打印输出

2017年5月9日　　18:32

个人资料

bluebelfast

访问： 84996次
积分： 1089
等级： BLOG 4
排名： 千里之外

原创： 20篇 转载： 24篇
译文： 1篇 评论： 10条

文章搜索

文章分类

Deep Learning (15)
CASA (5)
OpenCL (0)
场景感知/行为感知 (1)
Deep Learning 代码分析 (1)
Android 开发 (1)
linux 开发 (0)
语音识别 (0)
行为识别 (0)
场景分析 (0)
python (14)
matlab (0)
大数据与云计算 (7)
商业智能BI (2)
3D 视觉理解 (0)
3D 虚拟现实之3D robot 虚拟 (0)

文章存档

2016年05月 (1)
2015年12月 (1)
2015年04月 (5)
2015年03月 (1)
2015年01月 (1)

Restricted Boltzmann Machines

标签： deep learning 机器学习 深度学习 dnn

2013-12-11 08:43 7657人阅读 评论(0) 收藏 举报

分类：

Deep Learning (14)

目录(?) [+]

要清楚的理解Restricted Boltzam Machines（RBMs）还是需要大量数学和神经网络的先学基础的（例如数学上的动力系统理论，Hopfield网络，神经网络稳态分析，最优化等）。内容多而杂，在没有这些基础时我们怎样比较简单的能理解RBMs？这里我想通过总结几个大牛的博客和一些理解，尽量给出一个好理解的Restricted Boltzam Machines网络的学习分析。

1、Restricted Boltzamnn Machines 简介

Restricted Boltzamnn Machines（RBMs）是一种典型的随机性神经网络。这里所说的随机表示该神经网络神经元的状态/输出是以概率的方式生成的，例如神经元表示为二元（0,1），0表示该神经元未激活，1表示激活，则神经元处于激活或未激活状态/输出是以一定概率产生/计算的。即神经元的状态是一服从某概率分布的样本（这也是为什么需要sampling的原因）。

Restricted Boltzamnn Machines 由三个部分构成

1）观测单元（visible units）：后用v={v1,v2,...vN}或x表示，为网络的输入，例如图像像素值，语音的音频值等。

2）隐藏单元（hidden units）：后用h={h1,h2,...hM}表示，网络的隐藏单元，如想学习得到的抽象特征，类别等。

3）偏移单元（bias unit）：其状态始终处于激活状态，通过该单元可以对各个隐藏单元输出进行调节。

Restricted Boltzamnn Machines的Restricted是对网络连接进行了限制，如图1、2所示，RBMs无任意两两观测单元间和两两隐藏单元间的连接。图1、2给出了RBM 网络的几何拓扑结构。如果假设所有的节点都是随机二值变量节点（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)。

2、从网络拓扑结构看 Restricted Boltzamnn Machines

RBMs的几何拓扑结构，其表现为二部图/二分图。

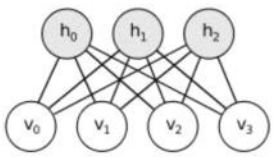

图1 RBMs网络拓扑

假设输入v为一幅图像的像素，RBMs可以表示为如下

关闭

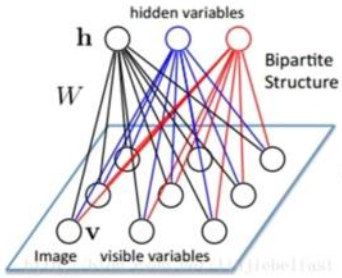

图2 RBMs 拓扑，输入v为图像像素

3、理解 Restricted Boltzamnn Machines

下面我们通过一个简单的电影例子来理解RBMs如何进行建模和学习。例如我们有6部电影：（哈利波特、阿凡达、指环王3、泰坦尼克、变形金刚、环太平洋）。并且我们事先要求用户告诉我们那部电影或那几部电影是他选择想观看的（用户的观看选择数据作为网络的输入，如果想看则输入为1，不想看为0。例如一个用户如果想看哈利波特、阿凡达，则输入为110000）。如果假设我们想通过一RBMs将6部电影进行建模，抽象为两个潜在的类别，即2个隐藏节点。我们可以设置有两个隐藏节点的RBMs，这两个隐藏节点是这些电影的一种抽象特征，例如可以是这些电影的分类类别信息。这两个类别可以是，例如：1）优秀的影片（例如哈利波特，变形金刚，指环王3），2）奥斯卡获奖影片（例如指环王3，泰坦尼克，阿凡达），也可以是其它类别，如适合年轻人的和适合老年人的。构建的RBMs网络有如下图所示的拓扑结构。

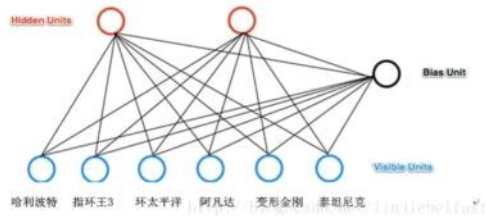

图3 2隐藏节点的RBMs网络

3.1 神经元的激活

RBMs和其它神经网络工作原理都是一样的，都是通过网络中神经元状态的变化，即通过在已知一些状态情况下，更新一些状态（例如二元状态有激活和未激活变更），来工作如分类、抽象、预测等任务，实现一动力系统。因而，这里我们重点讨论下RBMs网络中单个神经元状态是如何改变的过程。假设我们已知所构建的RBMs网络的连接权重W，状态单元i将通过下面的过程进行计算更新。(公式用latex表示)

1）计算第i个神经元的激励能量a_i=\sum_j w_{ij} x_j 。这里的求和计算了所有的连接到神经元i的第个j神经单元信息。w_{ij}是单元i和单元j的连接权重，x_j是第j个神经元的状态，这里因为假设是二元单元，因此只能为0或1。该激励能量可以理解为所有连接到神经元i的节点j都将自己的信息传递给神经元i，能量为这些信息之和。

2）记p_i=\sigma(a_i)。\sigma(x)=1/(1+exp(-x))为逻辑函数（为什么是\sigma(.)后面将讲到）。可以看到对于大的正激励能量a_i，p_i趋近于1,反之对于大的负激励能量，p_i趋近于0.

3）我们然后根据概率p_i，更新神经元i的状态为1，和以概率1-p_i将该单元的状态设置为0。

同样用以上所举例的电影例子进行列举，并假设我们所构建两个隐藏单元就是表示优秀影片和奥斯卡获奖影片。

- 如果小明告诉我们他观看电影的六个二元输入参数（例如111000），然后我们可以通过事先训练好的的RBMs推测哪个隐藏单元被激活（也就是，通过RBMs的隐藏单元状态来解释小明的喜好/选择是优秀影片还是奥斯卡获奖影片）。因而小明对这六个电影的选择给隐藏单元传递了信息，告诉他们自己状态如何更新。注意到，尽管小明声明他选择观看哈利波特、变形金刚和指环王3，这也并不保证"优秀影片"的隐藏单元始终被置为激活，只能说该状态设置为1的可能性是高概率的。这个可以用现实世界的情形来理解，在真实世界中小明想看这三部电影，使得我们可以推测他喜欢看优秀电影，但是也有小的可能性他可能是其它原因导致他想去看这三部电影。因此RBMs提供了我们对真实世界的一种建模的能力。
- 反过来看，如果我们知道一个人喜欢看"优秀影片"（即代表优秀影片的隐藏单元处于激活状态），我们然后能够通过RBMs，推知那些电影是这个人所愿意选择观看的。也就是让RBMs产生对电影的一种推荐。所以隐藏单元传送信息给每个电影观测单元，告诉他们如何更新他们的状态。再次注意到，如果"优秀影片"隐藏单元处于激活状态，并不能完全保证我们总是想看哈利波特、变形金刚和指环王3。

3.2 权重的学习

目录下"这一步具体怎么执行呢？需要下载吗？

我们如何训练/学习网络的连接权重W？假设我们有一批训练样本，每个训练样本是一个二元矢量，代表一个人对6部电影观看的选择与否（例：100101）。RBMs通过一个循环训练过程进行不断的权重更新，到达学习的目的，循环过程直至网络收敛为止。每次循环训练过程如下

1）获取一个训练样本，根据训练样本，设置网络的观测单元值。

2）用上述描述的逻辑激励策略更新隐藏单元状态：对于第j个隐藏单元，计算它的激励能量a_j=\sum_i w_{i,j} x_i，并根据概率\sigma(a_j)设置第j个隐藏单元为1和1-\sigma(a_j)概率设置为0，即第j个单元的状态是满足概率\sigma(a_j)的样本值。然后计算每条边e){ij} ，Positive(e_[ij})=x_i*x_j（也就是评估同时为激活的单元对）。

3）然后用类似的方法，重构每个观测单元，计算他的激励能量a_i，并更新他的状态。（注意，重构出的观测单元矢量可能和输入的观测单元矢量是有差别的）。然后再次更新隐藏单元状态，并对每条边计算Negative(e_{i,j})=x_i*x_j。

4）更新每条边e_{ij}的权重。w_{ij}=w_{ij}+L*(Positive(e_{ij})-Negative(e_ij}))。这里L是更新率（或称为更新步长）。

5）对每个训练样本重复以上操作。

该循环观测直到网络收敛为止。也就是训练样本和重构样本之差小于一个门限时，或达到某个最大循环次数时。为什么以上的更新是合理的？有如下理解

- 在第一个过程，Positive(e_ij})评价了通过训练样本，学习得到的第i和第j个单元间的关联，如果两个都为1，这表示该观测单元信息支撑了隐藏单元。
- 在重构过程，RBMs基于他现有的模型和隐藏单元状态预测了观测单元的值。Negative(e_{ij})评价了RBMs网络的预测能力。例如如果输入为1，通过重构过程重构的预测值也为1，则表示预测正确。

所以通过为每条边增加Positive(e_{ij})-Negative(e_ij})，我们想希望该网络有更好的预测能力，即预测结果与实际输入训练样本更加匹配/一致。例如，如果预测不一致，Negative(e_ij}会较大，Positive(e_{ij})-Negative(e_ij})较小或为负值，这表明该条边建立的单元i和j的关系是不可靠的，应该减小i和j的关系，即边的权重。这个就是前面在讲Deep Learning训练时谈到的Weak 和Sleep过程。

4、Restricted Boltzamnn Machines与Deep learning

为什么RBMs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$p(h|v)=p(h_1|v)\ldots p(h_n|v)$。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v1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。

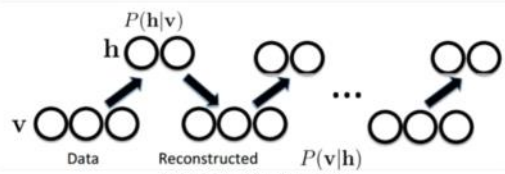

图4 RBMs的抽象与重构

5、形式化表示

下面我们将形式化/公式化的对RBMs进行描述。

5.1 基于能量的模型

同 Hopfield网络一样，RBMs网络也是基于能量模型研究的一类动力系统。基于能量的模型通过变化一能量函数，使得其网络达到一种理想的形态，例如最低能量或稳态。

能量函数是从动力系统中引入的概念。一个事物有相应的稳态，如在一个碗内的小球会停留在碗底，即使受到扰动偏离了碗底，在扰动消失后，它会回到碗底。学过物理的人都知道，稳态是它势能最低的状态。因此稳态对应与某一种能量的最低状态。将这种概念引用到神经网络中，构造了一种能量函数的定义。引进能量函数概念可以进一步加深对这一类动力系统性质的认识，可以把求稳态变成一个求极值与优化的问题，即可以理解为求能量函数最小值即网络的稳态，或动力系统的最优值。动力系统是一数学理论分支，如何构建动力系统的能量函数要学习的内容太多，要讲明白得开一门课，这里我们无法一 一道来，所我我们就先认定能量函数就是这个定义就行了。

通过能量函数，基于能量理论的概率模型定义了以下关于观测x的概率分布。

关闭

$$P(x) = \frac{e^{-\text{Energy}(x)}}{Z}.$$

(1)

很多时候，我们想引入一些隐藏单元增加模型的表现能力。因而定义以下的观测x和隐藏单元h的联合概率分布。

$$P(x,h) = \frac{e^{-\text{Energy}(x,h)}}{Z}$$

(2)

关于观测x的边沿概率为

$$P(x) = \sum_h \frac{e^{-\text{Energy}(x,h)}}{Z}.$$

(3)

很多情况下我们只关注条件概率。根据概率论定论，有P(h|x)=P(x,h)/P(x).因此

$$P(h \mid x) = \frac{e^{-Energy(x,h)}}{\sum_y e^{-Energy(x,h)}}$$

(4)

设θ为该能量模型的参数集，该能量模型的训练过程，将不断向着能量减少梯度方向变化参数集θ，最终达到势能最低即稳态，其能量变化趋势是对θ的导数有

$$\begin{aligned}
\frac{\partial \log P(x)}{\partial \theta} &= \frac{\partial \log \sum_h e^{-\text{Energy}(x,h)}}{\partial \theta} - \frac{\partial \log \sum_{x,h} e^{-\text{Energy}(x,h)}}{\partial \theta} \\
&= -\frac{1}{\sum_h e^{-\text{Energy}(x,h)}} \sum_h e^{-\text{Energy}(x,h)} \frac{\partial \text{Energy}(x,h)}{\partial \theta} \\
&\quad + \frac{1}{\sum_{x,h} e^{-\text{Energy}(x,h)}} \sum_{x,h} e^{-\text{Energy}(x,h)} \frac{\partial \text{Energy}(x,h)}{\partial \theta} \\
&= -\sum_h P(h|x) \frac{\partial \text{Energy}(x,h)}{\partial \theta} + \sum_{x,h} P(x,h) \frac{\partial \text{Energy}(x,h)}{\partial \theta}.
\end{aligned}$$

(5)

要求解（5），需要求解P(h|x),P(x,h),和能量函数对θ的导数。当跟定一个网络的能量函数时，求能量函数对θ的导数可以通过普通的求导过程（大学数学有讲）进行求解，因此比较简单。P(h|x),P(x,h)可以由（2）和（4）进行求得。为了求解(2),(4)和量函数对θ的导数，这里需要具体的x,h数值。注意到随机神经网络中单元x,h的值都是按照一定概率分布的样本值，因此需要按照一定概率分布，即P(h|x)和P(x|h)对x,h进行采样。在RBMs的训练中Hinton等采用了Gibbs sampling方法进行这里的x和h的采样，具体Gibbs sampling 请看MCMC & gibbs sampling。

5.2 Restricted Boltzamnn Machines

5.2.1 能量函数

RBMs的能量函数定义为

$$\text{Energy}(x,h) = -b'x - c'h - h'Wx$$

(6)

b为观测单元x的偏移，c为隐藏单元h的偏移，W为权重矩阵。

根据该能量函数带入公式（4），计算其P(h|x)为

$$\begin{aligned}
P(h|x) &= \frac{\exp(b'x + c'h + h'Wx)}{\sum_{\tilde{h}} \exp(b'x + c'\tilde{h} + \tilde{h}'Wx)} \\
&= \frac{\prod_i \exp(c_i h_i + h_i W_i x)}{\prod_i \sum_{\tilde{h}_i} \exp(c_i \tilde{h}_i + \tilde{h}_i W_i x)} \\
&= \prod_i \frac{\exp(h_i(c_i + W_i x))}{\sum_{\tilde{h}_i} \exp(\tilde{h}_i(c_i + W_i x))} \\
&= \prod_i P(h_i|x).
\end{aligned}$$

(7)

假设我们采用二元神经元，即取值为{0,1}两值的神经元构建RBMs，将能量函数（6），带入（7），计算隐藏单元h=1，即激活状态的输入概率为

$$P(h_i = 1|x) = \frac{e^{c_i + W_i x}}{1 + e^{c_i + W_i x}} = \text{sigm}(c_i + W_i x). \quad (8)$$

同理当给定h时x=1的概率值计算公式推导为如下

$$P(x_j = 1|h) = \text{sigm}(b_j + W'_j \cdot h) \quad (9)$$

5.2.2 用 Gibbs sampling 实现 RBM 训练

同样根据（5）式计算其网络模型参数变化值，并采用Gibbs sampling进行x和h的采样，二元RBMs的更新算法描述如下：

Algorithm 1

RBMupdate(x_1, ϵ, W, b, c)
This is the RBM update procedure for binomial units. It can easily adapted to other types of units.
x_1 is a sample from the training distribution for the RBM
ϵ is a learning rate for the stochastic gradient descent in Contrastive Divergence
W is the RBM weight matrix, of dimension (number of hidden units, number of inputs)
b is the RBM biases vector for hidden units
c is the RBM biases vector for input units

for all hidden units i **do**
- compute $Q(\mathbf{h}_{1i} = 1|x_1)$ (for binomial units, $\text{sigm}(b_i + \sum_j W_{ij} x_{1j})$)
- sample $\mathbf{h}_{1i}$ from $Q(\mathbf{h}_{1i}|x_1)$

end for
for all visible units j **do**
- compute $P(x_{2j} = 1|\mathbf{h}_1)$ (for binomial units, $\text{sigm}(c_j + \sum_i W_{ij} \mathbf{h}_{1i})$)
- sample x_{2j} from $P(x_{2j} = 1|\mathbf{h}_1)$

end for
for all hidden units i **do**
- compute $Q(\mathbf{h}_{2i} = 1|x_2)$ (for binomial units, $\text{sigm}(b_i + \sum_j W_{ij} x_{2j})$)

end for
- $W \leftarrow W + \epsilon(\mathbf{h}_1 x'_1 - Q(\mathbf{h}_{2\cdot} = 1|x_2)x'_2)$
- $b \leftarrow b + \epsilon(\mathbf{h}_1 - Q(\mathbf{h}_{2\cdot} = 1|x_2))$
- $c \leftarrow c + \epsilon(x_1 - x_2)$

References：

https://github.com/echen/restricted-boltzmann-machines

http://blog.csdn.net/zouxy09/article/details/8781396

Y．Bengio, Learningdeep architectures for AI, Foundations and Trends in Machine Learning 1(2)pages 1-127.

- Deep Learning 训练/学习... 2013-12-05 阅读 1194
- 对比传统模式识别方法理解... 2013-12-05 阅读 2189
- DNN与ANN的区别 2013-12-05 阅读 2618
- 机器学习基础unsupervise... 2013-12-05 阅读 2400

更多文章

参考知识库

.NET知识库
3846 关注 | 839 收录

算法与数据结构知识库
16073 关注 | 2320 收录

猜你在找

机器学习案例实战第四课-神经网络模型
python数据分析与机器学习实战
Python数据分析（机器学习）经典案例实战
Python数据分析实战：泰坦尼克号之灾与机器学习算法
统计机器学习入门——线性模型选择与正则化1
机器学习中的神经网络Neural Networks for Machine
训练Restricted Boltzmann MachineRBM十五个问题
受限玻尔兹曼机Restricted Boltzmann Machine RBM
Deeplearning5监测学习过程from A practical guide
Restricted Boltzman Machines for Collaborative

查看评论

暂无评论

您还没有登录,请[登录]或[注册]

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

全部主题 Hadoop AWS 移动游戏 Java Android iOS Swift 智能硬件 Docker
OpenStack VPN Spark ERP IE10 Eclipse CRM JavaScript 数据库 Ubuntu NFC
WAP jQuery BI HTML5 Spring Apache .NET API HTML SDK IIS Fedora XML
LBS Unity Splashtop UML components Windows Mobile Rails QEMU KDE Cassandra
CloudStack FTC coremail OPhone CouchBase 云计算 iOS6 Rackspace Web App
SpringSide Maemo Compuware 大数据 aptech Perl Tornado Ruby Hibernate ThinkPHP
HBase Pure Solr Angular Cloud Foundry Redis Scala Django Bootstrap

关闭