Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift

Sergey Ioffe
Google Inc.
kriz@cs.utoronto.ca

Christian Szegedy
Google Inc.
ilya@cs.utoronto.ca

摘要

训练深度神经网络的复杂性在于，每层输入的分布在训练过程中会发生变化，因为前面的层的参数会发生变化。通过要求较低的学习率和仔细的参数初始化减慢了训练，并且使具有饱和非线性的模型训练起来非常困难。我们将这种现象称为内部协变量转移，并通过标准化层输入来解决这个问题。我们的方法力图使标准化成为模型架构的一部分，并为每个训练小批量数据执行标准化。批标准化使我们能够使用更高的学习率，并且不用太注意初始化。它也作为一个正则化项，在某些情况下不需要 Dropout。将批量标准化应用到最先进的图像分类模型上，批标准化在取得相同的精度的情况下，减少了 14 倍的训练步骤，并以显著的差距击败了原始模型。使用批标准化网络的组合，我们改进了在 ImageNet 分类上公布的最佳结果：达到了 4.9% top-5 的验证误差（和 4.8%测试误差），超过了人类评估者的准确性。

1 引言

深度学习在视觉、语音等诸多方面显著提高了现有技术的水平。随机梯度下降（SGD）已经被证明是训练深度网络的有效方式，并且已经使用诸如动量（Sutskever 等，2013）和 Adagrad（Duchi 等人，2011）等 SGD 变种取得了最先进的性能。SGD 优化网络参数Θ，以最小化损失。

$$\Theta = \text{argmin}_\theta \frac{1}{N}\sum_{i=1}^{N} \ell(x_i, \Theta)$$

$x_{1\ldots N}$是训练数据集。使用 SGD，训练将逐步进行，在每一步中，我们考虑一个大小为 m 的小批量数据$x_{1\ldots m}$。通过计算$\frac{1}{m}\sum_{i=1}^{m}\frac{\partial \ell(x_i,\Theta)}{\partial \Theta}$,使用小批量数据来近似损失函数关于参数的梯度。使用小批量样本，而不是一次一个样本，在一些方面是有帮助的。首先，小批量数据的梯度损失是训练集上的梯度估计，其质量随着批量增加而改善。第二，由于现代计算平台提供的并行性，对一个批次的计算比单个样本计算 m 次效率更高。

虽然随机梯度是简单有效的，但它需要仔细调整模型的超参数，特别是优化中使用的学习速率以及模型参数的初始值。训练的复杂性在于每层的输入受到前面所有层的参数的影响——因此当网络变得更深时，网络参数的微小变化就会被放大。

层输入的分布变化是一个问题，因为这些层需要不断适应新的分布。当学习系统的输入分布发生变化时，据说会经历协变量转移（Shimodaira, 2000）。这通常是通过域适应（Jiang, 2008）来处理的。然而，协变量漂移的概念可以扩展到整个学习系统之外，应用到学习系统的一部分，例如子网络或一层。考虑网络计算

$$\ell = F_2(F_1(u, \Theta_1), \Theta_2)$$

F_1和F_2是任意变换，学习参数Θ_1，Θ_2以便最小化损失ℓ。学习Θ_2可以看作输入 $x = F_1(u, \Theta_1)$送入到子网络

$$\ell = F_2(x, \Theta_2)。$$

例如，梯度下降步骤

$$\Theta_2 \leftarrow \Theta_2 - \frac{\alpha}{m}\sum_{i=1}^{m}\frac{\partial F_2(x_i,\Theta_2)}{\partial \Theta_2}$$

(对于批大小 m 和学习率α) 与输入为 x 的单独网络F_2完全等价。因此，输入分布特性使训练更有效–例如训练数据和测试数据之间有相同的分布–也适用于训练子网络。因此 x 的分布在时间上保持固定是有利的。然后，$9 不必重新调整来补偿 x$分布的变化。

子网络输入的固定分布对于子网络外的层也有积极的影响。考虑一个激活函数为 $g(x)=\frac{1}{1+\exp(-x)}$的层，u 是层输 入，权重矩阵 W 和偏置向量 b 是要学习的层参数，$g(x)=\frac{1}{1+\exp(-x)}$。随着|x|的增加，$g'(x)$趋向于 0。这意味着对于 $x=Wu+b$ 的所有维度，除了那些具有小的绝对值之外，流向 u 的梯度将会消失，模型将缓慢的进行训练然而，由于 x 受 W,b 和下面所有层的参数的影响,训练期间那些参数的改变可能会将 x 的许多维度移动到非线性的饱和状态并减慢收敛。这个影响随着网络深度的增加而放大。在实践中，饱和问题和由此产生的梯度消失通常通过使用修正线性单元(Nair <Hinton, 2010) $ReLU(x)=\max(x,0)$，仔细的初始化(Bengio < Glorot, 2010; Saxe et Al.,2013)和小的学习率来解决。然而，如果我们能保证非线性输入的分布在网络训练时保持更稳定,那么优化器将不太可能陷入饱和状态,训练将加速。

我们把训练过程中深度网络内部结点的分布变化称为内部协变量转移。消除它可以保证更快的训练。我们提出了一种新的机制，我们称为为批标准化，它是减少内部协变量转移的一个步骤，这样做可以显著加速深度神经网络的训练。它通过标准化步骤来实现，标准化步骤修正了层输入的均值和方差。批标准化减少了梯度对参数或它们的初始值尺度上的依赖，对通过网络的梯度流动有有益的影响。这允许我们使用更高的学习率而没有发散的风险。此外， 批标准化使模型正则化并减少了对 Dropout(Srivastava et al.,2014)的需求。最后，批标准化通过阻止网络陷入饱和模式让使用饱和非线性成为可能。

在 4.2 小节，我们将批标准化应用到性能最好的 ImageNet 分类网络上，并且表明我们可以使用仅 7%的训练步骤来匹配其性能，并且可以进一步超过其准确性一大截。通过使用批标准化训练的网络的集合，我们取得了 top-5 错误率，其改进了 ImageNet 分类上已知的最佳结果。

2 减少内部协变量转变

由于训练过程中网络参数的变化，我们将内部协变量转移定义为网络激活分布的变化。为了改善训练，我们寻求减少内部协变量转移。随着训练的进行，通过固定层输入 x 的分布，我们期望提高训练速度。众所周知(LeCun et al. 1998b; Wiesler ” Ney, 2011)如果对网络的输入进行白化，网络训练将会收敛的更快–即输入线性变换为具有零均值和单位方差，并去相关。当每一层观察下面的层产生的输入时，实现每一层输入进行相同的白化将是有利的。通过白化每一层的输入，我们将采取措施实现输入的固定分布，消除内部协变量转移的不良影响。

我们考虑在每个训练步骤或在某些间隔来白化激活值,通过直接修改网络或根据网络激活值来更改优化方法的参数 (Wiesler et al, 2014; Raiko et al., 2012; Povey et al, 2014; Desjardins & Kavukcuoglu)。然而，如果这些修改分散在优化步骤中，那么梯度下降步骤可能会试图以要求标准化进行更新的方式来更新参数，这会降低梯度下降步骤的影响。例如，考虑一个层，其输入 u 加上学习到的偏置 b,通过减去在训练集上计算的激活值的均值对结果进行归一化：$\hat{x}=x-E[x], x=u+b, X=x_{1\ldots N}$是训练集上 x 值的集合，$E[x]=\frac{1}{N}\sum_{i=1}^{N}x_i$。如果梯度下降步骤忽略了 E[x]对 b 的依赖，那它将更新 $b\leftarrow b+\Delta b$,其中$\Delta b \propto -\partial \ell/\partial \hat{x}$。然后 $u+(b+\Delta b)-E[u+(b+\Delta b)]=u+b-E[u+b]$。因此，结合 b 的更新和接下来标准化中的改变会导致 层的输出没有变化，从而导致损失没有变化。随着训练的继续，b 将无限增长而损失保持不变。如果标准化不仅中心化而且缩放了激活值，问题会变得更糟糕。我们在最初的实验中已经观察到了这一点,当标准化参数在梯度下降步骤之外计算时,模型会爆炸。

上述方法的问题是梯度下降优化没有考虑到标准化中发生的事实。为了解决这个问题，我们希望确保对于任何参数值，网络总是产生具有所需分布的激活值。这样做将允许关于模型参数损失的梯度来解释标准化，以及它对模型参数Θ的依赖。设 x 为层的输入，将其看作向量，χ是这些输入在训练集上的集合。标准化可以写为变换

$$\hat{x}=\mathrm{Norm}(x,\mathcal{X})$$

它不仅依赖于给定的训练样本 x 而且依赖于所有样本$\mathcal{X}$——它们中的每一个都依赖于Θ, 如果 x 是由另一层生成的。对于反向传播，我们将需要计算 Jacobians $\frac{\partial \mathrm{Norm(x,X)}}{\partial \mathrm{x}}$和$\frac{\partial \mathrm{Norm(x,X)}}{\partial \mathcal{X}}$; 忽略后一项会导致上面描述的爆炸。在这个框架中，白化层输入是昂贵的，因为它要求计算协方差矩阵 $\mathrm{Cov[x]} = \mathrm{E}_{\mathrm{x}\in\mathcal{X}}[\mathrm{xx}^{\mathrm{T}}] - \mathrm{E[x]E[x]}^{\mathrm{T}}$和它的平方根倒数，从而生成白化的激活 $\mathrm{Cov[x]}^{-1/2}(\mathrm{x} - \mathrm{E[x]})$和这些变换进行反向传播的偏导数。这促使我们寻求一种替代方案，以可微分的方式执行输入标准化，并且在每次参数更新后不需要对整个训练集进行分析。

以前的一些方法 (例如 (Lyu'u'Simoncelli, 2008))使用通过单个训练样本计算的统计信息，或者在图像网络的 情况下，使用给定位置处不同特征图上的统计。然而，通过丢弃激活值绝对尺度改变了网络的表示能力。我们希望 通过对相对于整个训练数据统计信息的单个训练样本的激活值进行归一化来保留网络中的信息。

3 通过 Mini-Batch 统计进行标准化

由于每一层输入的整个白化是代价昂贵的并且不是到处可微分的，因此我们做了两个必要的简化。首先是我们将单独标准化每个标量特征，从而代替在层输入输出对特征进行共同白化，使其具有零均值和单位方差。对于具有 d 维输入 $\mathrm{x} = (\mathrm{x}^{(1)}...\mathrm{x}^{(\mathrm{d})})$的层，我们将标准化每一维

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

其中期望和方差在整个训练数据集上计算。如(LeCunetal.,1998b)中所示，这种标准化加速了收敛，即使特征没有去相关。 注意简单标准化层的每一个输入可能会改变层可以表示什么。例如，标准化 sigmoid 的输入会将它们约束到非线性 的线性状态。为了解决这个问题，我们要确保插入到网络中的变换可以表示恒等变换。为了实现这个，对于每一个激活值 $\mathrm{x}^{(\mathrm{k})}$,我们引入成对的参数$\gamma^{(\mathrm{k})}$,$\beta^{(\mathrm{k})}$,它们会归一化和移动标准化值：

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

这些参数与原始的模型参数一起学习，并恢复网络的表示能力。实际上，通过设置$\gamma^{(\mathrm{k})} = \sqrt{\mathrm{Var}[\mathrm{x}^{(\mathrm{k})}]}$和$\beta^{(\mathrm{k})} = \mathrm{E}[\mathrm{x}^{(\mathrm{k})}]$,我们可以重新获得原始的激活值，如果这是要做的最优的事。

每个训练步骤的批处理设置是基于整个训练集的，我们将使用整个训练集来标准化激活值。然而，当使用随机优化 时，这是不切实际的。因此，我们做了第二个简化：由于我们在随机梯度训练中使用小批量，每个小批量产生每次激活平均值和方差的估计。这样，用于标准化的统计信息可以完全参与梯度反向传播。注意，通过计算每一维的方差而不是联合协方差，可以实现小批量的使用；在联合情况下，将需要正则化，因为小批量大小可能小于白化的激活值的数量，从而导致单个协方差矩阵。

考虑一个大小为 m 的小批量数据$\mathcal{B}$。由于标准化被单独地应用于每一个激活，所以让我们集中在一个特定的激活$\mathrm{x}^{(\mathrm{k})}$,为了清晰忽略 k。在小批量数据里我们有这个激活的 m 个值，

$$\mathcal{B} = \{\mathrm{x}_{1...\mathrm{m}}\}.$$

设标准化值为$\hat{\mathrm{x}}_{1...\mathrm{m}}$,它们的线性变换为$\mathrm{y}_{1...\mathrm{m}}$。我们把变换

$$\mathrm{BN}_{\gamma,\beta}: \mathrm{x}_{1...\mathrm{m}} \rightarrow \mathrm{y}_{1...\mathrm{m}}$$

看作批标准化变换。我们在算法 1 中提出了 BN 变换。在算法中，为了数值稳定，ϵ是一个加到小批量数据方差上的常量。

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

BN 变换可以添加到网络上来操纵任何激活。在公式 $y = \mathrm{BN}_{\gamma,\beta}(x)$中，我们指出参数γ和β需要进行学习，但应该注意到在每一个训练样本中 BN 变换不单独处理激活。相反，$\mathrm{BN}_{\gamma,\beta}(x)$取决于训练样本和小批量数据中的其它样本。缩放和移动的值 y 传递到其它的网络层。标准化的激活值$\hat{x}$在我们的变换内部，但它们的存在至关重要。只要每个小批量的元素从相同的分布中进行采样，如果我们忽略ϵ,那么任何$\hat{x}$值的分布都具有期望为 0，方差为 1。这可以通过观察$\sum_{i=1}^{m}\hat{x}_i = 0$ 和$\frac{1}{m}\sum_{i=1}^{m}\hat{x}_i^2 = 1$ 看到，并取得预期。每一个标准化的激活值$\hat{x}^{(k)}$可以着作出线性变换$y^{(k)} = \gamma^{(k)}\hat{x}^{(k)} + \beta^{(k)}$组成的子网络的输入，接下来是原始网络的其它处理。所有的这些子网络输入都有固定的均值和方差，尽管这些标准化的$\hat{x}^{(k)}$的联合分布可能在训练过程中改变，但我们预计标准化输入的引入会加速子网络的训练，从而加速整个网络的训练。

在训练过程中我们需要通过这个变换反向传播损失ℓ的梯度，以及计算关于 BN 变换参数的梯度。我们使用的链式法则如下(简化之前):

$$\frac{\partial \ell}{\partial \widehat{x}_i} = \frac{\partial \ell}{\partial y_i}\cdot\gamma$$

$$\frac{\partial \ell}{\partial \sigma_{\mathcal{B}}^2} = \sum_{i=1}^{m}\frac{\partial \ell}{\partial \hat{x}_i}\cdot(x_i - \mu_{\mathcal{B}})\cdot\frac{-1}{2}(\sigma_{\mathcal{B}}^2 + \epsilon)^{-3/2}$$

$$\frac{\partial \ell}{\partial \mu_{\mathcal{B}}} = \sum_{i=1}^{m}\frac{\partial \ell}{\partial \widehat{x}_i}\cdot\frac{-1}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$$

$$\frac{\partial \ell}{\partial x_i} = \sum_{i=1}^{m}\frac{\partial \ell}{\partial \hat{x}_i}\cdot\frac{-1}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} + \frac{\partial \ell}{\partial \sigma_{\mathcal{B}}^2}\cdot\frac{2(x_i - \mu_{\mathcal{B}})}{m} + \frac{\partial \ell}{\partial \mu_{\mathcal{B}}}\cdot\frac{1}{m}$$

$$\frac{\partial \ell}{\partial \gamma} = \sum_{i=1}^{m}\frac{\partial \ell}{\partial y_i}\cdot\hat{x}_i$$

$$\frac{\partial \ell}{\partial \beta} = \sum_{i=1}^{m}\frac{\partial \ell}{\partial y_i}$$

因此，BN 变换是将标准化激活引入到网络中的可微变换。这确保了在模型训练时，层可以继续学习输入分布，表现出更少的内部协变量转移，从而加快训练。此外，应用于这些

标准化的激活上的学习到的仿射变换允许 BN 变换表示恒等变换并保留网络的能力。

3.1 批标准化网络的训练和推断

为了批标准化一个网络，根据算法 1，我们指定一个激活的子集，然后在每一个激活中插入 BN 变换。任何以前接收 x 作为输入的层现在接收 BN(x)作为输入。采用批标准化的模型可以使用批梯度下降，或者用小批量数据大小为 $m > 1$ 的随机梯度下降，或使用它的任何变种例如 Adagrad (Duchiet al,2011)进行训练。依赖小批量数据的激活值的标准化可以有效地训练，但在推断过程中是不必要的也是不需要的；我们希望输出只确定性地取决于输入。为此，一旦网络训练完成，我们使用总体统计来进行标准化

$$\hat{x} = \frac{x - E[x]}{\sqrt{Var[x] + \epsilon}}$$

,而不是小批量数据统计。跟训练过程中一样，如果忽略ϵ,这些标准化的激活具有相同的均值 0 和方差 1。我们使用无偏方差估计 $\mathrm{Var}[x] = \frac{m}{m-1} \cdot E_B[\sigma_B^2]$,其中期望是在大小为 m 的小批量训练数据上得到的，σ_B^2是其样本方差。使用这些值移动平均，我们在训练过程中可以眼踪模型的准确性。由于均值和方差在推断时是固定的，因此标准化是应用到每一个激活上的简单线性变换。它可以进一步由缩放γ和转移β组成，以产生代替 BN(x)的单线性变换。算法 2 总结了训练批标准化网络的过程。

Input: Network N with trainable parameters Θ;
subset of activations $\{x^{(k)}\}_{k=1}^{K}$

Output: Batch-normalized network for inference, $N_{\text{BN}}^{\text{inf}}$

1: $N_{\text{BN}}^{\text{tr}} \leftarrow N$ // Training BN network
2: **for** $k = 1 \ldots K$ **do**
3: Add transformation $y^{(k)} = \text{BN}_{\gamma^{(k)},\beta^{(k)}}(x^{(k)})$ to $N_{\text{BN}}^{\text{tr}}$ (Alg. 1)
4: Modify each layer in $N_{\text{BN}}^{\text{tr}}$ with input $x^{(k)}$ to take $y^{(k)}$ instead
5: **end for**
6: Train $N_{\text{BN}}^{\text{tr}}$ to optimize the parameters $\Theta \cup \{\gamma^{(k)}, \beta^{(k)}\}_{k=1}^{K}$
7: $N_{\text{BN}}^{\text{inf}} \leftarrow N_{\text{BN}}^{\text{tr}}$ // Inference BN network with frozen // parameters
8: **for** $k = 1 \ldots K$ **do**
9: // For clarity, $x \equiv x^{(k)}, \gamma \equiv \gamma^{(k)}, \mu_{\mathcal{B}} \equiv \mu_{\mathcal{B}}^{(k)}$, etc.
10: Process multiple training mini-batches $\mathcal{B}$, each of size m, and average over them:

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

11: In $N_{\text{BN}}^{\text{inf}}$, replace the transform $y = \text{BN}_{\gamma,\beta}(x)$ with $y = \frac{\gamma}{\sqrt{\text{Var}[x]+\epsilon}} \cdot x + \left(\beta - \frac{\gamma\,\text{E}[x]}{\sqrt{\text{Var}[x]+\epsilon}}\right)$
12: **end for**

Algorithm 2: Training a Batch-Normalized Network

3.2 批标准化卷积网络

批标准化可以应用于网络的任何激活集合。这里我们专注于仿射变换和元素级非线性组成的变换：

$$z = g(Wu + b)$$

其中 W 和 b 是模型学习的参数，$g(\cdot)$是非线性例如 sigmoid 或 ReLU。这个公式涵盖了全连接层和卷积层。我们在非线性之前通过标准化 $x = Wu + b$ 加入 BN 变换。我们也可以标准化层输入 u,但由于 u 可能是另一个非线性的输出，它的分布形状可能在训练过程中改变，并且限制其第一矩或第二矩不能去除协变量转移。相比之下，$Wu + b$ 更可能具有对称，非稀疏分布，即“更高斯”(Hyvärinen” 2000); 对其标准化可能产生具有稳定分布的激活。

注意，由于我们对 $Wu + b$ 进行标准化，偏置 b 可以忽略，因为它的效应将会被后面的中心化取消(偏置的作用会 归入到算法 1 的β)。因此，$z = g(Wu + b)$被

$$z = g\big(\text{BN}(Wu)\big)$$

替代,其中 BN 变换独立地应用到 $x = Wu$ 的每一维,每一维具有单独的成对学习参数$\gamma^{(k)}$,$\beta^{(k)}$。

另外，对于卷积层我们希望标准化遵循卷积特性–为的是同一特征映射的不同元素，在不同的位置，以相同的方式进行标准化。为了实现这个，我们在所有位置联合标准化了小批量数据中的所有激活。在算法 1 中，我们让 B 是跨越小批量数据的所有元素和空间位置的

特征图中所有值的集合–因此对于大小为 m 的小批量数据和大小为 $p \times q$ 的特征映射，我们使用有效的大小为$m' = |\mathcal{B}| = m \cdot pq$ 的小批量数据。我们每个特征映射学习一对参数 $\gamma^{(k)}$和$\beta^{(k)}$,而不是每个激活。算法 2 进行类似的修改，以便推断期间 BN 变换对在给定的特征映射上的每一个激活 应用同样的线性变换。

3.3 批标准化可以提高学习率

在传统的深度网络中，学习率过高可能会导致梯度爆炸或梯度消失，以及陷入差的局部最小值。批标准化有助于解决这些问题。通过标准化整个网络的激活值，在数据通过深度网络传播时，它可以防止层参数的微小变化被放大。例如，这使 sigmoid 非线性更容易保持在它们的非饱和状态，这对训练深度 sigmoid 网络至关重要，但在传统上很难实现。

批标准化也使训练对参数的缩放更有弹性。通常，大的学习率可能会增加层参数的缩放，这会在反向传播中放大梯 度并导致模型爆炸。然而，通过批标准化，通过层的反向传播不受其参数缩放的影响。实际上，对于标量 a,

$$\mathrm{BN}(\mathrm{Wu}) = \mathrm{BN}\big((\mathrm{aW})\mathrm{u}\big)$$

因此$\frac{\partial \mathrm{BN}((\mathrm{aW})\mathrm{u})}{\partial \mathrm{u}} = \frac{\partial \mathrm{BN}(\mathrm{Wu})}{\partial \mathrm{u}}$,因此标量不影响层的 Jacobian 行列式，从而不影响梯度传播。此外，$\frac{\partial \mathrm{BN}((\mathrm{aW})\mathrm{u})}{\partial(\mathrm{aW})} = \frac{1}{\mathrm{a}} \cdot \frac{\partial \mathrm{BN}(\mathrm{Wu})}{\partial \mathrm{W}}$因此更大的权重会导致更小的梯度，并且批标准化会稳定参数的增长。我们进一步推测，批标准化可能会导致雅可比行列式的奇异值接近于 1，这被认为对训练是有利的(Saxe et al.2013)。考虑具有标准化输入的两个连续的层，并且变换位于这些标准化向量之间：$\hat{z} = F(\hat{x})$。如果我们假设$\hat{x}$ 和$\hat{z}$是高斯分布且不相关的，那么 $F(\hat{x}) \approx J\hat{x}$是对给定模型参数的一个线性变换，$\hat{x}$和$\hat{z}$有单位方差，并且 $I = \mathrm{Cov}[\hat{z}] = J\mathrm{Cov}[\hat{x}]J^{T} = JJ^{T}$。因此，J 是正交的，其保留了反向传播中的梯度大小。尽管上述假设在现 实中不是真实的，但我们希望批标准化有助于梯度传播更好的执行。这有待于进一步研究。

4.实验

4.1. 随时间激活

为了验证内部协变量转移对训练的影响，以及批标准化对抗它的能力，我们考虑了在 MNIST 数据集上预测数字类别的问题(LeCunet al, 1998a)。我们使用非常简单的网络，28x28 的二值图像作为输入，以及三个全连接层，每层 100 个激活。每一个隐藏层用 sigmoid 非线性计算 $y = g(Wu + b)$,权重 W 初始化为小的随机高斯值。最后的隐藏层之后是具有 10 个激活(每类 1 个)和交叉熵损失的全连接层。我们训练网络 50000 次迭代，每份小批量数据中有 60 个样本。如第 3.1 节所述，我们在网络的每一个隐藏层后添加批标准化。我们对基准线和批标准化网络之间的比较感兴趣，而不是实现在 MNIST 上的最佳性能 (所描述的架构没有)。

图 1(a)显示了随着训练进行，两个网络在提供的测试数据上正确预测的分数。批标准化网络具有更高的测试准确率。为了调查原因，我们在训练过程中研究了原始网络 N 和批标准化网络N_{BN}^{tr}(Alg.2)中的 sigmoid 输入。在图 1(b,c)中，我们显示，对于来自每个网络的最后一个隐藏层的一个典型的激活，其分布如何演变。原始网络中的 分布随着时间的推移而发生显著变化，无论是平均值还是方差，都会使后面的层的训练复杂化。相比之下，随着训练的进行，批标准化网络中的分布更加稳定，这有助于训练。

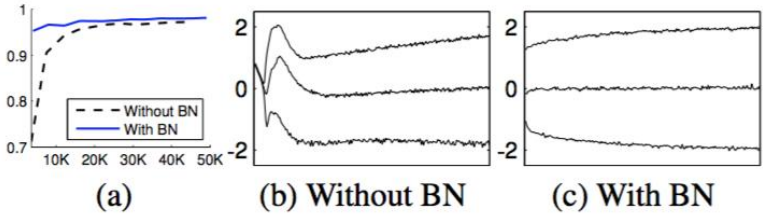

图 1。(a)使用批标准化和不使用批标准化训练的网络在 MNIST 上的测试准确率，以及训练的迭代次数。批标准化有助于网络训练的更快，取得更高的准确率。(b, c)典型的 sigmoid 在训练过程中输入分布的演变，显示为 15%，50%，85%。批标准化使分布更稳定并降低了

内部协变量转移。

4.2. ImageNet 分类

我们将批标准化化应用于在 ImageNet 分类任务 (Russakovsky 等，2014) 上训练的 Inception 网络的新变种(Szegedy 等，2014)。网络具有大量的卷积和池化层，和一个 softmax 层用来在 1000 个可能之中预测图像的类 别。卷积层使用 ReLU 作为非线性。与 (Szegedy 等人，2014 年) 中描述的网络的主要区别是 5×5 卷积层被两个连续的 3x3 卷积层替换，最多可以有 128 个滤波器。该网络包含 $13.6 \cdot 10^6$个参数，除了顶部的 softmax 层之外，没有全连接层。在其余的文本中我们将这个模型称为 Inception。训练在大型分布式架构 (Deanetal。,2012)上进行，10 个模型副本中的每一个都使用了 5 个并行步骤，使用异步带动量的 SGD (Sutskever 等，2013), 小批量数据大小为 32。随着训练进行，所有网络都通过计算验证准确率@1 来评估，即每幅图像使用单个裁剪图像，任 1000 个可能性中预测正确标签的概率。

在我们的实验中，我们评估了几个带有批标准化的 Inception 修改版本。在所有情况下，如第 3.2 节所述，批标准化 以卷积方式应用于每个非线性的输入，同时保持架构的其余部分不变。

4.2.1. 加速 BN 网络

将批标准化简单添加到网络中不能充分利用我们方法的优势。为此，我们进行了以下修改：

提高学习率。在批标准化模型中，我们已经能够从高学习率中实现训练加速，没有不良的副作用（第 3.3 节）。

删除丢弃。我们发现从 BN-Inception 中删除丢弃可以使网络实现更高的验证准确率。我们推测，批标准化提供了类似丢弃的正则化收益，因为对于训练样本观察到的激活受到了同一小批量数据中样本随机选择的影响。

更彻底地搅乱训练样本。我们启用了分布内部搅乱训练数据，这样可以防止同一个例子一起出现在小批量数据中。这导致验证准确率提高了约 1%，这与批标准化作为正则化项的观点是一致的：它每次被看到时都会影响一个样本，在我们的方法中内在的随机化应该是最有益的。

减少 L2 全中正则化。虽然在 Inception 中模型参数的 L2 损失会控制过拟合，但在修改的 BN-Inception 中，损失的权重减少了 5 倍。我们发现这提高了在提供的验证数据上的准确性。

加速学习率衰减。在训练 Inception 时，学习率呈指数衰减。因为我们的网络训练速度比 Inception 更快，所以我们将学习速度降低加快 6 倍。

删除局部响应归一化。虽然 Inception 和其它网络（Srivastava 等人，2014）从中受益，但是我们发现使用批标准化它是不必要的。

减少光照扭曲。因为批标准化网络训练更快，并且观察每个训练样本更少的次数，所以通过更少地扭曲它们，我们让训练器关注更多的“真实”图像。

4.2.2. 单网络分类

我们评估了下面的网络，所有的网络都在 LSVRC2012 训练数据上训练，并在验证数据上测试：

Inception: 在 4.2 小节开头描述的网络，以 0.0015 的初始学习率进行训练。

BN-Baseline: 每个非线性之前加上批标准化，其它的与 Inception 一样。

BN-x5: 带有批标准化的 Inception，修改在 4.2.1 小节中。初始学习率增加 5 倍到了 0.0075。原始 Inception 增加同样的学习率会使模型参数达到机器无限大。

BN-x30: 类似于 BN-x5,但初始学习率为 0.045 (Inception 学习率的 30 倍)。

BN-x5-Sigmoid: 类似于 BN-x5,但使用 sigmoud 非线性 $g(t) = \frac{1}{1+\exp(-x)}$来代替 ReLU。我

们也尝试训练带有 sigmoid 的原始 Inception，但模型保持在相当于机会的准确率。

在图 2 中，我们显示了网络的验证集准确率，作为训练步骤次数的函数。Inception 网络在 $31 \cdot 10^6$次训练步骤后达 到了 72.2%的准确率。图 3 显示，对于每个网络，达到同样的 72.2%准确率需要的训练步骤数量，以及网络达到的最大验证集准确率和达到该准确率的训练步骤数量。

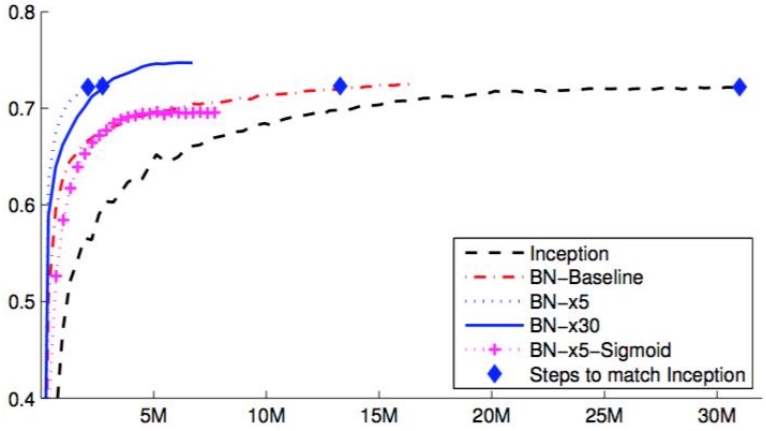

图 2。Inception 和它的批标准化变种在单个裁剪图像上的验证准确率以及训练步骤的数量。

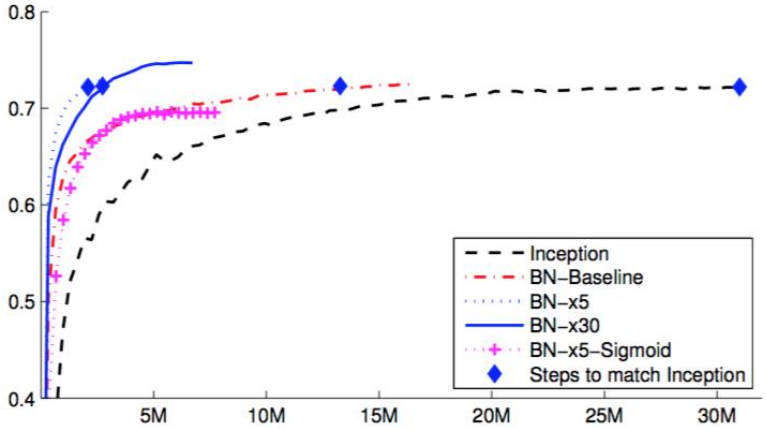

图 3。对于 Inception 和它的批标准化变种，达到 Inception 最大准确率(72.296)所需要的训练步骤数量，以及网络取得的最大准确率。

通过仅使用批标准化 (BN-Baseline),我们在不到 Inception 一半的训练步骤数量内将准确度与其相匹配。通过应 用 4.2.1 小节中的修改，我们显著提高了网络的训练速度。BN-x5 需要比 Inception 少 14 倍的步骤就达到了 72.2%的准确率。有趣的是，进一步提高学习率 (BN-x30) 使得该模型最初训练有点慢，但可以使其达到更高的最终准确率。这种现象是违反直觉的，应进一步调查。在 $6 \cdot 10^6$步骤之后，BN-x30 达到 74.8%的准确率，即比 Inception 达到 72.2%的准确率所需的步骤减少了 5 倍。

我们也证实了尽管训练这样的网络是众所周知的困难，但是当使用 sigmoid 作为非线性时，内部协变量转移的减少 允许具有批标准化的深层网络被训练。的确，BN-x5-Sigmoid 取得了 69.8%的准确率达。没有批标准化，使用 sigmoid 的 Inception 从未达到比 1/1000 准确率更好的结果。

4.2.3. 组合分类

目前在 ImageNet 大型视觉识别竞赛中报道的最佳结果是传统模型 (Wuetal。, 2015) 的 Deep Image 组合和 (He 等，2015) 的组合模型。后者报告了 ILSVRC 测试服务器评估的

4.94% 的 top-5 错误率。这里我们在测试服务器上报告 4.82% 的测试错误率。这提高了以前的最佳结果，并且根据 (Russakovsky 等，2014) 这超过了人类评估者的评估准确率。

对于我们的组合，我们使用了 6 个网络。每个都是基于 BN-x30 的，进行了以下一些修改：增加卷积层中的初始重 量；使用 Dropout(丢弃概率为 5%或 10%,而原始 Inception 为 40%); 模型最后的隐藏层使用非卷积批标准化。每个网络在大约 $6 \cdot 10^6$个训练步骤之后实现了最大的准确率。组合预测是基于组成网络的预测类概率的算术平均。组合和多裁剪图像推断的细节与 (Szegedyet al, 2014) 类似。

我们在图 4 中证实了批标准化使我们能够在 ImageNet 分类挑战基准上设置新的最佳结果。

我们在图 4 中证实了批标准化使我们能够在 ImageNet 分类挑战基准上设置新的最佳结果。

Model	Resolution	Crops	Models	Top-1 error	Top-5 error
GoogLeNet ensemble	224	144	7	-	6.67%
Deep Image low-res	256	-	1	-	7.96%
Deep Image high-res	512	-	1	24.88	7.42%
Deep Image ensemble	up to 512	-	-	-	5.98%
MSRA multicrop	up to 480	-	-	-	5.71%
MSRA ensemble	up to 480	-	-	-	4.94%*
BN-Inception single crop	224	1	1	25.2%	7.82%
BN-Inception multicrop	224	144	1	21.99%	5.82%
BN-Inception ensemble	224	144	6	20.1%	**4.82%***

图 4。批标准化 Inception 与以前的最佳结果在提供的包含 5 万张图像的验证集上的比较。组合结果是在测试集上由测试服务器评估的结果。BN-Inception 组合在验证集的 5 万张图像上取得了 4.9% top-5 的错误率。所有报道的其它结果是在验证集上。

5. 结论

我们提出了一个新的机制，大大加快了深度网络的训练。它是基于前提协变量转移的，已知其会使机器学习系统的训练复杂化，也适用于子网络和层，并且从网络的内部激活中去除它可能有助于训练。我们提出的方法从其标准化激活中获取其功能，并将这种标准化合并到网络架构本身。这确保了标准化可以被用来训练网络的任何优化方法进行恰当的处理。为了让深度网络训练中常用的随机优化方法可用，我们对每个小批量数据执行标准化，并通过标准化参数来反向传播梯度。批标准化每个激活只增加了两个额外的参数，这样做可以保持网络的表示能力。我们提出了一个算法，其用于构建，训练和执行推断批标准化网络。所得到的网络可以用饱和非线性进行训练，能更容忍增加的训练率，并且通常不需要丢弃来进行正则化。

仅仅将批标准化添加到了最新的图像分类模型中便在训练中取得了实质的加速。通过进一步提高学习率，删除丢弃和应用批标准化所提供的其它修改，我们只用了少部分的训练步骤就达到了以前的技术水平——然后在单网络图像分类中击败了最先进的技术。此外，通过组合多个使用批标准化训练的模型，我们在 ImageNet 上的表现显著优于最好的已知系统。

我们的方法与（Gülçehre＆Bengio，2013）的标准化层相似，尽管这两个方法解决的目标不同。批标准化寻求在整个训练过程中激活值的稳定分布，并且对非线性的输入进行归一化，因为这时更有可能稳定分布。相反，标准化层被应用于非线性的输出，这导致了更稀疏的激活。我们没有观察到非线性输入是稀疏的，无论是有批标准化还是没有批标准化。批标准化的其它显著差异包括学习到的缩放和转移允许 BN 变换表示恒等，卷积层处理以及不依赖于小批量数据的确定性推断。

在这项工作中，我们没有探索批标准化可能实现的全部可能性。我们的未来工作包括将我们的方法应用于循环神经网络（Pascanu et al.，2013），其中内部协变量转移和梯度消失或爆炸可能特别严重，这将使我们能够更彻底地测试假设标准化改善了梯度传播（第 3.3 节）。需要对批标准化的正则化属性进行更多的研究，我们认为这是 BN-Inception 中删除丢弃时我们观察到的改善的原因。我们计划调查批标准化是否有助于传统意义上的域自适应——即网络执行标准化是否能够更容易泛化到新的数据分布，也许仅仅是对总体均值和方差的重新计算（Alg.2）。最后，我们认为，该算法的进一步理论分析将允许更多的改进和应用。

6. 致谢

我们感谢 Vincent Vanhoucke 和 Jay Yagnik 的帮助和讨论，以及审稿人的深刻评论。

References

Bengio, Yoshua and Glorot, Xavier. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of AISTATS 2010, volume 9, pp. 249–256, May 2010.

Dean, Jeffrey, Corrado, Greg S., Monga, Rajat, Chen, Kai, Devin, Matthieu, Le, Quoc V., Mao, Mark Z., Ranzato, Marc'Aurelio, Senior, Andrew, Tucker, Paul, Yang, Ke, and Ng, Andrew Y. Large scale distributed deep networks. In NIPS, 2012.

Desjardins, Guillaume and Kavukcuoglu, Koray. Natural neural networks. (unpublished).

Duchi, John, Hazan, Elad, and Singer, Yoram. Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res., 12:2121 – 2159, July 2011. ISSN 1532-4435.

G¨ulc¸ehre, C¸ aglar and Bengio, Yoshua. Knowledge matters: Importance of prior information for optimization. CoRR, abs/1301.4083, 2013.

He, K., Zhang, X., Ren, S., and Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. ArXiv e-prints, February 2015.

Hyv¨arinen, A. and Oja, E. Independent component analysis: Algorithms and applications. Neural Netw., 13(4-5):411–430, May 2000.

Jiang, Jing. A literature survey on domain adaptation of statistical classifiers, 2008.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, November 1998a.

LeCun, Y., Bottou, L., Orr, G., and Muller, K. Efficient backprop. In Orr, G. and K., Muller (eds.), Neural Networks: Tricks of the trade. Springer, 1998b.

Lyu, S and Simoncelli, E P. Nonlinear image representation using divisive normalization. In Proc. Computer Vision and Pattern Recognition, pp. 1–8. IEEE Computer Society, Jun 23-28 2008. doi: 10.1109/CVPR.2008.4587821.

Nair, Vinod and Hinton, Geoffrey E. Rectified linear units improve restricted boltzmann machines. In ICML, pp.807–814. Omnipress, 2010.

Pascanu, Razvan, Mikolov, Tomas, and Bengio, Yoshua. On the difficulty of training recurrent neural networks. In Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013, pp. 1310–1318, 2013.

Povey, Daniel, Zhang, Xiaohui, and Khudanpur, Sanjeev. Parallel training of deep neural networks with natural gradient and parameter averaging. CoRR,abs/1410.7455, 2014.

Raiko, Tapani, Valpola, Harri, and LeCun, Yann. Deeplearning made easier by linear transformations in perceptrons. In International Conference on Artificial Intelligence and Statistics (AISTATS), pp. 924–932, 2012.

Russakovsky, Olga, Deng, Jia, Su, Hao, Krause, Jonathan, Satheesh, Sanjeev, Ma, Sean, Huang, Zhiheng, Karpathy, Andrej, Khosla, Aditya, Bernstein, Michael, Berg,Alexander C., and Fei-Fei, Li. ImageNet Large Scale Visual Recognition Challenge, 2014.

Saxe, Andrew M., McClelland, James L., and Ganguli, Surya. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. CoRR, abs/1312.6120, 2013.

Shimodaira, Hidetoshi. Improving predictive inference under covariate shift by weighting the log-likelihood function. Journal of Statistical Planning and Inference, 90(2):227–244, October 2000.

Srivastava, Nitish, Hinton, Geoffrey, Krizhevsky, Alex,Sutskever, Ilya, and Salakhutdinov, Ruslan. Dropout:A simple way to prevent neural networks from overfitting. J. Mach. Learn. Res., 15(1):1929–1958, January 2014.

Sutskever, Ilya, Martens, James, Dahl, George E., and Hinton, Geoffrey E. On the importance of initialization and momentum in deep learning. In ICML (3), volume 28 of JMLR Proceedings, pp. 1139–1147. JMLR.org, 2013.

Szegedy, Christian, Liu, Wei, Jia, Yangqing, Sermanet, Pierre, Reed, Scott, Anguelov, Dragomir, Erhan, Dumitru, Vanhoucke, Vincent, and Rabinovich, Andrew. Going deeper with convolutions. CoRR, abs/1409.4842, 2014.

Wiesler, Simon and Ney, Hermann. A convergence analysis of log-linear training. In Shawe-Taylor, J., Zemel, R.S., Bartlett, P., Pereira, F.C.N., and Weinberger, K.Q. (eds.), Advances in Neural Information Processing Systems 24, pp. 657–665, Granada, Spain, December 2011.

Wiesler, Simon, Richard, Alexander, Schl¨uter, Ralf, and Ney, Hermann. Mean-normalized stochastic gradient for large-scale deep learning. In IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 180–184, Florence, Italy, May 2014.

Wu, Ren, Yan, Shengen, Shan, Yi, Dang, Qingqing, and Sun, Gang. Deep image: Scaling up image recognition, 2015.