China · Beijing

Fregata:
轻量级大规模机器学习算法库

张夏天

Chief Data Scientist，TalkingData

ArchSummit
全 球 架 构 师 峰 会 2016

[北京站]

主办方 Geekbang›. 极客邦科技 InfoQ

大纲

- 大规模机器学习的挑战
- Fregata的优点
- GSA算法介绍
- GSA算法在Spark上的并行化
- 与MLLib的对比
- 如何使用Fregata
- Fregata的发展目标

大规模机器学习两个挑战

计算瓶颈

调参困难

经典算法的计算瓶颈

计算复杂度随数据规模超线性增长

	single	multi
LWLR	$O(mn^2 + n^3)$	$O(\frac{mn^2}{P} + \frac{n^3}{P'} + n^2 \log(P))$
LR	$O(mn^2 + n^3)$	$O(\frac{mn^2}{P} + \frac{n^3}{P'} + n^2 \log(P))$
NB	$O(mn + nc)$	$O(\frac{mn}{P} + nc \log(P))$
NN	$O(mn + nc)$	$O(\frac{mn}{P} + nc \log(P))$
GDA	$O(mn^2 + n^3)$	$O(\frac{mn^2}{P} + \frac{n^3}{P'} + n^2 \log(P))$
PCA	$O(mn^2 + n^3)$	$O(\frac{mn^2}{P} + \frac{n^3}{P'} + n^2 \log(P))$
ICA	$O(mn^2 + n^3)$	$O(\frac{mn^2}{P} + \frac{n^3}{P'} + n^2 \log(P))$
k-means	$O(mnc)$	$O(\frac{mnc}{P} + mn \log(P))$
EM	$O(mn^2 + n^3)$	$O(\frac{mn^2}{P} + \frac{n^3}{P'} + n^2 \log(P))$
SVM	$O(m^2n)$	$O(\frac{m^2n}{P} + n \log(P))$

Cheng T. Chu, Sang K. Kim, Yi A. Lin, Yuanyuan Yu, Gary R. Bradski, Andrew Y. Ng, Kunle Olukotun， **Map-Reduce for Machine Learning on Multicore, NIPS, 2006.**

梯度下降法

$$w := w - \eta \nabla Q(w)$$

随机梯度下降法

$$w := w - \eta \nabla Q_i(w)$$

三大计算瓶颈

IO开销

通信开销

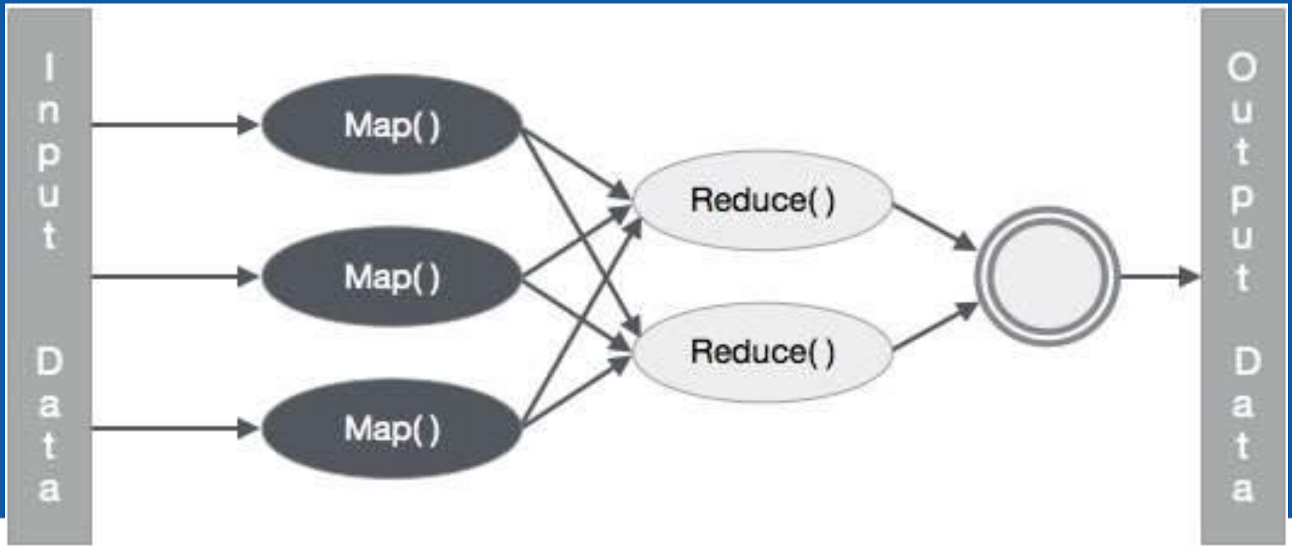
模型规模

Map Reduce

IO开销:　可通过内存/SSD加速来缓解

通信开销: 无法解决

模型规模: 无法解决

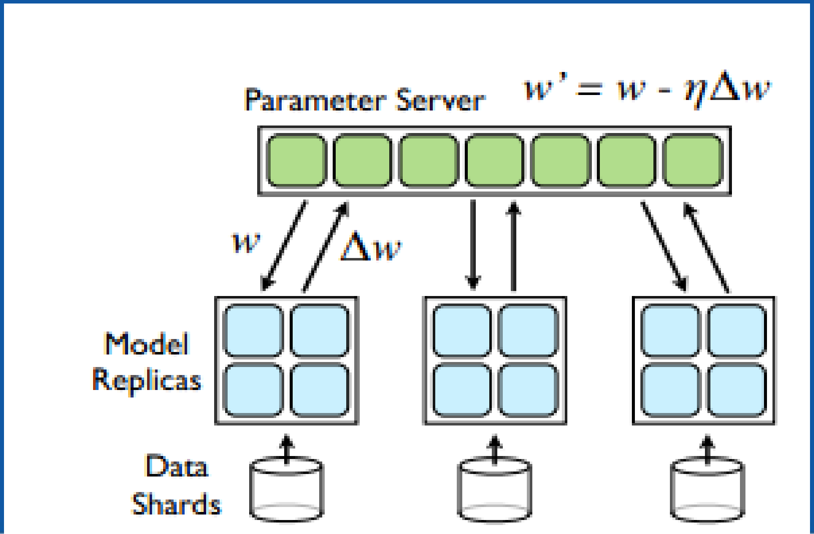


Parameter Server

IO开销:　可通过内存/SSD加速来缓解

通信开销: 通过异步更新部分缓解

模型规模: 分布式管理，解除了模型规模限制



调参困难

- 参数搜索空间大
- 对经验依赖比较大

知乎 搜索你感兴趣的内容… 首页 话题 发现 消息

数据挖掘 机器学习 计算机科学 深度学习（Deep Learning） 玄学（广义的）

修改

调参这事儿，为什么越干越觉得像老中医看病？ 修改

只能通过摸索调参的经历，让我觉得很不严谨很不舒服，这个过程真的好烦。

在不同的模型下都有不同的超参数，而每个超参的意义又不同。

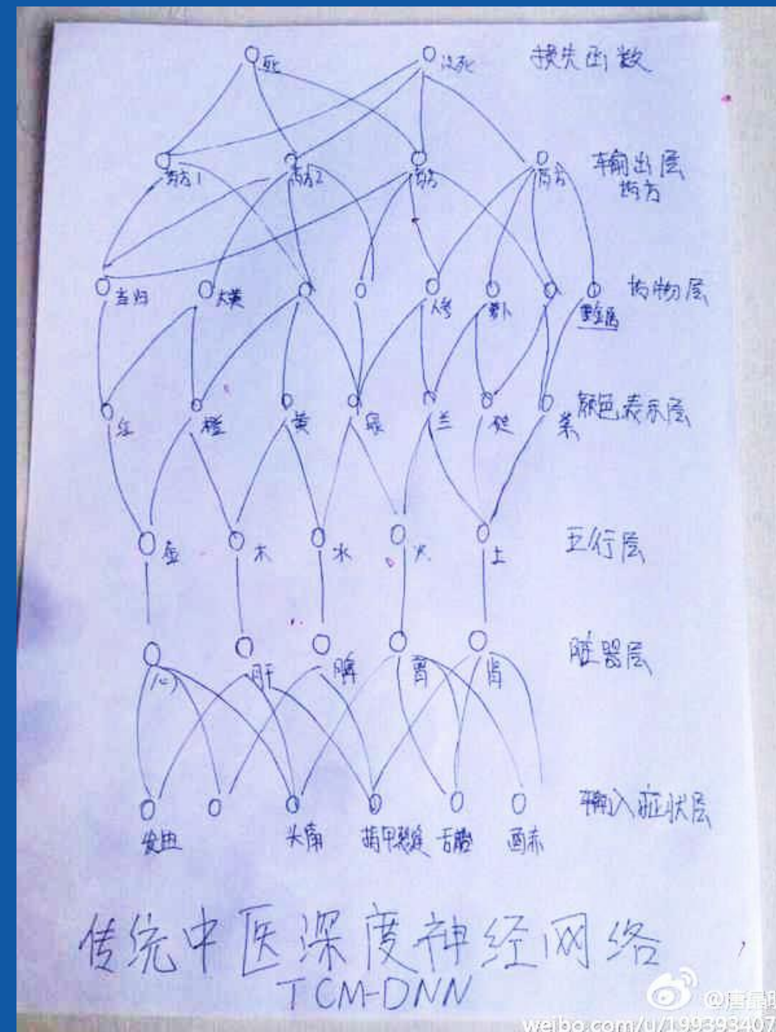
在不同实验结果下，调的参数和调的方向又都不一样。

有没有可能做出比较成熟的自动化调参方案呢？

丫的这拓码简直就是玄学啊？ 修改

13 条评论 分享 • 邀请回答 举报

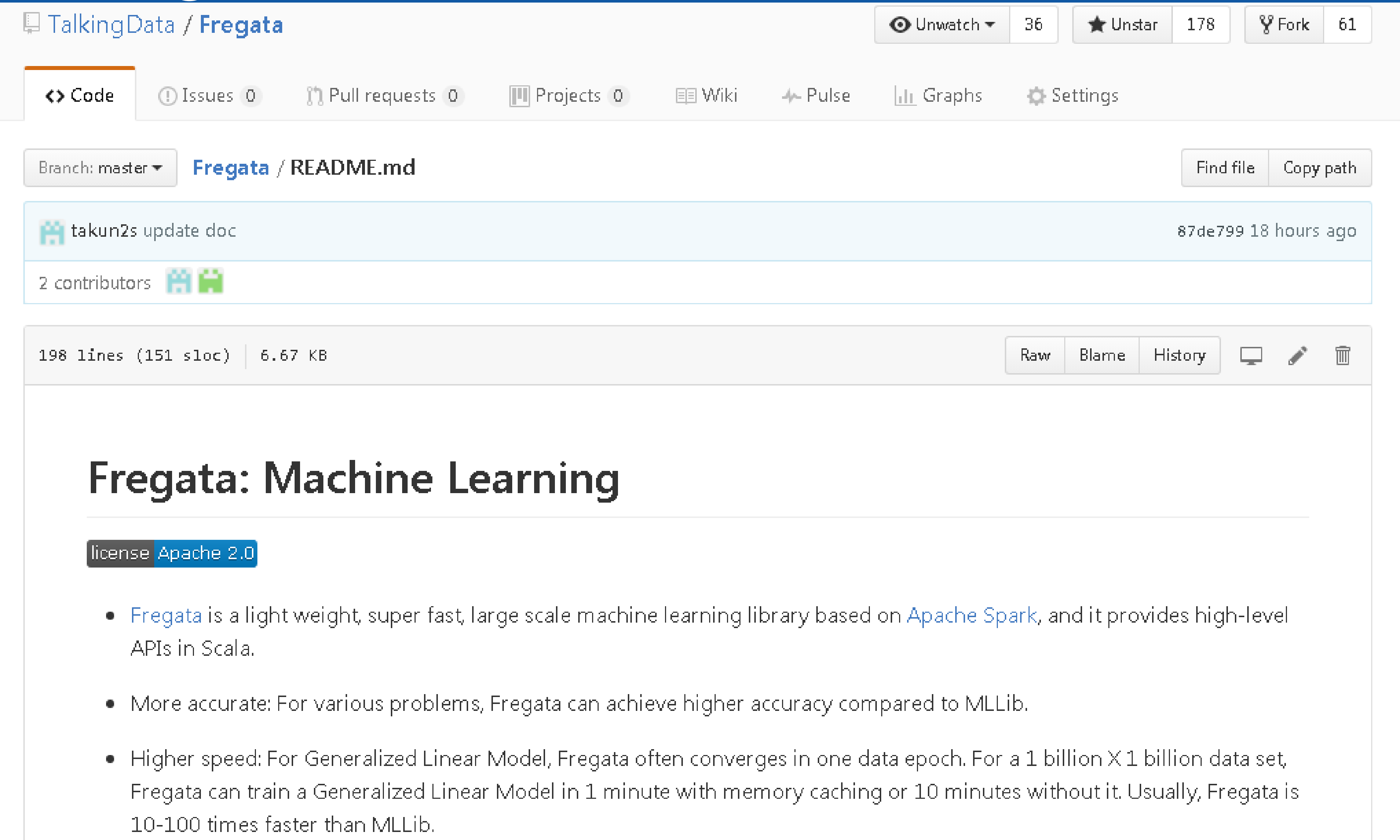
35 个回答 默认排序



大纲

- 大规模机器学习的挑战
- Fregata的优点
- GSA算法介绍
- GSA算法在Spark上的并行化
- 与MLLib的对比
- 如何使用Fregata
- Fregata的发展目标

Fregata项目 https://github.com/TalkingData/Fregata



Fregata项目

- 基于Spark，目前支持1.6
- 目前实现了四种算法
  - Logistic Regression
  - Combine Features Logistic Regression
  - Softmax
  - Random Decision Trees

Fregata项目的特点

速度快

- 只需要扫描一遍数据

调参容易

- LR和Softmax算法不需要调参
- RDT调参容易

大纲

- 大规模机器学习的挑战
- Fregata的优点
- GSA算法介绍
- GSA算法在Spark上的并行化
- 与MLLib的对比
- 如何使用Fregata
- Fregata的发展目标

https://arxiv.org/abs/1611.03608

Greedy Step Averaging优化方法

Greedy Step Averaging: A parameter-free stochastic optimization method

Xiatian Zhang*, Fan Yao*, and Yongjun Tian*

*TalkingData Technology(Beijing)Co,.Ltd, China,
Email: {xiatian.zhang, fan.yao, yongjun.tian}@tendcloud.com

November 14, 2016

Abstract

In this paper we present the greedy step averaging(GSA) method, a parameter-free stochastic optimization algorithm for a variety of machine learning problems. As a gradient-based optimization method, GSA makes use of the information from the minimizer of a single sample's loss function, and takes average strategy to calculate reasonable learning rate sequence. While most existing gradient-based algorithms introduce an increasing number of hyper parameters or try to make a trade-off between computational cost and convergence rate, GSA avoids the manual tuning of learning rate and brings in no more hyper parameters or extra cost. We perform exhaustive numerical experiments for logistic and softmax regression to compare our method with the other state of the art ones on 16 datasets. Results show that GSA is robust on various scenarios.

Keywords Optimization, algorithm, learning rate, parameter-free, self-adaptive, averaging strategy

GSA优点

SGD方法需要调学习率

衍生方法Adadelta, ADMM, SVRG同样存在着调参的问题，有些还需要付出更大的存储代价

GSA 方法不需要调参

GSA算法流程图

Algorithm 2 GSA algorithm in general

Require: Initial parameter ω_0, loss function $L(\omega) = \sum_{i=1}^{N} l_i(\omega)$

for t in $i \in [0, T]$ **do**

  Take a Training Sample (x_t, y_t);

  Compute Stochastic Gradient $g_t = \frac{\partial l_t}{\partial \omega}$;

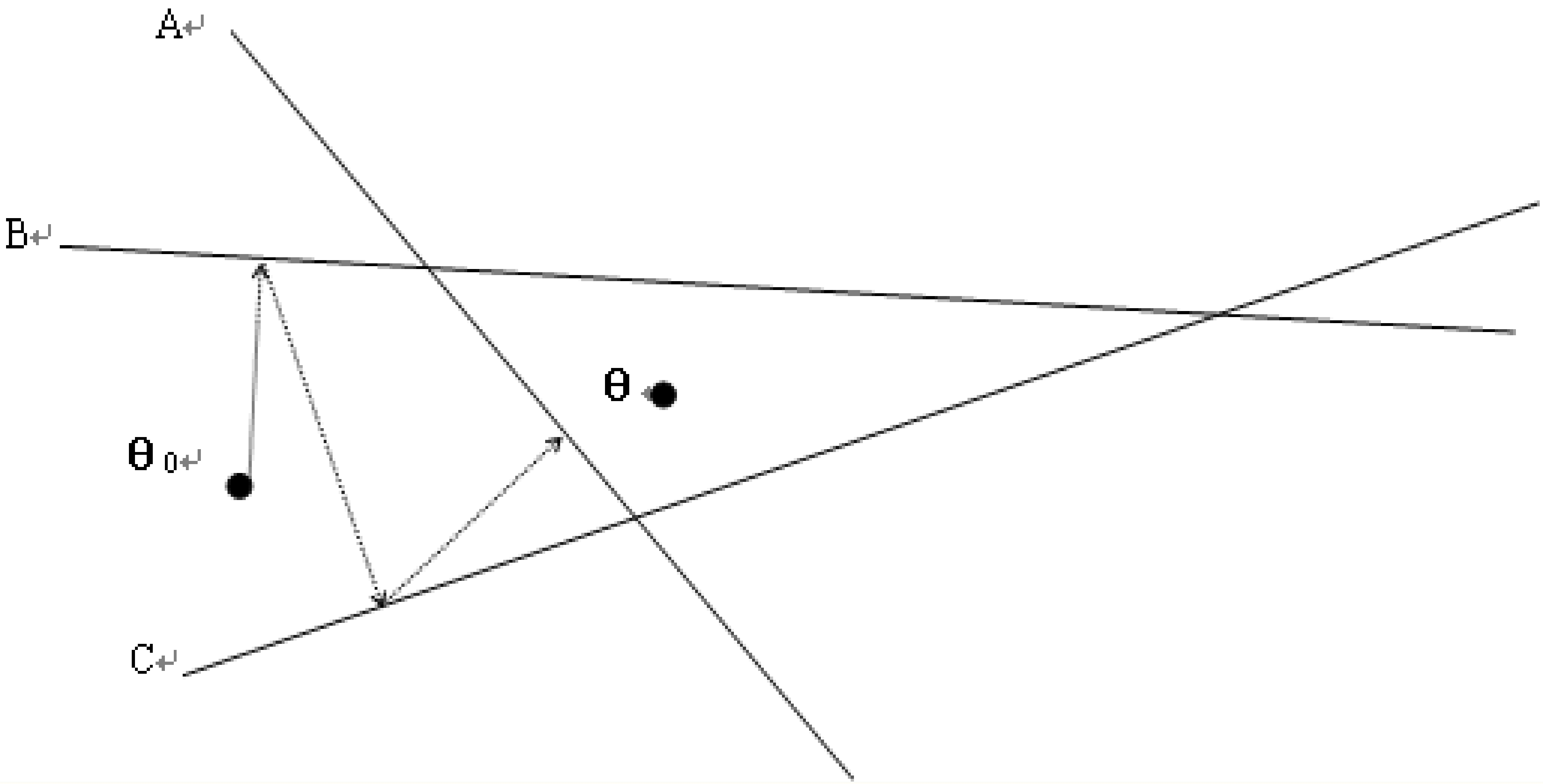
  Compute Greedy Step Size η_t by exact line search on ${}_t(\omega_t - \eta g_t)$;

  Compute Averaged Greedy Step Size $\bar{\eta} = mean(\eta_t)$;

  Apply Update $\omega_{t+1} = \omega_t - \bar{\eta} g_t$;

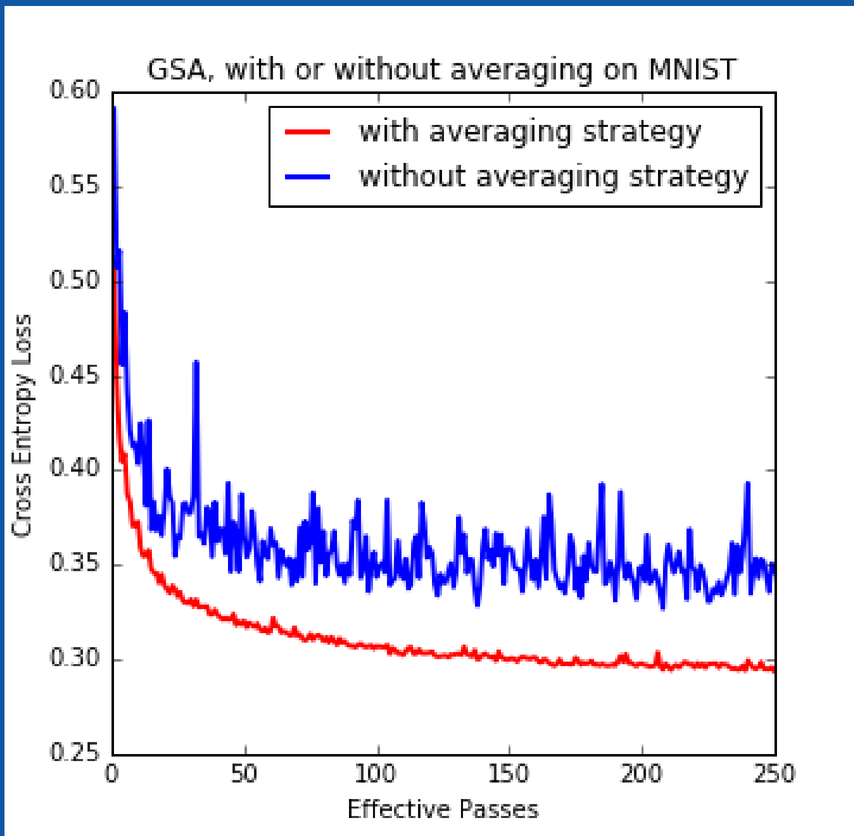
end for

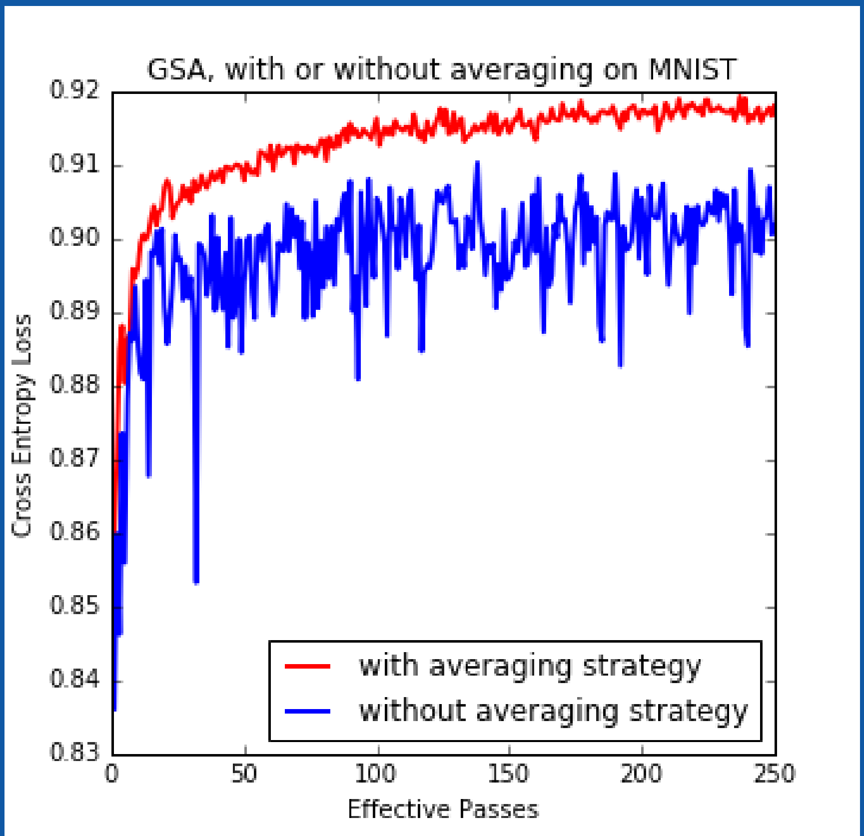
Greedy Step方法



Averaging 策略

$$E[\eta]_t = \frac{t-1}{t} E[\eta]_{t-1} + \frac{1}{t} \eta_t.$$





Logistic Regression & Softmax via GSA

Logistic Regression 学习率公式

$$\eta_{t,i} = \frac{\operatorname{sgn}(y_i - 0.5)\log(\hat{p}_1/\hat{p}_0) - \omega^{(t)} \cdot x_i}{x_i^T x_i (y_i - p_i)}.$$

Softmax学习率公式

$$\eta = \frac{-\hat{p}_k \sum_{j=1}^{L} e_j + e_k}{\hat{p}_k \sum_{j=1}^{L} e_j(1-b_j) + e_k - e e_k/b_k} \cdot \frac{1}{x^T x}$$

GSA LR & Softmax 实验 – 数据

Table 2: Dataset information

dataset	#instance(train/test)	#feature	#class
w1a	2477/47272	300	2
mnist.scale	60000	780	10
news20.scale	15935/3993	62061	20
aloi.scale	108000	128	1000
a9a	32561	123	2
breast-cancer_scale	683	10	2
gistte_scale	6000/1000	5000	2
madelon	2000/600	500	2
cod-rna	59535	8	2
url	2,396,130	3231961	2
letter.scale	15000/5000	16	26
dna.scale	2000/1186	180	3
sector.scale	6412/3207	55197	105
usps	7291/2007	256	10
protein	17766 /6621	357	3
rcv1.multiclass	15564/518571	47236	53

GSA LR & Softmax 实验 – 对比算法和总体结果

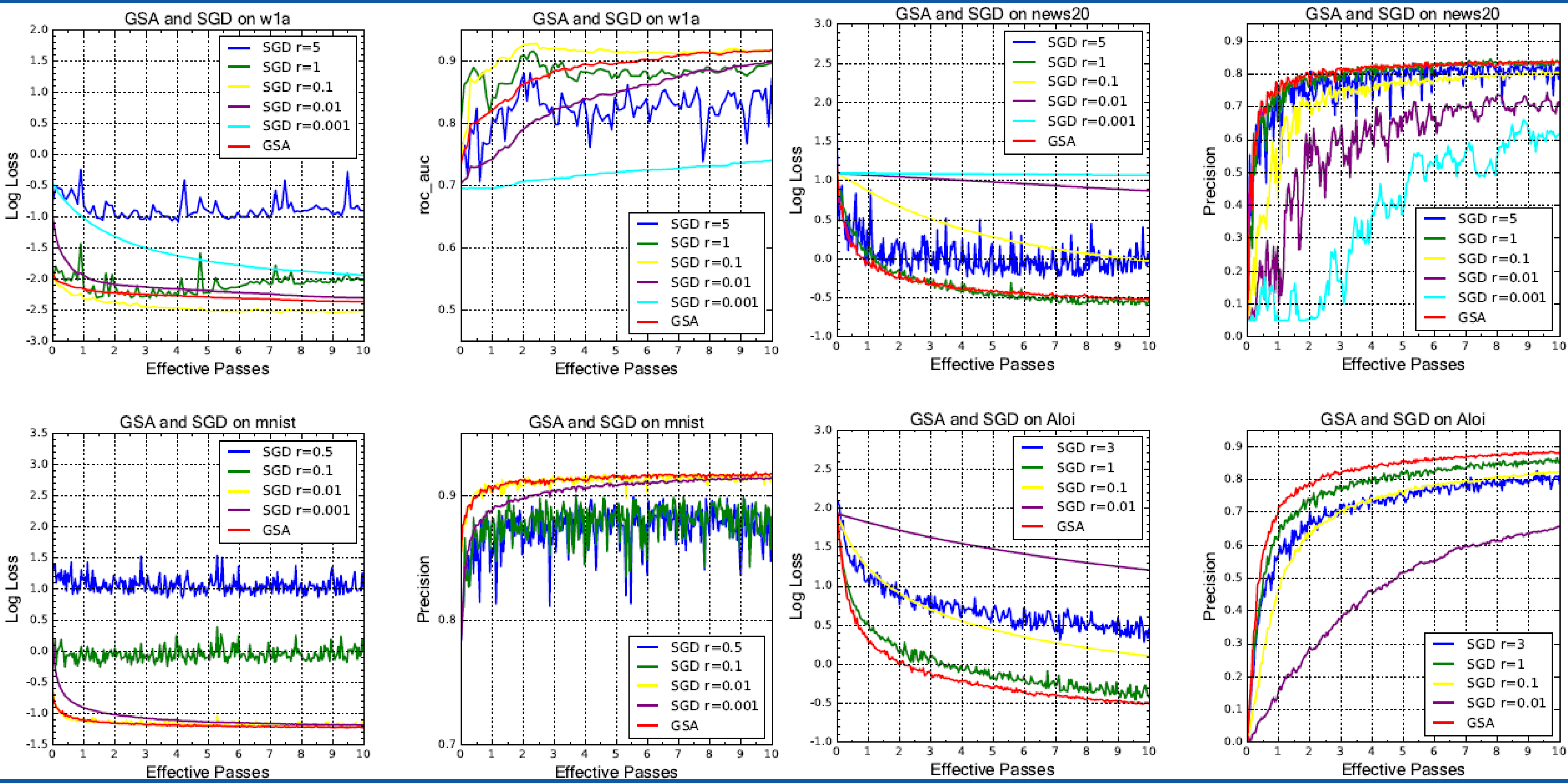
对比算法

SGD， Adadelta， SCSG

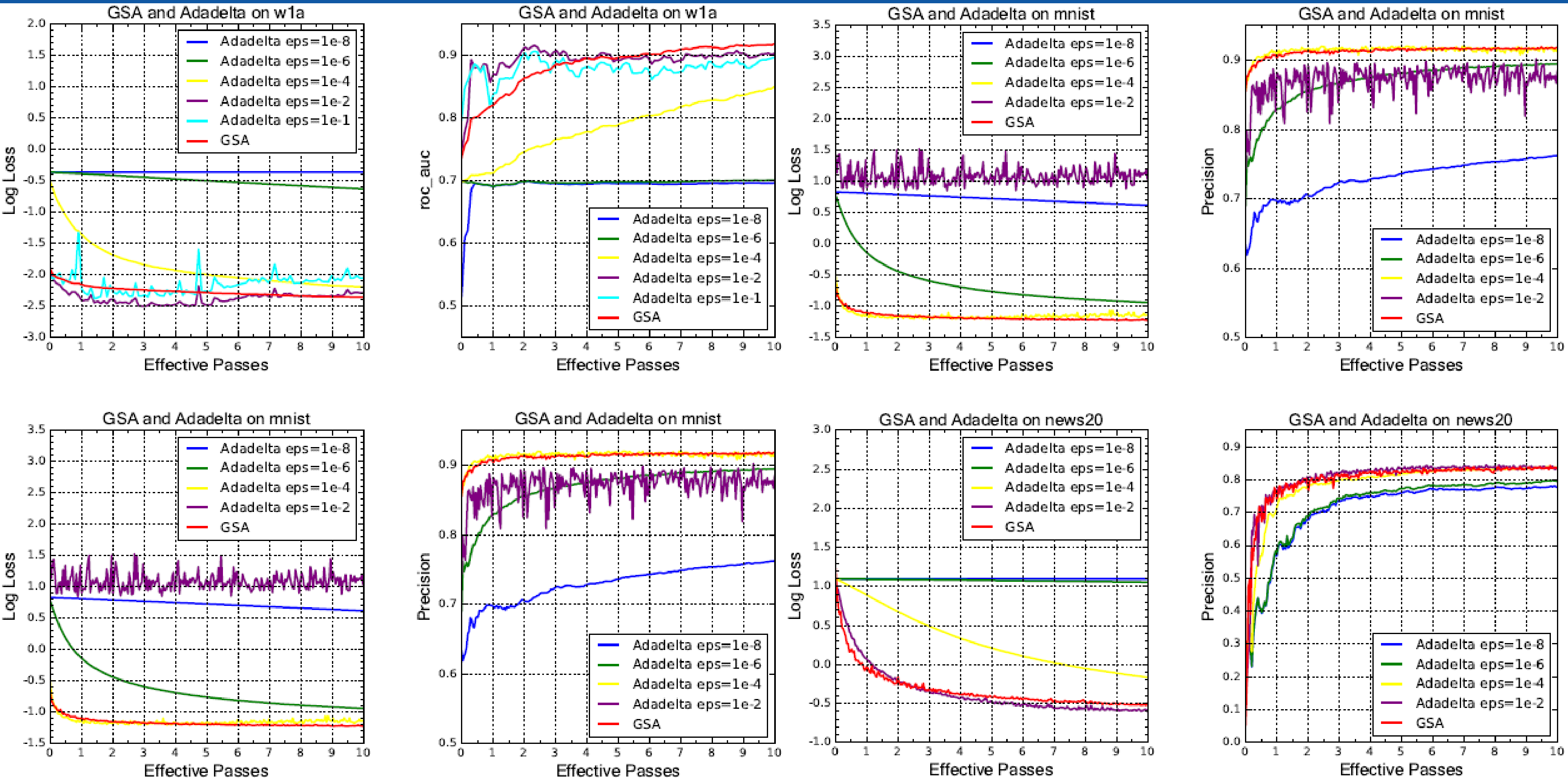
Table 1: Comparison of average performance

problem	metric	loss			prec.			auc		
		1	2	last	1	2	last	1	2	last
LR	mean(Err)	0.056	0.016	0.019	-0.016	-0.018	-0.011	-0.009	-0.005	-0.001
	#best out of 7	0	0	1	0	1	0	3	2	3
Softmax	mean(Err)	0.026	0.030	0.038	0.004	-0.004	-0.001	/	/	/
	#best out of 9	2	2	3	3	3	5	/	/	/

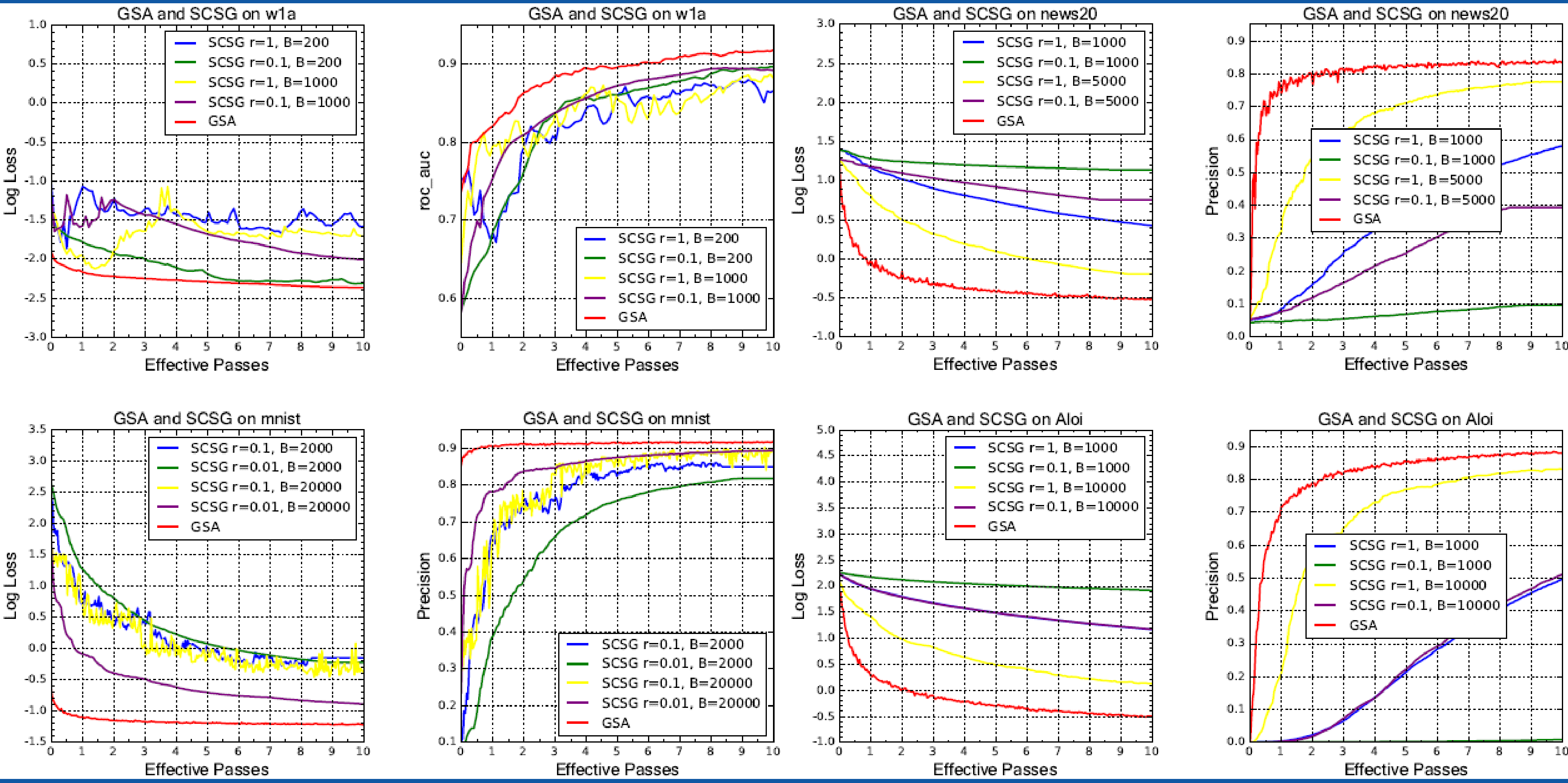
GSA LR & Softmax 实验 – GSA vs SGD



GSA LR & Softmax 实验 – GSA vs Adadelta



GSA LR & Softmax 实验 – GSA vs SCSG



大 纲

- 大规模机器学习的挑战
- Fregata的优点
- GSA算法介绍
- GSA算法在Spark上的并行化
- 与MLLib的对比
- 如何使用Fregata
- Fregata的发展目标

大规模机器学习并行化方法

梯度平均

$$w_t = w_{t-1} - \frac{\eta}{n}\sum_{i=0}^{n} \nabla Q_i(w_{t-1})$$

模型平均

$$w_t = \frac{1}{n}\sum_{i=0}^{n} w_{t-1,i}$$

结果平均

$$y_j = \frac{1}{m}\sum_{k=0}^{m} y_{j,k}$$

模型平均的收敛性

当N个样本均匀分配给m台机器训练出m个p维的模型，n=N/m时，对线性模型且当n>>p时逼近效果是比较好的，当p很大时，误差和m呈线性关系。对非线性模型，误差包含二阶项，可能会很大。

On the optimality of averaging in distributed statistical learning

Jonathan D. Rosenblatt†
Department of Industrial Engineering and Management, Ben Gurion University of the Negev, Be'er-Sheva, Israel
†Corresponding author. Email: johnros@bgu.ac.il

AND

Boaz Nadler
Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel
Email: boaz.nadler@weizmann.ac.il

[Received on 10 June 2015; revised on 7 February 2016; accepted on 5 April 2016]

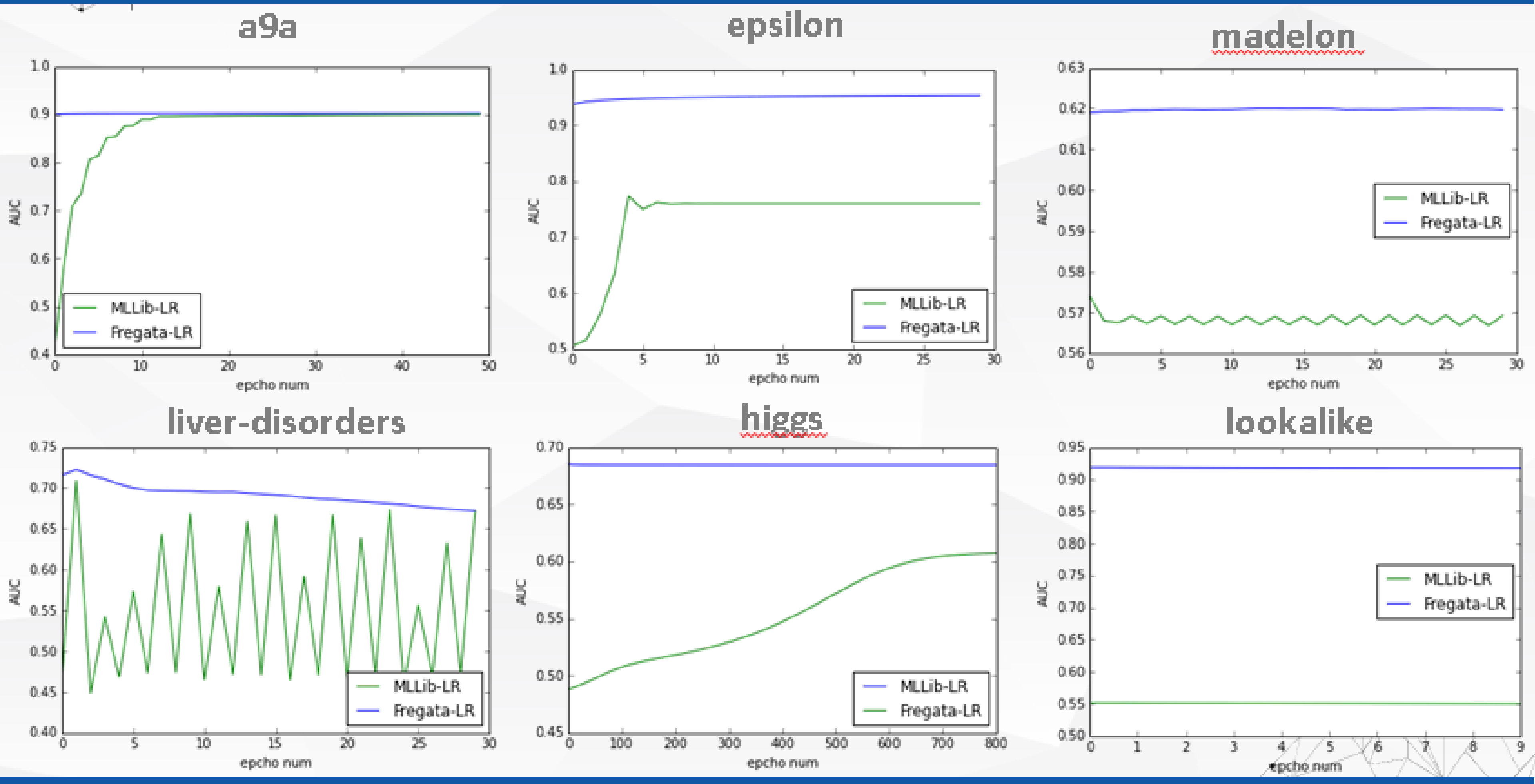
A common approach to statistical learning with Big-data is to randomly split it among m machines and learn the parameter of interest by averaging the m individual estimates. In this paper, focusing on empirical risk minimization or equivalently M-estimation, we study the statistical error incurred by this strategy. We consider two large-sample settings: first, a classical setting where the number of parameters p is fixed, and the number of samples per machine $n \to \infty$. Second, a high-dimensional regime where both $p, n \to \infty$ with $p/n \to \kappa \in (0, 1)$. For both regimes and under suitable assumptions, we present *asymptotically exact* expressions for this estimation error. In the fixed-p setting, we prove that to leading order averaging is *as accurate as* the centralized solution. We also derive the second-order error terms, and show that these can be non-negligible, notably for nonlinear models. The high-dimensional setting, in contrast, exhibits a qualitatively different behavior: data splitting incurs a first-order accuracy loss, which increases linearly with the number of machines. The dependence of our error approximations on the number of machines traces an interesting accuracy-complexity tradeoff, allowing the practitioner an informed choice on the number of machines to deploy. Finally, we confirm our theoretical analysis with several simulations.

Rosenblatt J D, Nadler B. **On the optimality of averaging in distributed statistical learning[J]. Information and Inference, 2016: iaw013 MLA**

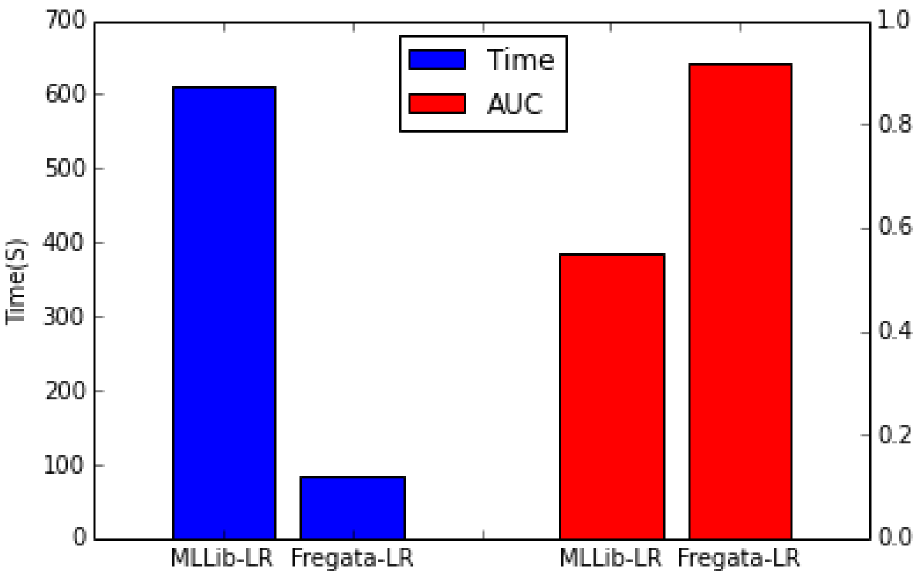
大纲

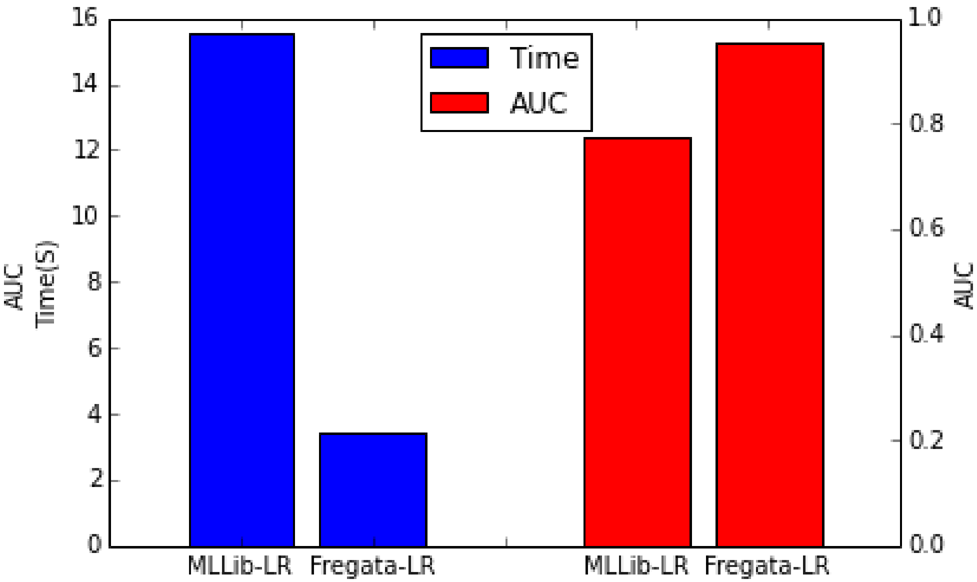
- 大规模机器学习的挑战
- Fregata的优点
- GSA算法介绍
- GSA算法在Spark上的并行化
- 与MLLib的对比
- 如何使用Fregata
- Fregata的发展目标

LR 实验结果1


a9a
epsilon
madelon
liver-disorders
higgs
lookalike
MLLib-LR
Fregata-LR
AUC
epcho num

LR实验结果2

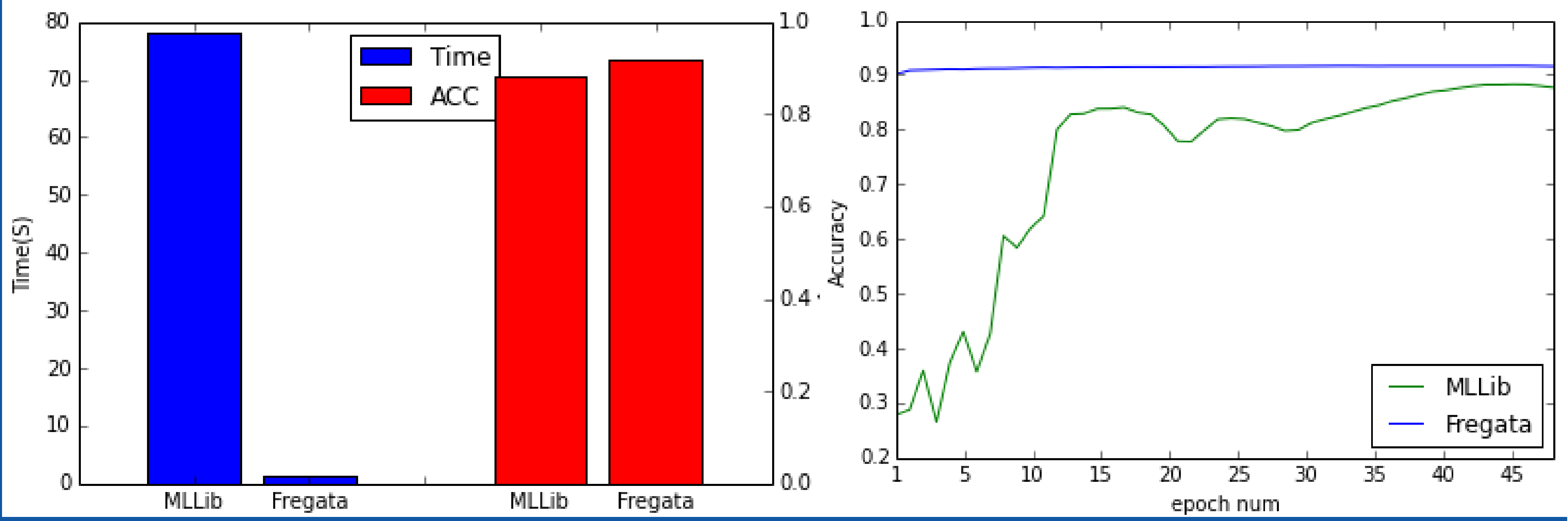




Lookalike
300million X 20 million dataset
0.01% postive class instances

Epsilon
400000X 2000 dataset

Softmax实验结果 - MNIST



大 纲

- 大规模机器学习的挑战
- Fregata的优点
- GSA算法介绍
- GSA算法在Spark上的并行化
- 与MLLib的对比
- 如何使用Fregata
- Fregata的发展目标

Maven配置

```
<dependency>
    <groupId>com.talkingdata.fregata</groupId>
    <artifactId>core</artifactId>
    <version>0.0.1</version>
</dependency>
<dependency>
    <groupId>com.talkingdata.fregata</groupId>
    <artifactId>spark</artifactId>
    <version>0.0.1</version>
</dependency>
```

SBT配置

```
libraryDependencies += "com.talkingdata.fregata" % "core" % "0.0.1"

libraryDependencies += "com.talkingdata.fregata" % "spark" % "0.0.1"
```

LR算法示例

```
import fregata.spark.data.LibSvmReader
import fregata.spark.metrics.classification.{AreaUnderRoc, Accuracy}
import fregata.spark.model.classification.LogisticRegression
import org.apache.spark.{SparkConf, SparkContext}
......
//加载数据
val (_, trainData)  = LibSvmReader.read(sc, trainPath, numFeatures.toInt)
val (_, testData)  = LibSvmReader.read(sc, testPath, numFeatures.toInt)
//训练模型
val model = LogisticRegression.run(trainData)
val pd = model.classPredict(testData)
//测试AUC指标
val auc = AreaUnderRoc.of( pd.map{
    case ((x,l),(p,c)) =>
      p -> l
  })
```

大纲

- 大规模机器学习的挑战
- Fregata的优点
- GSA算法介绍
- GSA算法在Spark上的并行化
- 与MLLib的对比
- 如何使用Fregata
- Fregata的发展目标

Fregata的目标

轻量级

高性能

易使用

THANKS

邮箱：xiatian.zhang@gmail.com

微博：张夏天_机器学习