基于差异合并的分布式随机梯度下降算法

陈振宏[1),2)] 兰艳艳[1)] 郭嘉丰[1)] 程学旗[1)]

[1)](中国科学院计算技术研究所网络数据科学与技术重点实验室 北京 100190)

[2)](中国科学院大学 北京 100190)

摘 要 大规模随机梯度下降算法是近年来的热点研究问题,提高其收敛速度和性能具有重要的应用价值.大规模随机梯度下降算法可以分为数据并行和模型并行两大类.在数据并行算法中,模型合并是一种比较常用的策略.目前,基于模型合并的随机梯度下降算法普遍采用平均加权方式进行合并,虽然取得了不错的效果,但是,这种方式忽略了参与合并的模型的内在差异性,最终导致算法收敛速度慢,模型的性能及稳定性较差.针对上述问题,该文在分布式场景下,提出了基于模型差异进行合并的策略,差异性主要体现在两方面,各模型在其训练数据上错误率的差异和训练不同阶段模型合并策略的差异.此外,该文对合并后的模型采用规范化技术,将其投射到与合并前模型Frobenius范数相同的球体上,提高了模型的收敛性能.作者在Epsilon、RCV1-v2和URL 3个数据集上,验证了提出的基于差异合并的分布式随机梯度下降算法相对于平均加权方式具有收敛速度更快、模型性能更好的性质.

关键词 分布式;随机梯度下降;规范化;模型合并;社交网络;社会计算

中图法分类号 TP311 **DOI号** 10.11897/SP.J.1016.2015.02054

Distributed Stochastic Gradient Descent with Discriminative Aggregating

CHEN Zhen-Hong[1),2)] LAN Yan-Yan[1)] GUO Jia-Feng[1)] CHENG Xue-Qi[1)]

[1)](*Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100190)

[2)](*University of Chinese Academy of Sciences, Beijing* 100190)

Abstract Large scale stochastic gradient descent has been a popular research topic. It is important to improve its convergence speed and performance in practice. Currently, large scale stochastic gradient descent algorithms can be roughly divided into two classes, data-parallel and model parallel. In data-parallel, model aggregating is a popular strategy. State-of-the-art model aggregating methods simply average different models together and it has been proved to be effective. However, the simply average method may neglect the differences between different models, thus may slow down the convergence speed and harm the performance of the algorithm. In this paper, we propose a distributed stochastic gradient descent algorithm with discriminative aggregating to avoid the above problems. Our discriminative aggregating algorithm differs to existing average method in two aspects. Firstly, we take into account the training performance of each model on training data. Secondly, we exploit the dynamic importance of different models as the training process moves on. Meanwhile, to further improve model performance, we project the combined model onto the surface sphere of the local model respectively, using Frobenius norm. Experimental

收稿日期:2014-08-19;**最终修改稿收到日期**:2015-03-25.**本课题得到国家"九七三"重点基础研究发展规划项目基金**(2012CB316303,2014CB340401)、**国家"八六三"高技术研究发展计划项目子课题基金**(2012AA011003)、**国家自然科学基金重点基金**(61232010)、**国家自然科学基金杰出青年学者基金**(61203298,61003166)**资助**.**陈振宏**,**男**,1988**年生**,**博士研究生**,**主要研究方向为大规模机器学习、分布式系统、社交网络**.E-mail:chenzhenhong@software.ict.ac.cn.**兰艳艳**,**女**,1982**年生**,**博士**,**副研究员**,**主要研究方向为机器学习、排序学习、统计学习理论、数据挖掘**.**郭嘉丰**,**男**,1980**年生**,**博士**,**副研究员**,**主要研究方向为互联网搜索与挖掘、用户数据挖掘、机器学习、社交网络**.**程学旗**,**男**,1971**年生**,**博士**,**研究员**,**博士生导师**,**主要研究领域为网络科学与社会计算、互联网搜索与挖掘、网络信息安全、分布式系统与大型仿真平台**.

results on Epsilon, RCV1-v2 and URL data sets verify the fast convergence rate and high performance of the proposed algorithm.

Keywords distributed; stochastic gradient descent; standardized; model combine; social networks; social computing

1 引　言

机器学习算法中的随机梯度下降算法由于使用简单、收敛速度快、效果可靠等优点得到了普遍应用，但是该算法需要在训练数据上不断迭代，遍历多遍数据，这在数据规模较大以及单台机器计算能力有限的情况下，算法执行效率往往比较低.

在大数据背景下，分布式随机梯度下降算法得到了广泛的研究，提高其收敛速度和性能具有重要的应用价值. 根据文献[1]，我们可以把当前的大规模随机梯度下降算法分为两类：数据并行和模型并行. 在数据并行算法研究中，之前的工作[2-6]将数据随机划分分布到不同机器上，通过在多个机器上同时独立地执行随机梯度下降算法，将得到的多个模型进行合并作为最终的模型或者下一次迭代的初始模型，这种策略取得了很好的效果.

目前，基于模型合并的分布式随机梯度下降算法普遍采用平均加权方式，平等对待每台机器上得到的模型. 这样的合并方式存在以下两个缺点：第一，采用平均加权的方式，忽略了各模型内在的差异性，合并得到的模型不能很好地反映全局数据间的差异特点，导致学习收敛速度变慢；第二，将合并得到的模型作为下一轮迭代计算的初始模型时，从每台机器的角度来看，相当于在模型空间中根据自己当前的模型和远程机器上的模型搜索下一个更好的点，直接平均加权合并得到的模型相对本地模型变化比较大，一定程度上影响了模型性能. 虽然文献[2]尝试基于模型错误率进行合并，但其实验结果表明基于模型错误率的合并策略和平均加权合并的策略效果基本没有差别.

针对上述问题，本文提出了一种基于差异合并的分布式随机梯度下降算法 DA-DSGD(Distributed Stochastic Gradient Descent with Discriminative Aggregating)，通过两种策略来提升分布式随机梯度下降算法的收敛速度和模型性能：(1) 基于性能的加权合并. 与平均加权方式和简单的基于模型错误率进行合并的策略不同，参与合并的各个模型的权重综合考虑了其在所在机器上已使用训练数据的误差以及整个学习过程的进度. 特别是随着学习过程的推进，给予性能较好的远程模型更高的权重，从而能更好地捕捉全局数据的特点，保证模型的稳定性；(2) 合并模型的规范化. 对于加权合并后的模型，我们使用规范化技术使得合并后的模型与本地模型的 Frobenius 范数相同，即两个模型处于模型空间的同一个球体表面上，模型合并相当于利用全局信息修正了本地模型的方向，从而提高本地模型的性能.

我们在分布式环境下，通过实验验证了上述合并策略的有效性，使用上述合并策略的算法相对平均加权收敛速度更快更稳定，而且算法能够达到更优的性能.

本文第 2 节介绍相关工作；第 3 节介绍本文使用的 Pegasos 学习算法以及分布式通信框架；第 4 节详细阐述本文提出的基于差异合并的分布式随机梯度下降算法 DA-DSGD；第 5 节通过实验验证本文提出的算法的有效性；第 6 节对本文进行总结并讨论下一步研究的方向.

2 相关工作

分布式随机梯度下降算法是近几年来的一个热点研究问题. 本文关注基于模型合并的分布式随机梯度下降算法，由于篇幅限制，我们并不详细介绍共享内存多核并行下的随机梯度下降算法[7-9]和基于参数服务器的算法[1,9-12]等相关工作. 基于模型合并是处理大规模数据的一种常用策略，特别是在完全分布式的环境下，例如 p2p 网络中，大部分算法普遍采用模型合并的方式进行训练[13-14].

文献[3]在 MapReduce 和 Bigtable 框架下验证了 DGD(Distributed Gradient Descent)、IPM(Iterative Parameter Mixtures)和 DAT(Distributed Asynchronous Training)等算法的性能. DGD 算法在 Map 阶段，每个机器获取最新模型，然后计算各自机器上数据的梯度，Reduce 阶段将 Map 阶段得到的梯度加到一起更新模型，重复多次直到模型收敛. 与 DGD 不同，IPM 算法[2-3]在 Map 阶段各机器独立地执行一遍随机梯度下降，Reduce 阶段将 Map 阶

段得到的模型进行平均加权合并，得到下一轮迭代的初始模型. DGD 和 IPM 算法在每一轮迭代后需要同步更新全局模型，为避免同步带来的额外时间开销，DAT 算法在计算的时候每个机器在每一轮迭代开始时异步地获取全局模型，然后在本地机器上独立地运行随机梯度下降算法，并用每一轮结束时得到的模型与这一轮初始模型的差对全局模型进行异步地更新. DAT 是一种简单的基于参数服务器实现的分布式随机梯度下降算法. DAT 相对 DGD 和 IPM 虽然节省了同步的时间开销，但是 DAT 算法中每个机器上的模型不是最新的模型，导致异步更新时使用的梯度不能很好的反应当前全局模型的梯度，每一轮迭代的收敛速度相对较慢[11].

由于网络延迟以及同步所需的额外开销，MapReduce计算框架并不太适合随机梯度下降这种需要在数据上迭代多次、顺序更新模型的算法. 文献[5]提出的 PSGD(Parallel Stochastic Gradient Descent)算法与 IPM 算法类似，但 PSGD 算法只有一轮的 MapReduce，在 Map 阶段，每个机器独立地执行完整的随机梯度下降算法，直到模型收敛；在 Reduce 阶段，对每个机器上的模型进行平均加权合并，得到最终的模型.

PSGD 相对 IPM 算法大大地降低了机器间的通信开销，但是由于每台机器都只使用本地的数据，训练过程没能够利用全局数据信息提高本地模型的性能. 介于 IPM 和 PSGD 之间，文献[15]提出的 BM-DSGD(Distributed Stochastic Gradient Descent with Butterfly Mixing)算法采用蝴蝶状通信方式，去除参数服务器中心节点，每轮迭代中各个节点独立地执行随机梯度下降算法，然后将模型仅发送给下一个通信节点，同时每个节点平均加权合并本地模型与接收到的模型，这种方式较 PSGD 加强了全局数据在训练过程中对于本地模型的作用，蝴蝶型的通信方式与 IPM 相比又降低了通信代价，因此具有良好的性能，在分布式随机梯度方法的研究中成为一个热点. 其具体通信机制及优点将在 3.2 小节中进行详细介绍.

综上所述，据我们目前调研的结果，大部分分布式梯度下降算法在做模型合并时都只是简单地进行平均加权，忽略了模型之间的差异性，而且平均加权合并得到的模型在模型空间中的分布范围较大，可能会影响算法收敛的速度和性能. 本文以 BM-DSGD 算法的通信框架为基础，在进行模型合并时利用各个模型在其机器上数据的性能，同时考虑整个学习算法的进度对模型进行加权合并. 另外，通过限制结果模型在模型空间的分布范围，很好地提高了模型的收敛速度和性能.

3 基本学习算法与通信机制

本文采用 Pegasos[16] 作为每个节点上的基本学习算法，因为 Pegasos 是一种使用随机梯度下降算法求解支持向量机(Support Vector Machine)原始问题的知名算法，而支持向量机是被广泛使用和研究的机器学习算法. 本文提出的 DA-DSGD 算法执行时各机器节点间的通信计算采用与 BM-DSGD[15] 算法相同的方式. 本节中，我们分别介绍 Pegasos 和 BM-DSGD 算法.

3.1 Pegasos 算法

支持向量机是目前广泛使用的机器学习算法之一，对于给定的训练数据集 $S=\{(x_i,y_i)\}_{i=1}^m$，其中 $x_i\in R^n$，$y_i\in\{+1,-1\}$，支持向量机可以形式化为式(1)定义的最小化问题. Pegasos 使用随机梯度下降直接求解式(1)，具体求解步骤如算法 1 所示.

$$\min_w \frac{\lambda}{2}\|w\|^2+\frac{1}{m}\sum_{(x,y)\in S}\max\{0,1-y\langle w,x\rangle\} \tag{1}$$

算法 1. Pegasos Algorithm.

输入：S,λ,T,k

输出：w_{T+1}

1. INITIALIZE: Choose w_1 s.t. $\|w_1\|\leq 1/\sqrt{\lambda}$
2. FOR $t=1,2,\cdots,T$
3. Choose $A_t\subseteq S$, where $|A_t|=k$
4. Set $A_t^+=\{(x,y)\in A_t: y\langle w_t,x\rangle<1\}$
5. Set $\eta_t=\frac{1}{\lambda t}$
6. Set $w_{t+\frac{1}{2}}=(1-\eta_t\lambda)w_t+\frac{\eta_t}{k}\sum_{(x,y\in A_t^+)}yx$
7. Set $w_{t+1}=\min\left\{1,\frac{1/\sqrt{\lambda}}{\|w_{t+\frac{1}{2}}\|}\right\}w_{t+\frac{1}{2}}$

Pegasos 算法输入参数为迭代次数 T 和用来计算梯度的样本数 k，每一轮迭代分两步：第 1 步选择训练集 S 中的 k 个样本组成集合 A_t，对 A_t 中导致目标函数产生非零损失的数据点计算梯度更新模型 w；第 2 步将得到的 w 映射到集合 $B=\{w:\|w\|\leq 1/\sqrt{\lambda}\}$ 中.

3.2 BM-DSGD 通信机制

BM-DSGD(Butterfly Mixing Distributed Stochastic Gradient Descent)采用蝴蝶状通信方式，如

图 1 所示. 假设集群中有 2^n 个机器，这里以 $n=2$ 为例进行说明. BM-DSGD 算法每一轮迭代分两步，每个节点先独立地执行一遍随机梯度下降算法，然后将得到的模型按图 1 的通信方式发送给对应节点，同时将收到的来自其他节点的模型与本地模型进行平均加权，合并得到的模型作为本地节点下一次迭代的初始模型. 例如第 1 轮通信时，节点 1 与节点 2 互发模型，节点 3 与节点 4 互发模型；第 2 轮通信节点 1 与节点 3 互发模型，节点 2 与节点 4 互发模型. BM-DSGD 算法每经过 n 轮迭代，使每台机器上的数据信息会传播到整个集群上.

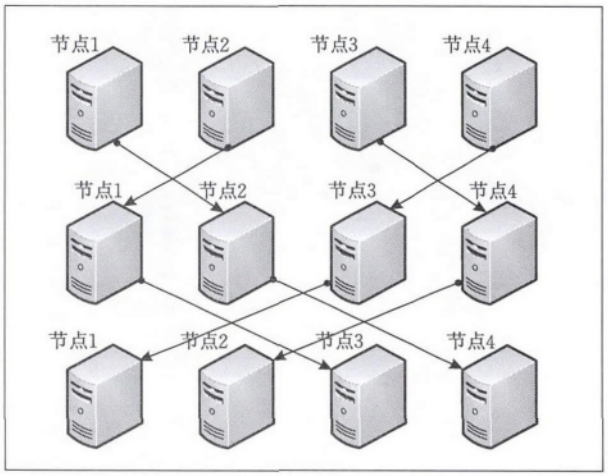

图 1　BM-DSGD 蝴蝶状通信结构示意

相对 DGD[3] 和 IPM[2-3] 算法来说，BM-DSGD 避免了同步更新全局模型带来的网络开销及中心节点瓶颈限制，每个节点同时进行模型合并，大大降低了每一轮迭代的通信开销；相对 PSGD 算法，BM-DSGD 在迭代的过程中能利用不同机器上的数据信息，提高算法的收敛速度和性能；相对基于参数服务器实现的大规模随机梯度下降算法，BM-DSGD 是一种更加分布式化的实现，每个节点独立地保存最新的模型，能并行地进行模型应用. 特别地，在 p2p 网络中，例如传感器网络和移动手机网络中，基于模型合并的 BM-DSGD 是一种典型且普遍应用的训练框架[13-14].

4　DA-DSGD 算法

通过上述相关工作的回顾，我们发现，目前基于模型合并的分布式随机梯度下降算法在进行模型合并时基本采用平均加权方式，忽略了模型之间的差异性，合并结果相对本地模型变化较大，作为下一轮迭代的初始模型可能导致模型收敛速度较慢，而提高大规模随机梯度下降算法的收敛速度和性能具有重要的应用价值，能极大地节省计算资源和训练时间. 针对这个问题，本文提出基于差异合并的分布式随机梯度下降算法，DA-DSGD(Distributed Stochastic Gradient Descent with Discriminative Aggregating)，算法具体步骤如算法 2 所示. 核心思想是在合并时充分考虑每个模型在其所在机器上已使用训练数据的误差以及整个学习过程的进度这两项差异，从而加快算法收敛速度；同时使用规范化技术使得合并后的模型与本地模型的 Frobenius 范数相同，降低合并给模型带来的巨大变化，提高学习精度.

算法 2.　DA-DSGD 算法.

输入：S,λ,PT,k,T,N

输出：w_T

1. Shuffle S and randomly split into N disjoint sets $S^1,S^2,\cdots,S^N$
2. INITIALIZE: for all nodes $i=1,2,\cdots,N$, choose w_0^i, s. t. $\|w_0^i\|\leq 1/\sqrt{\lambda}$
3. FOR $t=1,2,\cdots,T$ PARALLEL DO
4. $\quad w_t^i=\text{PegasosIterateOnce}(S^i,\lambda,PT,k,w_{t-1}^i)$
5. $\quad \varepsilon_i=$Average 0−1 loss of local seen training data on node i
6. $\quad j=\text{NextCommunicateNode}(i,t,N)$
7. $\quad$ send (w_t^i,ε_i) to node j and receive (w_t^j,ε_j) from node j
8. $\quad \mu_i=\ln\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$, $\mu_j=\ln\left(\frac{1-\varepsilon_j}{\varepsilon_j}\right)$
9. $\quad w_{t'}^i=\frac{\mu_i}{\mu_i+\mu_j}w_t^i+\frac{\mu_j(\mu_i+\mu_j)}{\mu_i+\mu_j(\mu_i+\mu_j)}w_t^j$
10. $\quad w_t^i=\frac{\|w_t^i\|}{\|w_{t'}^i\|}w_{t'}^i$
11. ENDFOR
12. $w_T=\sum_{i=1}^{N}\frac{\mu_i}{\sum_{j=1}^{N}\mu_j}w_T^i$

过程 1.　$\text{PegasosIterateOnce}(S^i,\lambda,PT,k,w_{t-1}^i)$.

1. Set $w_{t-1,0}^i=w_{t-1}^i$
2. FOR $pt=1,2,\cdots,PT$
3. Choose $A_{pt}\subseteq S$, where $|A_{pt}|=k$
4. $\quad$ Set $A_{pt}^+=\{(x,y)\in A_{pt}:y\langle w_{t-1,pt-1}^i,x\rangle<1\}$
5. $\quad$ Set $\eta_{t-1,pt}=\frac{1}{\lambda((t-1)\times PT+pt)}$
6. $\quad$ Set $w_{t-1,pt-\frac{1}{2}}^i=(1-\eta_{t-1,pt}\lambda)w_{t-1,pt-1}^i+\frac{\eta_{t-1,pt}}{k}\sum_{(x,y\in A_{pt}^+)}yx$
7. $\quad$ Set $w_{t-1,pt}^i=\min\left\{1,\frac{1/\sqrt{\lambda}}{\|w_{t-1,pt-\frac{1}{2}}^i\|}\right\}w_{t-1,pt-\frac{1}{2}}^i$
8. ENDFOR
9. RETURN $w_t^i=w_{t-1,PT}^i$

过程 2. NextCommunicateNode(i,t,N).

1. $t'=(t-1)\%\ \log_2 N+1$
2. $j=i+2^{t'-1}$
3. $lower\_bound=0$
4. $range=2^{t'}$
5. $bias=0$
6. WHILE $bias<N$
7. IF $i<=bias+range$
8. $lower\_bound=bias$
9. BREAK
10. ENDIF
11. $bias+=range$
12. ENDWHILE
13. IF $j>lower\_bound+step$
14. $j=lower\_bound+j\%\ range$
15. ENDIF
16. RETURN j

DA-DSGD 算法输入参数中 S 为训练数据集，λ 为 Pegasos 算法目标函数中模型参数 w 的正则因子，PT 为本地节点每一轮迭代中 Pegasos 算法的迭代次数，k 是 Pegasos 算法每一轮迭代中用来计算梯度的训练样本数，T 是 DA-DSGD 算法每个节点的迭代次数，N 是集群中参与计算的节点数. 其中，PT 的值用来调节本地计算时间和网络通信时间的平衡，显然太小的 PT 值会使算法的大部分时间开销都在网络通信上.

具体地，训练数据被随机均匀划分到 N 个节点上，每个节点随机初始化模型参数 w_0^i，实验中我们限制初始模型 w_0^i 的 Frobenius 范数为 1. 每一轮迭代中，各计算节点调用函数 PegasosIterateOnce 执行 PT 次 Pegasos 算法的内部迭代运算，然后在本地已经被使用过的训练数据上计算模型的平均错误率. 算法 2 的步骤 7 中 NextCommunicateNode 函数根据当前迭代次数以及蝴蝶状通信网络结构选择通信节点，并互相发送节点上的模型和错误率，各节点将接收到的模型及其错误率与本地模型进行差异化合并. 最后收集合并各机器上的模型得到 DA-DSGD 算法的最终输出模型.

DA-DSGD 与 BM-DSGD 算法的主要差别在于 BM-DSGD 采用平均加权方式进行模型合并，而 DA-DSGD 对模型进行差异化合并. DA-DSGD 借鉴集成学习算法 AdaBoost[17] 的权重计算公式，根据模型的错误率使用算法 2 中步骤 8 公式得到模型的基本权重；同时，考虑到算法不同阶段模型表现的差异以及对全局数据信息的利用情况，我们选择算法 2 中步骤 9 的公式在步骤 8 的基础上重新对权重进行计算，得到最终合并使用的权重（具体做法在下一段落解释）. 此外，在每轮迭代的最后一步（算法 2 步骤 10），我们将合并得到的模型投射到与本地节点合并前模型的 Frobenius 范数相同的球面上，实验结果表明这么做能够使最终模型的误差更小，模型也更稳定.

本文使用的模型差异化计算公式能够很好地刻画学习算法不同阶段应该采取的合并策略. 算法 2 步骤 9 中本地模型 w_t^i 和接收到的模型 w_t^j 的权重变化情况如图 2 所示.

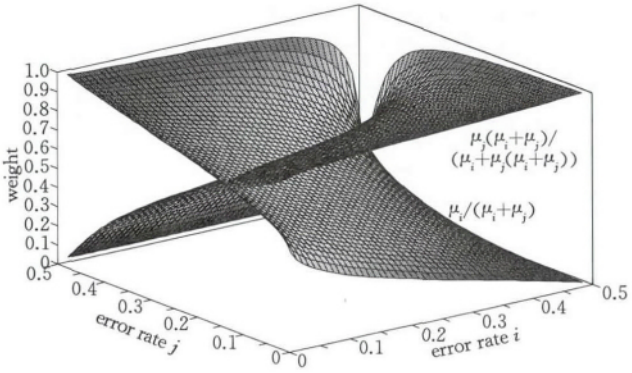

图 2 DA-DSGD 模型合并权重示意

其中，“error rate i”和“error rate j”坐标分别表示模型 w_t^i 和 w_t^j 在其已使用训练数据上的误差率，也就是算法 2 中的 ε_i 和 ε_j. 从图 2 可以看到，在训练刚开始的时候，各个节点上模型的误差率相对较大，例如在 0.4～0.5 之间，模型还不能很好地建模训练数据，来自其他节点的模型对本地数据的建模效果也很差，此时，如果本地模型误差率与接收到的模型误差率接近，DA-DSGD 就给予本地模型更大的权重，即 $\frac{\mu_i}{\mu_i+\mu_j}>\frac{\mu_j(\mu_i+\mu_j)}{\mu_i+\mu_j(\mu_i+\mu_j)}$；当接收到的模型误差率显著低于本地模型误差率时，DA-DSGD 给予接收到的模型较大的权重. 而在算法训练后期，各模型表现趋于稳定，在各节点数据属于独立同分布的假设下，各模型的训练误差率比较接近，此时误差率也相对较小，DA-DSGD 给予接收到的模型更大的权重，原因在于此时来自其他节点的模型相对本地模型包含的数据信息更大，能更好地建模全局训练数据.

DA-DSGD 相对 BM-DSGD 在模型合并时采用了差异化的合并策略，虽然需要额外计算各节点的错误率，但是在分布式环境下，这些开销相对每次迭代中的通信开销小得多. 所以本文并不讨论 DA-DSGD 与 BM-DSGD 在运行时间上的差异. 另外，DA-DSGD 差异化合并的思想能方便地扩展到 MapReduce 等

其他通信框架上.

5　实验分析

本文在 Epsilon、RCV1-v2[18] 和 URL[19] 3 个分类数据集上对比 DA-DSGD 与 BM-DSGD 算法的收敛速度和最终模型的性能，采用二分类而不是多分类任务进行实验主要是因为多分类问题可以转换为二分类问题，同时，这也是大部分相关工作采用的做法[13-14]. 另外，为了探究差异化合并和规范化技术对模型收敛速度及性能的影响，我们对 BM-DSGD 合并后的模型使用 DA-DSGD 采用的规范化技术，得到 SBM-DSGD，对 DA-DSGD 去除规范化步骤，得到 UDA-DSGD，并对这些算法的性能及收敛性进行对比实验. 下面分别介绍实验使用的数据集和实验结果.

5.1　数据集

Epsilon 是人工构造的数据集，来自 2008 年 Pascal 大规模学习竞赛，是二分类问题. Epsilon 数据集包含 400 000 条训练数据和 100 000 条测试数据，共 2000 个特征，特征预处理时先按特征进行 z-score 归一化，再对每条数据按长度为 1 进行归一化.

RCV1-v2 数据集共 804 414 篇文档，我们使用 CCAT 类别对其进行二分类，使用其中的 621 392 篇文档作为训练数据，183 022 篇文档作为测试数据，文档用词的向量空间模型表示，去除停用词并进行词干还原，特征使用 tf-idf 计算方式，每个文档向量按长度为 1 进行归一化. 为降低特征数量，我们去除在训练集中文档频率小于 5 的词项，最终的特征数为 118 840.

URL 数据集共 2 396 130 条数据，3 231 961 个特征，用于判断 URL 是否为恶意链接，是二分类问题. 特征主要有词法特征和基于主机的特征两大类，都是从网页 url 抽取的，不涉及网页具体内容. 词法特征是对网页 url 按分隔符切分得到词项，再用词袋模型表示，基于主机的特征包括域名注册日期、注册者、登记者、主机地理位置信息、IP 地址前缀等，关于该数据集的更加详细的说明请参考文献[19]. 我们对该数据集进行随机划分，划分后训练数据集有 2 255 171 条数据，测试集有 140 959 条数据.

5.2　实验结果

我们对 N 取不同的值，$N=16,32,64$，分别进行实验，验证提出的算法的有效性. 我们发现 N 取不同值时得到的实验结论是一致的，于是，我们对 $N=16$ 的实验结果进行详细分析，验证 DA-DSGD 相对 BM-DSGD 在性能和收敛速度上的优势. 实验中，迭代次数 T 取足够大的值，保证观察到算法的收敛情况. 参考文献[15] BM-DSGD 和文献[16] Pegasos 算法的实验，我们取 $PT=100$，$k=10$，$\lambda=10^{-4}$.

5.2.1　DA-DSGD 与 BM-DSGD 两种算法的性能及收敛速度对比

针对 N 的不同取值，我们得到图 3 的实验结果，可以看到，本文提出的算法在 Epsilon、RCV1-v2 和 URL 3 个数据集上相对 BM-DSGD 算法收敛速度更快、性能更好.

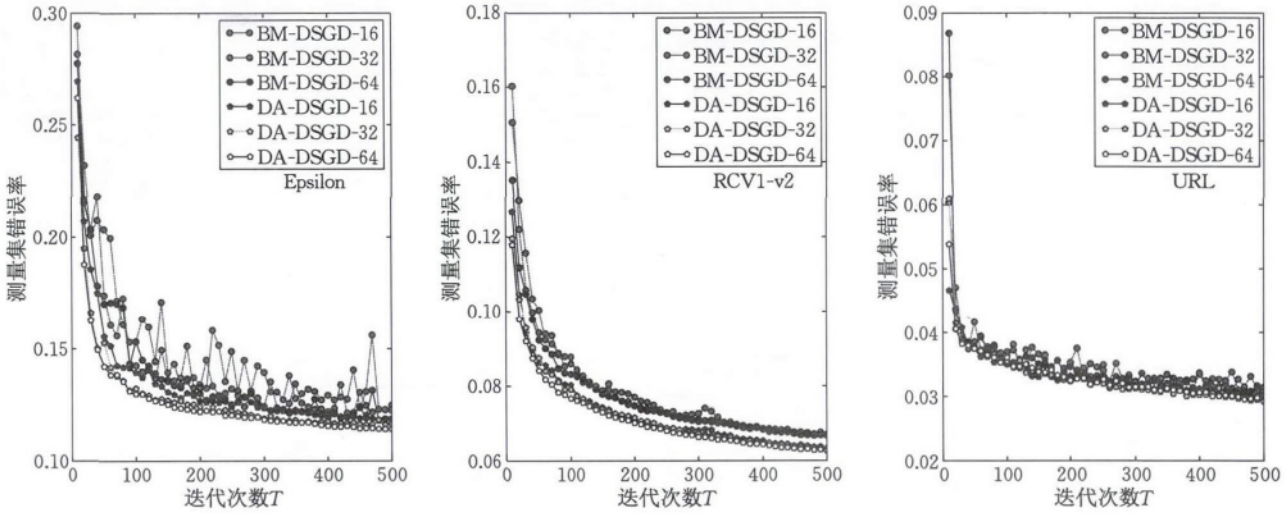

图 3　$N=16,32,64$ 时算法的测试集错误率

下面我们以 $N=16$ 为例，进行详细的分析. 首先我们对比 DA-DSGD 与 BM-DSGD 两种算法的性能. 为此，我们计算测试集错误率随迭代次数 T 的变化，在 3 个数据集上的测试集错误率如图 4 所示.

可以看出，DA-DSGD 在测试集上的性能明显优于 BM-DSGD 的性能. 例如 $T=300$ 时，在 3 个数据集上我们的方法比 BM-DSGD 的性能分别提高了 1.86%、0.49%和 0.08%.

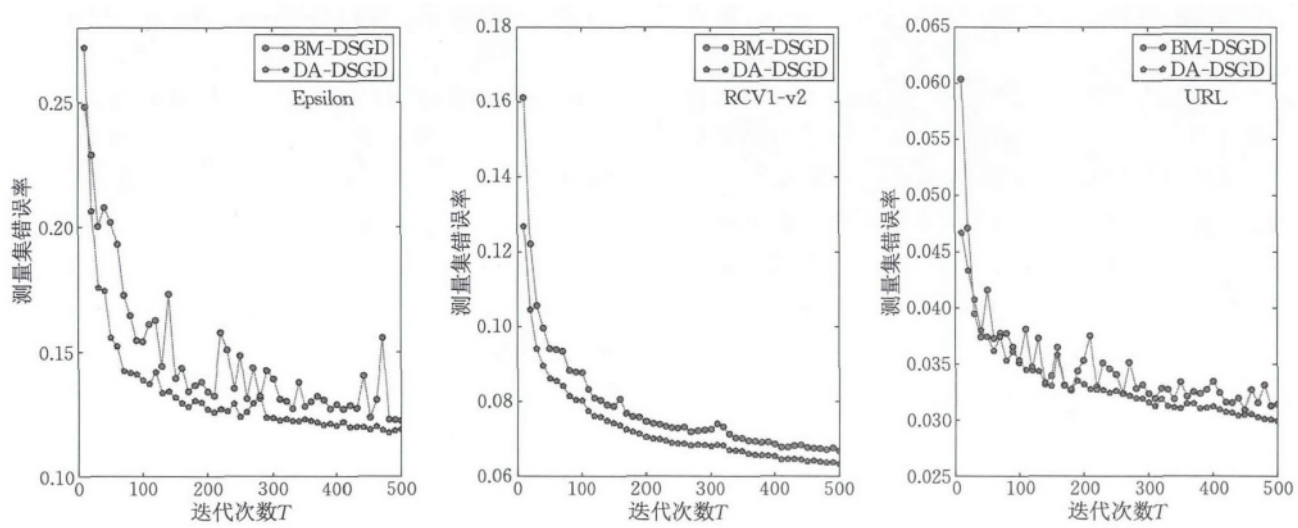

图 4 测试集错误率

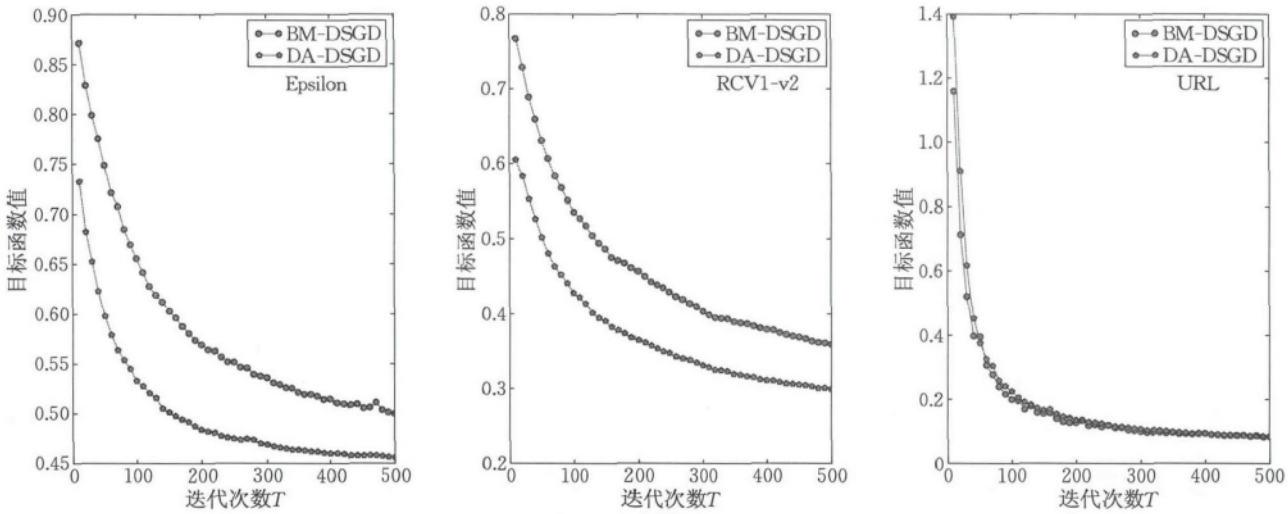

图 5 Pegasos 目标函数值

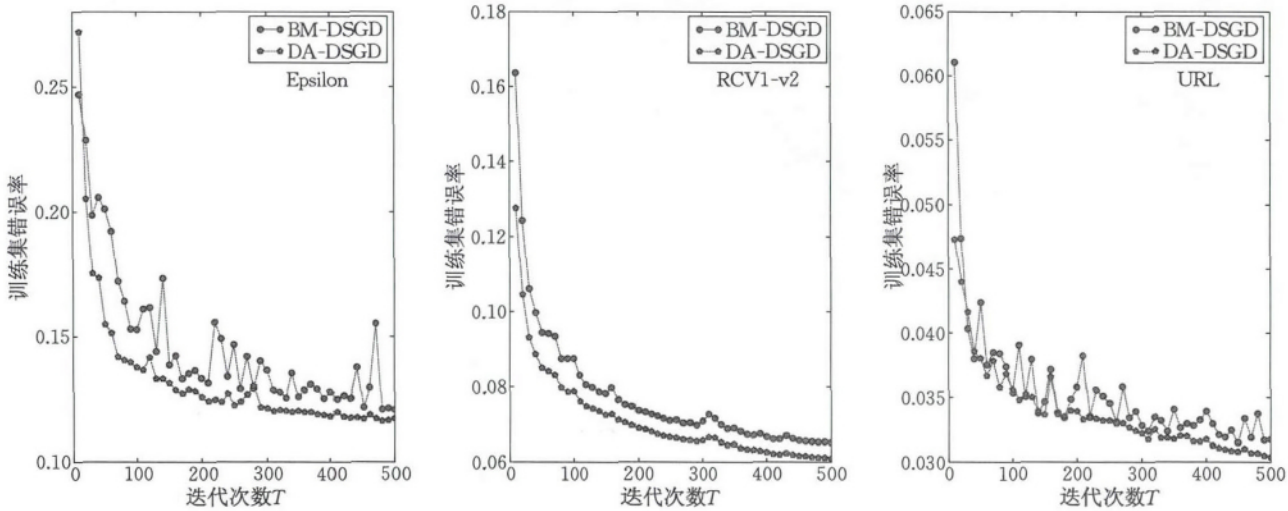

图 6 训练集错误率

下面我们对比 DA-DSGD 与 BM-DSGD 两种算法的收敛速度. 为此，我们分别计算 Pegasos 目标函数值和训练集错误率随迭代次数 T 的变化，注意这些结果都是在全部训练数据上进行计算的. Pegasos 目标函数值和训练集错误率在 3 个数据集上的结果分别如图 5 和图 6 所示. 可以看到，DA-DSGD 算法相对 BM-DSGD 算法收敛得更快. 具体地，DA-DSGD仅用 300 轮到 400 轮的迭代其目标函数值和训练误差就基本收敛了，而 BM-DSGD 目标函数值虽然基本收敛了，但是其训练错误率还有很大的波动，当我们将迭代次数 T 增加到 1000 次时，发现 BM-DSGD 算法在 Epsilon 和 URL 两个数据集上的训练错误率依然没有收敛.

5.2.2　差异化合并和规范化两个因素的深度分析

下面我们进一步分析 DA-DSGD 中差异化合并和规范化这两个因素对收敛速度和收敛值的影响. 我们对 DA-DSGD 去除规范化，采用与 DA-DSGD 相同的合并权重计算公式，但是用权重归一化取代合并结果规范化，得到 UDA-DSGD. 同时，对 BM-DSGD 的平均加权合并结果进行规范化，得到 SBM-DSGD. 本文实验结果中测试集错误率和训练集错误率曲线趋势基本一致，所以在接下来的分析中我们使用测试集错误率来比较各种算法的收敛速度和收敛值，探究差异化合并和规范化两个因素的不同作用. 在 Epsilon、RCV1-v2 和 URL 的 3 个数据集上不同算法的实验结果如图 7 所示.

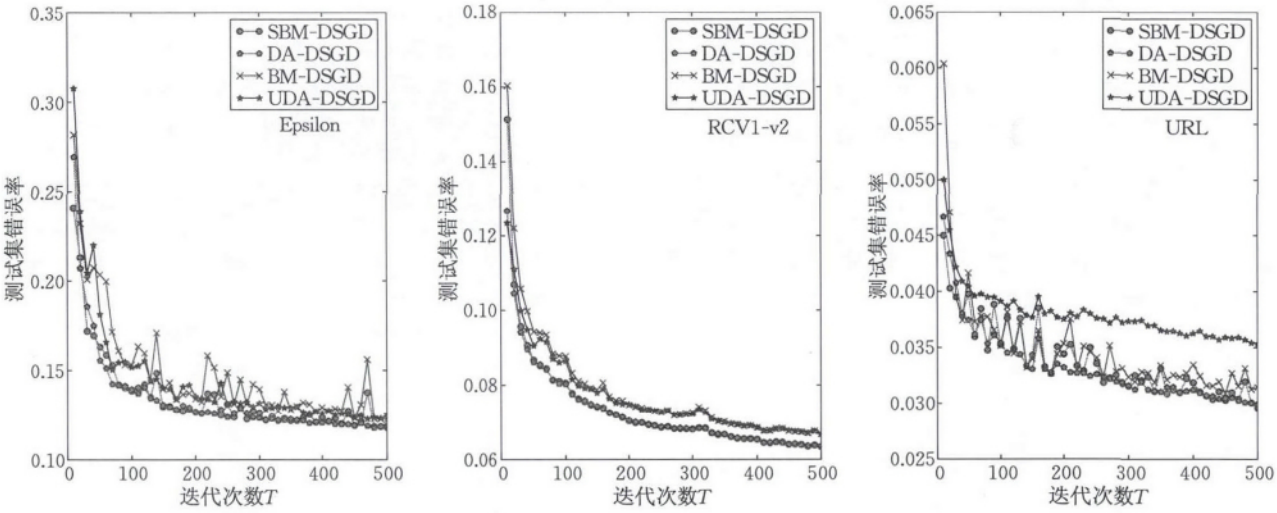

图 7　测试集错误率

首先我们研究差异化模型合并因素的影响，为此我们对比 DA-DSGD 和 SBM-DSGD 算法，同时对比 UDA-DSGD 和 BM-DSGD 算法. 通过实验结果，我们发现 DA-DSGD 相对 SBM-DSGD 收敛速度更快，随着迭代次数的增加，DA-DSGD 很快就收敛了，但是 SBM-DSGD 还有一定程度的波动，但是两者最终收敛值一致. 对比 UDA-DSGD 和 BM-DSGD 我们得到了相同的结果. 3 个数据集上得到的结果也基本一致，相对 RCV1-v2 数据集，Epsilon 和 URL 数据集上的差异更明显，部分原因可能在于 RCV1-v2 上 CCAT 的分类任务相对容易，算法基本都能很快收敛.

接下来研究规范化技术对模型收敛速度和收敛值的影响，为此我们对比 DA-DSGD 和 UDA-DSGD 算法，同时对比 SBM-DSGD 和 BM-DSGD 算法. 在 3 个数据集上的实验结果说明，通过使用规范化技术，DA-DSGD 相对 UDA-DSGD、SBM-DSGD 相对 BM-DSGD 均收敛到更好的值.

综合以上实验结果表明，差异化合并能够提高模型收敛速度，而规范化技术能使模型收敛到更好的值.

5.2.3　实验小结

上述对 DA-DSGD 算法实验结果的分析和对比，用实验验证了差异化模型合并策略和规范化技术能很好地提高模型的收敛速度和性能，使模型随着迭代次数的增加，其性能有更稳定和更好的表现.

另外，我们也尝试使用不同的权重计算方式，直接对模型在其所在机器上使用过的训练数据的准确率进行线性加权，实验结果与平均加权合并方式接近，因为每个节点上的模型从训练数据属于独立同分布的假设上来看，其准确率应该是接近的，直接线性加权的方式不能很好地捕捉模型之间的差异性.

另外，最终模型合并时对 DA-DSGD 算法根据各节点模型按性能仅取前 K 个进行合并，能进一步稍微提高最终模型的性能.

6 总 结

本文针对分布式随机梯度下降算法普遍采用平均加权方式进行模型合并存在的收敛速度慢和最终模型性能较差的问题，提出了基于模型性能进行差异化合并的策略，同时对合并得到的模型进行规范化，使得到的模型能更好地利用全局数据信息，提高收敛速度和性能. 实验结果证明，差异化合并策略相对平均加权方式，能提高模型收敛速度，同时，规范化技术的使用，使模型收敛到了更好的点.

后续研究工作中，我们打算使用逻辑回归等其他学习算法，同时考虑 MapReduce 等分布式计算框架，验证本文提出的差异化合并策略和规范化技术的普遍有效性. 另外，对于本文使用的差异化加权方式，我们将更细致的研究算法迭代的不同阶段其加权机制对模型收敛速度的影响，同时，我们也会研究其他形式的权重计算方式.

参 考 文 献

[1] Dean J, Corrado G, Monga R, et al. Large scale distributed deep networks//Proceedings of the Conference on Neural Information Processing Systems. Lake Tahoe, USA, 2012: 1223-1231

[2] McDonald R, Hall K, Mann G. Distributed training strategies for the structured perceptron//Proceedings of the Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Los Angeles, USA, 2010: 456-464

[3] Hall K B, Gilpin S, Mann G. MapReduce/Bigtable for distributed optimization//Proceedings of the Conference on Neural Information Processing Systems. Workshop on Learning on Cores, Clusters and Clouds. Vancouver, Canada, 2010

[4] Mann G, McDonald R T, Mohri M, et al. Efficient large-scale distributed training of conditional maximum entropy models//Proceedings of the Conference on Neural Information Processing Systems. Vancouver, Canada, 2009: 1231-1239

[5] Zinkevich M, Weimer M, Smola A J, et al. Parallelized stochastic gradient descent//Proceedings of the Conference on Neural Information Processing Systems. Vancouver, Canada, 2010: 4

[6] Kleiner A, Talwalkar A, Sarkar P, et al. The big data bootstrap //Proceedings of the International Conference on Machine Learning. Edinburgh, UK, 2012

[7] Louppe G, Geurts P. A zealous parallel gradient descent algorithm//Proceedings of the Conference on Neural Information Processing Systems. Workshop on Learning on Cores, Clusters and Clouds. Vancouver, Canada, 2010

[8] Niu F, Recht B, Ré C, et al. Hogwild !: A lock-free approach to parallelizing stochastic gradient descent// Proceedings of the Conference on Neural Information Processing Systems. Granada, Spain, 2011: 693-701

[9] Langford J, Smola A J, Zinkevich M. Slow learners are fast//Proceedings of the Conference on Neural Information Processing Systems. Vancouver, Canada, 2009: 2331-2339

[10] Dai W, Wei J, Zheng X, et al. Petuum: A framework for iterative-convergent distributed. arXiv preprint arXiv: 1312.7651, 2013

[11] Ho Q, Cipar J, Cui H, et al. More effective distributed ML via a stale synchronous parallel parameter server//Proceedings of the Conference on Neural Information Processing Systems. Lake Tahoe, USA, 2013: 1223-1231

[12] Li M, Andersen D G, Park J W, et al. Scaling distributed machine learning with the parameter serve//Proceedings of the Operating Systems Design and Implementation. Broomfield, USA, 2014: 1332-1340

[13] Ormándi R, Hegedüs I, Jelasity M. Asynchronous peer-to-peer data mining with stochastic gradient descent//Proceedings of the Euro-Par 2011 Parallel Processing. Bordeaux, France, 2011: 528-540

[14] Ormándi R, Hegedüs I, Jelasity M. Gossip learning with linear models on fully distributed data. Concurrency and Computation: Practice and Experience, 2013, 25(4): 556-571

[15] Zhao H, Canny J F. Communication-Efficient Distributed Stochastic Gradient Descent with Butterfly Mixing. Location: Department of Electrical Engineering and Computer Sciences University of California, Berkeley, USA, 2012

[16] Shalev-Shwartz S, Singer Y, Srebro N, et al. Pegasos: Primal estimated sub-gradient solver for svm. Mathematical Programming, 2011, 127(1): 3-30

[17] Freund Y, Schapire R E. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 1997, 55(1): 119-139

[18] Lewis D D, Yang Y, Rose T G, et al. Rcv1: A new benchmark collection for text categorization research. The Journal of Machine Learning Research, 2004, 5(12): 361-397

[19] Ma J, Saul L K, Savage S, et al. Identifying suspicious URLs: An application of large-scale online learning// Proceedings of the 26th Annual International Conference on Machine Learning. Montreal, Canada, 2009: 681-688

CHEN Zhen-Hong, born in 1988, Ph. D. candidate. His research interests include large scale machine learning, distributed system and social networks.

LAN Yan-Yan, born in 1982, Ph. D., associate professor. Her research interests include machine learning, learning to rank, statistical learning theory and data mining.

GUO Jia-Feng, born in 1980, Ph. D., associate professor. His research interests include Web search and data mining, user mining, machine learning and social networks.

CHENG Xue-Qi, born in 1971, Ph. D., professor, Ph. D. supervisor. His main research interests include network science and social computing, Web search and data mining, web information security, distributed system and large scale simulation platform.

Background

Stochastic gradient descent has been one of the most popular optimization technic among the machine learning communities, due to its simplicity and fast convergence. However, it is naturally a sequential algorithm. How to parallelize stochastic gradient descent has been an appealing and challenging research topic, especially in this big data era.

A number of distributed gradient descent optimization strategies have been proposed, including distributed gradient descent, asynchronous gradient descent and parameter mixing. Among these methods, parameter mixing strategy is appealing for its low resource consumption in terms of CPU time and network traffic. It has also been shown that parameter mixing keeps similar theoretical convergence rates and accuracies as the corresponding single machine algorithms, in the tasks of training conditional maximum entropy models and structured perceptron. On the other hand, when model parameters become too large to be held by one single machine, asynchronous methods implemented with parameter server exhibits powerful computing scalabilities.

This paper focuses on the parameter mixing method for large scale stochastic gradient descent, in a full distributed environment. State-of-the-art parameter mixing methods simply average different models together and it has been proved to be effective. However, the simply average method may neglect the differences between different models, thus may slow down the convergence speed and harm the performance of the algorithm. Though model aggregating with error rates had been considered, its performance was empirically proved to about the same as average methods.

In this paper, we propose a distributed stochastic gradient descent algorithm with discriminative aggregating to avoid the above potential problem, to further improve the convergence and performance of the parameter mixing method. Our discriminative aggregating algorithm differs to existing average method in two aspects. Firstly, we take into account the training performance of each model on training data. Secondly, we exploit the dynamic importance of different models as the training process moves on. Meanwhile, to further improve model performance, we project the combined model onto the surface sphere of the local model respectively, using Frobenius norm. Our experimental results on Epsilon, RCV1-v2 and URL data sets verify the fast convergence rate and high performance of the proposed algorithm.

This work is supported by the National Basic Research Program(973 Program) of China under Grant Nos. 2012CB316303 and 2014CB340401, the National High Technology Research and Development Program (863 Program) of China under Grant No. 2012AA011003, the National Natural Science Foundation of China Key Program under Grants No. 61232010, and the National Natural Science Foundation of China for Distinguished Young Scholars under Grants Nos. 61203298 and 61003166.