1 实验背景及内容

随机梯度法是机器学习中非常重要的训练算法，对于机器学习中出现的大规模训练问题，随机梯度法又演化出迷你批量法(mini-batch)及其各种各样的用以减小梯度中噪声的一阶方法，为了提高随机梯度法的收敛速度，研究者也尝试了各种二阶方法。也讨论了深度学习中出现的非凸最优化问题的算法设计和目前的研究工作。

描述用于分类的 Logistic 回归模型，并就文本分类数据 RCV1 测试随机梯度法、mini-batch 方法和 LBFGS 方法的效率。

2 Logistic 回归模型介绍

逻辑回归(Logistic Regression)是一种基于概率的二分类模型。逻辑回归的类别预测函数和损失函数可以用多种等价假设引出，比如论文[22]以 p(Y=y|x)的 logit 变换等于 yp(w,x)为假设，推导得到逻辑回归的类别预测函数：

$$\ln\left(\frac{\mathbb{P}(Y=y|x)}{1-\mathbb{P}(Y=y|x)}\right)=yp(w,x).$$

下面从概率论和信息论的角度给出一个新的思路。考虑一个二分类问题，标签 y∈{0，1}，有 N 个从{X，Y}样本空间采样得到的数据对。

$$\{X,Y\}=\{xi\in R^m, yi\in\{0,1\}\}, i=1,2,\dots,N$$

假设 x 关于 y 的后验分布属于关于参数 wi 和 bi 指数分布族：

$$p(xk|yk=i)\propto \exp(wi^T xk+bi), i\in\{0,1\}$$

由 bayes 公式则可推出 p(y|x)的条件概率（不妨假设 p(xk=i)=0.5，i∈{0, 1}，因为可以并入 exp(bi)的 bi 分量中成为新的 bi'）：

$$p(yk=1|xk)=\frac{p(xk|yk=1)p(yk=1)}{p(xk|yk=1)p(yk=1)+p(xk|yk=0)p(yk=0)}$$

$$p(yk=1|xk)\propto\frac{1}{1+\exp\{(w0-w1)^T xk+(b0-b1)\}}$$

$$p(yk=0|xk) \propto \frac{1}{1+\exp\{(w1-w0)^T xk+(b1-b0)\}}$$

令 w = w1 – w0, b = b1 – b0，则可得到如下条件概率表达形式：

$$p(y_k=1|x_k)=\frac{1}{1+\exp\left(-(w^T x_k+b)\right)}$$

在类别预测的时候，当$p(yk=1|xk) \geq 0.5$的时候就是类别 1，反之则是类别 0。因此逻辑回归获得的决策边界$\{x|w^T x+b=0\}$是一个超平面（又称为线性边界），称$p(w,x)={w_1}^T x+w_0$为预测函数族。

而模型的损失函数就是要让$p(y|x;w,b)$条件分布与目标分布$q(y|x)$的“距离”尽可能小，目标分布$q(yk=1|xk)=yk, k=1,2,..,N$。信息论中一般用 KL 散度和交叉熵来衡量两个分布之间的差异，这里使用二元交叉熵（Binary cross entropy）作为两个分布的“距离”，得到的经验损失函数(empirical loss function)如下：

$$\hat{f}(\boldsymbol{w})=-\frac{1}{N}\sum_{i=1}^{N} yi*\log\big(p(yi=1|xi;\boldsymbol{w})\big)+(1-yi)*\log\left(p(yi=0|xi;\boldsymbol{w})\right)$$

则逻辑回归即可转化成如下无约束优化问题：

$$minimize\ \hat{f}(\boldsymbol{w})$$

之所以说$\hat{f}(\boldsymbol{w})$是经验损失函数，因为它依赖于训练集的 N 个样本：

$$\min_{w\in\mathcal{W}}\ \hat{f}(w),\quad \text{where}\quad \hat{f}(w):=\frac{1}{n}\sum_{i=1}^{n}\ell(p(w,x_i),y_i).$$

而期望损失函数(expected loss function)是对 x 和 y 的联合概率密度求积分：

$$\min_{w\in\mathcal{W}}\ f(w),\quad \text{where}\quad f(w):=\int_{\mathcal{X}\times\mathcal{Y}}\ell(p(w,x),y)dP(x,y)=\mathbb{E}[\ell(p(w,x),y)],$$

可以看到经验损失函数是关于期望损失函数的无偏估计量，而且当 N 很大时，由大数定理可得，经验损失函数依概率收敛于期望损失函数，可以近似代替期望损失函数。

值得注意的时，逻辑回归的经验损失函数$\hat{f}(\boldsymbol{w})$是关于 **w** 的凸函数，$\hat{f}(\boldsymbol{w})$的

稳定点是全局极小点。

除了二分类问题之外，逻辑回归还能解决多标签分类问题。多标签分类问题的标签 y∈{0, 1}^{d} 其中 d 代表标签的个数。相应的经验损失函数可以看成 d 个逻辑回归二分类损失函数的和取平均，本次大作业关于 RCV1 的文本分类问题就是一个**多标签分类问题**。

3 过拟合与正则化

3.1 过拟合

令$\hat{f}(\hat{w})$代表经验损失函数的极小值，极小值点为$\hat{w}$。$f(w_*)$代表期望损失函数极小值。两者差值的绝对值有一个上界：

$$|\hat{f}(\hat{w}) - f(w_*)| \leq |\hat{f}(\hat{w}) - f(\hat{w})| + |f(\hat{w}) - f(w_*)|.$$

其中不等式右边第一项表示在训练集上表现很好的参数$\hat{w}$，在整个样本空间上的表现与在训练集上的表现的差距。第二项表示期望损失$f(\hat{w})$与$f(w_*)$的差距。

机器学习领域更关注不等式右边第一项，它衡量的是训练模型的泛化效果（在训练集上表现很好的参数，在样本空间的表现如何），当这个值很大时(一般$f(\hat{w}) - \hat{f}(\hat{w}) > 0$ ）称这个模型在训练集上过拟合(ovrefit)了。一个通用的上界为，我们至少有$1-\delta$的概率可以确定：

$$|\hat{f}(w) - f(w)| \leq \mathcal{O}\left(\sqrt{\frac{C + \log(\frac{1}{\delta})}{n}}\right) \quad \text{for} \quad w \in \mathcal{W},$$

C 是一个衡量预测函数族$p(w,*)$复杂度的标量，n 是训练集样本的个数。可以看到当训练集样本增大，预测函数复杂度减小时可以缩小$f(\hat{w})$与$f(w_*)$的差距，防止过拟合。预测函数族复杂度有多种测度，在分类任务中比较常用的是 VC 维度（Vapnik-Chervonenkis dimension）。对一个线性预测函数族，比如逻辑回归，VC 维度即参数的个数。

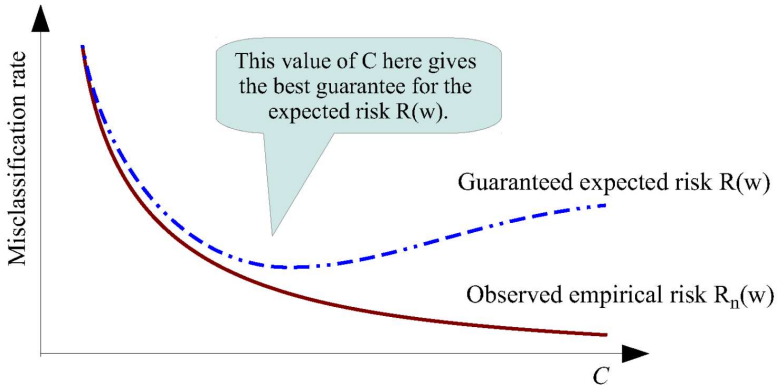
对于一般的预测函数族，有一个通用的结论：如果输入空间 X 被一个半径为 R_x 的二范数球约束，预测函数族 p(**w**, x)的 w 参数也被一个半径为 R_w 的二范数球约束，那么当损失函数是光滑的时候，C 的上界可以通过 $O(R_x^2 R_w^2)$得到。

3.2 正则化

在不增长训练集数量的情况下，使得模型泛化能力提高，$|f(\hat{w}) - \hat{f}(\hat{w})|$减小的技术称为正则化。通过前面的分析，一个对所有模型通用的正则化方法是在经验损失函数后面加一个正则化函数 r(w)，用来约束参数。比较常用的正则化函数有二范数、一范数等。

$$\min_{w \in \mathbf{R}^d} F(w), \quad \text{where} \quad F(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(p(w, x_i), y_i) + \lambda r(w)$$

这种技巧由 vapnik 提出，称为结构风险最小化(Structural Risk Minimization)。定义预测函数族 p(**w**, x)组成的集合为假设空间 H（Hypothesis space），选择一个结构，即一个受约束的函数族组成的集合。比如我们可以定义这样的假设空间子集$H_c \coloneqq \{h \in H : \Omega(\mathrm{h}) \leq \mathrm{C}\}$，其中 Ω 称为偏好函数，C 是一个参数。当 C 增大时，相当于约束条件**松弛**，则增大后的经验损失函数最优值将不大于之前的最优值。然而由前面的结论，此时预测函数的复杂度增加，经验损失函数和期望损失函数的间隙将增大，如下图所示：



在经验损失函数后面加上正则化函数 Rn(h)＋λΩ(h)相当于用 lagrange 函数求解约束优化问题。

$$minimize\ Rn(h) \quad s.t.\ \Omega(\mathrm{h}) \leq C$$

本次逻辑回归的经验损失函数加入二范数正则化函数，除了降低预测函数复杂度提高泛化性能之外，保证了损失函数是可微凸函数，也可以保证模型有最优值。另外，一般来说引入正则化项需要对超参数 λ 进行交叉验证选出最好的取值，这不是本次实验的重点，λ 取定值 0.01。

4 RCV1 数据集介绍

RCV1(Reuters Corpus Volume)是路透社提供的用于文本主题分类的数据集。该数据集 xi 是一个 47236 维的用于描述文本的特征向量（feature vector），yi 是一个 103 维的标签向量（label vector），每个样本在相应类别中的值为 1，在其他类别中的值为 0。一个文本可以属于多个类别，也就是标签向量可以有多处为 1。

数据集一共有 804414 个数据对，其中 23149 个数据对是训练集，781265 个数据对是测试集。本次大作业采用 RCV 数据集中的训练集进行数值试验。

5 实验结果及分析

RCV1 文本分类问题优化模型如下，其中 sigmoid 代表对向量每个分量 xi 取 1/(1+exp(-xi))。

$$P(Y|x) = \text{sigmoid}(Wx+b), \quad W \in R^{103\times 47236}, \ b \in R^{103}$$

经验损失函数的定义为：

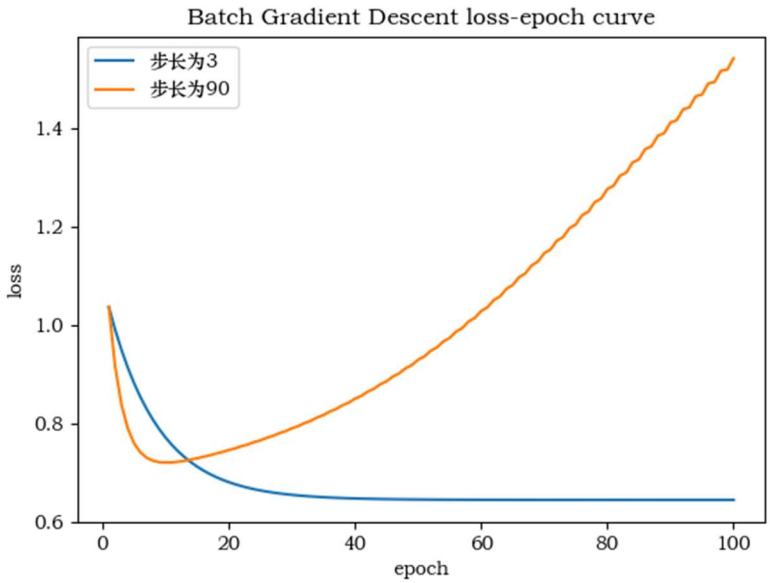
$$F(\boldsymbol{w}) = -\frac{1}{23149 * 103}\sum_{i=1}^{23149}\sum_{j=1}^{103} yij * logP(yij|xi;\boldsymbol{w}) + (1-yij) * \log\big(1-P(yij|xi;\boldsymbol{w})\big) + 0.01\,||\boldsymbol{w}||_2^2$$

5.1 批量梯度下降法（Batch Gradient descent）

使用经验损失函数作为目标函数进行梯度下降，进行非精确线搜索（如步长满足 Armijo 准则等）产生的步长序列$\{\alpha_k\}$可以保证大范围收敛性。然而，在大规

模机器学习问题上，当 N 很大时进行非精确线搜索的计算代价比较高。

取而代之的使用固定步长的梯度下降法，理论证明当取一个足够小的步长定值 α 时，算法依然可以收敛。当经验损失函数 F 是严格凸函数时，可以达到线性收敛速率，而 F 是非严格凸函数时，可以达到次线性收敛速率。下面是在 RCV1 上使用固定步长梯度下降法的结果。



在机器学习中训练集所有样本均进行训练后称为一个 epoch，使用 BGD 进行优化时，一个 Epoch 对应一次参数的更新。从图中可以看到当步长为 3 的时候损失函数可以收敛到极小点，且根据 loss 算得的收敛阶数为 1，速率常数<1,大约在 0.85 附近，说明 BGD 的收敛速率是线性的。而当步长为 90 的时候，BGD 的训练损失函数先下降后上升发散。以上数值实验说明固定步长的 BGD 方法，在步长较小的时候对严格凸函数可以达到线性收敛的收敛速率，而当步长较大则可能不收敛发散。

5.2 随机梯度下降法（Stochastic Gradient Descent）

虽然固定步长的梯度下降法省去了进行线搜索的复杂度，但是每次参数的更新的计算代价非常大。Robbins 和 Monro 发明了随机梯度下降法（Stochastic gradient descent），相较于梯度下降法参数更新 O(Nd)的计算复杂度，随机梯度下降法的计算复杂度只有 O(d)。虽然运算的复杂度降低了，但 SGD 的收敛速率比

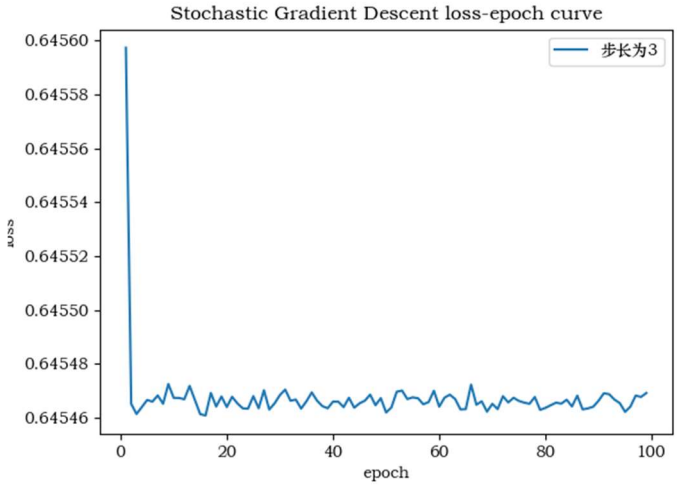
BGD 慢，一般是次线性收敛速度。另外，对一个强凸函数 SGD 也不能保证收敛到全局极小点，只能保证收敛到全局极小点附近。

SGD 的更新如下所示，其中 Sk 是一个从训练集中随机采样得到的势为 1 的集合。

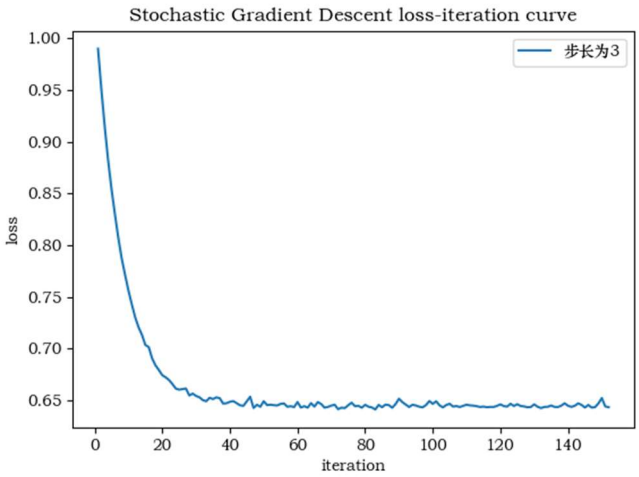
$$\nabla_{S_k} F(w) = -\frac{1}{|S_k|} \sum_{i \in S_k} \left(\frac{1}{1 + e^{y_i (w^T x_i)}} \right) y_i x_i + 2\lambda w,$$

$$w_{k+1} \leftarrow w_k - \alpha_k \nabla_{S_k} F(w_k).$$

下图为 RCV1 数据集上使用固定步长 SGD 的 loss-epoch 曲线，固定步长取 3。由于一个 epoch 要更新 23149 次参数，SGD 相对于 BGD 可以用更少的 epoch 到达全局极小点附近。



机器学习中一个 itertion 代表一次参数的更新，下图为取第一个 epoch 的前 150 个 iteration 计算的经验损失得到的 loss-iteration 曲线，可以看到由于随机梯度下降法每次迭代的方向并不一定是经验损失函数下降方向，loss 会出现震荡。



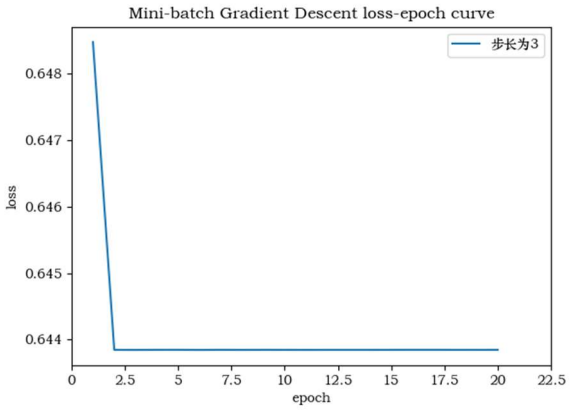
5.3 mini-batch 梯度下降法

mini-batch 梯度下降法综合了 SGD 和 BGD 的优势，它既不需要一次迭代取训练集所有样本，又通过增大采样集合 Sk 的势减小梯度的方差（也称为减小梯度的噪声）。

$$\nabla_{S_k} F(w) = -\frac{1}{|S_k|} \sum_{i \in S_k} \left(\frac{1}{1 + e^{y_i (w^T x_i)}} \right) y_i x_i + 2\lambda w,$$

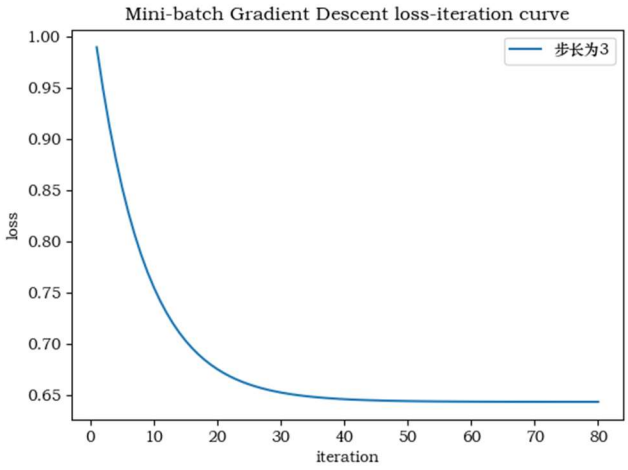
$$w_{k+1} \leftarrow w_k - \alpha_k \nabla_{S_k} F(w_k).$$

本次实验取 Sk 采样集合的势为 32，步长为 3，获得的 loss-epoch 曲线如下：



取第一个 epoch 的前 80 个 iteration 得到的 loss-iteration 曲线如下所示，可以看到 mini-batch 梯度下降法比 SGD 的下降趋势更加平滑，弱化了了 SGD 由于梯

度方差过大导致的震荡问题。



5.4 L-BFGS 方法

首先介绍 BFGS 方法，BFGS 方法是一种拟牛顿法，拟牛顿法的思路是用一个对称正定矩阵且满足拟牛顿方程的矩阵来近似代替 Hessian 矩阵，其中拟牛顿方程如下：

$$B^{(k+1)}\left(x^{(k+1)}-x^{(k)}\right)=g^{(k+1)}-g^{(k)}$$

令$s^{(k)}=x^{(k+1)}-x^{(k)}, y^{(k)}=g^{(k+1)}-g^{(k)}$,当满足曲率条件即${s^{(k)}}^T y^{(k)}>0$时拟牛顿方程一定有解，曲率条件在步长满足 Wolfe 条件或者强 Wolfe 条件下时对任意函数成立。而由于带正则项的逻辑回归经验损失函数是可微严格凸函数，故曲率条件一定成立。

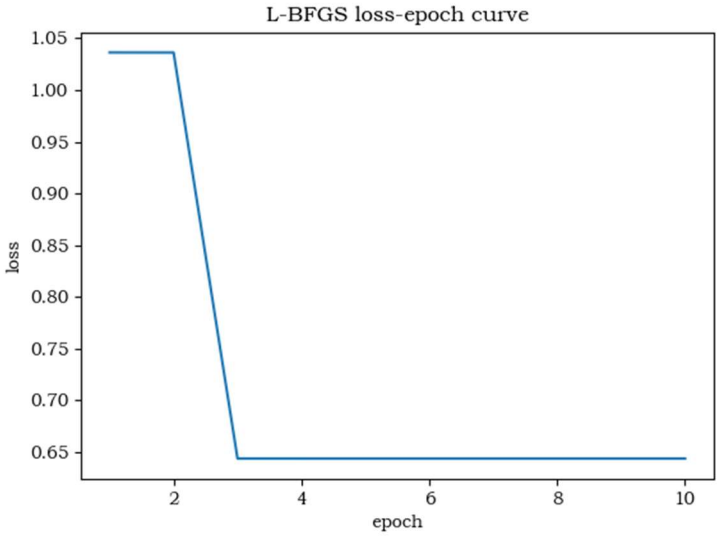
当曲率条件成立时，对称矩阵有 n(n+1)/2 个自由度，拟牛顿方程减去 n 个自由度，正定性要求 n 个额外条件，这时矩阵 B 仍有很大的自由度。BFGS 的要求$B^{(k+1)}$和$B^{(k)}$之间相差一个秩二矩阵。得到 BFGS 矫正公式如下：

$$B_{k+1}^{-1}=\left(I-\frac{s_k y_k^T}{s_k^T y_k}\right)B_k^{-1}\left(I-\frac{y_k s_k^T}{y_k^T s_k}\right)+\frac{s_k s_k^T}{y_k^T s_k},$$

令$H^{(k)}$为$B^{(k)}$的逆矩阵，BFGS 算法中每次迭代计算$H^{(k+1)}$需要$H^{(k)}$，存储空间复杂度为$O(N^2)$，对于参数巨大的模型，存储空间将是非常大的。L-BFGS 的基

本思想就是通过存储前 m 次迭代的少量数据来替代$H^{(k)}$。首先存储下所有的$s^{(k)}$和$y^{(k)}$，要计算$H^{(M)}$就用单位矩阵和存储下的$y^{(k)}$，$s^{(k)}k = \{1, 2, .., M\}$计算，用时间换空间的办法让存储复杂度变为$O(N * M)$。但当迭代次数$M$很大时，内存存不下所有的$y^{(k)}$和$s^{(k)}$，这时只能丢掉一些数据。L-BFGS 需要设置存储向量数 w，当 s 和 y 迭代超过 w 时,就会扔掉第一个 s 和 y，加上新的 s 和 y，以此类推。虽然损失了精度，但可以保证存储空间复杂度为$O(N * w)$。L-BFGS 算法是对 BFGS 算法的又一次近似。L-BGFS 收敛速度是超线性收敛。

L-BFGS 在大规模机器学习问题上又衍生出随机 L-BFGS 法，本次数值试验采取对经验损失函数（整个 Batch）进行 L-BFGS 优化，不采用线搜索方法，且设置存储向量数为 100，得到的损失曲线如下所示：



可以看到经过 3 次参数更新后就达到了全局极小点，L-BFGS 的收敛速率是超线性的。

6 总结

本次最优化大作业，用 RCV 数据集上带二范数正则项的逻辑回归的经验损失函数（凸函数）作为目标函数进行不同优化方法的测试。

在收敛速率方面，终止条件为函数值距离最优函数值小于 0.001（最优值取 L-BFGS 第十次迭代的函数值）。L-BFGS 需要 3 次迭代，BGD 需要 90 次迭代，

mini-Batch Gradient descent 需要 70 次迭代，SGD 需要 46 次迭代。L-BFGS 由于利用了曲率信息收敛速率是超线性的，是最快的；其次固定步长的 SGD，然后是固定步长 mini-batch GD，最后是线性收敛的固定步长 BGD。比较神奇的是，BGD 虽然每次更新都能保证是下降方向，但是收敛到全局极小点的速度并没有 SGD 和 mini-batch GD 方法快。

在空间复杂度方面，BGD 和 L-BFGS 占用的内存空间较大，都需要用整个训练集的样本进行梯度更新，除此之外，L-BFGS 还需要存储额外 s 和 y 的数据信息。Mini-batch GD 和 SGD 一次参数更新只利用训练集中部分的样本，因此每次参数更新的计算复杂度较小。

综合收敛速度、存储空间复杂度、收敛曲线平稳程度来看，mini-batch GD 的收敛效果快且下降平稳的，但是 batch-size 是超参数，一般需要多次试验对比获得。

论文

[1] V. N. Vapnik and A. Y. Chervonenkis. Theory of Pattern Recognition. Nauka, Moscow, 1974. German Translation: Theorie der Zeichenerkennung, Akademie–Verlag, Berlin, 1979.