批标准化如何帮助神经网络

刘子源[1)]

[1)](清华大学 电子工程系, 北京 100084)

摘 要 批标准化是一种自适应的重参数化方法，通过控制神经网络内部激励的均值及方差，以解决训练深层模型的困难。批标准化一经提出便取得了巨大的成功，现在已经是各种神经网络的标配。本文试图从优化的角度，对批标准化如何改进神经网络进行解释。通过研究发现，批标准化主要通过规范网络内部激励以及平滑网络优化空间来帮助神经网络的收敛。

关键词 神经网络；优化；批标准化；深度学习

How Does Batch Normalization Help Neural Network

Ziyuan Liu[1)]

[1)](Department of Electrical Engineering, Tsinghua University, Beijing 100084, China)

Abstract Batch standardization is an adaptive reparameterization method, which solves the difficulty of training deep models by controlling the mean and variance of the internal activation of the neural network. Batch standardization has achieved great success as soon as it was proposed, and it is now the default configuration of various neural networks. This article attempts to explain how batch standardization can improve neural networks from the perspective of optimization. Through research, it is found that batch standardization mainly helps the convergence of neural networks by regulating the internal activations of the network and smoothing the network optimization landscape.

Key words Neural Network; Optimization; Batch Normalization; Deep Learning

1 背景

众所周知，标准化神经网络输入（使其变为零均值及方差稳定）对神经网络的训练很有帮助[1]。基于这一思想，随着神经网络和深度学习在 21 世纪的飞速发展，Ioffe 和 Szegedy 提出了批标准化[2]（Batch Normalization），目的是为了解决神经网络训练复杂，收敛速度慢等问题。该方法一经提出，便在众多神经网络模型中得以应用，并且有效地提升了网络的性能并降低网络的训练时间[3,4]。然而尽管批标准化大量被使用，其背后的原理并没有被完全研究清楚。Ioffe 等人认为批标准化降低了网络内部的“Covariate Shift”现象[2]，但是最近有研究从其他优化角度考虑这个问题，并认为批标准化的有效是源自于其使得整个空间更加平滑的特性。

本文在对有关批标准化的文献的调研基础上，站在优化的视角对神经网络和批标准化进行了分析，并试图从优化的角度给出对批标准化如何改进神经网络进行解释。本文剩余的章节安排如下：第二节介绍神经网络中常用的优化问题；第三节给出批标准化的定义，并介绍其优点；第四节从优化的角度对批标准化试图进行解释；第五节总结文章的内容。

刘子源，性别男，2000 年生，博士学位在读，学生，主要研究领域为深度学习 E-mail: liuziyua22@mails.tsinghua.edu.cn.
第 1 作者手机号码：13001911233，E-mail: liuziyua22@mails.tsinghua.edu.cn

2 神经网络中的优化问题

人工神经网络（尤其是深度学习算法）在许多情况下都涉及到优化问题。例如，最常见的是在有监督学习中利用数值优化方法来求解目标函数的最优值从而找到最优的网络参数；或者是无监督学习中求解形如 $min_S \sum_{k=1}^{K} \sum_{x \in S_k} ||x - \mu_k||_2^2$ 的 k 均值聚类算法；或者是在强化学习中求解最优的策略方程。这些问题都可以归为机器学习的范畴，首先建立模型假设，之后再利用优化算法求解目标函数的最大或最小值。

另外，神经网络中也有一些子问题可以建模成一个优化问题。神经网络结构搜索（Neural Architecture Search, NAS），或者是样本选取中都有用到优化算法。

本节重点关注并介绍神经网络目标函数的优化算法，并重点分析其中最常用的算法——批量随机梯度下降算法[5]（Mini-Batch Stochastic Gradient Descent，下文简称 SGD）。目前，常用在神经网络中的优化算法可以分为两类，分别是：（1）一阶优化方法和（2）二阶优化方法。这些算法的过程中都需要利用到目标函数的梯度信息。

但是，用于神经网络模型训练的优化算法与传统的优化算法有所不同。在机器学习中，人们通常关注某个指标 P。这个指标通常定义在测试集中并且是不可解的（即使对于线性分类器而言，精确地优化 0-1 损失也是一个 NP-H 问题）[Marcotte and Savard, 1992]，因此，我们只能够利用一个代理函数（通常被称为代价函数）$J(\boldsymbol{\theta})$ 来间接地优化 P。这就与直接优化目标 P 的传统优化算法不同。通常，代价函数可写为训练集上的平均

$$J(\boldsymbol{\theta}) = \mathbb{E}_{(\boldsymbol{x},y)\sim\hat{p}_{data}} L(f(\boldsymbol{x};\boldsymbol{\theta}), y)$$

其中 L 是每个样本的损失函数，$f(\boldsymbol{x};\boldsymbol{\theta})$ 是输入 $\boldsymbol{x}$ 是所预测的输出，$\hat{p}_{data}$ 是数据的经验分布。

要使用优化方法来解决机器学习问题，则需要将机器学习问题转换为一个优化问题。其中最简单的方法就是：最小化训练集上的期望损失（经验风险）：

$$\min \frac{1}{N}\sum_{i=1}^{N} L(f(\boldsymbol{x}^{(i)};\boldsymbol{\theta}), y^{(i)})$$

其中 N 表示训练样本的数目。

2.1 一阶优化算法

近年来，随机梯度优化方法及其衍生出的自适应算法正在广泛应用于神经网络中，并且已经在各个领域取得了出色的成绩。其中，SDG 是最基础、最重要、同样是最被广泛应用的算法。SGD 与传统的梯度下降方法不同之处在于，SGD 每次只是按照数据生成分布抽取 m 个小批量（独立同分布）样本，通过计算小批量样本的均值，可以得到梯度的无偏估计。

SGD 在每次迭代中利用参数更新梯度时，只考虑当前小批量中的 m 个样本，而不用计算所有的训练样本。SGD 梯度更新公式如下：

$$\boldsymbol{\theta}' = \boldsymbol{\theta} - \eta\frac{1}{m}\nabla_{\boldsymbol{\theta}}\sum_{i} L(f(\boldsymbol{x}^{(i)};\boldsymbol{\theta}), \boldsymbol{y}^{(i)})$$

其中 η 为算法采用的学习率（需要人工设定）。

SGD 相较于传统梯度下降算法（即沿着整个训练集的梯度方向下降，下文简称梯度下降算法）有两个显著的优点。（1）**计算复杂度低**：其每次迭代只需要使用 m 个小批量样本。在每次迭代中，梯度下降的计算复杂度为 $O(ND)$，而 SGD 计算复杂度仅为 $O(mD)$（考虑线性回归模型，其中 D 为输入的维度，即 $\boldsymbol{x}^{(i)} = (x_1^{(i)}, \cdots, x_D^{(i)})$）。虽然 SGD 增加了收敛所需的迭代次数，但是当 N 远大于 m 时（这正是现今大部分机器学习的场景），增加的迭代次数在计算复杂度的大幅减小下可以忽略不计。（2）**更容易收敛到全局最优值**：由于神经网络是非凸的，梯度下降算法使用的固定的梯度迭代方法，容易陷入一个局部极小值点。而 SGD 中梯度的随机震荡会使得优化函数跳出当前的局部最优。因此，与梯度下降方法相比，SGD 可以有效的降低计算复杂度、提升收敛速度并且收敛更加稳定。

但是，SGD 存在一个严重的问题，正是因为 SGD 在用小批量样本来估计训练机梯度时引入了噪声，从而使得**梯度方向震荡**，这个现象在小批量样本很小时尤为严重。这会导致当模型下降到达极小点时，SGD 由于梯度估计引入的噪声源并不会消失，导致模型在极小点附近持续震荡。当学习率较大时，甚至可能会离开最优的极小点。为了使得算法稳定收敛，必须选择一个合适的学习率。

另外，与很多利用梯度的迭代优化算法一样，**SGD 对于初始值的选择也很敏感**[6]。参数初始值的选取可以决定算法是否收敛，有些初始点十分不稳定，可能会使得模型参数过大且 SGD 无法收敛[7]。当学习收敛时，一个“好的”初始点可以使学习更快得收敛，或是收敛到一个代价更低的点。并且，初始点的选取也会极大的影响泛化误差。

在实践中，SGD 一般会选择线性衰减的学习率直到第 τ 次迭代：

$$\eta_k = (1-\alpha)\eta_0 + \alpha\eta_\tau$$

其中 $\alpha = \frac{k}{\tau}$。在第 τ 次迭代后，一般使 η 保持常数。

由于 SGD 需要人工设置学习率等参数，后续又衍生出了如 AdaGrad，RMSProp 等自适应学习率算。另外，为了加快学习速度，动量[8,9](Momentum) 的概念也被广泛用于神经网络中，其通过积累之前梯度的平均移动方向，并沿着该方向继续移动。

2.2 二阶优化算法

为了进一步利用梯度的信息，二阶优化方法也逐渐被用于神经网络中。与一阶方法相比，二阶方法使用二阶导数改进了优化，但同时也导致了更加复杂的计算。

牛顿法是二阶梯度方法的一个代表，其基于二阶泰勒级数展开在某点 $\boldsymbol{\theta}_0$ 附近来近似 $J(\boldsymbol{\theta})$。通过求解这个函数的临界点，可以得到牛顿参数的更新规则：

$$\boldsymbol{\theta}' = \boldsymbol{\theta}_0 - \boldsymbol{H}^{-1}\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0)$$

其中 $\boldsymbol{H}$ 是 J 相对于 $\boldsymbol{\theta}$ 的 Hessian 矩阵在 $\boldsymbol{\theta}_0$ 处的估计。但是，由于深度学习中目标函数的表面通常是非凸的，因此不能直接使用牛顿法。例如，在靠近鞍点处，牛顿法实际上会导致更新朝错误的方向移动。这种情况通常使用正则化 Hessian 矩阵来避免，例如可以在 Hessian 矩阵对角线上增加常数 α，正则化更新变为

$$\boldsymbol{\theta}' = \boldsymbol{\theta}_0 - [H(f(\boldsymbol{\theta}_0)) + \alpha \boldsymbol{I}]^{-1}\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0)$$

如果应用到大型神经网络中，牛顿法还有一个更大的挑战，就是其相比于一阶方法显著的计算负担。Hessian 矩阵中元素数目是神经网络中参数的平方，这对于当今动辄百万千万级别参数（甚至上亿参数，如 GPT-3[10]）的神经网络来说计算负担是不可接受的。

正是由于 Hessian 矩阵巨大的计算量，有一些避免计算 Hessian 矩阵（Hessian-free）的二阶方法被提出，如共轭梯度、BFGS、L-BFGS 算法等。总的来说，二阶方法比一阶方法利用了更多梯度的信息，因此可以使得网络参数更快地收敛。但是由于其较大的计算复杂度，在实际应用中很少有大规模神经网络使用二阶的优化方法。

3 批标准化

批标准化（下文称为 BatchNrom）并不是一个优化算法，而是一个自适应的重参数化的方法，试图解决训练非常深模型的困难[11]。在神经网络的层面，BatchNorm 通过在神经网络的两个层（layer）之间添加一个额外的标准化操作来控制数据的均值和方差，以稳定输入的分布，从而减少网络内部的“Covariate Shift”[12]（通常指样本分布在不同数据集中不一致的现象[13]，后文简称 ICS）[1]。

BatchNorm 计算过程可以写为

$$y = g(\hat{h}), \hat{h} = \gamma\frac{h - \mu_{\mathcal{B}}}{\sigma_{\mathcal{B}}} + \beta \; and \; h = \boldsymbol{w}^T\boldsymbol{x}$$

[1]考虑迁移学习的场景，设源域 S 和目标域 T 的数据均记为 X，标签均为 Y，两个域的条件分布相同，即 $P_s(Y|X=x) = P_t(Y|X=x), \forall x \in \mathcal{X}$，但是两个域的边缘分布不一致，即 $P_s(X) \neq P_t(X)$

其中，y 是一个神经元的输出，$g(\cdot)$ 表示某个激活函数，h 和 $\hat{h}$ 分别是 BatchNrom 前和后的隐藏值（即激活函数的输入值），$\boldsymbol{x}$ 和 $\boldsymbol{w}$ 分别是网络某一层的输入和网络参数。在 BatchNrom 中，$\mu_{\mathcal{B}}$ 和 $\sigma_{\mathcal{B}}$ 分别是 h 的均值和标准差，他们通过一个小批量样本来针对每个神经元独立估计。γ 和 β 分别是缩放和平移的参数。其算法如下面的算法流程图所示。

算法 1 BatchNorm

输入：$h_1, \cdots, h_m \; based \; on \; a \; miniBatch \; \mathcal{B} = \{x_{1\cdots m}\}, \gamma, \beta$
输出：$\{y_i = BN_{\gamma,\beta}(h_i)\}$
$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} h_i$ // 小批量均值
$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(h_i - \mu_{\mathcal{B}})^2$ // 小批量方差
$\hat{h}_i \leftarrow \frac{h_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}}$ // 标准化
$y_i \leftarrow \gamma\hat{h}_i + \beta \equiv BN_{\gamma,\beta}(h_i)$ // 缩放和平移

在 BatchNorm 方法中，最后的缩放和平移操作是为了确保 BatchNorm 变换具有表示一个恒等变换的能力，从而恢复原网络的表示能力。当 $\gamma^{(k)} = \sqrt{Var[h^{(k)}]}$ 和 $\beta^{(k)} = E[h^{(k)}]$ 时，我们可以恢复出原始的激活值（activation），其中 k 代表隐藏层的第 k 个维度。

3.1 批标准化的优点

虽然 SGD 已经被证明是一个简单高效的一阶优化方法，但是 SGD 要求对于模型参数（尤其是学习率和初始化）有很高的要求（2.1 节中说明）。

(1) 更高的学习率。在传统的无 BatchNorm 神经网络中，一个过大的学习率会导致梯度爆炸或消失[6]。在激活函数前插入 BatchNorm 层，可以防止部分参数微小的变化导致数据在深度网络传播的过程中被放大。例如，BatchNorm 可以使得 Sigmoid 非线性激活函数的输入更多的保持在非饱和区域，以防止反向传播的梯度消失。另外，BatchNorm 可以降低训练过程对于参数的大小依赖性。一般来说，更高的学习率会导致参数的数值变大，从而放大反向传播的梯度，导致梯度爆炸。然而加上 BatchNorm 后，反向传播并不会受参数大小的影响。比如，考虑一个缩放因子 $\alpha > 1$

$$BN(Wu) = BN((aW)u)$$

因此，$\frac{\partial BN((\alpha W)u)}{\partial u} = \frac{\partial BN(Wu)}{\partial u}$，即两者的反向梯度传播相同。而且，$\frac{\partial BN((\alpha W)u)}{\partial(\alpha W)} = \frac{1}{\alpha}\frac{\partial BN(Wu)}{\partial W}$，也就是说更大参数本身的梯度更新会变得更小。

(2) 更强的泛化能力。实践上，在添加了 BatchNorm 的 Inception 网络[14]（下文称 BN-Inception 网

络）上移除 Dropout 层，可以使网络达到更高的验证准确率。Ioffe 等人推测，这可能是由于 BatchNorm 提供了与 Dropout 类似的正则化收益[2]。另外，BN-Inception 网络中减少损失函数中 L2 正则项的权重也可以提升验证的准确度。因此，从这些实验结果推测，BatchNorm 一定程度上提升了网络的泛化表示能力。

(3) 降低对参数初始化的依赖。考虑一个有 Sigmoid 激活函数的网络层 $z = g(Wu + b)$，其中 u 是输入，W 为参数矩阵，b 是偏置向量，$g(x) = \frac{1}{1+exp(-x)}$。随着输入的变大，激活函数的导数会趋近于 0（即 $\lim_{|x|\to\infty} g(x) = 0$ ）。这样如果输入 u 处于 Sigmoid 的饱和区，对于参数 W 和 b 的更新就会非常慢。另外，输入的值会受到前层网络的影响，对于前层网络参数的训练很可能会导致当前层的输入进去饱和区，大大降低模型的收敛速度。这个现象随着网络的加深会变得更加明显。除了 ReLU[15]，初始化策略[6,7,16] 以及小学习率等方法外，BatchNorm 可以很好的解决此问题。因为 BatchNorm 中会把激活函数的输入控制在一个标准范围内，强制使其远离 Sigmoid 的饱和区，故而可以降低对模型参数初始化策略的依赖。

4 优化角度的解释

由于批标准化的种种优点，其被提出后就取得了巨大的成功，已经是所有深度学习实践上的默认配置（截止目前，BatchNorm 原文已经有 2.6 万次引用）。但是，BatchNorm 为何有效在其被提出的几年内并没有被完全研究清楚，研究者对于 BatchNorm 的理解不够深入。而且，尽管有很多 BatchNorm 的变形被提出，但是他们都没有在本质上理解为何 BatchNorm 能够有效工作。

4.1 子网络输入分布不变

目前，最广泛被接受的解释就是 BatchNorm 可以改进神经网络 ICS 的现象，这也是 BatchNorm 被提出的原始动机。不严谨地说，ICS 指的是是由于前层网络参数更新而导致的子网络输入分布的变化。Ioffe 等人认为这种子网络输入的持续变化会对神经网络的训练带来负面的影响，BatchNorm 最初的提出也正是为了改善这一问题。

早在 1998 年，Lecun 等人就提出网络输入的分布的平稳性（均值为 0、方差不变、尽量不相关）有利于加速神经网络收敛[1]。因此，如果我们将每一个标准化后的激励 $\hat{x}^{(k)}$ 都视作是一个子网络的输入（这个子网络包括了一个线性变换 $y^{(k)} = \gamma^{(k)}\hat{x}^{(k)} + \beta^{(k)}$，以及后续的其他网络部分），经过 BatchNorm 规范化后，每一个子网络的输入都具有较为固定的均值和方差。虽然 BatchNorm 中的缩放和平移参数会随学习改变，但在网络的多个位置引入规范化操作后，会使得子网络的训练加速，从而加速整个网络的训练。

4.2 平滑优化空间

BatchNorm 降低 ICS 并使得子网络输入更加稳定的观点自 Ioffe 等人提出后被大部分学者所接受，但是这个角度并没有从本质上解释 BatchNorm 的作用。或者说，还有一个问题有待回答：ICS 的降低本质上如何改进网络的训练过程？Santurkar 等人的研究表明，BatchNorm 的成功与降低 ICS 关系不大，而是由于 BatchNorm 使得整个优化空间（Optimization Landscape）更加平滑，使得梯度方向更加稳定，从而加速了训练[17]。

Santurkar 等人通过对比添加了 BatchNorm 的神经网络（BN-VGG）以及在 BN-VGG 上的每个 BatchNorm 输出后添加一个随机噪声的网络（如图1所示），发现 ICS 与网络的收敛速度并无直接关系（如图2中实验结果所示）。

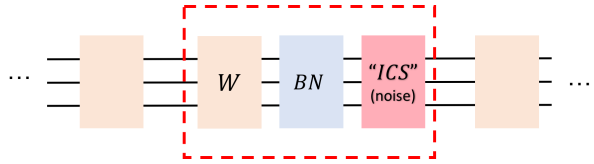


图 1 添加噪声的 BatchNorm 网络示意图

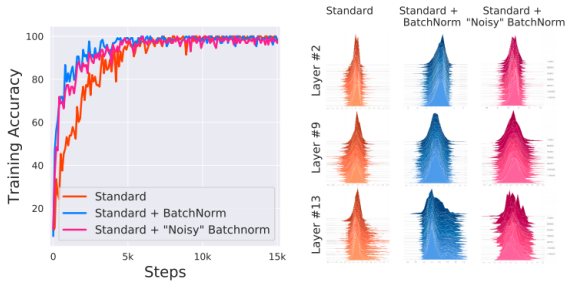
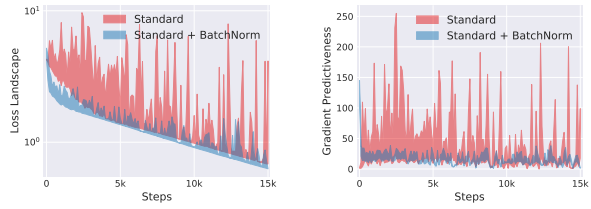


图 2 ICS 与网络表现的关系：比较了没有 BatachNorm 的 VGG 网络（Standard），有 BatchNorm 的 VGG 网络 (Standard + BatchNorm）以及在激励上刻意加噪声以加大 ICS 的 NB-VGG 网络（Standard + "Noisy" BatchNorm）。从图中可以看出有噪声的 BN-VGG 网络与无噪声的 BN-VGG 网络收敛速度相当，均好于无 BatchNorm 的 VGG 网络，但是有噪声的 BN-VGG 网络的 ICS 肉眼看上去明显要比无 BatchNorm 的 VGG 网络要大。



(a) loss landscape (b) gradient predictiveness

图 3 VGG 网络的优化空间分析。在某一个训练阶段，衡量损失的变化程度 (a) 以及沿着梯度方向的梯度 $l2$ 改变量 (b)。从 a 和 b 两个图中可以看出有 BatchNorm 的网络在这些度量上有明显的改善，因此 BatchNorm 的网络有着更优的优化空间。

通过对优化空间的分析，Santurkar 等人认为 BatchNorm 起作用的关键点在于：**BatchNorm 对网络的重参数化使得整个优化空间更平滑**。BatchNorm 提升了损失函数的 Lipschizness[2]，也就是说网络损失和参数均会以一个更小的幅度进行改变。由于损失曲面上的一些局部极小点或鞍点会使得基于梯度的优化方法变得不稳定，而 BatchNorm 使得损失曲面更加平滑可以有效改善这些问题，因此使得训练更加稳定。另外，由于整个优化空间更加平滑，梯度也会更加稳定，SGD 算法可以采用更大的学习率并同样收敛。

5 总结

批标准化作为一个重参数方法，通过标准化网络内部的激励来改进网络的训练，使得 SGD 等算法可以用更高的学习率，并使得网络的泛化能力更加并且降低网络对于参数初始化的依赖。研究表明，批参数化的成功主要来自于两个方面。分别是对于网络内部激励的控制，即使得子网络的输入分布不变；以及使得整个优化空间更加平滑。

参考文献

[1] LECUN Y A, BOTTOU L, ORR G B, et al. Efficient backprop[M]// Neural networks: Tricks of the trade. [S.l.]: Springer, 2012: 9-48.

[2] IOFFE S, SZEGEDY C. Batch normalization: Accelerating deep network training by reducing internal covariate shift[C]//International conference on machine learning. [S.l.]: PMLR, 2015: 448-456.

[3] HE K, ZHANG X, REN S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2016: 770-778.

[4] SILVER D, SCHRITTWIESER J, SIMONYAN K, et al. Mastering the game of go without human knowledge[J]. nature, 2017, 550(7676):354-359.

[5] BOTTOU L. Online algorithms and stochastic approximations[M/OL]// SAAD D. Online Learning and Neural Networks. Cambridge, UK: Cambridge University Press, 1998. http://leon.bottou.org/papers/bottou-98x.

[6] GLOROT X, BENGIO Y. Understanding the difficulty of training deep feedforward neural networks[C]//Proceedings of the thirteenth international conference on artificial intelligence and statistics. [S.l.]: JMLR Workshop and Conference Proceedings, 2010: 249-256.

[7] HE K, ZHANG X, REN S, et al. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification[C]//Proceedings of the IEEE international conference on computer vision. [S.l.: s.n.], 2015: 1026-1034.

[8] POLYAK B T. Some methods of speeding up the convergence of iteration methods[J]. Ussr computational mathematics and mathematical physics, 1964, 4(5):1-17.

[9] SUTSKEVER I, MARTENS J, DAHL G, et al. On the importance of initialization and momentum in deep learning[C]//International conference on machine learning. [S.l.]: PMLR, 2013: 1139-1147.

[10] BROWN T B, MANN B, RYDER N, et al. Language models are few-shot learners[J]. 2020.

[11] GOODFELLOW I, BENGIO Y, COURVILLE A, et al. Deep learning: volume 1[M]. [S.l.]: MIT press Cambridge, 2016.

[12] SHIMODAIRA H. Improving predictive inference under covariate shift by weighting the log-likelihood function[J]. Journal of statistical planning and inference, 2000, 90(2):227-244.

[13] JIANG J. A literature survey on domain adaptation of statistical classifiers[J]. URL: http://sifaka. cs. uiuc. edu/jiang4/domainadaptation/survey, 2008, 3:1-12.

[14] SZEGEDY C, LIU W, JIA Y, et al. Going deeper with convolutions [C]//Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2015: 1-9.

[15] NAIR V, HINTON G E. Rectified linear units improve restricted boltzmann machines[C]//Icml. [S.l.: s.n.], 2010.

[16] SAXE A M, MCCLELLAND J L, GANGULI S. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks[J]. arXiv preprint arXiv:1312.6120, 2013.

[17] SANTURKAR S, TSIPRAS D, ILYAS A, et al. How does batch normalization help optimization?[J]. arXiv preprint arXiv:1805.11604, 2018.

[2]函数 f 是 L-Lipschitz 的，如果 $|f(x_1) - f(x_2)| \leqslant L||x_1 - x_2||, \forall x_1, x_2$