《On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima》-ICLR2017文章阅读

([ICLR 17] https://arxiv.org/pdf/1609.04836.pdf. Very early empirical works about the relationship between sharp minima and generalization.)

这篇文章主要关注的问题是：在深度学习任务中，使用大批量时模型的泛化能力(Generalization gap)会下降。作者对这一现象做出了解释，并用实验验证了观点。作者认为是因为大的batchsize训练使得目标函数倾向于收敛到sharp minima，而小的batchsize则倾向于收敛到一个flat minima。最后作者还讨论了几种策略，以试图帮助大批量方法消除这种泛化差距。

1. 背景介绍

深度学习的主要目标是通过最小化一个损失函数来训练模型，而这个损失函数在深度学习中通常都是非凸的。要求解的问题的数学形式如下：

$$\min_{x\in\mathbb{R}^n} f(x) := \frac{1}{M}\sum_{i=1}^{M} f_i(x).$$

SGD及其变体是通常使用的优化算法，它用于找到一个函数的最小值。从一个随机初始点开始，然后不断地沿着函数梯度的反方向更新，因为在深度学习问题中使用整个数据集会很耗时，所以每一步只选择一个或一小批样本来计算梯度，其基本公式如下：

$$x_{k+1} = x_k - \alpha_k \left(\frac{1}{|B_k|} \sum_{i\in B_k} \nabla f_i(x_k) \right),$$

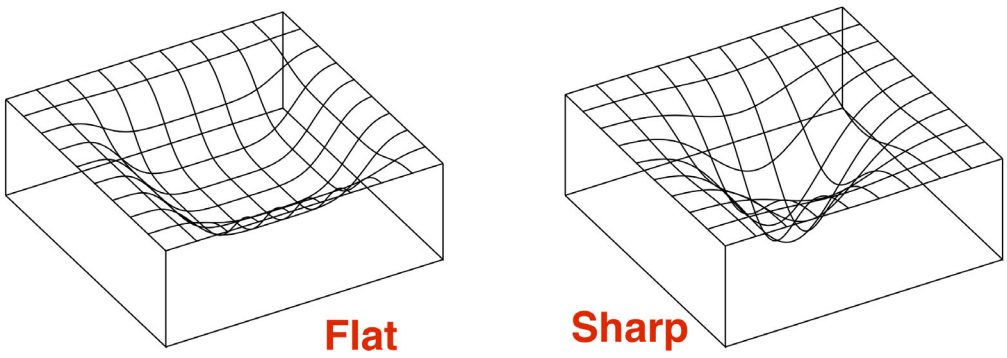
其中$|B_k|$是batchsize的大小，通常$|B_k| \in \{32, 64, \ldots, 512\}$。
SGD每次只考虑一小部分数据，引入了随机性，具有以下已被验证的理论性质：
(a) 对于强凸函数，这些方法能够保证收敛到最小值点；对于非凸函数，能够保证收敛到稳定点;
(b) 能够避免鞍点。鞍点是函数在某些方向是局部最小值，而在其他方向是局部最大值的点;
(c) 对输入数据的稳健性。这意味着算法的性能并不会因为输入数据的微小变化（例如添加了噪声）而发生大的改变。
但是SGD也有一个缺点：并行化困难(更新是串行的)。一种常见的方法是增大batchsize，然而这就导致了Generation gap的出现。
作者解释Generation gap的出现与大batchsize最终得到的解的尖锐度(sharpness)有关，即得到的解可能在某些方向上变化非常敏感（表现为尖锐），这可能会导致模型的泛化性能下降。

2. Large-Batch方法的缺点

大批量（large-batch，LB）方法训练深度学习模型时，观察到了泛化差距。但是，大批量方法通常得到的训练函数值和小批量（small-batch，SB）方法类似。可能的原因有：
(i) 大批量方法过度拟合了模型;
(ii) 大批量方法被鞍点所吸引;
(iii) 大批量方法缺乏小批量方法的探索性质，倾向于聚焦在距离初始点最近的最小值上;
(iv) 小批量和大批量方法收敛到具有不同泛化属性的质量不同的最小值。这篇论文中的数据支持了后两个猜想。

The lack of generalization ability is due to the fact that large-batch methods tend to converge to *sharp minimizers* of the training function. These minimizers are characterized by a significant number of large positive eigenvalues in $\nabla^2 f(x)$, and tend to generalize less well. In contrast, small-batch methods converge to *flat minimizers* characterized by having numerous small eigenvalues of $\nabla^2 f(x)$. We have observed that the loss function landscape of deep neural networks is such that large-batch methods are attracted to regions with sharp minimizers and that, unlike small-batch methods, are unable to escape basins of attraction of these minimizers.

作者认为LB之所以出现Generalization Gap问题，原因是LB训练时候更容易收敛到sharp minima，而SB则更容易收敛到flat minima。并且LB不容易从这些sharp minima中出来。



模型在训练过程中，如果找到的是一个平坦的极值点，那么模型在这个点附近的表现将不会太差，也就是说模型对输入数据的小变化不敏感，因此具有更好的泛化能力。反之，如果找到的是一个尖锐的极值点，那么模型在这个点附近的表现可能会差很多，也就是说模型对输入数据的小变化非常敏感，泛化能力差。因此，尽管尖锐的极值点可能会使得训练误差更小，但是在测试数据上的表现可能会更差。

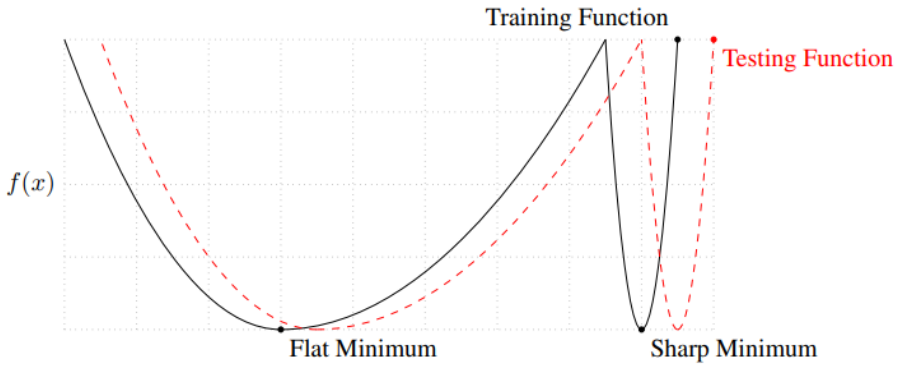
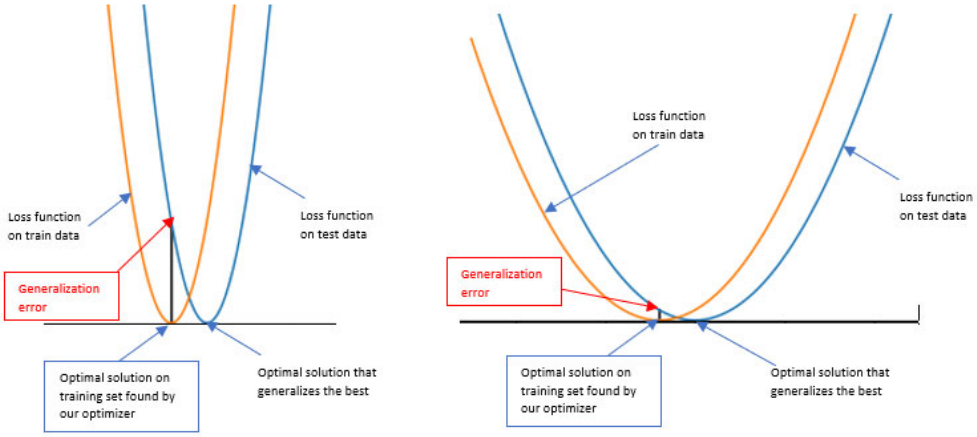


Figure 1: A Conceptual Sketch of Flat and Sharp Minima. The Y-axis indicates value of the loss function and the X-axis the variables (parameters)



Left : Sharp minima vs Right : Flat minima

接下来作者进行了一些实验来验证观点，选用的网络结构和数据集如下表：

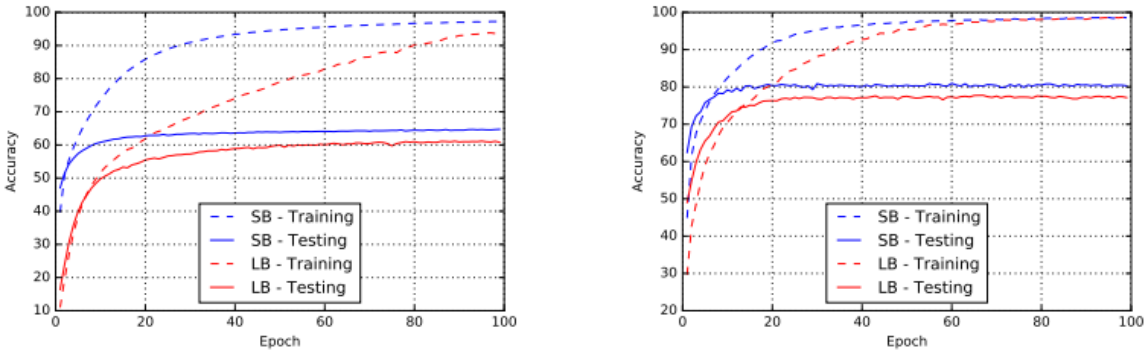
Table 1: Network Configurations

Name	Network Type	Architecture	Data set
F_1	Fully Connected	Section B.1	MNIST (LeCun et al., 1998a)
F_2	Fully Connected	Section B.2	TIMIT (Garofolo et al., 1993)
C_1	(Shallow) Convolutional	Section B.3	CIFAR-10 (Krizhevsky & Hinton, 2009)
C_2	(Deep) Convolutional	Section B.4	CIFAR-10
C_3	(Shallow) Convolutional	Section B.3	CIFAR-100 (Krizhevsky & Hinton, 2009)
C_4	(Deep) Convolutional	Section B.4	CIFAR-100

其中C1和C3是AlexNet结构，C2和C4是VGGNet结构。文章的目的是探索LB和SB方法的最小值的性质，而不是追求state-of-art或者是时间消耗，所以作者选用testing accuracy来表示实验结果。
对于LB方法，作者选用的batchsize为training data的10%，对于SB方法，作者选用256作为batchsize。优化器使用ADAM（ADAGRAD、SGD、adaQN等几个方法得到的结论是类似的）。损失函数使用的是交叉熵形式。最终的结果如下：

	Training Accuracy		Testing Accuracy	
Name	SB	LB	SB	LB
F_1	$99.66\% \pm 0.05\%$	$99.92\% \pm 0.01\%$	$98.03\% \pm 0.07\%$	$97.81\% \pm 0.07\%$
F_2	$99.99\% \pm 0.03\%$	$98.35\% \pm 2.08\%$	$64.02\% \pm 0.2\%$	$59.45\% \pm 1.05\%$
C_1	$99.89\% \pm 0.02\%$	$99.66\% \pm 0.2\%$	$80.04\% \pm 0.12\%$	$77.26\% \pm 0.42\%$
C_2	$99.99\% \pm 0.04\%$	$99.99\% \pm 0.01\%$	$89.24\% \pm 0.12\%$	$87.26\% \pm 0.07\%$
C_3	$99.56\% \pm 0.44\%$	$99.88\% \pm 0.30\%$	$49.58\% \pm 0.39\%$	$46.45\% \pm 0.43\%$
C_4	$99.10\% \pm 1.23\%$	$99.57\% \pm 1.84\%$	$63.08\% \pm 0.5\%$	$57.81\% \pm 0.17\%$

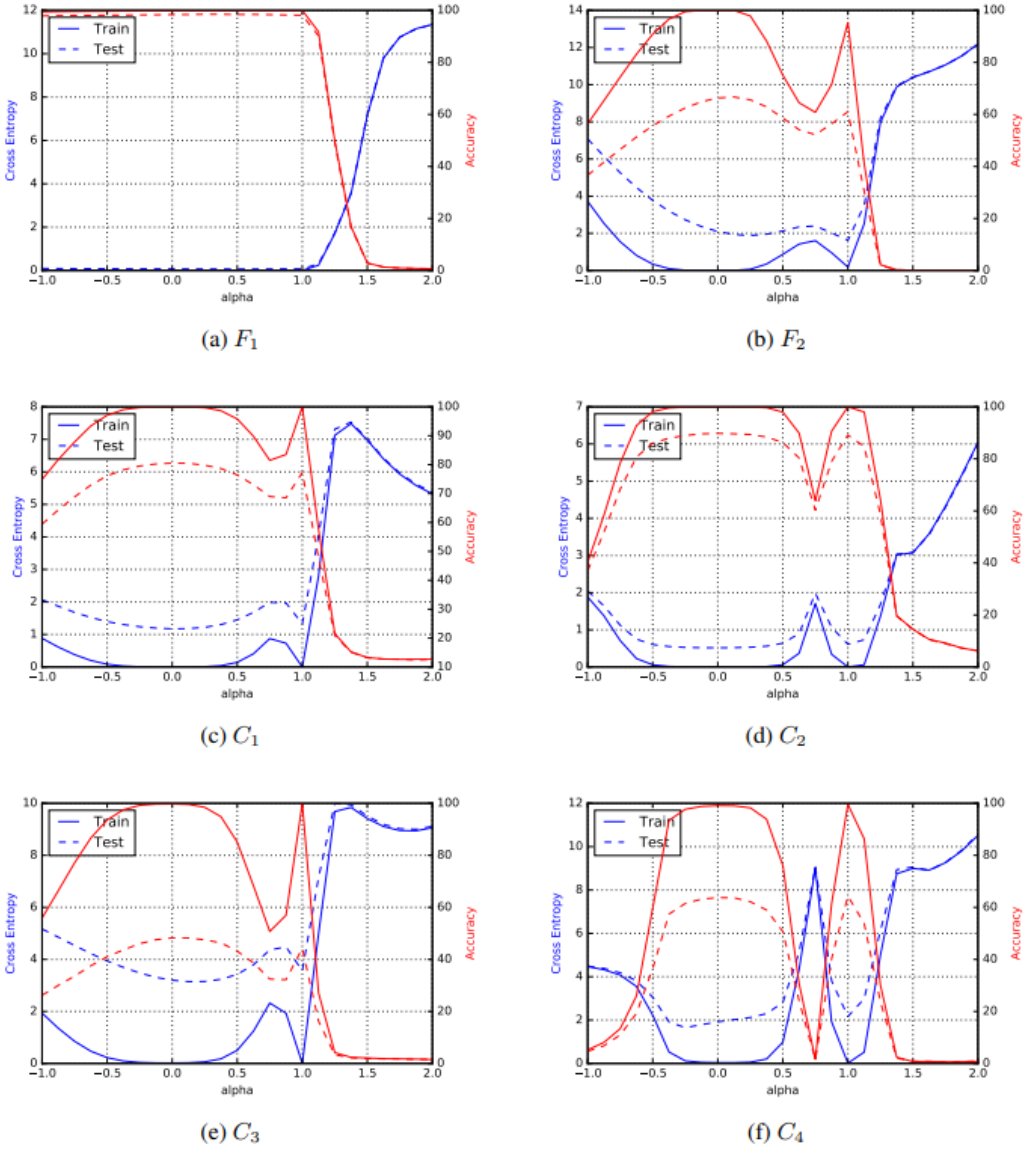
可以看到，SB和LB的training accuracy并无明显差异，而LB的testingaccuracy均明显低于SB，且LB方法出现了Generalization gap。
作者还强调了LB的Generalization gap出现的原因不是过拟合，在F2和C1网络的训练-测试曲线上（这两个网络可以代表其它的网络）可以看到，因此，旨在防止模型过拟合的提前停止策略并不能帮助减少泛化差距。



(a) Network F_2 (b) Network C_1

Figure 2: Training and testing accuracy for SB and LB methods as a function of epochs.

作者接下来绘制了一维的参数曲线，用x_s和x_l分别表示使用小批量和大批量训练时通过ADAM优化算法得到的解。然后，他们绘制了损失函数在训练和测试数据集上沿着包含这两个点的线段的变化。具体来说，对于α在$[-1, 2]$范围内，他们绘制了函数$f\left(\alpha x_l^* + (1-\alpha)x_s^*\right)$的变化，并在这些中间点上绘制了分类精度。



(a) F_1 (b) F_2

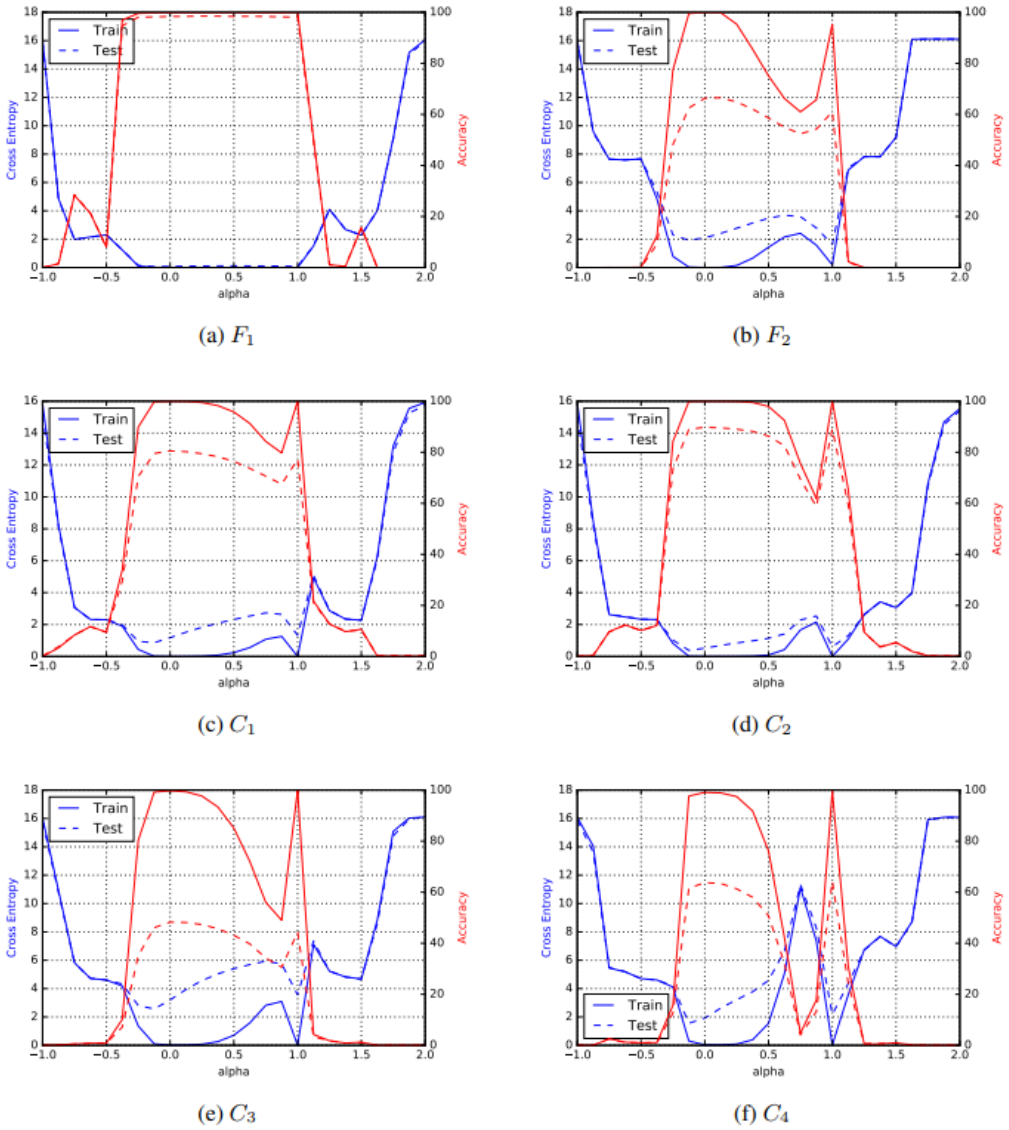
(c) C_1 (d) C_2

(e) C_3 (f) C_4

Figure 3: Parametric Plots – Linear (Left vertical axis corresponds to cross-entropy loss, f, and right vertical axis corresponds to classification accuracy; solid line indicates training data set and dashed line indicated testing data set); $\alpha = 0$ corresponds to the SB minimizer and $\alpha = 1$ to the LB minimizer.

可以看到LB方法的最优值明显比SB方法的最优值锋利（sharpness）。

作者还绘制了一个非线性切分的曲线图，$f\left(\sin\left(\frac{\alpha\pi}{2}\right)x_l^* + \cos\left(\frac{\alpha\pi}{2}\right)x_s^*\right)$，结果如下：



(a) F_1 (b) F_2

(c) C_1 (d) C_2

(e) C_3 (f) C_4

Figure 7: Parametric Plots – Curvilinear (Left vertical axis corresponds to cross-entropy loss, f, and right vertical axis corresponds to classification accuracy; solid line indicates training data set and dashed line indicated testing data set); $\alpha = 0$ corresponds to the SB minimizer while $\alpha = 1$ corresponds to the LB minimizer

此时的LB与SB方法的锐度（sharpness）区别也很明显。

Sharpness可以由目标函数的海森矩阵计算得到，但是计算代价过大。因此，作者使用了一个不完美但在计算上可行的方法，它基于探索解的一个邻域，并计算函数f在该邻域中的最大值，然后使用那个值来衡量给定局部最小值处的训练函数的敏感性。作者还采用了一种策略来避免对函数进行最大化求解时被局部子空间误导（当函数f在某个局部区域（比如高维空间$\mathbb{R}^n$的一个小子空间）中达到一个很大的值，但在整个空间$\mathbb{R}^n$中，这个值可能并不代表函数f的整体行为。这可能会导致我们错误地认为我们找到了一个全局最优点，而实际上这只是一个局部最优点，或者在全局范围内并不是最优的）。作者在$\mathbb{R}^n$和低维子空间中同时进行最大化求解，这样就可以观察到函数在不同维度和方向的行为，从而避免只在一个特定的小子空间中进行观察，导致的可能的误导。作者给出了一个矩阵$A(n*p)$的概念，A是一个在全空间随机抽样产生的矩阵，列数p就是随机产生的子空间维数，文中设置为$p = 100$。

局部区域（约束集）：

Specifically, let $\mathcal{C}_\epsilon$ denote a box around the solution over which the maximization of f is performed, and let $A \in \mathbb{R}^{n\times p}$ be the matrix defined above. In order to ensure invariance of sharpness to problem dimension and sparsity, we define the constraint set $\mathcal{C}_\epsilon$ as:

$$\mathcal{C}_\epsilon = \{z \in \mathbb{R}^p : -\epsilon(|(A^+x)_i| + 1) \leq z_i \leq \epsilon(|(A^+x)_i| + 1) \quad \forall i \in \{1, 2, \cdots, p\}\}, \tag{3}$$

where A^+ denotes the pseudo-inverse of A. Thus ϵ controls the size of the box. We can now define our measure of sharpness (or sensitivity).

Sharpness：

Metric 2.1. *Given $x \in \mathbb{R}^n$, $\epsilon > 0$ and $A \in \mathbb{R}^{n\times p}$, we define the $(\mathcal{C}_\epsilon, A)$-sharpness of f at x as:*

$$\phi_{x,f}(\epsilon, A) := \frac{(\max_{y\in\mathcal{C}_\epsilon} f(x + Ay)) - f(x)}{1 + f(x)} \times 100. \tag{4}$$

式子（4）定义了函数f在点x处的锐度。就是在局部区域C_ϵ内，函数f的最大值与其在x点处的值之差的比率。如果这个值很大，说明函数在这个区域内变化很快，因此锐度高。如果这个值小，那么函数在这个区域内变化较慢，因此锐度低。

Table3、4是文中对于六个网络结构的优化过程中的sharpness的计算结果。其中table3表示了整个空间上的结果，而table4则是子空间（100维）。

Table 3: Sharpness of Minima in Full Space; ϵ is defined in (3).

	$\epsilon = 10^{-3}$		$\epsilon = 5 \cdot 10^{-4}$	
	SB	LB	SB	LB
F_1	1.23 ± 0.83	205.14 ± 69.52	0.61 ± 0.27	42.90 ± 17.14
F_2	1.39 ± 0.02	310.64 ± 38.46	0.90 ± 0.05	93.15 ± 6.81
C_1	28.58 ± 3.13	707.23 ± 43.04	7.08 ± 0.88	227.31 ± 23.23
C_2	8.68 ± 1.32	925.32 ± 38.29	2.07 ± 0.86	175.31 ± 18.28
C_3	29.85 ± 5.98	258.75 ± 8.96	8.56 ± 0.99	105.11 ± 13.22
C_4	12.83 ± 3.84	421.84 ± 36.97	4.07 ± 0.87	109.35 ± 16.57

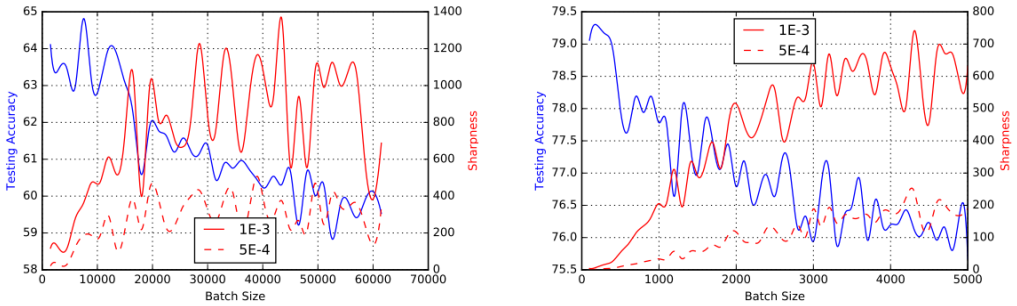
Table 4: Sharpness of Minima in Random Subspaces of Dimension 100

	$\epsilon = 10^{-3}$		$\epsilon = 5 \cdot 10^{-4}$	
	SB	LB	SB	LB
F_1	0.11 ± 0.00	9.22 ± 0.56	0.05 ± 0.00	9.17 ± 0.14
F_2	0.29 ± 0.02	23.63 ± 0.54	0.05 ± 0.00	6.28 ± 0.19
C_1	2.18 ± 0.23	137.25 ± 21.60	0.71 ± 0.15	29.50 ± 7.48
C_2	0.95 ± 0.34	25.09 ± 2.61	0.31 ± 0.08	5.82 ± 0.52
C_3	17.02 ± 2.20	236.03 ± 31.26	4.03 ± 1.45	86.96 ± 27.39
C_4	6.05 ± 1.13	72.99 ± 10.96	1.89 ± 0.33	19.85 ± 4.12

作者还指出，sharp minima并不出现在所有方向上，根据作者的实验观察，只有5%的子空间上会出现，在其他方向上，minima相对比较flat。

3. SB方法的优点

在选择batchsize时，会存在一个阈值，超过该阈值，模型的质量就会下降。

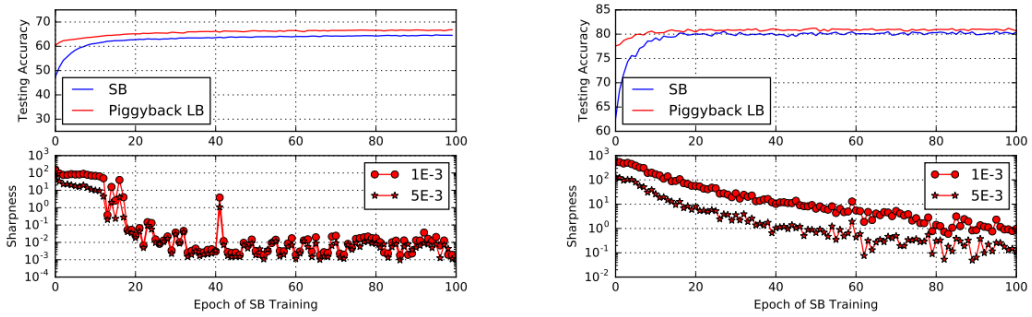


(a) F_2 (b) C_1

Figure 4: Testing Accuracy and Sharpness v/s Batch Size. The X-axis corresponds to the batch size used for training the network for 100 epochs, left Y-axis corresponds to the testing accuracy at the final iterate and right Y-axis corresponds to the sharpness of that iterate. We report sharpness for two values of ϵ: 10^{-3} and $5 \cdot 10^{-4}$.

SB方法在计算步长时使用噪声梯度，梯度中的噪声会把迭代结果从sharp的最小值吸引区域推出，鼓励它向较flat的最小值方向移动。当批量大小大于上述阈值时，随机梯度中的噪声不足以使迭代结果从初始区域跳出，从而导致收敛到一个sharp minima。

作者进行了实验来进行探索：用256的小batch训练一个网络100个epoch，每一个epoch以后保留迭代的结果，将这100个结果作为LB大一个起始训练点，称之为piggybacked LB。



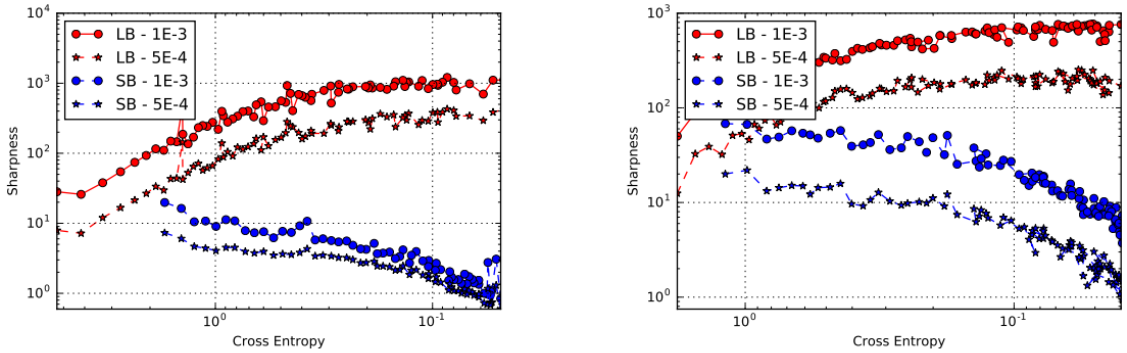
(a) F_2 (b) C_1

Figure 5: Warm-starting experiments. The upper figures report the testing accuracy of the SB method (blue line) and the testing accuracy of the warm started (piggybacked) LB method (red line), as a function of the number of epochs of the SB method. The lower figures plot the sharpness measure (4) for the solutions obtained by the piggybacked LB method v/s the number of warm-starting epochs of the SB method.

从结果来看，初始几个epoch时，由于还未完成优化过程，两者的test accuracy都比较低，且sharpness都较高。另一方面，当达到一定数量的epoch后，test accuracy会提高，大批量迭代结果的sharpness也会下降。这种情况似乎是在SB方法结束了探索阶段并找到了一个flat的最小值后发生的；之后，大批量方法能够向这个方向收敛，从而获得良好的测试精度。

还有一个推测是LB方法倾向于被初始点x_0附近的minima吸引，SB方法则会相对远离这些初始点附近的minima，作者的实验支持了这个猜想，他们观察到：小batch的最优值与起始点的距离是大batch的最优点与起始点距离的3到10倍。

作者还给出了sharpness随loss变化的曲线：



(a) F_2 (b) C_1

Figure 6: Sharpness v/s Cross Entropy Loss for SB and LB methods.

对于损失函数的较大值，即接近初始点的值，SB和LB方法的sharpness相似。然而，随着损失函数的减小，LB方法对应的迭代结果的sharpness快速增加，而小批量方法的sharpness初始阶段相对保持恒定，然后减小，最终到一个flat的minimum。

4. 尝试改进LB方法

4.1. Data Augmentation

Table 6: Effect of Data Augmentation

	Testing Accuracy		Sharpness (LB method)	
	Baseline (SB)	Augmented LB	$\epsilon = 10^{-3}$	$\epsilon = 5 \cdot 10^{-4}$
C_1	$83.63\% \pm 0.14\%$	$82.50\% \pm 0.67\%$	231.77 ± 30.50	45.89 ± 3.83
C_2	$89.82\% \pm 0.12\%$	$90.26\% \pm 1.15\%$	468.65 ± 47.86	105.22 ± 19.57
C_3	$54.55\% \pm 0.44\%$	$53.03\% \pm 0.33\%$	103.68 ± 11.93	37.67 ± 3.46
C_4	$63.05\% \pm 0.5\%$	$65.88 \pm 0.13\%$	271.06 ± 29.69	45.31 ± 5.93

从结果来看，经过数据增强的LB方法可以达到和SB方法差不多的预测精度，但是sharpness仍然较高，表明数据增强并没有改变LB方法存在的sharp minima的现象。

4.2. Conservative Training

有文章已经证明优化下面一个式子可以使得SGD算法的LB方法的收敛率得到改善：

$$x_{k+1} = \arg\min_x \frac{1}{|B_k|} \sum_{i \in B_k} f_i(x) + \frac{\lambda}{2} \|x - x_k\|_2^2$$

这种策略的动机是，在大批量方法的背景下，在移动到下一个批量之前更好地利用一个批量。作者将相同的框架应用于深度学习，但是没有理论保证。

Table 7: Effect of Conservative Training

	Testing Accuracy		Sharpness (LB method)	
	Baseline (SB)	Conservative LB	$\epsilon = 10^{-3}$	$\epsilon = 5 \cdot 10^{-4}$
F_1	$98.03\% \pm 0.07\%$	$98.12\% \pm 0.01\%$	232.25 ± 63.81	46.02 ± 12.58
F_2	$64.02\% \pm 0.2\%$	$61.94\% \pm 1.10\%$	928.40 ± 51.63	190.77 ± 25.33
C_1	$80.04\% \pm 0.12\%$	$78.41\% \pm 0.22\%$	520.34 ± 34.91	171.19 ± 15.13
C_2	$89.24\% \pm 0.05\%$	$88.495\% \pm 0.63\%$	632.01 ± 208.01	108.88 ± 47.36
C_3	$49.58\% \pm 0.39\%$	$45.98\% \pm 0.54\%$	337.92 ± 33.09	110.69 ± 3.88
C_4	$63.08\% \pm 0.10\%$	62.51 ± 0.67	354.94 ± 20.23	68.76 ± 16.29

从结果来看，与data augmentation类似，LB方法可以达到和SB方法差不多的预测精度，但是sharpness仍然较高。

4.3. Robust Training

该方法旨在优化一个最坏情况下的cost，而不是常规的cost。数学形式如下：

$$\min_{x} \phi(x) := \max_{||\Delta x|| \leqslant \epsilon} f(x + \Delta x).$$

图示：

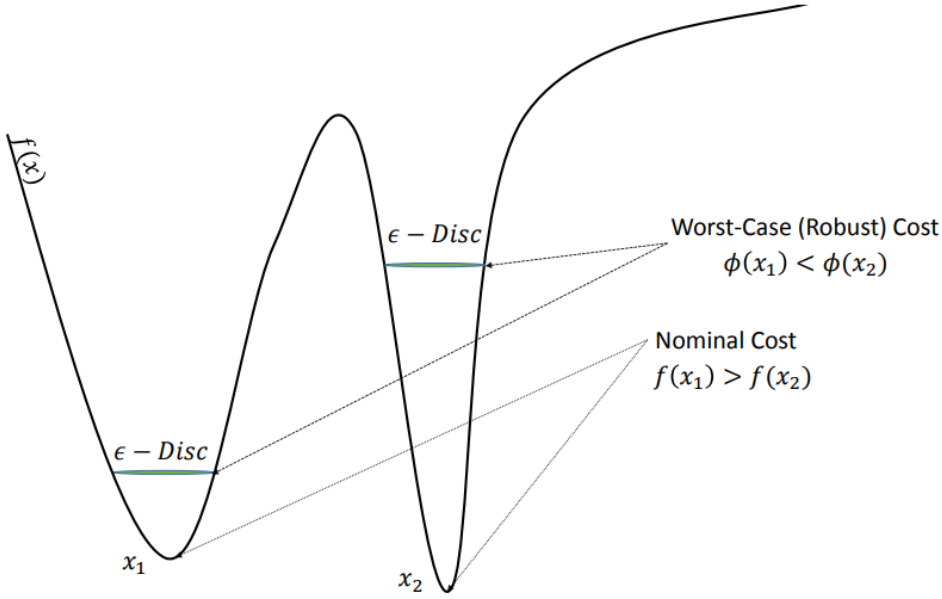


Figure 8: Illustration of Robust Optimization

在深度学习中，robustness的概念可以分为两个方面：即数据的robustness和优化过程的robustness。数据的robustness是吧CNN当成一个统计模型，而优化过程的robustness则是把CNN当成一个黑箱模型，而对于优化过程的robustness实际上就是对抗学习adversarial training。

对抗学习不同于数据增强对数据进行的那些常规操作，它是有目标的的增强对抗样本。但是在本文作者的实验中，对抗学习的泛化性能以及sharpness和baseline非常接近。因此，这种方法也不能够解决sharp minima的现象。

5. 总结与讨论

文章通过实验证明了收敛到sharp minima会导致深度学习的大批量方法的泛化性能不佳。并且，作者也给出了三个解决LB方法问题的方法，但是根据作者的实验，这几种方法虽然一定程度上提高了网络的泛化性能，但是sharp minima的现象依然存在。另一种可能的解决方案包括使用动态采样，其中批量大小随着迭代的进行而逐渐增加（Byrd et al., 2012; Friedlander & Schmidt, 2012）。文中的piggybacked LB实验表明这种方法可能的可行性。

最后，文中也抛出了几个问题：

（a）是否可以证明大批量（LB）方法通常会收敛到深度学习训练函数的sharp minima？（文中作者只提供了实验数据支持）

（b） 优化过程的sharp和flat两种minimum的密度如何？

（c） 能否设计出适合于LB方法特性的各种任务的神经网络架构？

（d） 能够找到一种合适的初始化方法使得大batch的方法能够收敛到flat minimum上面去？

（e）是否有可能通过算法或调控手段引导LB方法避开sharp minima？