像，声音和文本。深度学习是无监督学习的一种。

深度学习的概念源于人工神经网络的研究。含多隐层的多层感知器就是一种深度学习结构。深度学习通过组合低层特征形成更加抽象的高层表示属性类别或特征，以发现数据的分布式特征表示。

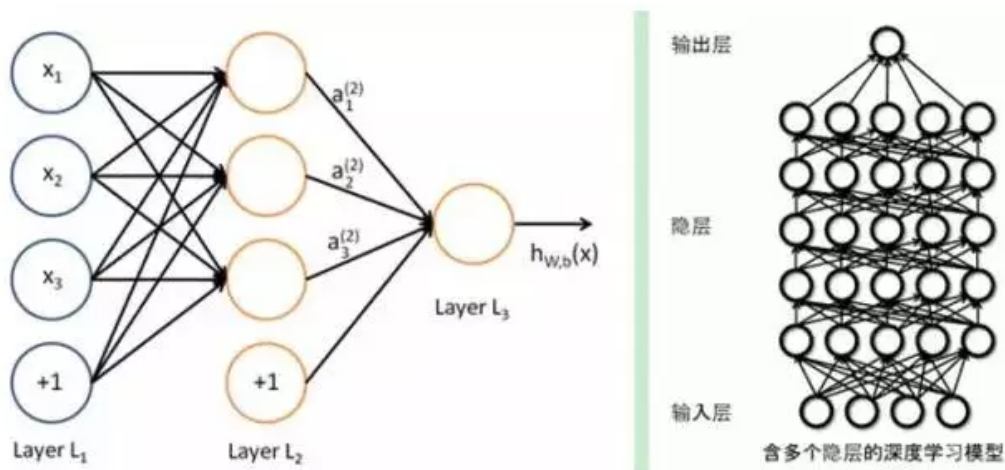
Deep learning 本身算是 machine learning 的一个分支，简单可以理解为 neural network 的发展。大约二三十年前，neural network 曾经是 ML 领域特别火热的一个方向，但是后来确慢慢淡出了，原因包括以下几个方面：

- 1) 比较容易过拟合，参数比较难 tune，而且需要不少 trick；
- 2) 训练速度比较慢，在层次比较少（小于等于 3）的情况下效果并不比其它方法更优；

所以中间有大约 20 多年的时间，神经网络被关注很少，这段时间基本上是 SVM 和 boosting 算法的天下。但是，一个痴心的老先生 Hinton，他坚持了下来，并最终（和其它人一起 Bengio、Yann.lecun 等）提成了一个实际可行的 deep learning 框架。

Deep learning 与传统的神经网络之间有相同的地方也有很多不同。

二者的相同在于 deep learning 采用了神经网络相似的分层结构，系统由包括输入层、隐层（多层）、输出层组成的多层网络，只有相邻层节点之间有连接，同一层以及跨层节点之间相互无连接，每一层可以看作是一个 logistic regression 模型；这种分层结构，是比较接近人类大脑的结构。



而为了克服神经网络训练中的问题，DL 采用了与神经网络很不同的训练机制。传统神经网络中，采用的是 back propagation 的方式进行，简单来讲就是采用迭代的算法来训练整个网络，随机设定初值，计算当前网络的输出，然后根据当前输出和 label 之间的差去改变前面各层的参数，直到收敛（整体是一个梯度

下降法)。而 deep learning 整体上是一个 layer-wise 的训练机制。这样做的原因是因为，如果采用 back propagation 的机制，对于一个 deep network (7 层以上)，残差传播到最前面的层已经变得太小，出现所谓的 gradient diffusion (梯度扩散)。这个问题我们接下来讨论。

八、Deep learning 训练过程

8.1、传统神经网络的训练方法为什么不能用在深度神经网络

BP 算法作为传统训练多层网络的典型算法，实际上对仅含几层网络，该训练方法就已经很不理想。深度结构 (涉及多个非线性处理单元层) 非凸目标代价函数中普遍存在的局部最小是训练困难的主要来源。

BP 算法存在的问题:

- (1) 梯度越来越稀疏: 从顶层越往下, 误差校正信号越来越小;
- (2) 收敛到局部最小值: 尤其是从远离最优区域开始的时候 (随机值初始化会导致这种情况的发生);
- (3) 一般, 我们只能用有标签的数据来训练: 但大部分的数据是没标签的, 而大脑可以从没有标签的数据中学习;

8.2、deep learning 训练过程

如果对所有层同时训练, 时间复杂度会太高; 如果每次训练一层, 偏差就会逐层传递。这会面临跟上面监督学习中相反的问题, 会严重欠拟合 (因为深度网络的神经元和参数太多了)。

2006 年, hinton 提出了在非监督数据上建立多层神经网络的一个有效方法, 简单的说, 分为两步, 一是每次训练一层网络, 二是调优, 使原始表示 x 向上生成的高级表示 r 和该高级表示 r 向下生成的 x' 尽可能一致。方法是:

- 1) 首先逐层构建单层神经元, 这样每次都是训练一个单层网络。
- 2) 当所有层训练完后, Hinton 使用 wake-sleep 算法 进行调优。

将除最顶层的其它层间的权重变为双向的, 这样最顶层仍然是一个单层神经网络, 而其它层则变为了 图模型。向上的权重用于“认知”, 向下的权重用于“生成”。然后使用 Wake-Sleep 算法调整所有的权重。让认知和生成达成一致, 也就是保证生成的最顶层表示能够尽可能正确的复原底层的结点。比如顶层的一个结点表示人脸, 那么所有人脸的图像应该激活这个结点, 并且这个结果向下生成的图像应该能够表现为一个大概的人脸图像。Wake-Sleep 算法分为醒 (wake) 和睡 (sleep) 两个部分。

1) **wake 阶段**: 认知过程, 通过外界的特征和向上的权重 (认知权重) 产生每一层的抽象表示 (结点状态), 并且使用梯度下降修改层间的下行权重 (生成权重)。也就是“如果现实跟我想象的不一样, 改变我的权重使得我想象的东西就是这样的”。

2) **sleep 阶段**: 生成过程, 通过顶层表示 (醒时学得的概念) 和向下权重, 生成底层的状态, 同时修改层间向上的权重。也就是“如果梦中的景象不是我脑中的相应概念, 改变我的认知权重使得这种景象在我看来就是这个概念”。

deep learning 训练过程具体如下:

1) **使用自下上升非监督学习 (就是从底层开始, 一层一层的往顶层训练)**:

采用无标定数据 (有标定数据也可) 分层训练各层参数, 这一步可以看作是一个无监督训练过程, 是和传统神经网络区别最大的部分 (这个过程可以看作是 **feature learning** 过程)

具体的, 先用无标定数据训练第一层, 训练时先学习第一层的参数 (这一层可以看作是得到一个使得输出和输入差别最小的三层神经网络的隐层), 由于模型 **capacity** 的限制以及稀疏性约束, 使得得到的模型能够学习到数据本身的结构, 从而得到比输入更具有表示能力的特征; 在学习得到第 **n-1** 层后, 将 **n-1** 层的输出作为第 **n** 层的输入, 训练第 **n** 层, 由此分别得到各层的参数;

2) **自顶向下的监督学习 (就是通过带标签的数据去训练, 误差自顶向下传输, 对网络进行微调)**:

基于第一步得到的各层参数进一步 **fine-tune** 整个多层模型的参数, 这一步是一个有监督训练过程; 第一步类似神经网络的随机初始化初值过程, 由于 **DL** 的第一步不是随机初始化, 而是通过学习输入数据的结构得到的, 因而这个初值更接近全局最优, 从而能够取得更好的效果; 所以 **deep learning** 效果好很大程度上归功于第一步的 **feature learning** 过程。

I 九、Deep Learning 的常用模型或者方法

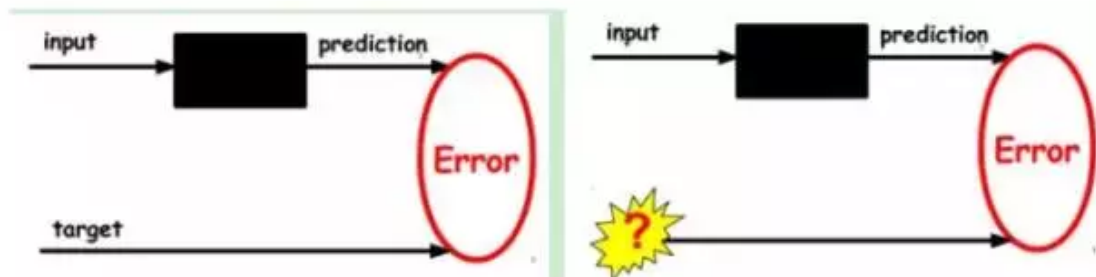
9.1、AutoEncoder 自动编码器

Deep Learning 最简单的一种方法是利用人工神经网络的特点, 人工神经网络 (**ANN**) 本身就是具有层次结构的系统, 如果给定一个神经网络, 我们假设其输出与输入是相同的, 然后训练调整其参数, 得到每一层中的权重。自然地, 我们就得到了输入 **I** 的几种不同表示 (每一层代表一种表示), 这些表示就是特征。自动编码器就是一种尽可能复现输入信号的神经网络。为了实现这种复现, 自动

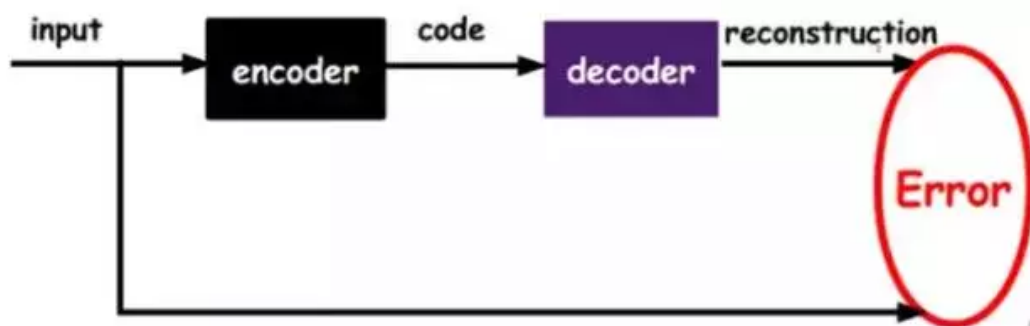
编码器就必须捕捉可以代表输入数据的最重要的因素，就像 PCA 那样，找到可以代表原信息的主要成分。

具体过程简单的说明如下：

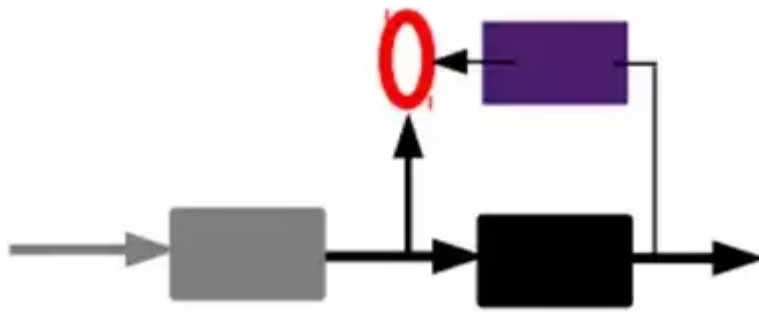
1) 给定无标签数据，用非监督学习学习特征：



在我们之前的神经网络中，如第一个图，我们输入的样本是有标签的，即（input, target），这样我们根据当前输出和 target（label）之间的差去改变前面各层的参数，直到收敛。但现在我们只有无标签数据，也就是右边的图。那么这个误差怎么得到呢？



如上图，我们将 input 输入一个 encoder 编码器，就会得到一个 code，这个 code 也就是输入的一个表示，那么我们怎么知道这个 code 表示的就是 input 呢？我们加一个 decoder 解码器，这时候 decoder 就会输出一个信息，那么如果输出的这个信息和一开始的输入信号 input 是很像的（理想情况下就是一样的），那很明显，我们就有理由相信这个 code 是靠谱的。所以，我们就通过调整 encoder 和 decoder 的参数，使得重构误差最小，这时候我们就得到了输入 input 信号的第一个表示了，也就是编码 code 了。因为是无标签数据，所以误差的来源就是直接重构后与原输入相比得到。



2) 通过编码器产生特征，然后训练下一层。这样逐层训练：

那上面我们就得到第一层的 **code**，我们的重构误差最小让我们相信这个 **code** 就是原输入信号的良好表达了，或者牵强点说，它和原信号是一模一样的（表达不一样，反映的是一个东西）。那第二层和第一层的训练方式就没有差别了，我们将第一层输出的 **code** 当成第二层的输入信号，同样最小化重构误差，就会得到第二层的参数，并且得到第二层输入的 **code**，也就是原输入信息的第二个表达了。其他层就同样的方法炮制就行了（训练这一层，前面层的参数都是固定的，并且他们的 **decoder** 已经没用了，都不需要了）。



3) 有监督微调：

经过上面的方法，我们就可以得到很多层了。至于需要多少层（或者深度需要多少，这个目前本身就没有一个科学的评价方法）需要自己试验调了。每一层都会得到原始输入的不同表达。当然了，我们觉得它是越抽象越好了，就像人的视觉系统一样。

到这里，这个 **AutoEncoder** 还不能用来分类数据，因为它还没有学习如何去连结一个输入和一个类。它只是学会了如何去重构或者复现它的输入而已。或者说，它只是学习获得了一个可以良好代表输入的特征，这个特征可以最大程度上代表原输入信号。那么，为了实现分类，我们就可以在 **AutoEncoder** 的最顶

的编码层添加一个分类器（例如罗杰斯特回归、SVM 等），然后通过标准的多层神经网络的监督训练方法（梯度下降法）去训练。

也就是说，这时候，我们需要将最后层的特征 **code** 输入到最后的分类器，通过有标签样本，通过监督学习进行微调，这也分两种，一个是只调整分类器（黑色部分）：



另一种：通过有标签样本，微调整整个系统：（如果有足够多的数据，这个是最好的。end-to-end learning 端对端学习）

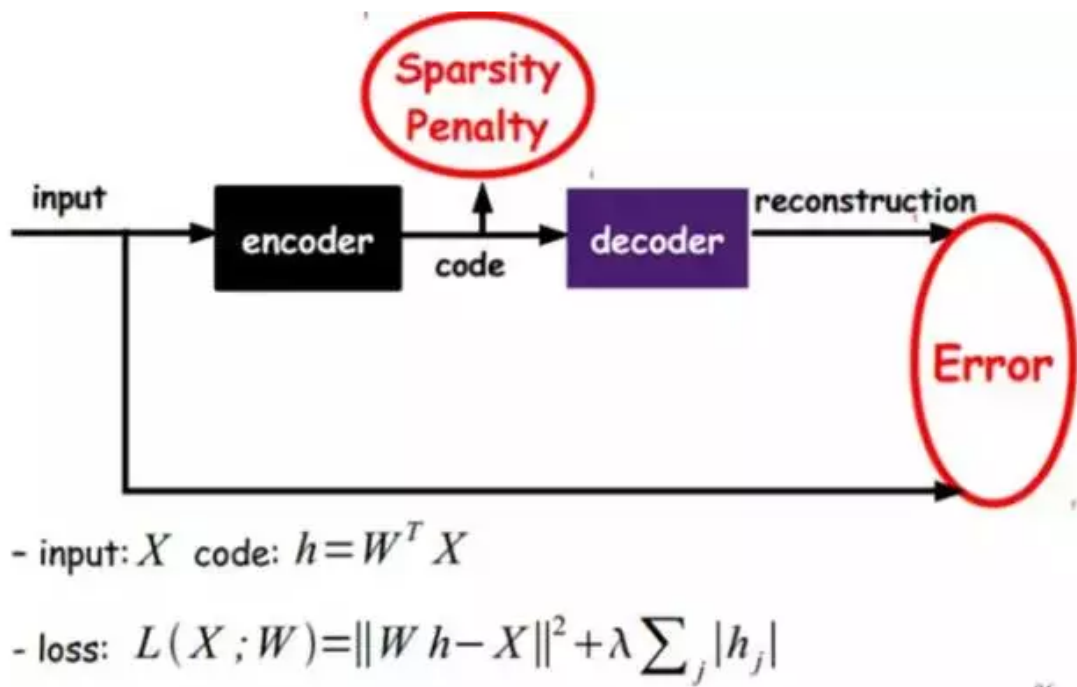


一旦监督训练完成，这个网络就可以用来分类了。神经网络的最顶层可以作为一个线性分类器，然后我们可以用一个更好性能的分类器去取代它。在研究中可以发现，如果在原有的特征中加入这些自动学习得到的特征可以大大提高精确度，甚至在分类问题中比目前最好的分类算法效果还要好！

AutoEncoder 存在一些变体，这里简要介绍下两个：

Sparse AutoEncoder 稀疏自动编码器：

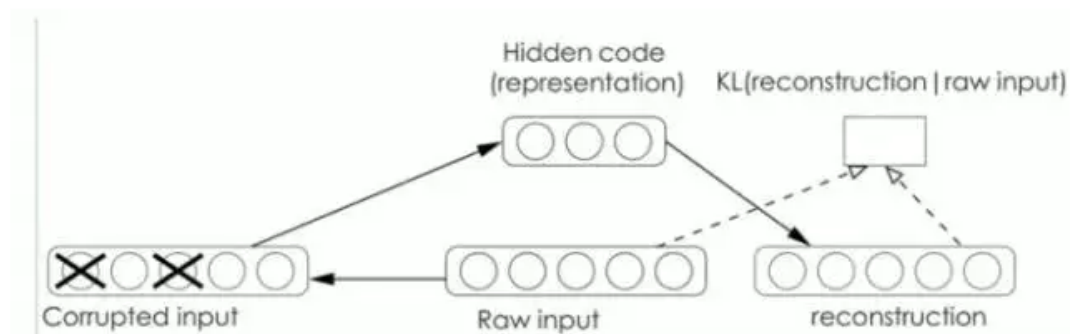
当然，我们还可以继续加上一些约束条件得到新的 Deep Learning 方法，如：如果在 **AutoEncoder** 的基础上加上 L1 的 **Regularity** 限制（L1 主要是约束每一层中的节点中大部分都要为 0，只有少数不为 0，这就是 **Sparse** 名字的来源），我们就可以得到 **Sparse AutoEncoder** 法。



如上图，其实就是限制每次得到的表达 **code** 尽量稀疏。因为稀疏的表达往往比其他的表达要有效（人脑好像也是这样的，某个输入只是刺激某些神经元，其他的大部分神经元是受到抑制的）。

Denoising AutoEncoders 降噪自动编码器:

降噪自动编码器 DA 是在自动编码器的基础上，**训练数据加入噪声**，所以自动编码器必须学习去去除这种噪声而获得真正的没有被噪声污染过的输入。因此，这就迫使编码器去学习输入信号的更加鲁棒的表达，这也是它的泛化能力比一般编码器强的原因。DA 可以通过梯度下降算法去训练。



9.2、Sparse Coding 稀疏编码

如果我们把输出必须和输入相等的限制放松，同时利用线性代数中基的概念，即 $O = a_1 * \phi_1 + a_2 * \phi_2 + \dots + a_n * \phi_n$ ， ϕ_i 是基， a_i 是系数，我们可以得到这样一个优化问题：

$$\min |I - O|$$

其中 I 表示输入， O 表示输出。
通过求解这个最优化式子，我们可以求得系数 a_i 和基 ϕ_i ，这些系数和基就是输入的另外一种近似表达。

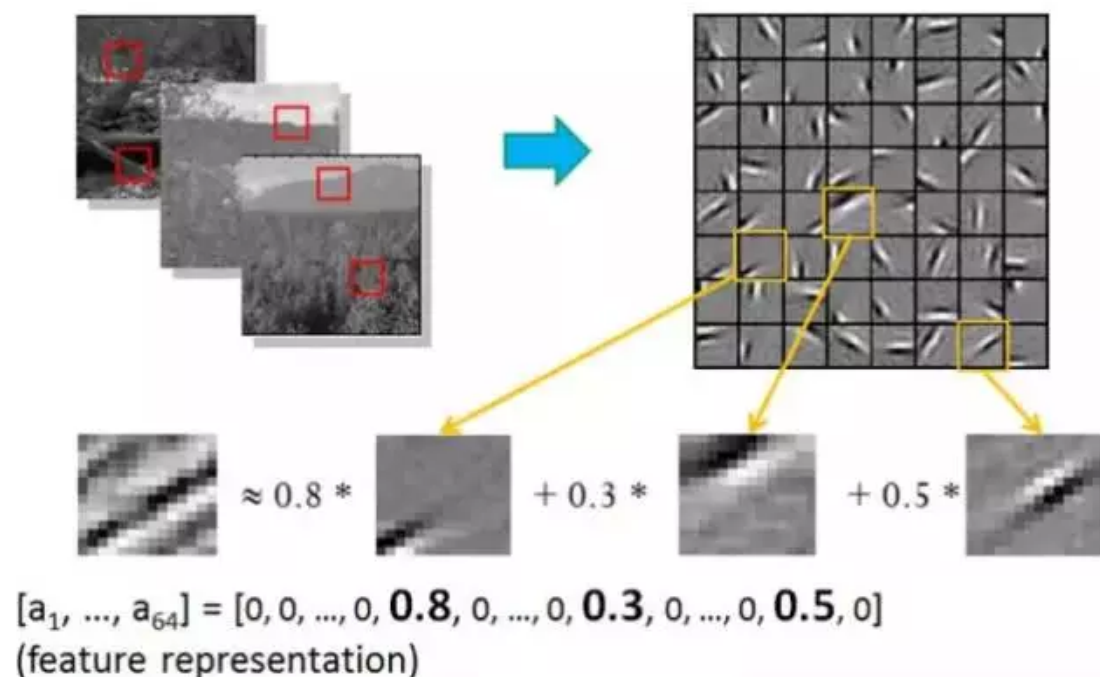
$$x = \sum_{i=1}^k a_i \phi_i$$

因此，它们可以用来表达输入 I ，这个过程也是自动学习得到的。如果我们在上述式子上加上 L1 的 **Regularity** 限制，得到：

$$\min |I - O| + \lambda (|a_1| + |a_2| + \dots + |a_n|)$$

这种方法被称为 **Sparse Coding**。通俗的说，就是将一个信号表示为一组基的线性组合，而且要求只需要较少的几个基就可以将信号表示出来。“稀疏性”定义为：只有很少的几个非零元素或只有很少的几个远大于零的元素。要求系数 a_i 是稀疏的意思就是说：对于一组输入向量，我们只想有尽可能少的几个系数远大于零。选择使用具有稀疏性的分量来表示我们的输入数据是有原因的，因为绝大多数的感官数据，比如自然图像，可以被表示成少量基本元素的叠加，在图像中这些基本元素可以是面或者线。同时，比如与初级视觉皮层的类比过程也因此得到了提升（人脑有大量的神经元，但对于某些图像或者边缘只有很少的神经元兴奋，其他都处于抑制状态）。

稀疏编码算法是一种无监督学习方法，它用来寻找一组“超完备”基向量来更高效地表示样本数据。虽然形如主成分分析技术（PCA）能使我们方便地找到一组“完备”基向量，但是这里我们想要做的是找到一组“超完备”基向量来表示输入向量（也就是说，基向量的个数比输入向量的维数要大）。超完备基的好处是它们能更有效地找出隐含在输入数据内部的结构与模式。然而，对于超完备基来说，系数 a_i 不再由输入向量唯一确定。因此，在稀疏编码算法中，我们另加了一个评判标准“稀疏性”来解决因超完备而导致的退化（degeneracy）问题。



比如在图像的 Feature Extraction 的最底层要做 Edge Detector 的生成，那么这里的工作就是从 Natural Images 中 randomly 选取一些小 patch，通过这些 patch 生成能够描述他们的“基”，也就是右边的 $8 \times 8 = 64$ 个 basis 组成的 basis，然后给定一个 test patch，我们可以按照上面的式子通过 basis 的线性组合得到，而 sparse matrix 就是 a ，下图中的 a 中有 64 个维度，其中非零项只有 3 个，故称“sparse”。

这里可能大家会有疑问，为什么把底层作为 Edge Detector 呢？上层又是什么呢？这里做个简单解释大家就会明白，之所以是 Edge Detector 是因为不同方向的 Edge 就能够描述出整幅图像，所以不同方向的 Edge 自然就是图像的 basis 了……而上一层的 basis 组合的结果，上上层又是上一层的组合 basis……（就是上面第四部分的时候咱们说的那样）

Sparse coding 分为两个部分：

1) **Training 阶段**：给定一系列的样本图片 $[x_1, x_2, \dots]$ ，我们需要学习得到一组基 $[\phi_1, \phi_2, \dots]$ ，也就是字典。

稀疏编码是 k-means 算法的变体，其训练过程也差不多（EM 算法的思想：如果要优化的目标函数包含两个变量，如 $L(W, B)$ ，那么我们可以先固定 W ，调整 B 使得 L 最小，然后再固定 B ，调整 W 使 L 最小，这样迭代交替，不断将 L 推向最小值。

训练过程就是一个重复迭代的过程，按上面所说，我们交替的更改 a 和 ϕ 使得下面这个目标函数最小。

$$\min_{a, \phi} \sum_{i=1}^m \left\| x_i - \sum_{j=1}^k a_{i,j} \phi_j \right\|^2 + \lambda \sum_{i=1}^m \sum_{j=1}^k |a_{i,j}|$$

每次迭代分两步：

a) 固定字典 $\Phi[k]$ ，然后调整 $a[k]$ ，使得上式，即目标函数最小（即解 LASSO 问题）。

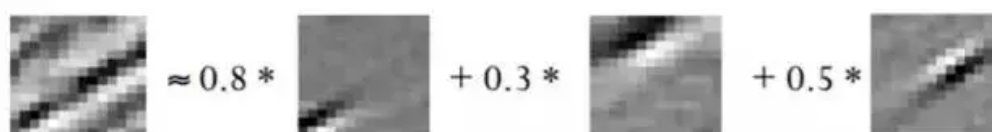
b) 然后固定住 $a[k]$ ，调整 $\Phi[k]$ ，使得上式，即目标函数最小（即解凸 QP 问题）。

不断迭代，直至收敛。这样就可以得到一组可以良好表示这一系列 x 的基，也就是字典。

2) Coding 阶段：给定一个新的图片 x ，由上面得到的字典，通过解一个 LASSO 问题得到稀疏向量 a 。这个稀疏向量就是这个输入向量 x 的一个稀疏表达了。

$$\min_a \sum_{i=1}^m \left\| x_i - \sum_{j=1}^k a_{i,j} \phi_j \right\|^2 + \lambda \sum_{i=1}^m \sum_{j=1}^k |a_{i,j}|$$

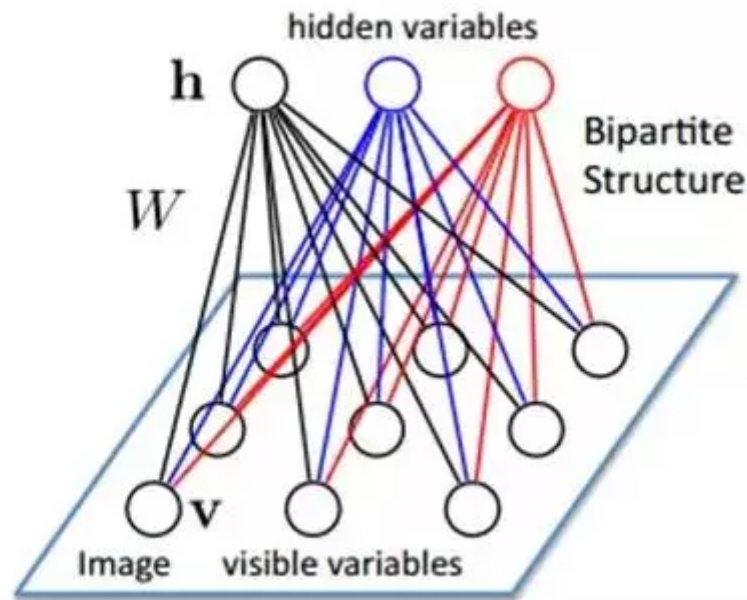
例如：



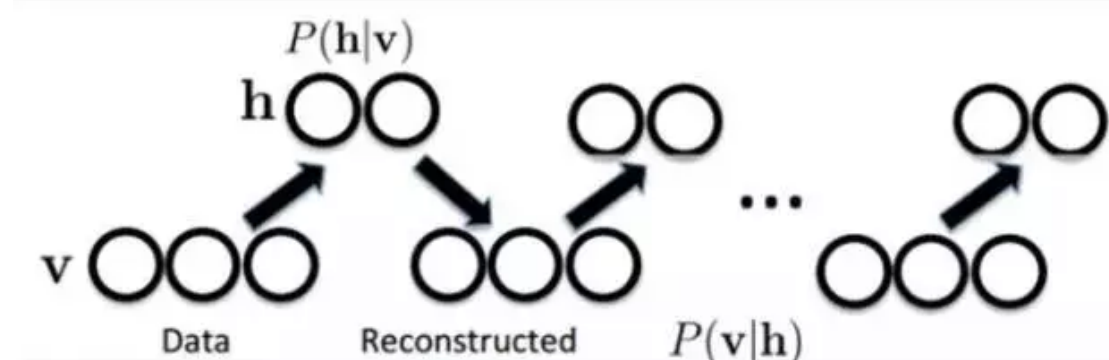
Represent x_i as: $a_i = [0, 0, \dots, 0, \mathbf{0.8}, 0, \dots, 0, \mathbf{0.3}, 0, \dots, 0, \mathbf{0.5}, \dots]$

9.3、Restricted Boltzmann Machine (RBM)限制波尔兹曼机

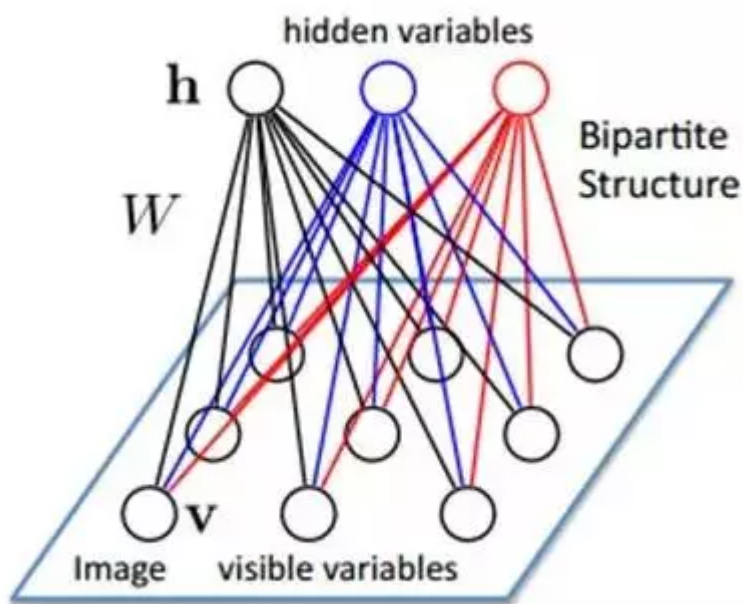
假设有一个二部图，每一层的节点之间没有链接，一层是可视层，即输入数据层 (v)，一层是隐藏层(h)，如果假设所有的节点都是随机二值变量节点（只能取 0 或者 1 值），同时假设全概率分布 $p(v,h)$ 满足 Boltzmann 分布，我们称这个模型是 Restricted Boltzmann Machine (RBM)。



下面我们来看看为什么它是 Deep Learning 方法。首先，这个模型因为是二部图，所以在已知 v 的情况下，所有的隐藏节点之间是条件独立的（因为节点之间不存在连接），即 $p(h|v)=p(h_1|v)...p(h_n|v)$ 。同理，在已知隐藏层 h 的情况下，所有的可视节点都是条件独立的。同时又由于所有的 v 和 h 满足 Boltzmann 分布，因此，当输入 v 的时候，通过 $p(h|v)$ 可以得到隐藏层 h ，而得到隐藏层 h 之后，通过 $p(v|h)$ 又能得到可视层，通过调整参数，我们就是要使得从隐藏层得到的可视层 v_1 与原来的可视层 v 如果一样，那么得到的隐藏层就是可视层另外一种表达，因此隐藏层可以作为可视层输入数据的特征，所以它就是一种 Deep Learning 方法。



如何训练呢？也就是可视层节点和隐节点间的权值怎么确定呢？我们需要做一些数学分析。也就是模型了。



联合组态（joint configuration）的能量可以表示为：

$$E(\mathbf{v}, \mathbf{h}; \theta) = - \sum_{ij} W_{ij} v_i h_j - \sum_i b_i v_i - \sum_j a_j h_j$$

$\theta = \{W, a, b\}$ model parameters.

而某个组态的联合概率分布可以通过 Boltzmann 分布（和这个组态的能量）来确定：

$$P_{\theta}(\mathbf{v}, \mathbf{h}) = \frac{1}{Z(\theta)} \exp(-E(\mathbf{v}, \mathbf{h}; \theta)) = \frac{1}{Z(\theta)} \underbrace{\prod_{ij} e^{W_{ij} v_i h_j}}_{\text{partition function}} \underbrace{\prod_i e^{b_i v_i} \prod_j e^{a_j h_j}}_{\text{potential functions}}$$

$$Z(\theta) = \sum_{\mathbf{h}, \mathbf{v}} \exp(-E(\mathbf{v}, \mathbf{h}; \theta))$$

因为隐藏节点之间是条件独立的（因为节点之间不存在连接），即：

$$P(\mathbf{h}|\mathbf{v}) = \prod_j P(h_j|\mathbf{v})$$

然后我们可以比较容易（对上式进行因子分解 **Factorizes**）得到在给定可视层 \mathbf{v} 的基础上，隐层第 j 个节点为 1 或者为 0 的概率：

$$P(h_j = 1|\mathbf{v}) = \frac{1}{1 + \exp(-\sum_i W_{ij}v_i - a_j)}$$

同理，在给定隐层 \mathbf{h} 的基础上，可视层第 i 个节点为 1 或者为 0 的概率也可以容易得到：

$$P(\mathbf{v}|\mathbf{h}) = \prod_i P(v_i|\mathbf{h}) \quad P(v_i = 1|\mathbf{h}) = \frac{1}{1 + \exp(-\sum_j W_{ij}h_j - b_i)}$$

给定一个满足独立同分布的样本集： $D=\{\mathbf{v}(1), \mathbf{v}(2), \dots, \mathbf{v}(N)\}$ ，我们需要学习参数 $\theta=\{\mathbf{W}, \mathbf{a}, \mathbf{b}\}$ 。

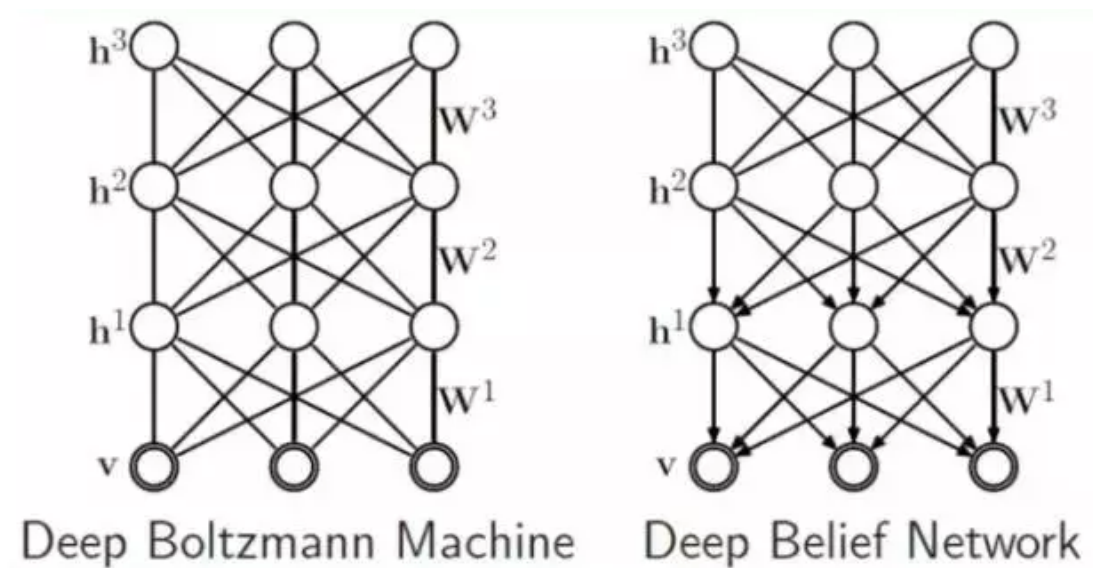
我们最大化以下对数似然函数（最大似然估计：对于某个概率模型，我们需要选择一个参数，让我们当前的观测样本的概率最大）：

$$L(\theta) = \frac{1}{N} \sum_{n=1}^N \log P_{\theta}(\mathbf{v}^{(n)}) - \frac{\lambda}{N} \|\mathbf{W}\|_F^2$$

也就是对最大对数似然函数求导，就可以得到 L 最大时对应的参数 \mathbf{W} 了。

$$\frac{\partial L(\theta)}{\partial W_{ij}} = E_{P_{data}}[v_i h_j] - E_{P_{\theta}}[v_i h_j] - \frac{2\lambda}{N} W_{ij}$$

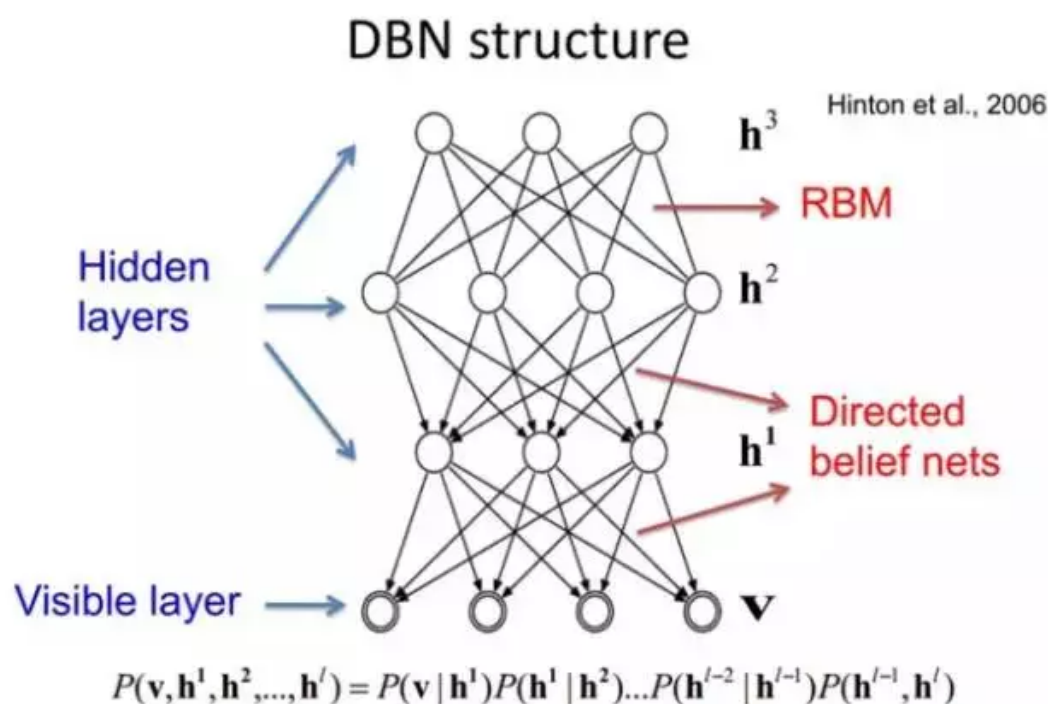
如果，我们把隐藏层的层数增加，我们可以得到 **Deep Boltzmann Machine (DBM)**；如果我们在靠近可视层的部分使用贝叶斯信念网络（即有向图模型，当然这里依然限制层中节点之间没有链接），而在最远离可视层的部分使用 **Restricted Boltzmann Machine**，我们可以得到 **Deep Belief Net (DBN)**。



9.4、Deep Belief Networks 深信度网络

DBNs 是一个概率生成模型，与传统的判别模型的神经网络相对，生成模型是建立一个观察数据和标签之间的联合分布，对 $P(\text{Observation}|\text{Label})$ 和 $P(\text{Label}|\text{Observation})$ 都做了评估，而判别模型仅仅而已评估了后者，也就是 $P(\text{Label}|\text{Observation})$ 。对于在深度神经网络应用传统的 BP 算法的时候，DBNs 遇到了以下问题：

- (1) 需要为训练提供一个有标签的样本集；
- (2) 学习过程较慢；
- (3) 不适当的参数选择会导致学习收敛于局部最优解。



DBNs 由多个限制玻尔兹曼机（Restricted Boltzmann Machines）层组成，一个典型的神经网络类型如图三所示。这些网络被“限制”为一个可视层和一个隐层，层间存在连接，但层内的单元间不存在连接。隐层单元被训练去捕捉在可视层表现出来的高阶数据的相关性。

首先，先不考虑最顶构成一个联想记忆（associative memory）的两层，一个 DBN 的连接是通过自顶向下的生成权值来指导确定的，RBMs 就像一个建筑块一样，相比传统和深度分层的 sigmoid 信念网络，它能易于连接权值的学习。

最开始的时候，通过一个非监督贪婪逐层方法去预训练获得生成模型的权值，非监督贪婪逐层方法被 Hinton 证明是有效的，并被其称为对比分歧（contrastive divergence）。

在这个训练阶段，在可视层会产生一个向量 v ，通过它将值传递到隐层。反过来，可视层的输入会被随机的选择，以尝试去重构原始的输入信号。最后，这些新的可视的神经元激活单元将前向传递重构隐层激活单元，获得 h （在训练过程中，首先将可视向量值映射给隐单元；然后可视单元由隐层单元重建；这些新可视单元再次映射给隐单元，这样就获取新的隐单元。执行这种反复步骤叫做吉布斯采样）。这些后退和前进的步骤就是我们熟悉的 Gibbs 采样，而隐层激活单元和可视层输入之间的相关性差别就作为权值更新的主要依据。

训练时间会显著的减少，因为只需要单个步骤就可以接近最大似然学习。增加进网络的每一层都会改进训练数据的对数概率，我们可以理解为越来越接近能

量的真实表达。这个有意义的拓展，和无标签数据的使用，是任何一个深度学习应用的决定性的因素。

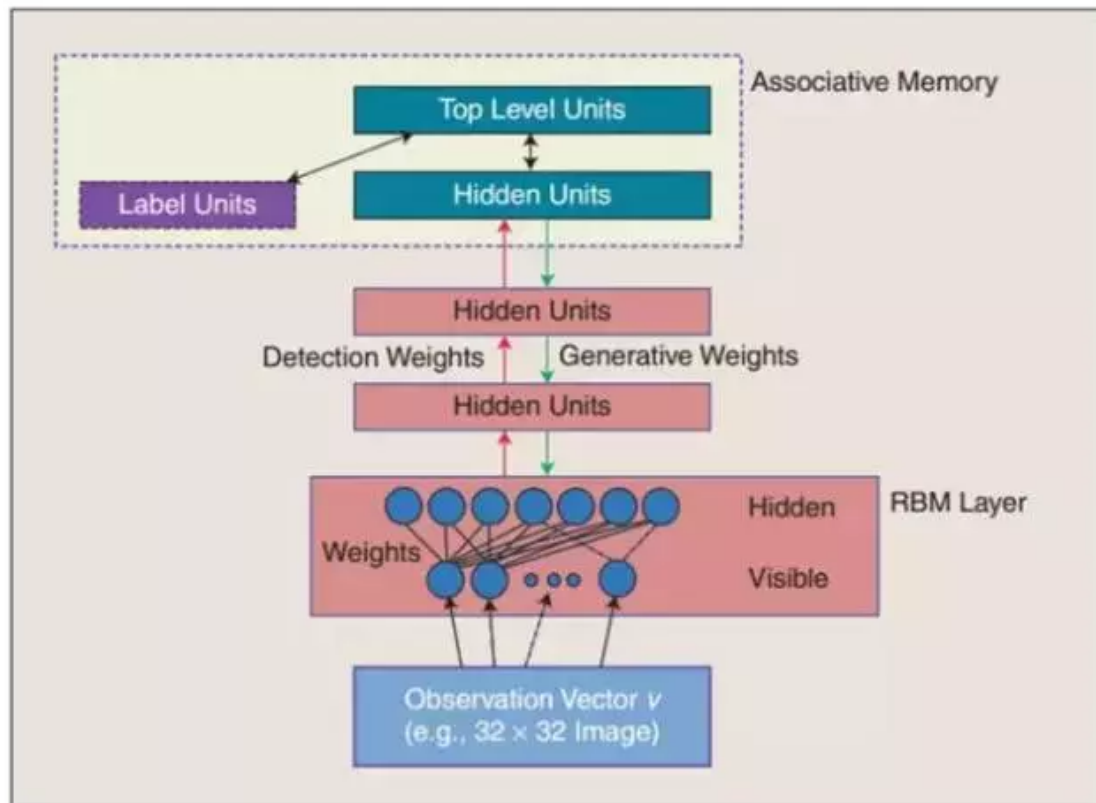


FIGURE 3 Illustration of the Deep Belief Network framework.

在最高两层，权值被连接到一起，这样更低层的输出将会提供一个参考的线索或者关联给顶层，这样顶层就会将其联系到它的记忆内容。而我们最关心的，最后想得到的就是判别性能，例如分类任务里面。

在预训练后，DBN 可以通过利用带标签数据用 BP 算法去对判别性能做调整。在这里，一个标签集将被附加到顶层（推广联想记忆），通过一个自下向上的，学习到的识别权值获得一个网络的分类面。这个性能会比单纯的 BP 算法训练的网络好。这可以很直观的解释，DBNs 的 BP 算法只需要对权值参数空间进行一个局部的搜索，这相比前向神经网络来说，训练是要快的，而且收敛的时间也少。

DBNs 的灵活性使得它的拓展比较容易。一个拓展就是卷积 DBNs（Convolutional Deep Belief Networks(CDBNs)）。DBNs 并没有考虑到图像的 2 维结构信息，因为输入是简单的从一个图像矩阵一维向量化的。而 CDBNs 就是考虑到了这个问题，它利用邻域像素的空域关系，通过一个称为卷积 RBMs 的模型区达到生成模型的变换不变性，而且可以容易得变换到高维图像。DBNs 并

没有明确地处理对观察变量的时间联系的学习上,虽然目前已经有这方面的研究,例如堆叠时间 RBMs,以此为推广,有序列学习的 **dubbed temporal convolution machines**,这种序列学习的应用,给语音信号处理问题带来了一个让人激动的未来研究方向。

目前,和 DBNs 有关的研究包括堆叠自动编码器,它是通过用堆叠自动编码器来替换传统 DBNs 里面的 RBMs。这就使得可以通过同样的规则来训练产生深度多层神经网络架构,但它缺少层的参数化的严格要求。与 DBNs 不同,自动编码器使用判别模型,这样这个结构就很难采样输入采样空间,这就使得网络更难捕捉它的内部表达。但是,降噪自动编码器却能很好的避免这个问题,并且比传统的 DBNs 更优。它通过在训练过程添加随机的污染并堆叠产生场泛化性能。训练单一的降噪自动编码器的过程和 RBMs 训练生成模型的过程一样。

十、总结与展望

1) Deep learning 总结

深度学习是关于自动学习要建模的数据的潜在(隐含)分布的多层(复杂)表达的算法。换句话说,深度学习算法自动的提取分类需要的低层次或者高层次特征。高层次特征,一是指该特征可以分级(层次)地依赖其他特征,例如:对于机器视觉,深度学习算法从原始图像去学习得到它的一个低层次表达,例如边缘检测器,小波滤波器等,然后在这些低层次表达的基础上再建立表达,例如这些低层次表达的线性或者非线性组合,然后重复这个过程,最后得到一个高层次的表达。

Deep learning 能够得到更好地表示数据的 **feature**,同时由于模型的层次、参数很多, **capacity** 足够,因此,模型有能力表示大规模数据,所以对于图像、语音这种特征不明显(需要手工设计且很多没有直观物理含义)的问题,能够在大规模训练数据上取得更好的效果。此外,从模式识别特征和分类器的角度, **deep learning** 框架将 **feature** 和分类器结合到一个框架中,用数据去学习 **feature**,在使用中减少了手工设计 **feature** 的巨大工作量(这是目前工业界工程师付出努力最多的方面),因此,不仅仅效果可以更好,而且,使用起来也有很多方便之处,因此,是十分值得关注的一套框架,每个做 **ML** 的人都应该关注了解一下。

当然, **deep learning** 本身也不是完美的,也不是解决世间任何 **ML** 问题的利器,不应该被放大到一个无所不能的程度。

2) Deep learning 未来

深度学习目前仍有大量工作需要研究。目前的关注点还是从机器学习的领域借鉴一些可以在深度学习使用的方法特别是降维领域。例如：目前一个工作就是稀疏编码，通过压缩感知理论对高维数据进行降维，使得非常少的元素的向量就可以精确的代表原来的高维信号。另一个例子就是半监督流行学习，通过测量训练样本的相似性，将高维数据的这种相似性投影到低维空间。另外一个比较鼓舞人心的方向就是 **evolutionary programming approaches**（遗传编程方法），它可以通过最小化工程能量去进行概念性自适应学习和改变核心架构。

Deep learning 还有很多核心的问题需要解决：

（1）对于一个特定的框架，对于多少维的输入它可以表现得较优（如果是图像，可能是上百万维）？

（2）对捕捉短时或者长时间的时间依赖，哪种架构才是有效的？

（3）如何对于一个给定的深度学习架构，融合多种感知的信息？

（4）有什么正确的机理可以去增强一个给定的深度学习架构，以改进其鲁棒性和对扭曲和数据丢失的不变性？

（5）模型方面是否有其他更为有效且有理论依据的深度模型学习算法？

探索新的特征提取模型是值得深入研究的内容。此外有效的可并行训练算法也是值得研究的一个方向。当前基于最小批处理的随机梯度优化算法很难在多计算机中进行并行训练。通常办法是利用图形处理单元加速学习过程。然而单个机器 **GPU** 对大规模数据识别或相似任务数据集并不适用。在深度学习应用拓展方面，如何合理充分利用深度学习在增强传统学习算法的性能仍是目前各领域的研究重点。