Deep Boltzmann Machine

Chen Gong

21 May 2020

目录

1 Introduction

本章介绍的是深度玻尔兹曼机 (Deep Boltzmann Machines, DBM), 应该算是玻尔兹曼机系列的最后一个模型了。我们前面介绍的三种玻尔兹曼机和今天将要介绍的深度玻尔兹曼机的概率图模型如下图所示, 从左往右分别是深度信念网络 (Deep Belief Network), 限制玻尔兹曼机 (Restricted Boltzmann Machine，RBM)，和 DBM，玻尔兹曼机 (General Boltzmann Machine，BM)：

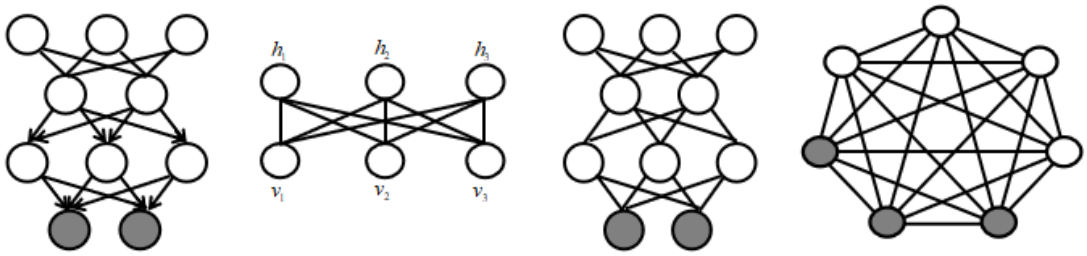

图 1: 四种玻尔兹曼机的概率图模型

显然，深度玻尔兹曼机和深度信念网络的区别就仅仅在于有向和无向上。其中，RBM，DBM 和 BM 都是玻尔兹曼机，而 DBN 和玻尔兹曼机就不太一样，实际上是一个混合模型，最上面是 RBM，而下面的部分都是有向图。

2 Boltzmann Machine 的发展历史

2.1 Boltzmann Machine

最早于 1983 年诞生的是 BM，其概念是来源于 Hopfield Network，这个 Hopfield Network 来源于统计物理学中的 Ising Model。看来，机器学习和统计物理还挺有缘的，记得前面见过的吉布斯分布（玻尔兹曼分布）吗，也是来源于统计物理学，包括强化学习中的很多概念也是。BM 提出了以后就给出了 learning rules。Learning rules 就是一个简单的随机梯度上升（Stochastic Gradient Ascend，SGA），SGA 的学习规则为：

$$\triangle w_{ij} = \alpha \left[\underbrace{\mathbb{E}_{P_{\text{data}}}[v_i h_j]}_{\text{Postive phase}} - \underbrace{\mathbb{E}_{P_{\text{model}}}[v_i h_j]}_{\text{Negative phase}} \right] \tag{1}$$

$$\begin{cases} P_{\text{data}} = P_{\text{data}}(v,h) = P_{\text{data}}(v) \cdot P_{\text{model}}(h|v) \\ P_{\text{model}} = P_{\text{model}}(v,h) \end{cases} \tag{2}$$

其中，$P_{\text{data}}(v)$ 表示由 N 个样本组成的经验分布，就是我们的数据，而 $P_{\text{model}}(h|v)$ 是由模型得出的后验分布，$P_{\text{model}}(v,h)$ 是联合概率分布，也就是模型本身。分布的计算都是通过 MCMC 采样来完成的，其缺点也很明显，就是无法解决过于复杂的问题，很容易遇到收敛时间过长的问题。所以，后来为了简化模型，提出了只有两层的 RBM 模型。

2.2 Restricted Boltzmann Machine

RBM 模型相对比较简单。但是 Hinton 老爷子当时不以为然，觉得 RBM 模型太简单了，表达力不够好。并于 2002 年，提出了对比散度（CD）算法，这个算法在之前的 “直面配分函数” 那章已经做了非常详细的介绍。基于对比散度的采样，实际上就是变了形的 Gibbs 采样，牺牲了部分精度来提高效率，核心目的就是让采样更加高效。同时，CD 算法也给了普通的 Boltzmann 机的学习算法一些借鉴。然后，后续又发展出了概率对比散度（PCD），一种变形的对比散度，采用的 Variational inference 来对 $P_{\text{model}}(v,h)$ 进行近似，从而 RBM 的 Learning 问题可以得到有效的解决方式，有兴趣的同学可以自己查阅。

2.3 Deep Belief Network

因为，RBM 的表达能力较弱，所以最简单的思路就是通过叠加多个 RBM 来增加其层数，从而增加表达能力。但是，增加层数得到的不是 Deep Boltzmann Machines，而是 Deep Belief Network，具体请详细阅读 “深度信念网络”。DBN 虽然预训练上是叠加两个 RBM 而成，但是表现形式并不是玻尔兹曼机。又因为其不是玻尔兹曼机，所以不能用（Stochastic Gradient Ascend，SGA）法来解决。DBN 的求解思路为：

$$\begin{cases} \text{Pre-training (Staking RBM)} \\ \text{Fine-training} \begin{cases} \text{Wake-Sleep 无标签} \\ \text{BP 有标签} \end{cases} \end{cases}$$

算法需要求解的是每一层的权值。第一步则是通过预训练来得到每层的初始值。在后续的 Fine-training 中，无标签的情况等价于 Wake-Sleep 算法求解，如果有标签的话大家觉得是不是和神经网络很像，采用 BP 算法求解。

2.4 Deep Boltzmann Machine

2008 年以后，诞生了 Deep Boltzmann Machine，显然这与 DBN 有很大的不同之处。在之前介绍的解决 Boltzmann Machines 的 SGA 算法，不能解决大规模处理的问题，在 DBM 的求解中的能力大打折扣。很多研究者都想找到高效的 learning rules。其中较好的想法是，先通过预训练来找到一些比较好的权值，然后再使用 SGA。大致流程可做如下描述：

$$\begin{cases} \text{Pre-training (Staking RBM)} \\ \text{SGA} \end{cases}$$

如果没有这个预训练的话，效果非常的不好，时间非常的长。因为，权值的初始值没有任何参考，直接就训练太弱了。关于 DBM 的联合训练方法，就是不通过预训练的方法，这章不作过多介绍。

2.5 小结

本小节介绍了四种 Boltzmann machine 的发展各自的优缺点等。下一小节主要介绍 Hinton 提出的权值初始化预训练的方法，如何将一层层的 RBM 进行叠加得到最终的 DBM，并介绍其与 DBN 中的预训练有什么不一样。

3 预训练

3.1 DBN 中 RBM 结合方法的缺陷

预训练这一章介绍的是，如何叠加两个 RBM，DBN 和 DBM 的不同之处在于如何融合不同的 RBM。实际上 DBN 和 DBM 的每一层的训练都是一样的，唯一的不同就在于各层训练好之后，如何 combine。首先回顾一下 RBM，假如现在只有一层，其他层我们都不考虑：

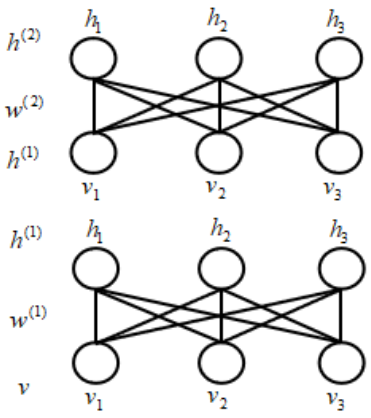

图 2: 限制玻尔兹曼机的概率图模型

首先，哪些是已知的呢? 数据的分布 $P_{\text{data}(v)}$ 是知道的。那么在每一步迭代过程中，可以求出 Log-Likelihood 梯度，利用对比散度采样（CD-K）算法，就可以把 $w^{(1)}$ 学习出来，而且学习到的 $w^{(1)}$ 还不错，当然这个只是近似的，精确的求不出来，主要原因是后验分布是近似采样得到的。有关 RBM 的参数学习在“直面配分函数”那章，已经做了详细的描述，这里不再多说了。

我们来表达一下这个模型：

$$\begin{aligned} P(v) &= \sum_{h^{(1)}} P(h^{(1)}, v) = \sum_{h^{(1)}} P(h^{(1)}) P(v|h^{(1)}) \\ &= \sum_{h^{(1)}} P(h^{(1)}; w^{(1)}) P(v|h^{(1)}, w^{(1)}) \end{aligned} \tag{3}$$

其中参数是 $w^{(1)}$，为什么呢? 因为 $P(h^{(1)}) = \sum_v P(v, h)$，而 v 和 h 之间显示是靠 $w^{(1)}$ 来连接的，所以 $P(h^{(1)})$ 一定是和 $w^{(1)}$ 相关的。

第一个 RBM 求出来以后，那么第二个 RBM 怎么求呢? 实际上这里的 $h^{(1)}$ 都是我们假设出来的。那么下一层 RBM 连样本都没有，怎么办呢? 我们可以将 $h^{(1)}$ 当成是下一个 RBM 的样本，那么采样得到 $h^{(1)}$ 好采吗? 当然简单啦！我们要求的分布为：

$$P(h^{(1)}|v; w^{(1)})$$

其中，$v, w^{(1)}$ 都是已知的，而且由于 RBM 的良好性质，$h^{(1)}$ 的节点之间都是相互独立的，那么：

$$P(h^{(1)}|v; w^{(1)}) = \prod_{i=1}^{3} P(h_i^{(1)}|v; w^{(1)}) \tag{4}$$

而 $P(h_i^{(1)}=1|v;w^{(1)})=\sigma(\sum_{j=1}^{3}w_{ij}v_j)$，而 $P(h_i^{(1)}=0|v;w^{(1)})=1-\sigma(\sum_{j=1}^{3}w_{ij}v_j)$，既然概率值我们都算出来了，从 0/1 分布中采样实在是太简单了。

现在，已经将 $h^{(1)}$ 作为下一层 RBM 的样本，那么怎么构造 $h^{(2)}$ 呢？我们这样构造，

$$P(h^{(1)};w^{(2)})=\sum_{h^{(2)}}P(h^{(1)},h^{(2)};w^{(2)})$$

DBM 中把两个 RBM 简单的挤压成了一个，这个挤压方式很简单，就是用 $P(h^{(1);w^{(2)}})$ 来代替 $P(h^{(1);w^{(1)}})$，为什么呢？因为，公式（3）的主要难点是要得到 $P(h^{(1)})$ 的概率分布，通过加一层的方式，用第二个 RBM 来对 $P(h^{(1)})$ 进行建模，这时替换过后 $P(h^{(1)})$ 是和 $w^{(2)}$ 相关了，而公式（3）中是和 $w^{(1)}$ 相关的。这样对 $h^{(1)}$ 进行建模时，只用到了一层的权重。而大家想想真实的 $P(h^{(1)})$ 到底和什么有关，写出来就知道了：

$$P(h^{(1)})=\sum_{v,h^{(2)}}P(h^{(1)},h^{(2)},v)$$

看到这个公式，用屁股想都知道肯定和 $w^{(1)}$ 和 $w^{(2)}$ 都有关。所以，应该表达为 $P(h^{(1)};w^{(1)},w^{(2)})$。那么，这样直接压缩的方式是有问题的，DBN 采用的就是这种方法（箭头怎么来的，不解释了，“深度信念网络”那一节有详细的解释。），那么 DBN 中对 $h^{(1)}$ 进行建模时，只用到了一层的权重，无论用哪一层，肯定是不恰当的。

那么，这样就给了我们一个启示，可不可以同时用到 $P(h^{(1)};w^{(1)})$ 和 $P(h^{(1)};w^{(2)})$，对他们两做一个平均，就没有那么极端。简单的处理就是取一个平均数就可以了。

启发：用 $P(h^{(1)};w^{(1)})$ 和 $P(h^{(1)};w^{(2)})$ 几何平均近似 $P(h^{(1)};w^{(1)},w^{(2)})$，这样肯定更加适合，但是怎么平均，是简单的相加除 2 吗？后续会详细的说明。

3.2 Double counting problem

真正要求的是：$P(h^{(1)};w^{(1)},w^{(2)})$。

而目的的直觉是：用 $P(h^{(1)};w^{(1)})$ 和 $P(h^{(1)};w^{(2)})$ 几何平均近似 $P(h^{(1)};w^{(1)},w^{(2)})$。

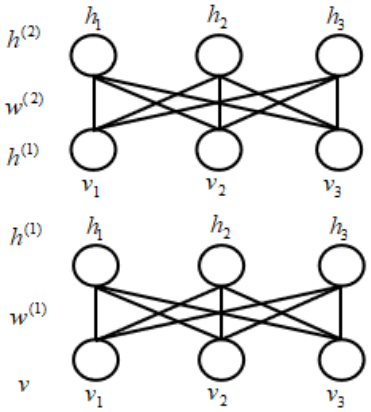

图 3: 限制玻尔兹曼机的概率图模型

其中，

$$\begin{cases} P(h^{(1)};w^{(1)}) = \sum_v P(v,h^{(1)};w^{(1)}) = \sum_v P(v)P(h^{(1)}|v;w^{(1)}) \\ P(h^{(1)};w^{(2)}) = \sum_{h^{(2)}} P(h^{(1)}\ h^{(2)};w^{(2)}) = \sum_{h^{(2)}} P(h^{(2)})P(h^{(2)}|h^{(1)};w^{(1)}) \end{cases}$$

模型中真实存在的只有 v，而 $h^{(1)}$ 实际上是不存在的，这是我们假设出来的。上一小节讲到了 DBN 的结合两个 RBM 的思路中，用 $w^{(2)}$ 来代替 $w^{(1)}$，只用到了一层参数。换句话说，只用 $P(h^{(1)};w^{(2)})$ 来近似真实的 $P(h^{(1)};w^{(1)},w^{(2)})$，而舍弃了 $P(h^{(1)};w^{(1)})$。很自然的可以想到，想要结合 $P(h^{(1)};w^{(1)})$ 和 $P(h^{(1)};w^{(2)})$，那么下一个问题就是怎么结合?

实际上，$\sum_v P(v)P(h^{(1)}|v;w^{(1)})$ 这个分布我们是求不出来的。通常是用采样的方法近似求解，假设观测变量集合为：$v \in V, |V| = N$。那么有：

$$\begin{aligned} \sum_v P(v)P(h^{(1)}|v;w^{(1)}) =& \mathbb{E}_{P(v)}[P(h^{(1)}|v;w^{(1)})] \\ \approx& \frac{1}{N}\sum_{v\in V} P(h^{(1)}|v;w^{(1)}) \end{aligned} \tag{5}$$

其中，$\frac{1}{N}\sum_{v\in V} P(h^{(1)}|v;w^{(1)})$ 也被称为 Aggregate Posterior（聚合后验），代表着用 N 个样本来代替分布。同样，关于 $\sum_{h^{(2)}} P(h^{(1)}\ h^{(2)};w^{(2)})$ 可以得到：

$$\sum_{h^{(2)}} P(h^{(1)}\ h^{(2)};w^{(2)}) \approx \frac{1}{N}\sum_{h^{(2)}\in H} P(h^{(2)}|h^{(1)};w^{(2)}) \tag{6}$$

然后，N 个 $h^{(1)}$ 再根据 $w^{(2)}$ 采样出 N 个 $h^{(2)}$，实际上，在 learning 结束之后，知道 $w^{(1)}$ 和 $w^{(2)}$ 的情况下，采样是很简单的。这样，从底向上采样，就可以计算出各层的后验，合并很简单：

$$\frac{1}{N}\sum_{v\in V} P(h^{(1)}|v;w^{(1)}) + \frac{1}{N}\sum_{h^{(2)}\in H} P(h^{(2)}|h^{(1)};w^{(2)}) \tag{7}$$

即可。这样采样看着倒还是很合理的，那么到底好不好呢？有什么样的问题呢?

这样会导致，Double Counting 的问题，也就是重复计算。

假设 V：是样本集合，$v \in V$；H：才是样本集合，$h^2 \in H$。采样是从下往上依次进行采样，所以，h^1, h^2 都依赖于 v。所以，在计算 h^1, h^2 的过程中都用到了 v 相当于把样本利用了两次。那么，重复计算会带来怎样的副作用呢?

下面举一个例子：

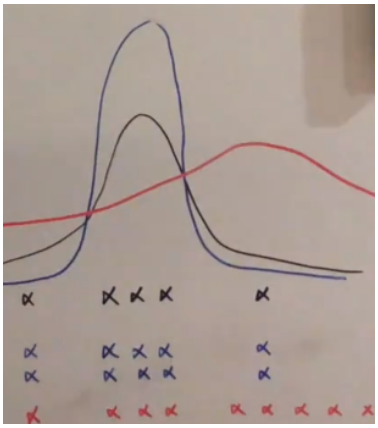

图 4: double counting 问题举例

假如红色的是真实分布，我们实际采样得到的是黑色的样本。简单的假设为高斯分布，利用极大似然估计得到黑色的分布。然后不同的利用黑色的样本，从中采样，采样结果重合，不停的重复利用，会导致所表达的分布越来越尖。从而使得偏差很大，所以简单的结合并不行。

3.3 预训练总结

上一小节，我们介绍了 double counting 问题。实际上在玻尔兹曼机这个系列中，我们可以计算的只有 RBM，其他版本的玻尔兹曼机我们都搞不定。所以，就想办法可不可以将 RBM 叠加来得到具有更好的表达能力的 RBM 模型，于是第一次简单尝试诞生的就是 DBN，DBN 除了顶层是双向的，其他层都是单向的。中国武学中讲究“任督二脉”，DBN 就像只打通了一半，另一半是不通的。很自然，我们想把另外一半也打通，这么模型的表示能力就更强一些。

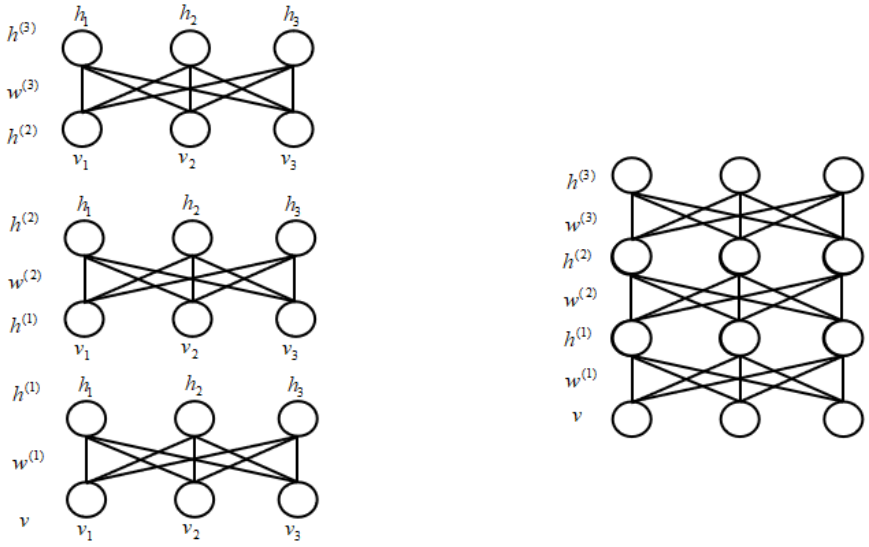

图 5: 利用分层训练 RBM 得到 DBM 的参数

那么，最开始就是将每层 RBM 的参数 $w^{(1)}$，$w^{(2)}$，$w^{(3)}$ 训练出来之后分别赋予 DBM 中的 $w^{(1)}$，$w^{(2)}$，$w^{(3)}$。但是，这样就会有只用到了一层的参数，所以解决方案是取平均值。比如，在分层的 RBM 训练中，求的是 $2w^{(1)}$，$2w^{(2)}$，$2w^{(3)}$，再将权值 $w^{(1)}$，$w^{(2)}$，$w^{(3)}$ 赋予 DBM，相对于每一次只用到了一半的权重。那么，对于中间层 $h^{(1)}$ 来说，就相对于同时用到了 $w^{(1)}$，$w^{(2)}$ 这个在前面，我们已经详细的讲过了。

然而，问题又来了，v 和 $h^{(3)}$ 只和一层相连。那么，除以 2 感觉上有一些不对。给出的处理方法是，在 RBM 的分层学习中，对于 v 从上往下是 $w^{(1)}$，从下往上是 $2w^{(1)}$，这个问题就解决了。这时不是一个简单的 RBM 了，我们称之为“RBM”，哈哈哈哈。其近似的概率图模型可以这样认为，来帮助我们理解：

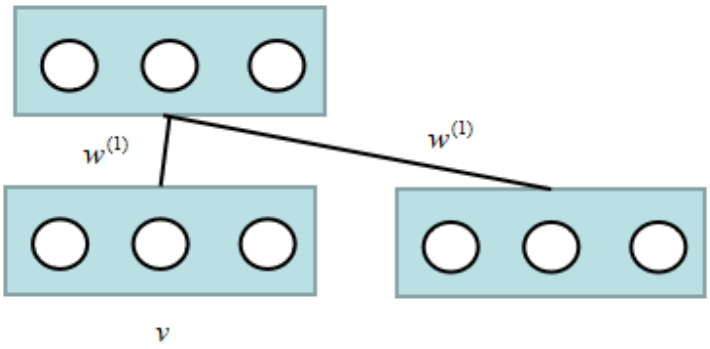

图 6: 利用分层训练 RBM 得到 DBM 的参数

那么，进到 $h^{(1)}$ 有两个 $w^{(1)}$，也就是 $2w^{(1)}$，而进到 v 只有一个 $w^{(1)}$。这就等价于从上往下是 $w^{(1)}$，从下往上是 $2w^{(1)}$。除了，第一层和最后一层，其他层的都是学习的两倍权值，第一层和最后一层则是“RBM”。

从直觉上看，感觉这一系列演变会让模型的性能越来越好。实际上，可以通过数学证明，DBM 的 ELBO、比 DBN 要高，而且 DBM 的层数越多，每叠加一层 RBM，ELBO 都会更高。

4 本章小结

这章中比较完善的讲解了 DBM 的通过，将 DBM 拆解为若干个 RBM，然后对 RBM 进行分层的来进行预训练，并将其预训练得到的权重直接赋予 DBM 的方法。实际上，这只是最经典的训练 DBM 的方法，近些年诞生了很多直接训练的方法，效果也很不错，现在基本也没有人用预训练了。但是，这种基本思想还是很值得学习的。

本章的主要内容，介绍了四种玻尔兹曼模型的发展，并重点介绍了 DBN 模型的不足之处（对中间层建模用到一层的参数），从而引出了 DBM 模型；介绍了 DBM 模型中的 Double counting 问题，并对其进行了详细的解释；最后对 DBM 模型进行了总结，并解释了边界层的处理方法。

实际上 DBM 模型的演变都非常的 Intuitive，不需要太多的数学证明也可以理解。个人觉得这样的思想在科研中很重要，很多情况都是有一个 intuitive 的想法，然后实验发现确实 work，最后寻找证明的方法。（个人愚见）