

## Batch Normalization 导读

原创

2016年05月24日 19:08:07

标签: 深度学习 / BatchNorm / 梯度消失

44243

/\* 版权声明：可以任意转载，转载时请标明文章原始出处和作者信息 .\*/

author: 张俊林

Batch Normalization 作为最近一年来 DL 的重要成果，已经广泛被证明其有效性和重要性。目前几乎已经成为 DL 的标配，任何有志于学习 DL 的同学们朋友们雷迪斯俺的詹特曼们都应该好好学一学 BN。BN 倒过来看就是 NB，从这个技术确实很 NB，虽然有些细节处理还解释不清其理论原因，但是实践证明好用才是真的好，别忘了 DL 的鼻祖 Geoffrey Hinton 对深层网络做 Pre-Train 开始就是一个经验领先于理论分析的偏经验的一门学问。

如何理解 BatchNorm? 请参考论文: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift。因为对于部分基础不是太好的同学们朋友们雷迪斯俺的詹特曼们可能阅读理解有一定障碍，所以本文是为了更容易理解 BN 而做的一番导读。由于本人水平也很有限，假设导游导错了路，那么..... 您就认倒霉好了，毕竟这是免费的导游您说是不，期望别太高，“任何对其它人或者事物报以极高期望的人是这个世界上最不幸福的人”，这是出自我的非名人的名言，所以“降低期望是通向幸福之路”，这也是我的名言。

机器学习领域有个很重要的假设：IID 独立同分布假设，就是假设训练数据和测试数据是满足相同分布的，这是通过训练数据获得的模型能够在测试集获得好的效果的一个基本保障。而 BatchNorm 是干啥的呢？BatchNorm 就是在深度神经网络训练过程中使得每一层神经网络的输入保持相同分布的。OK，BN 讲完了，再见。

嗯，这么讲步子迈得有点大，我们放慢脚步，把学习率调小一点，一步一步接近理解 BN 的最优解。

为什么深度神经网络随着网络深度加深，训练起来越困难，收敛越来越慢？这是个在 DL 领域很接近本质的好问题。很多论文都是解决这个问题的，比如 ReLU 激活函数，再比如 Residual Network，BN 本质上也是解释并从某个不同的角度来解决这个问题的。

### | “Internal Covariate Shift” 问题

从论文名字可以看出，BN 是用来解决 “Internal Covariate Shift” 问题的，那么首先得理解什么是 “Internal Covariate Shift”？

论文首先说明 Mini-Batch SGD 相对于 One Example SGD 的两个优势：梯度更新方向更准确；并行计算速度快；（本文作者：为啥要说这些？因为 BatchNorm 是基于 Mini-Batch SGD 的，所以先夸下 Mini-Batch SGD，当然也是大实话）；

然后吐槽下 SGD 训练的缺点：超参数调起来很麻烦。（本文作者：作者隐含意思是用我大 BN 就能解决很多 SGD 的缺点：用了大 BN，妈妈再也不用担心我的调参能力啦）

接着引入 covariate shift 的概念：如果 ML 系统实例集合 <X,Y> 中的输入值 X 的分布老是变，这不符合 IID 假设啊，那您怎么让我稳定的学规律啊，这不得引入迁移学习才能搞定吗，我们的 ML 系统还得去学习怎么迎合这种分布变化啊。

对于深度学习这种包含很多隐层的网络结构，在训练过程中，因为各层参数老在变，所以每个隐层都会面临 covariate shift 的问题，也就是在训练过程中，隐层的输入分布老是变来变去，这就是所谓的 “Internal Covariate Shift”，Internal 指的是深层网络的隐层，是发生在网络内部的事情，而不是 covariate shift 问题只发生在输入层。

然后提出了 BatchNorm 的基本思想：能不能让每个隐层节点的激活输入分布固定下来呢？这样就避免了 “Internal Covariate Shift” 问题了。

BN 不是凭空拍脑袋拍出来的好点子，它是有启发来源的：之前的研究表明如果在图像处理中对输入图像进行白化（Whiten）操作的话——所谓白化，就是对输入数据分布变换到 0 均值，单位方差的正态分布——那么神经网络会较快收敛，那么 BN 作者就开始推论了：图像是深度神经网络的输入层，做白化能加快收敛，那么其实对于深度网络来说，其中某个隐层的神经元是下一层的输入，意思是其实深度神经网络的每一个隐层都是输入层，不过是相对下一层来说而已，那么能不能对每个隐层都做白化呢？这就是启发 BN 产生的原初想法，而 BN 也确实就是这么做的，可以理解为对深层神经网络每个隐层神经元的激活值做简化版本的白化操作。

### | BatchNorm 的本质思想

BN 的基本思想其实相当直观：因为深层神经网络在做非线性变换前的激活输入值（就是那个  $x=WU+B$ ，U 是输入）随着网络深度加深或者在训练过程中，其分布逐渐发生偏移或者变动，之所以训练收敛慢，一般是整体分布逐渐往非线性函数的取值区间的上下限两端靠近（对于 Sigmoid 函数来说，意味着激活输入值  $WU+B$  是大的负值或正值），所以这导致后向传播时低层神经网络的梯度消失，这是训练深层神经网络收敛越来越慢的本质原因，而 BN 就是通过一定的规范化手段，把每层神经网络任意神经元这个输入值的分布强行拉回到均值为 0 方差为 1 的标准正太分布而不是萝莉分布（哦，是正态分布），其实就是把越来越偏的分布强制拉回比较标准的分布，这样使得激活输入值落在非线性函数对输入比较敏感的区域，这样输入的小变化就会导致损失函数较大的变化，意思是这样让梯度变大，避免梯度消失问题产生，而且梯度变大意味着学习收敛速度快，能大大加快训练速度。

THAT'S IT。其实一句话就是：对于每个隐层神经元，把逐渐向非线性函数映射后向取值区间极限饱和和区靠拢的输入分布强制拉回到均值为 0 方差为 1 的比较标准的正态分布，使得非线性变换函数的输入值落入对输入比较敏感的区域，以此避免梯度消失问题。因为梯度一直都能保持比较大的状态，所以很明显对神经网络的参数调整效率比较高，就是变动大，就是说向损失函数最优值迈动的步子大，也就是说收敛地快。NB 说到底就是这么个机制，方法很简单，道理很深刻。



张俊林博客

博客专家

原创	粉丝	喜欢	评论
112	2112	372	445

等级: 博客 6	访问量: 111万+
积分: 9367	排名: 2424

#### 广告

博主最新文章	更多文章
2017年AI技术盘点：关键进展与趋势	
通过不断重置学习率来逃离局部极值点	
深度学习中的注意力机制(2017版)	
机器码农：深度学习自动编程	
极深网络（ResNet/DenseNet）：Skip Connection为何有效及其它	

### | 文章分类

搜索引擎	36篇
自然语言处理	33篇
大数据	7篇
深度学习	33篇
随笔	21篇
SALSA算法	1篇
展开	

### | 文章存档

2018年2月	1篇
2017年12月	2篇
2017年6月	1篇
2017年3月	2篇
2016年11月	1篇
2016年10月	2篇
展开	

### | 博主热门文章

自然语言处理中的Attention Model：是什么及为什么	69949
机器码农：深度学习自动编程	47408
Batch Normalization 导读	44166
深度学习与自然语言处理之五：从RNN到LSTM	43402
深度学习在搜索和推荐领域的应用	39805
自然语言处理中CNN模型几种常见的Max Pooling操作	37694
使用深度学习打造智能聊天机器人	28329
使用Word Embedding构造简洁有效的文本摘要系统	25516
深度学习计算模型中“门函数（Gating Function）”的作用	23920
搜索引擎索引之如何建立索引	23907

### 联系我们

请扫描二维码联系客服

webmaster@csdn.net

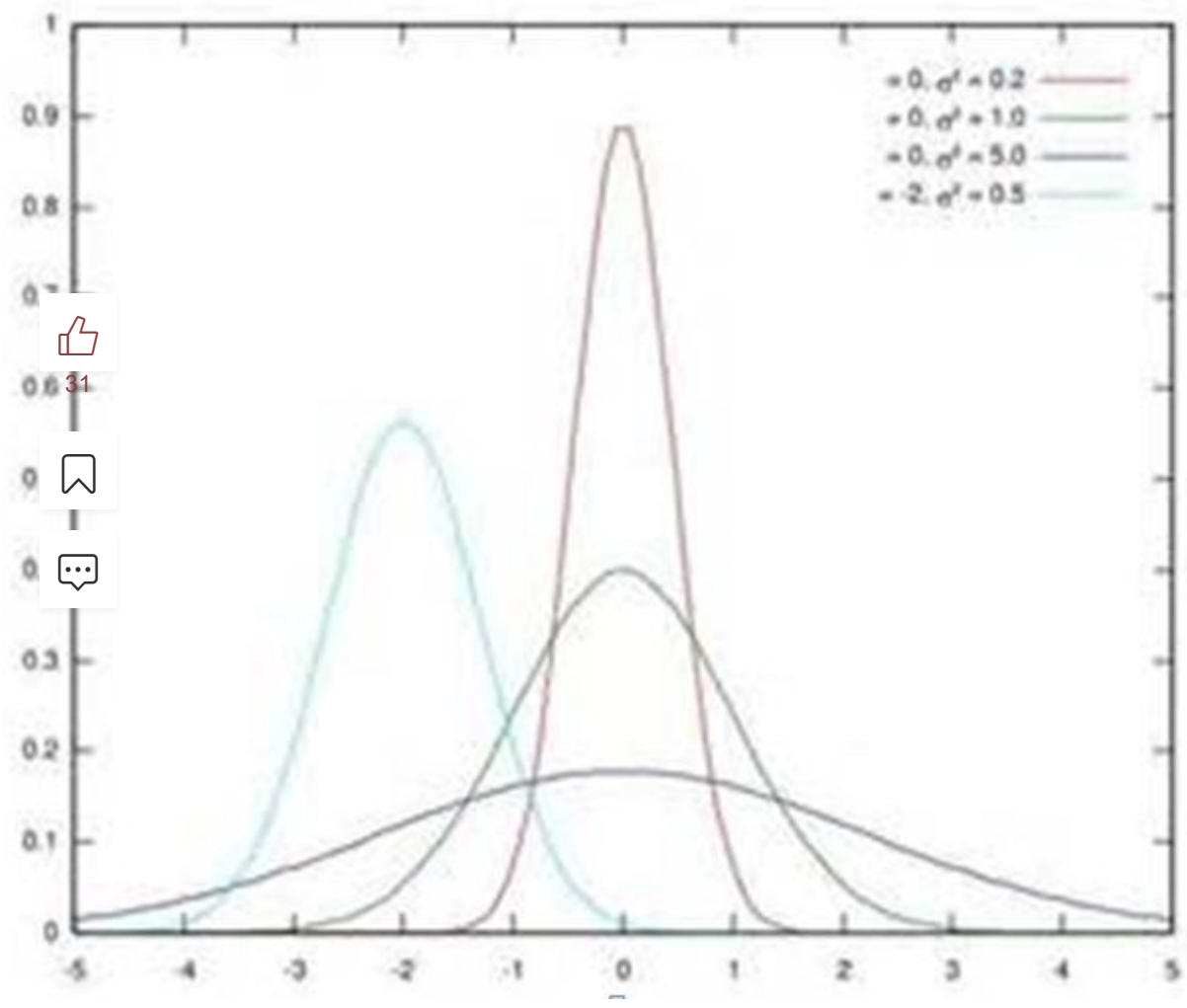


图1. 几个正态分布

假设某个隐层神经元原先的激活输入 $x$ 取值符合正态分布，正态分布均值是-2，方差是0.5，对应上图中最左端的浅蓝色曲线，通过BN后转换为均值为0，方差是1的正态分布（对应上图中的深蓝色图形），意味着什么，意味着输入 $x$ 的取值正态分布整体右移2（均值的变化），图形曲线更平缓了（方差增大的变化）。这个图的意思是，BN其实就是把每个隐层神经元的激活输入分布从偏离均值为0方差为1的正态分布通过平移均值压缩或者扩大曲线尖锐程度，调整为均值为0方差为1的正态分布。

那么把激活输入 $x$ 调整到这个正态分布有什么用？

首先我们看下均值为0，方差为1的标准正态分布代表什么含义：

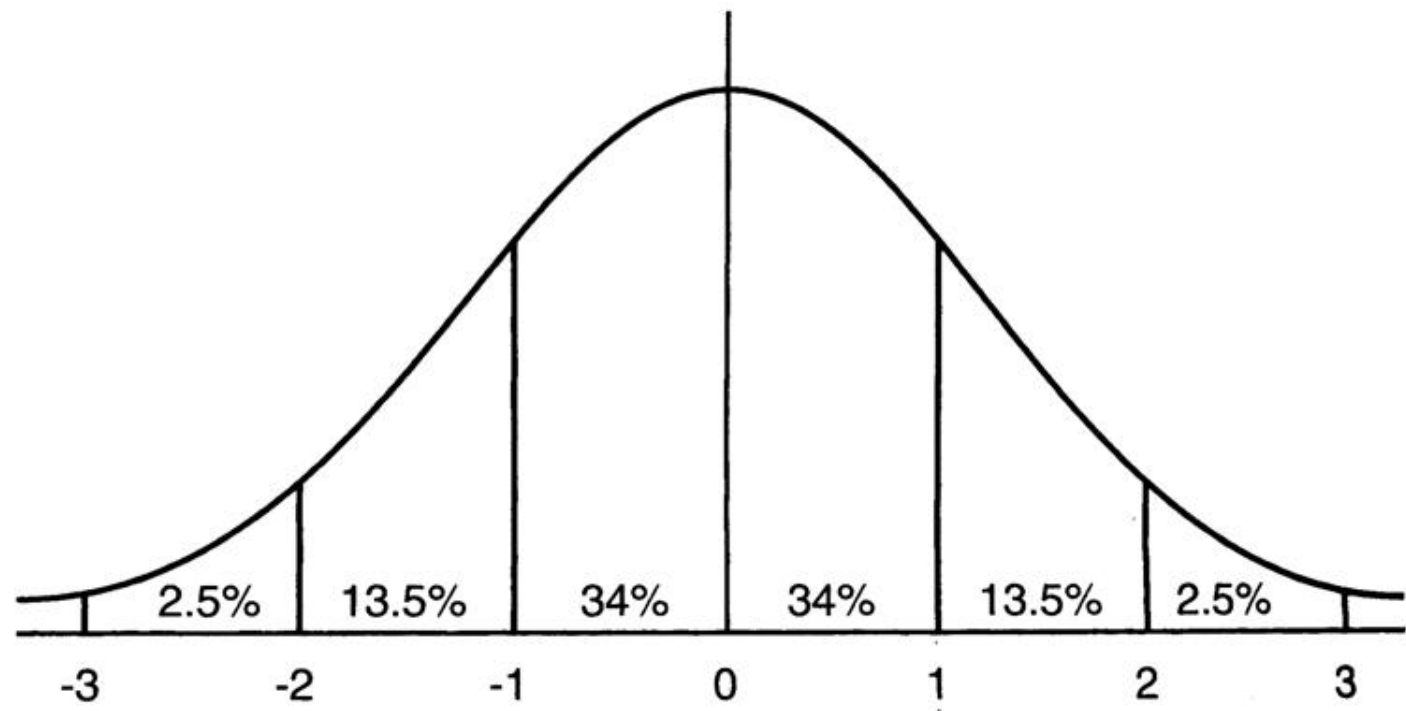


图2.均值为0方差为1的标准正态分布图

这意味着在一个标准差范围内，也就是说64%的概率 $x$ 其值落在 $[-1,1]$ 的范围内，在两个标准差范围内，也就是说95%的概率 $x$ 其值落在了 $[-2,2]$ 的范围内。那么这又意味着什么？我们知道，激活值 $x=WU+B$ , $U$ 是真正的输入， $x$ 是某个神经元的激活值，假设非线性函数是sigmoid，那么看下sigmoid( $x$ )其图形：

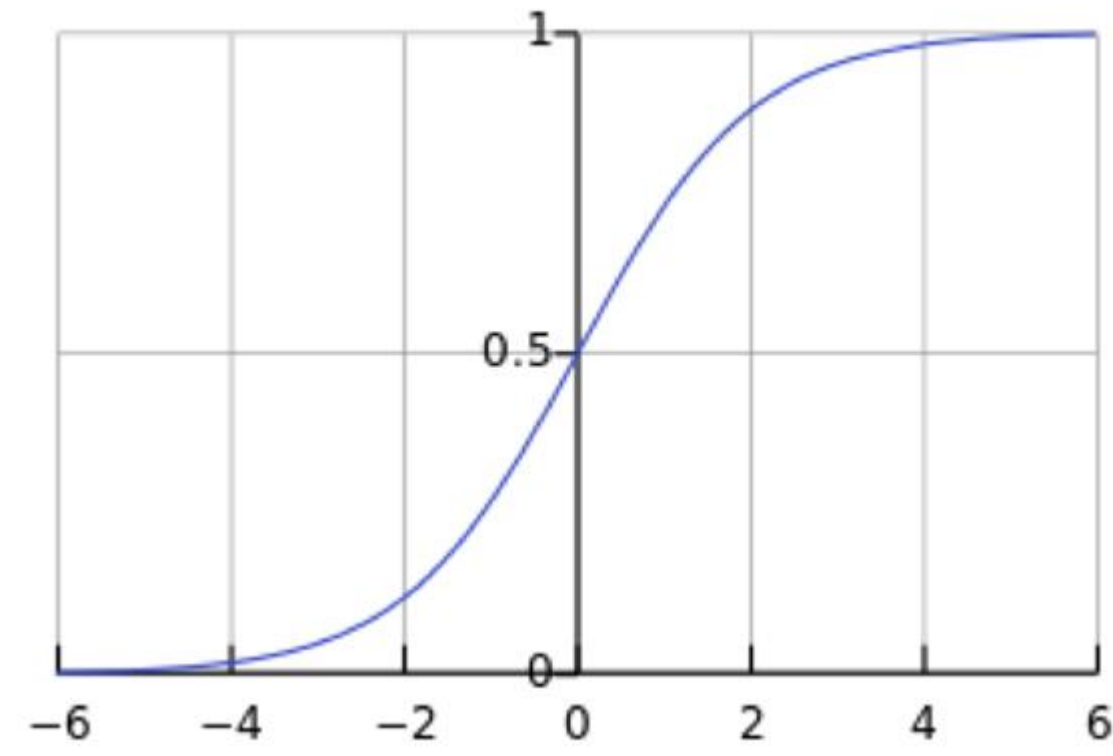


图3. Sigmoid( $x$ )

及sigmoid( $x$ )的导数为： $G'=f(x)*(1-f(x))$ ，因为 $f(x)=\text{sigmoid}(x)$ 在0到1之间，所以 $G'$ 在0到0.25之间，其对应的图如下：



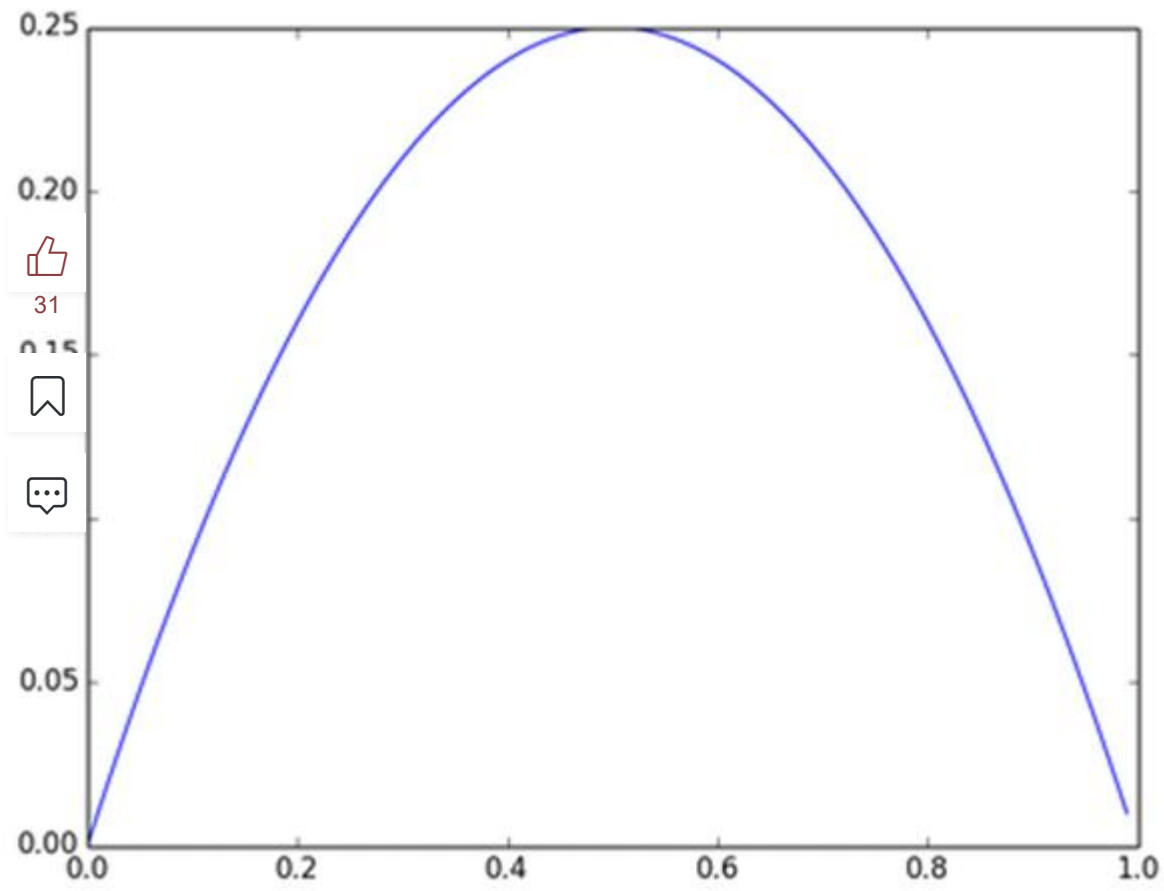


图4. Sigmoid(x)导数图

假设没有经过BN调整前x的原先正态分布均值是-6，方差是1，那么意味着95%的值落在了[-8,-4]之间，那么对应的Sigmoid（x）函数的值明显接近于0，这是典型的梯度饱和区，在这个区域里梯度变化很慢，为什么是梯度饱和区？请看下sigmoid(x)如果取值接近0或者接近于1的时候对应导数函数取值，接近于0，意味着梯度变化很小甚至消失。而假设经过BN后，均值是0，方差是1，那么意味着95%的x值落在了[-2,2]区间内，很明显这一段是sigmoid(x)函数接近于线性变换的区域，意味着x的小变化会导致非线性函数值较大的变化，也即是梯度变化较大，对应导数函数图中明显大于0的区域，就是梯度非饱和区。

从上面几个图应该看出来BN在干什么了吧？其实就是把隐层神经元激活输入 $x=WU+B$ 从变化不拘一格的正态分布通过BN操作拉回到了均值为0，方差为1的正态分布，即原始正态分布中心左移或者右移到以0为均值，拉伸或者缩减形态形成以1为方差的图形。什么意思？就是说经过BN后，目前大部分Activation的值落入非线性函数的线性区内，其对应的导数远离导数饱和区，这样来加速训练收敛过程。

但是很明显，看到这里，稍微了解神经网络的读者一般会提出一个疑问：如果都通过BN，那么不就跟把非线性函数替换成线性函数效果相同了？这意味着什么？我们知道，如果是多层的线性函数变换其实这个深层是没有意义的，因为多层线性网络跟一层线性网络是等价的。这意味着网络的表达能力下降了，这也意味着深度的意义就没有了。所以BN为了保证非线性的获得，对变换后的满足均值为0方差为1的x又进行了scale加上shift操作( $y=scale*x+shift$ )，每个神经元增加了两个参数scale和shift参数，这两个参数是通过训练学习到的，意思是通过scale和shift把这个值从标准正态分布左移或者由移一点并长胖一点或者变瘦一点，每个实例挪动的程度不一样，这样等价于非线性函数的值从正中心周围的线性区往非线性区动了动。核心思想应该是想找到一个线性和非线性的较好平衡点，既能享受非线性的较强表达能力的好处，又避免太靠非线性区两头使得网络收敛速度太慢。当然，这是我的理解，论文作者并未明确这样说。但是很明显这里的scale和shift操作是会有争议的，因为按照论文作者论文里写的理想状态，就会又通过scale和shift操作把变换后的x调整回未变换的状态，那不是饶了一圈又绕回去原始的“Internal Covariate Shift”问题里去了吗，感觉论文作者并未能够清楚地解释scale和shift操作的理论原因。

|训练阶段如何做BatchNorm

上面是对BN的抽象分析和解释，具体在Mini-Batch SGD下做BN怎么做？其实论文里面这块写得很清楚也容易理解。为了保证这篇文章完整性，这里简单说明下。

假设对于一个深层神经网络来说，其中两层结构如下：

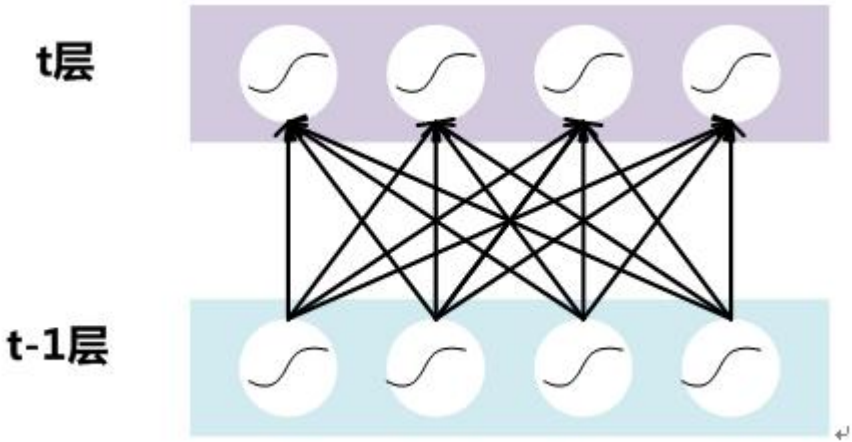
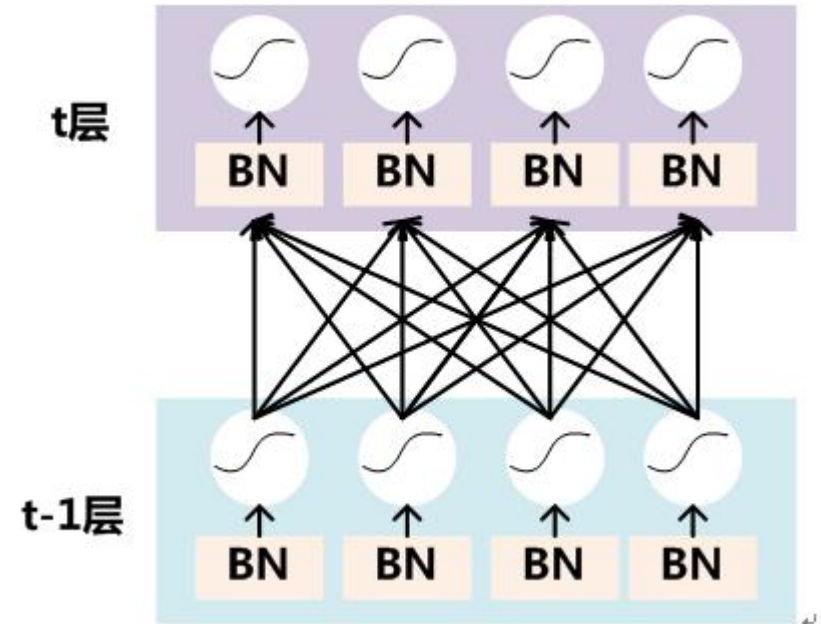


图5. DNN其中两层

要对每个隐层神经元的激活值做BN，可以想象成每个隐层又加上了一层BN操作层，它位于 $X=WU+B$ 激活值获得之后，非线性函数变换之前，其图示如下：



对于Mini-Batch SGD来说，一次训练过程里面包含m个训练实例，其具体BN操作就是对于隐层内每个神经元的激活值来说，进行如下变换：

$$\widehat{x}^{(k)} = \frac{x^{(k)} - \mathbb{E}[x^{(k)}]}{\sqrt{\text{Var}[x^{(k)}]}}$$

要注意，这里t层某个神经元的x(k)不是指原始输入，就是说不是t-1层每个神经元的输出，而是t层这个神经元的激活值，即U+B，这里的U才是t-1层神经元的输出。

变换的意思是：某个神经元对应的原始的激活x通过减去mini-Batch内m个实例获得的m个激活x求得的均值E(x)并除以求得的方差Var(x)来进行转换。

上文说过经过这个变换后某个神经元的激活x形成了均值为0，方差为1的正态分布，目的是把值往后续要进行的非线性变换的线性区拉动，增大导数值，增强反向传播信息流动性，加快训练收敛速度。但是这样会导致网络表达能力下降，为了防止这一点，每个神经元增加两个调节参数（scale和shift），这两个参数是通过训练来学习到的，用来对变换后的激活反变换，使得网络表达能力增强，即对变换后的激活进行如下的scale和shift操作，这其实是变换的反操作：

$$y^{(k)} = \gamma^{(k)}\widehat{x}^{(k)} + \beta^{(k)}.$$

BN其具体操作流程，如论文中描述的一样：

**Input:** Values of  $x$  over a mini-batch:  $\mathcal{B} = \{x_1 \dots x_m\}$ ;  
Parameters to be learned:  $\gamma, \beta$   
**Output:**  $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{ mini-batch mean}$$
$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{ mini-batch variance}$$
$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{ normalize}$$
$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{ scale and shift}$$

过程非常清楚，就是上述公式的流程化描述，这里不解释了，直接应该能看懂。

|BatchNorm的推理过程

BN在训练的时候可以根据Mini-Batch里的若干训练实例进行激活数值调整，但是在推理（inference）的过程中，很明显输入就只有一个实例，看不到Mini-Batch其它实例，那么这时候怎么对输入做BN呢？因为很明显一个实例是没法算实例集合求出的均值和方差的。这可如何是好？这可如何是好？这可如何是好？

既然没有从Mini-Batch数据里可以得到的统计量，那就想其它办法来获得这个统计量，就是均值和方差。可以用从所有训练实例中获得的统计量来代替Mini-Batch里面m个训练实例获得的均值和方差统计量，因为本来打算用全局的统计量，只是因为计算量等太大所以才会用Mini-Batch这种简化方式的，那么在推理的时候直接用全局统计量即可。

决定了获得统计量的数据范围，那么接下来的问题是如何获得均值和方差的问题。很简单，因为每次做Mini-Batch训练时，都会有那个Mini-Batch里m个训练实例获得的均值和方差，现在要全局统计量，只要把每个Mini-Batch的均值和方差统计量记住，然后对这些均值和方差求其对应的数学期望即可得出全局统计量，即：

$$\mathbb{E}[x] \leftarrow \mathbb{E}_{\mathcal{B}}[\mu_{\mathcal{B}}]$$
$$\text{Var}[x] \leftarrow \frac{m}{m-1} \mathbb{E}_{\mathcal{B}}[\sigma_{\mathcal{B}}^2]$$

有了均值和方差，每个隐层神经元也已经有对应训练好的Scaling参数和Shift参数，就可以在推导的时候对每个神经元的激活数据计算NB进行变换了，在推理过程中进行NB采取如下方式：

$$y = \frac{\gamma}{\sqrt{\text{Var}[x] + \epsilon}} \cdot x + \left( \beta - \frac{\gamma \mathbb{E}[x]}{\sqrt{\text{Var}[x] + \epsilon}} \right)$$

这个公式其实和训练时

$$y^{(k)} = \gamma^{(k)}\widehat{x}^{(k)} + \beta^{(k)}.$$

是等价的，通过简单的合并计算推导就可以得出这个结论。那么为啥要写成这个变换形式呢？我猜作者这么写的意思是：在实际运行的时候，按照这种变体形式可以减少计算量，为啥呢？因为对于每个隐层节点来说：

$$\frac{\gamma}{\sqrt{\text{Var}[x] + \epsilon}} \quad \frac{\gamma \mathbb{E}[x]}{\sqrt{\text{Var}[x] + \epsilon}}.$$

都是固定值。这样这两个值可以先计算好存起来。在推理的时候直接用就行了。这样比原始的公式每一步骤都



|BatchNorm的好处

BatchNorm为什么NB呢，关键还是效果好。不仅仅极大提升了训练速度，收敛过程大大加快，还能增加分类效果，一种解释是这是类似于Dropout的一种防止过拟合的正则化表达方式，所以不用Dropout也能达到相当的效果。另外调参过程也简单多了，对于初始化要求没那么高，而且可以使用大的学习率等。总而言之，经过这么简单的替换，带来的好处多得很，这也是为何现在BN这么快流行起来的原因。



扫一扫关注微信号：“布洛卡区”，深度学习在自然语言处理等智能应用的技术研讨与科普公众号。

目前您尚未登录，请 登录 或 注册 后参与评论

- wc996789331

2018-03-09 20:16

#19楼

深度好文

回复
- wexplorer

2018-01-18 18:11

#18楼

打扰了， $\gamma$ 和 $\beta$ 先由 $X(k)$ 的分布初始化，之后通过反向传播更新吗？谢谢！

回复
- wexplorer

2018-01-18 15:34

#17楼

NBNB

1条回复 回复
- 查看 22 条热评

深度学习中的Batch Normalization

whitesilence 2017年07月21日 17:06 9478

在看 ladder network(<https://arxiv.org/pdf/1507.02672v2.pdf>) 时初次遇到batch normalization（BN）. 文中说BN能加速收敛等好...

[深度学习] Batch Normalization算法介绍

lhanchao 2017年04月21日 11:34 6742

很早就打算写这篇博客了，最近遇到的问题比较多，所以拖了又拖，今天问题似乎解决了，等着程序运行的时候再来回顾一下Batch Normalization算法。Batch Normalization是2...

Batch Normalization 的原理解读

ZhikangFu 2016年11月29日 14:13 3796

1:motivation 作者认为：网络训练过程中参数不断改变导致后续每一层输入的分布也发生变化，而学习的过程又要使每一层适应输入的分布，因此我们不得不降低学习率、小心地初始化。作者将分布发生变化称...

深度学习（二十九）Batch Normalization 学习笔记

近年来深度学习捷报连连，声名鹊起，随机梯度下架成了训练深度网络的主流方法。尽管随机梯度下降法，将对于训练深度网络，简单高效，但是它有个毛病，就是需要我们人为的去选择参数，比如学习率、参数初始化等，这些...

hjimce 2016年03月12日 17:00 90906

Batch Normalization 学习笔记

leayc 2017年08月28日 11:31 984

本文是对批标准化方法的一则学习笔记

论文笔记-Batch Normalization

u012816943 2016年06月17日 11:30 7024

Batch Normalization：减弱 internal covariate shift，使训练加快，并且可以不再用dropout和LRN。...

关于Batch Normalization的另一种理解

Alchipmunk 2017年01月11日 16:00 4513

Batch Norm可谓深度学习中非常重要的技术，不仅可以使训练更深的网络变容易，加速收敛，还有一定正则化的效果，可以防止模型过拟合。在很多基于CNN的分类任务中，被大量使用。但我最近在图像超分辨...

解读Batch Normalization

elaine\_bao 2016年03月14日 22:22 37272

本文转载自：<http://blog.csdn.net/shuzfan/article/details/50723877> 本次所讲的内容为Batch Normalization，简称BN，来源于《

life\_is\_amazing

2016年07月06日 15:36

3020

Batch Normalization理解


原文地址：<http://blog.csdn.net/malefactor/article/details/51476961> 作者：张俊林 Batch Normalization作为...

shuzfan

2016年02月23日 16:03

14675

解读Batch Normalization

目录  I-Motivation 2-Normalization via Mini-Batch Statistics 测试 BN before or after Activation 3-Expe...

我读Batch Normalization

论文地址：Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift 论...

Batch Normalization简介

2017年11月24日 17:20 3.36MB 

下载

guoyuhaoaaa

2017年02月07日 10:39

581

batch-normalization 技术分析

这两天重新回顾了一下batch-normalization技术，主要参考了论文《Batch Normalization: Accelerating Deep Network Training by R...

pan5431333

2017年09月21日 15:47

1195

总结2: Batch Normalization反向传播公式推导及其向量化

1. 背景介绍上周学习了吴恩达的Deep Learning专项课程的第二门课，其中讲到了Batch Normalization（BN）。但在课程视频中，吴恩达并没有详细地给出Batch Normali...

meanme

2015年09月23日 14:47

18827

Batch Normalization 简单理解

1：背景由于在训练神经网络的过程中，每一层的 params是不断更新的，由于params的更新会导致下一层输入的分布情况发生改变，所以这就要求我们进行权重初始化，减小学习率。这个现象就叫做intern...

《Batch Normalization Accelerating Deep Network Training by Reducing Inte...

《Batch Normalization Accelerating Deep Network Training by Reducing Internal Covariate Shift》阅读笔记...

happynear

2015年03月13日 12:42

43022

qq\_34484472

2017年09月14日 16:38

936

Batch Normalization详解

Batch Normalization详细介绍

mao\_xiao\_feng

2017年01月10日 22:10

7504

【机器学习】covariate shift现象的解释

一、什么是covariate shift？ 在论文中经常碰到covariate shift这个词，网上相关的中文解释比较少 你可能会在介绍深度学习Batch Normalization方法的论文到中看...

Tcorpion

2018年01月04日 19:35

324

Tensorflow中dropout和batch normalization

直接上代码探讨：

```
def batch_norm_layer(x, train_phase, scope_bn):##x is input-tensor, train_phase is tf.Varia...
```

u010402786

2016年04月24日 15:33

8383

深入浅出——深度学习中的Batch Normalization使用

《Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shi