正则化超限学习机的最大分划广义交替方向乘子法

侯秀聪,赖晓平,曹九稳
(杭州电子科技大学人工智能研究院,浙江杭州 310018)

摘 要: 借助交替方向乘子法(Alternating Direction Method of Multipliers,ADMM),将多变量正则化最小二乘拟合问题,分解为多个可并行执行的标量优化问题,并引入可调步长因子加速算法,得到一个高度并行的最大分划广义 ADMM 算法,并应用于正则化超限学习机. 建立了算法的收敛条件,分析了算法的计算复杂度,通过基准现实数据集实验与新近文献方法——最大分划松弛 ADMM 进行了收敛率比较. 在 GPU 并行加速实验中,基于最大分划广义 ADMM 的正则化超限学习机获得的大 GPU 加速比,表明了该算法的高度并行性.

关键词: 机器学习; 超限学习机; 大数据; 并行学习; 交替方向乘子法
中图分类号: TP18 **文献标识码:** A **文章编号:** 0372-2112 (2021)04-0625-06
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.12263/DZXB.20200310

A Maximally Split Generalized ADMM for Regularized Extreme Learning Machines

HOU Xiu-cong, LAI Xiao-ping, CAO Jiu-wen
(*Artificial Intelligence Institute, Hangzhou Dianzi University, Hangzhou, Zhejiang* 310018, *China*)

Abstract: By virtue of the alternating direction method of multipliers (ADMM), the multivariate regularized least-squares model fitting problem is decomposed into multiple univariate subproblems that are solvable in parallel. By introducing a tunable step size to accelerate the algorithm, a highly parallel maximally split generalized ADMM (MS-GADMM) is developed for the regularized extreme learning machine (RELM). The convergence condition of the MS-GADMM is established and the computational complexity of the MS-GADMM-based RELM is analyzed. Through experiments on real-world benchmark datasets, the MS-GADMM is compared with a maximally split relaxed ADMM recently presented in the literature. In the GPU implementation experiments, the MS-GADMM has obtained very large GPU acceleration ratios, which demonstrates the high parallelism of the proposed MS-GADMM-based RELM.

Key words: machine learning; extreme learning machine; big data; parallel learning; alternating direction method of multipliers

1 引言

用于单隐层前馈神经网络(Single Hidden-Layer Feedforward Neural Network, SLFN)训练的超限学习机(Extreme Learning Machine, ELM)[1,2],以其快速的学习能力而受到研究者的广泛关注,并在各类工业领域中得到广泛应用[3~6]. 正则化 ELM (Regularized ELM, RELM)是为提高泛化能力提出的一个 ELM 扩展,它将输出权矩阵的 F-范数加到原始 ELM 的目标函数中,不仅最小化训练误差平方和,而且最小化输出权系数平方和[2]. 学习速度快是 ELM 的一个重要特性. 然而,数据体量和维度的不断增大,ELM 仍面临计算量和存储量过大的挑战,并行/分布式算法是解决这些挑战的一条重要途径[7]. 文献[8~11]利用并行/分布式计算软/硬件环境,对基于矩阵逆的 ELM 进行并行化改造,提出了并行/分布式 ELM 算法. 但基于矩阵逆的算法,可扩展性不强,只适应于二次型学习.

交替方向乘子法(Alternating Direction Method of

收稿日期:2020-03-30;修回日期:2020-07-10;责任编辑:李勇锋
基金项目:国家自然科学基金(No. U1909209)

Multipliers, ADMM)[12,13], 由于其优良并行结构和良好收敛性能, 已成功应用于信号处理和机器学习等领域的凸优化问题[14~20]. 文献[18]将 ADMM 应用于最小二乘拟合, 提出 ELM 的最大分划 ADMM 算法(Maximally Split ADMM, MS-ADMM), 每个迭代步对向量变量的更新, 都可按其标量分量独立进行, 并行性很高; 与松弛技术结合得到的最大分划松弛 ADMM(Maximally Split and Relaxed ADMM, MS-RADMM), 提高了收敛速度. 文献[19]将其推广到一般多分块情形, 研究了收敛率与模型分块数的关系, 提高了根据可用计算资源选择合适模型分块的灵活性. 文献[20]则推广到约束最小二乘情形, 得到具有高度并行和扩展性的二维有限脉冲响应数字滤波器设计算法.

文献[18]的 MS-RADMM 中, 将正则化最小二乘问题的正则项合并到二次损失函数中, 把问题转化成不含正则项的最小二乘问题, 未充分利用与 ADMM 相适应的模型结构, 导致算法收敛变慢. 另外, 文献[18]的松弛技术, 不能按原始机理推广到正则项不为零的情况, 需对其加速机制有更广义和合理的解释.

本文考虑基于 ADMM 的 RELM 算法, 但不像文献[18]那样把正则项合并到损失函数, 而是将其作为与损失函数平行的另一项目标函数来处理, 在模型拟合的 ADMM 框架下[12], 导出 MS-ADMM. 重点通过使 ADMM 中 $\boldsymbol{x}$-更新的步长因子可调, 得到 MS-ADMM 的一个加速变体, 即最大分划广义 ADMM(Maximally Split Generalized ADMM, MS-GADMM). 在建立收敛性之后, 将算法应用到单隐层前向网络的训练, 提出 RELM 的 MS-GADMM. 通过基准现实数据集上实验及与现有算法的比较, 验证算法的快速收敛性, 低计算复杂度和高度并行性.

2 正则化最小二乘拟合问题的 MS-GADMM

针对凸模型拟合问题[12]

$$\min_{\boldsymbol{x}} f(\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b})+r(\boldsymbol{x}) \tag{1}$$

其中 $\boldsymbol{A}=[\boldsymbol{a}_1,\boldsymbol{a}_2,\cdots,\boldsymbol{a}_N]\in\mathrm{R}^{M\times N}$ 是模型数据矩阵, $\boldsymbol{b}\in\mathrm{R}^M$ 是目标输出向量, $\boldsymbol{x}=[x_1,x_2,\cdots,x_N]^{\mathrm{T}}\in\mathrm{R}^N$ 是模型系数向量, $f(\cdot)$ 是凸损失函数, $r(\boldsymbol{x})=r_1(x_1)+\cdots+r_N(x_N)$ 是凸的可分正则化函数, 文献[18]给出了 MS-ADMM:

$$x_n^{k+1}=\arg\min_{x_n}\left\{r_n(x_n)+\frac{\rho}{2}\|\boldsymbol{a}_n x_n-\boldsymbol{z}^k+\boldsymbol{u}^k\|_2^2\right\} \tag{2a}$$

$$\boldsymbol{y}^{k+1}=\bar{\boldsymbol{A}}\,\boldsymbol{x}^{k+1} \tag{2b}$$

$$\boldsymbol{z}^{k+1}=\arg\min_{\boldsymbol{z}}\left\{f(N\boldsymbol{z}-N\bar{\boldsymbol{b}})+\frac{1}{2}N\rho\|\boldsymbol{y}^{k+1}-\boldsymbol{z}+\boldsymbol{u}^k\|_2^2\right\} \tag{2c}$$

$$\boldsymbol{u}^{k+1}=\boldsymbol{u}^k+\boldsymbol{y}^{k+1}-\boldsymbol{z}^{k+1} \tag{2d}$$

其中 $\rho>0$ 是惩罚因子, $\bar{\boldsymbol{A}}=N^{-1}\boldsymbol{A}$, $\bar{\boldsymbol{b}}=N^{-1}\boldsymbol{b}$, $\boldsymbol{z}$ 为辅助向量, $\boldsymbol{u}$ 为乘子向量, $\boldsymbol{y}$ 为模型输出, 式(2a)中 $n=1,2,\cdots,N$.

本文考虑的正则化最小二乘拟合问题

$$\min_{\boldsymbol{x}}\frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2+\frac{1}{2}\gamma^2\|\boldsymbol{x}\|_2^2 \tag{3}$$

其中 $\gamma^2>0$ 为正则化因子, 是问题(1)的一个特例, 即

$$f(\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b})=\frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2, r(\boldsymbol{x})=\frac{1}{2}\gamma^2\|\boldsymbol{x}\|_2^2 \tag{4}$$

将式(4)对应的 $f(\cdot)$ 及 $r_n(x_n)$ 代入式(2), 可得到问题(3)的 MS-ADMM:

$$x_n^{k+1}=x_n^k-N^{-1}\frac{\bar{\rho}^{-1}\bar{\gamma}^2x_n^k+\bar{\boldsymbol{a}}_n^{\mathrm{T}}(\bar{\boldsymbol{A}}\boldsymbol{x}^k+\boldsymbol{u}^k-\boldsymbol{z}^k)}{\rho^{-1}\bar{\gamma}^2+\|\bar{\boldsymbol{a}}_n\|^2} \tag{5a}$$

$$\boldsymbol{z}^{k+1}=(1+\bar{\rho})^{-1}[\bar{\boldsymbol{b}}+\bar{\rho}(\bar{\boldsymbol{A}}\boldsymbol{x}^{k+1}+\boldsymbol{u}^k)] \tag{5b}$$

$$\boldsymbol{u}^{k+1}=\boldsymbol{u}^k+\bar{\boldsymbol{A}}\boldsymbol{x}^{k+1}-\boldsymbol{z}^{k+1} \tag{5c}$$

其中, $\bar{\gamma}=N^{-1}\gamma$, $\bar{\rho}=N^{-1}\rho$, 式(5a)中 $n=1,2,\cdots,N$.

注意到式(5a)对 $\boldsymbol{x}^k$ 的更新可写成

$$x_n^{k+1}=x_n^k+N^{-1}q_n^k \tag{6a}$$

$$q_n^k=-\frac{\bar{\rho}^{-1}\bar{\gamma}^2x_n^k+\bar{\boldsymbol{a}}_n^{\mathrm{T}}(\bar{\boldsymbol{A}}\boldsymbol{x}^k+\boldsymbol{u}^k-\boldsymbol{z}^k)}{\rho^{-1}\bar{\gamma}^2+\|\bar{\boldsymbol{a}}_n\|^2} \tag{6b}$$

与无约束优化迭代算法的一般框架类比, $\boldsymbol{q}^k=[q_1^k,q_2^k,\cdots,q_N^k]^{\mathrm{T}}$ 相当于一个搜索方向, N^{-1} 则相当于一个搜索步长, 而增大搜索步长有可能加速算法收敛. 受此启发, 让 MS-ADMM 式(5)中 $\boldsymbol{x}$-更新的步长因子可调, 即用可调步长因子 $\bar{\alpha}$ 代替固定步长 N^{-1}. 当 $\bar{\alpha}>N^{-1}$ 时, 有望得到比 $\bar{\alpha}=N^{-1}$ 时更快的收敛速度. 用 $\bar{\alpha}$ 代替式(5a)中的 N^{-1} 后, MS-ADMM 式(5)变成:

$$x_n^{k+1}=x_n^k-\bar{\alpha}\frac{\bar{\rho}^{-1}\bar{\gamma}^2x_n^k+\bar{\boldsymbol{a}}_n^{\mathrm{T}}(\bar{\boldsymbol{A}}\boldsymbol{x}^k+\boldsymbol{u}^k-\boldsymbol{z}^k)}{\rho^{-1}\bar{\gamma}^2+\|\bar{\boldsymbol{a}}_n\|^2} \tag{7a}$$

$$\boldsymbol{z}^{k+1}=(1+\bar{\rho})^{-1}[\bar{\boldsymbol{b}}+\bar{\rho}(\bar{\boldsymbol{A}}\boldsymbol{x}^{k+1}+\boldsymbol{u}^k)] \tag{7b}$$

$$\boldsymbol{u}^{k+1}=\boldsymbol{u}^k+\bar{\boldsymbol{A}}\boldsymbol{x}^{k+1}-\boldsymbol{z}^{k+1} \tag{7c}$$

其中式(7a)中 $n=1,2,\cdots,N$.

称算法式(7)为正则化最小二乘问题(3)的 MS-GADMM, 它是 MS-ADMM 式(5)的一个加速变体. 从形式上看, 这里的加速方法与文献[18]的松弛技术类似, 都是让 $\boldsymbol{x}$-更新的步长因子可调. 但文献[18]是从放大 $\boldsymbol{a}_n x_n-\boldsymbol{z}_n=0$ 的约束残差(即对约束 $\boldsymbol{a}_n x_n-\boldsymbol{z}_n=0$ 进行松弛)出发得到最终算法, 正则项为 0 时算法的导出是正确的. 但当正则项不为 0 时, 由这种松弛技术难以导出简单实用的 ADMM 加速变体. 本文直接从增大 $\boldsymbol{x}$-更新步长因子的思想出发, 得到的 MS-GADMM, 既适合有正则项也适合无正则项的最小二乘拟合问题.

对于 MS-GADMM 式(7)的合理性, 有如下定理.

定理 1 对给定的参数 $\bar{\alpha}$ 和 $\bar{\rho}$, 如果 MS-GADMM 式(7)在 $\lim_{k\to\infty}\boldsymbol{x}^k=\boldsymbol{x}^\infty$ 意义下收敛, 则 $\boldsymbol{x}^\infty$ 一定是正则化最小二乘拟合问题(3)的解, 即

$$\boldsymbol{x}^\infty=(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\gamma^2\mathbf{I}_N)^{-1}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b} \tag{8}$$

其中 $\mathbf{I}_N$ 表示 N 阶单位矩阵.

由定理 1 知,只要 $\bar{\alpha}$ 和 $\bar{\rho}$ 选择得当,MS-GADMM 式(7)收敛后得到的 $\boldsymbol{x}^{\infty}$,与问题(3)基于矩阵逆的解析解是相同的. 但算法是否收敛及算法的收敛速度,依赖于参数 $\bar{\alpha}$ 和 $\bar{\rho}$ 的取值.

将 MS-GADMM 写成以下线性系统形式:

$$\begin{bmatrix} \bar{\boldsymbol{x}}^{k+1}-\bar{\boldsymbol{x}}^{\infty} \\ \bar{\boldsymbol{u}}^{k+1}-\bar{\boldsymbol{u}}^{\infty} \end{bmatrix} = \boldsymbol{Q}\begin{bmatrix} \bar{\boldsymbol{x}}^{k}-\bar{\boldsymbol{x}}^{\infty} \\ \bar{\boldsymbol{u}}^{k}-\bar{\boldsymbol{u}}^{\infty} \end{bmatrix} \tag{9}$$

其中

$$\bar{\boldsymbol{x}}^{k}=\boldsymbol{D}^{-1}\boldsymbol{x}^{k},\bar{\boldsymbol{u}}^{k}=\boldsymbol{D}\bar{\boldsymbol{A}}^{\mathrm{T}}\boldsymbol{u}^{k} \tag{10a}$$

$$\boldsymbol{D}=\mathrm{diag}\{d_1,d_2,\cdots,d_N\} \tag{10b}$$

$$d_n=\sqrt{[N^{-1}\bar{\rho}^{-1}\bar{\gamma}^2+\|\bar{\boldsymbol{a}}_n\|_2^2]^{-1}}>0 \tag{10c}$$

$$\boldsymbol{Q}=\begin{bmatrix} \boldsymbol{B} & -\bar{\alpha}(1-\bar{\rho})\mathbf{I}_N \\ \boldsymbol{GB}/(1+\bar{\rho}) & [\mathbf{I}_N-\bar{\alpha}(1-\bar{\rho})\boldsymbol{G}]/(1+\bar{\rho}) \end{bmatrix} \tag{10d}$$

$$\boldsymbol{B}=\mathbf{I}_N-\bar{\alpha}(\bar{\rho}^{-1}\bar{\gamma}^2\boldsymbol{D}^2+\boldsymbol{G}),\boldsymbol{G}=\boldsymbol{D}\bar{\boldsymbol{A}}^{\mathrm{T}}\bar{\boldsymbol{A}}\boldsymbol{D} \tag{10e}$$

根据线性系统理论[21],可以得到定理 2.

定理 2　MS-GADMM 式(7)在"$\lim_{k\to\infty}\boldsymbol{x}^k=\boldsymbol{x}^{\infty}$"意义下收敛的充要条件是,矩阵 $\boldsymbol{Q}$ 的特征值均在单位圆内.

3　基于 MS-GADMM 的 RELM

3.1　RELM

对于一个 L-类分类问题,假设训练样本集为 $\{(\boldsymbol{v}_m,\boldsymbol{t}_m),m=1,2,\cdots,M\}$,其中 $\boldsymbol{v}_m=[v_{m1},v_{m2},\cdots,v_{mP}]^{\mathrm{T}}$ 是 P 维样本输入,$\boldsymbol{t}_m=[t_{m1},t_{m2},\cdots,t_{mL}]$ 是 L 维目标输出. SLFN 有 N 个隐节点,隐节点的激活函数均为 $g(\cdot)$. 样本输入为 $\boldsymbol{v}_m$ 时,SLFN 的模型输出 $\boldsymbol{o}_m=[o_{m1},o_{m2},\cdots,o_{mL}]$ 为

$$\boldsymbol{o}_m=\boldsymbol{g}(\boldsymbol{v}_m^{\mathrm{T}}\boldsymbol{W}+\boldsymbol{s})\boldsymbol{\Theta},\quad m=1,2,\cdots,M \tag{11}$$

其中 $\boldsymbol{W}=[\boldsymbol{w}_1,\boldsymbol{w}_2,\cdots,\boldsymbol{w}_N]\in\mathrm{R}^{P\times N}$ 是隐节点的输出权矩阵,$\boldsymbol{\Theta}=[\boldsymbol{\theta}_1,\boldsymbol{\theta}_2,\cdots,\boldsymbol{\theta}_L]\in\mathrm{R}^{N\times L}$ 是输出节点与隐节点的连接权矩阵,$\boldsymbol{s}=[s_1,s_2,\cdots,s_N]\in\mathrm{R}^{1\times N}$ 是隐节点的偏置向量,

$$\boldsymbol{g}(\boldsymbol{v}_m^{\mathrm{T}}\boldsymbol{W}+\boldsymbol{s})=[g(\boldsymbol{v}_m^{\mathrm{T}}\boldsymbol{w}_1+s_1),\cdots,g(\boldsymbol{v}_m^{\mathrm{T}}\boldsymbol{w}_N+s_N)] \tag{12}$$

是第 m 个样本输入激发的隐节点输出. ELM 中,隐节点的输入权 $\boldsymbol{W}$ 和偏置 $\boldsymbol{s}$ 是随机产生后确定不变的,输出权 $\boldsymbol{\Theta}$ 是最小化模型输出 $\boldsymbol{O}=\boldsymbol{H\Theta}$ 与目标输出 $\boldsymbol{T}$ 之间的误差平方和得到的,其中 $\boldsymbol{H}$ 为 SLFN 的隐层输出矩阵,表示如下

$$\boldsymbol{H}=[\boldsymbol{g}^{\mathrm{T}}(\boldsymbol{v}_1^{\mathrm{T}}\boldsymbol{W}+\boldsymbol{s}),\cdots,\boldsymbol{g}^{\mathrm{T}}(\boldsymbol{v}_M^{\mathrm{T}}\boldsymbol{W}+\boldsymbol{s})]^{\mathrm{T}} \tag{13a}$$

$$\boldsymbol{T}=[\boldsymbol{t}_1^{\mathrm{T}},\boldsymbol{t}_2^{\mathrm{T}},\cdots,\boldsymbol{t}_M^{\mathrm{T}}]^{\mathrm{T}} \tag{13b}$$

为了增强所训练神经网络的泛化性能,RELM[2]在最小化训练误差平方和的同时,也最小化输出节点权系数的平方和. 即

$$\min_{\boldsymbol{\Theta}}\frac{1}{2}\|\boldsymbol{H\Theta}-\boldsymbol{T}\|_{\mathrm{F}}^2+\frac{1}{2}\gamma^2\|\boldsymbol{\Theta}\|_{\mathrm{F}}^2 \tag{14}$$

其中 $\gamma^2>0$ 是正则化参数,$\|\cdot\|_{\mathrm{F}}$ 表示矩阵的 Frobenius 范数. 解析求出问题(14)的解,为

$$\boldsymbol{\Theta}^{*}=\begin{cases}(\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}+\gamma^2\mathbf{I}_N)^{-1}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{T}, & M\geqslant N\\ \boldsymbol{H}^{\mathrm{T}}(\boldsymbol{H}\boldsymbol{H}^{\mathrm{T}}+\gamma^2\mathbf{I}_M)^{-1}\boldsymbol{T}, & M<N\end{cases} \tag{15}$$

当 M 和 N 很大时,基于矩阵逆的方法计算量很大. 本文应用前述具有高度并行结构的 MS-GADMM,来并行求解 RELM 问题(14).

3.2　RELM 的 MS-GADMM

把目标输出 $\boldsymbol{T}$ 按类分划,即 $\boldsymbol{T}=[\bar{\boldsymbol{t}}_1,\bar{\boldsymbol{t}}_2,\cdots,\bar{\boldsymbol{t}}_L]$,其中 $\bar{\boldsymbol{t}}_1=[t_{1l},t_{2l},\cdots,t_{Ml}]^{\mathrm{T}}\in\mathrm{R}^M$ 为 $\boldsymbol{T}$ 的第 l 列. 注意到

$$\|\boldsymbol{H\Theta}-\boldsymbol{T}\|_{\mathrm{F}}^2=\sum_{l=1}^{L}\|\boldsymbol{H}\boldsymbol{\theta}_l-\bar{\boldsymbol{t}}_l\|_2^2 \tag{16a}$$

$$\|\boldsymbol{\Theta}\|_{\mathrm{F}}^2=\sum_{l=1}^{L}\|\boldsymbol{\theta}_l\|_2^2 \tag{16b}$$

以矩阵 $\boldsymbol{\Theta}$ 为优化变量的正则化最小二乘问题(14),等价于 l 个以权向量 $\boldsymbol{\theta}_l$ 为优化变量的问题:

$$\min_x\frac{1}{2}\|\boldsymbol{H}\boldsymbol{\theta}_l-\bar{\boldsymbol{t}}_l\|_2^2+\frac{1}{2}\gamma^2\|\boldsymbol{\theta}_l\|_2^2,\quad l=1,2,\cdots,L \tag{17}$$

对于每个 l,问题(17)就是问题(3). 基于此及算法式(7),在得到隐层输出 $\boldsymbol{H}$ 后,输出层权矩阵优化的 MS-GADMM 如下. 即分别对 $l=1,2,\cdots,L$,执行迭代:

$$\theta_{nl}^{k+1}=\theta_{nl}^{k}-\bar{\alpha}\frac{\bar{\rho}^{-1}\bar{\gamma}^2\theta_{nl}^{k}+\bar{\boldsymbol{h}}_n^{\mathrm{T}}\boldsymbol{v}_l^{k}}{N^{-1}\bar{\rho}^{-1}\bar{\gamma}^2+\|\bar{\boldsymbol{h}}_n\|^2},\quad n=1,2,\cdots,N \tag{18a}$$

$$y_{ml}^{k+1}={}_m\bar{\boldsymbol{h}}\boldsymbol{\theta}_l^{k+1},\quad m=1,2,\cdots,M \tag{18b}$$

$$z_{ml}^{k+1}=\frac{1}{1+\bar{\rho}}\bar{t}_{ml}+\frac{\bar{\rho}}{1+\bar{\rho}}[y_{ml}^{k+1}+u_{ml}^{k}],\quad m=1,2,\cdots,M \tag{18c}$$

$$\boldsymbol{v}_{ml}^{k+1}=y_{ml}^{k+1}-z_{ml}^{k+1},u_{ml}^{k+1}=u_{ml}^{k}+\boldsymbol{v}_{ml}^{k+1},\quad m=1,2,\cdots,M \tag{18d}$$

$$\boldsymbol{v}_{ml}^{k+1}=\boldsymbol{v}_{ml}^{k+1}+u_{ml}^{k+1},\quad m=1,2,\cdots,M \tag{18e}$$

其中 $\boldsymbol{\theta}_l^k=[\theta_{1l}^k,\theta_{2l}^k,\cdots,\theta_{Nl}^k]^{\mathrm{T}}$,$y_l^k=[y_{1l}^k,y_{2l}^k,\cdots,y_{Ml}^k]^{\mathrm{T}}$,$z_l^k=[z_{1l}^k,z_{2l}^k,\cdots,z_{Ml}^k]^{\mathrm{T}}$,$u_l^k=[u_{1l}^k,u_{2l}^k,\cdots,u_{Ml}^k]^{\mathrm{T}}$,$\upsilon_l^k=[\upsilon_{1l}^k,\upsilon_{2l}^k,\cdots,\upsilon_{Ml}^k]^{\mathrm{T}}$,${}_m\bar{\boldsymbol{h}}$ 及 $\bar{\boldsymbol{h}}_n$ 是 $\bar{\boldsymbol{H}}=N^{-1}\boldsymbol{H}$ 的第 m 行和第 n 列.

注意到,RELM 的 MS-GADMM 算法式(18),对每个 $l=1,2,\cdots,L$ 是完全独立的,因此对 l 的循环是完全可并行执行的. 算法每个迭代步对向量 $\boldsymbol{\theta}_l^k,\boldsymbol{y}_l^k,\boldsymbol{z}_l^k,\boldsymbol{u}_l^k$ 的更新,也是按其标量分量独立进行的,其每个迭代步的所有标量更新,即(18a)对 n 的循环以及(18b)~(18e)对 m 的循环,都是可并行执行的. 但由于(18b)计算 y_{ml}^{k+1} 要用到(18a)中算出的 $\boldsymbol{\theta}_l^{k+1}$ 的所有分量,(18a)计算 θ_{nl}^{k+1} 要用到 $\boldsymbol{v}_l^k$ 的所有分量,所以对每个 l,不能把式(18)整个迭代过程分解成完全独立的子过程.

然而,由于在(18a)~(18e)的每个迭代,对变量的更新都是标量化的,有很好的并行性,易在 MATLAB 环境下用 GPU 并行实现. 并行实现后,MS-GADMM 的运算效率将大幅提高,有助于解决大数据环境下正则化

ELM 的计算瓶颈.

3.3 算法的计算复杂度

将迭代过程中不发生变化的量事先算出,比如 $d_n=[\rho^{-1}\bar{\gamma}^2+\|\bar{\boldsymbol{h}}_n\|^2]^{-1/2}$,$\bar{\boldsymbol{\phi}}_n=\bar{\alpha}d_n^2\bar{\boldsymbol{h}}_n$,$y_n=(1-\bar{\alpha}\rho^{-1}\bar{\gamma}^2d_n^2)$,其中 $\rho^{-1}=N^{-1}\bar{\rho}^{-1}$,并令 $\sigma_{ml}=\bar{t}_{ml}/(1/\bar{\rho})$,$\beta=\bar{\rho}/(1+\bar{\rho})$. 容易分析出,RELM 的 MS-GADMM 算法,每个由迭代步(18a)到迭代步(18e)的循环需 $L(2MN+M+N)$ 个浮点乘法运算和 $L(2MN+4M)$ 个浮点加法运算. 如果能够完全并行执行每个迭代步的所有标量更新,则每个循环需 $M+N+2$ 个浮点乘法运算和 $M+N+4$ 个浮点加法运算,略小于文献[18]的 MS-RADMM.

4 仿真实验及结果分析

本节在 5 个基准现实数据集上,对 RELM 的 MS-GADMM 进行仿真,并与本文的 MS-ADMM 及文献[18]的 MS-RADMM 进行性能对比. 仿真中实现算法的软件平台为 Matlab 2017,硬件平台是 CPU 为 Intel(R) Core(TM) i7-8700K@3.7GHz,内存为 64GB 的台式计算机. 5 个数据集分别是,Gisette 数据集①(2 类,6000 个训练样本,1000 个测试样本),20Newsgroup 数据集(20 类,11307 个训练样本,7539 个测试样本),USPS 数据集②(10 类,7291 个训练样本,2007 个测试样本),NORB 数据集③(5 类,24300 个训练样本,24300 个测试样本)和 MNIST 数据集④(10 类,35000 个训练样本,35000 个测试样本).

4.1 收敛性能比较实验

首先在 USPS 和 20Newsgroup 两个数据集上,分别用 MS-RADMM,MS-ADMM 以及 MS-GADMM 进行实验. 由于本文重点关注的是算法的并行性能和收敛速度,实验中没有对模型参数进行细致优化. 本节取 $N=10000$,隐节点激活函数为 Sigmoid 函数,正则化参数 $\gamma^2=10^3$. 算法的参数取值如表 1 示,其中 MS-RADMM 的参数值是根据文献[18]的方法计算得到的最优值,另两个算法则是以 MS-RADMM 参数值作为初值,在其附近多次试验调整得到的最好参数.

图 1、图 2 是算法训练过程中,目标函数值 $f(\boldsymbol{\Theta})$ 与最优值 $f(\boldsymbol{\Theta}^*)$ 之间的相对偏差 $D_r(k)$ 的收敛曲线,其中

$$f(\boldsymbol{\Theta})=\frac{1}{2}\|\boldsymbol{H}\boldsymbol{\Theta}-\boldsymbol{T}\|_{\mathrm{F}}^2+\frac{1}{2}\gamma^2\|\boldsymbol{\Theta}\|_{\mathrm{F}}^2 \tag{19}$$

$$D_r(k)=\left|f(\boldsymbol{\Theta}^k)/f(\boldsymbol{\Theta}^*)-1\right| \tag{20}$$

表 1 三种算法的参数取值

数据集	MS-ADMM	MS-GADMM		MS-RADMM	
	$\bar{\rho}$	$\bar{\rho}$	$\bar{\alpha}$	$\hat{\rho}$	$\hat{\alpha}$
USPS	0.0038	0.0060	2.40×10^{-4}	0.0123	3.109×10^{-4}
20Newsgroup	0.0043	0.0060	2.30×10^{-4}	0.0106	2.8542×10^{-4}

由图 1、图 2 可以清晰地看出,在 2 个数据集上,3 种算法都是线性收敛的. MS-GADMM 的收敛比率比 MS-ADMM 小(曲线斜率的绝对值更大),表明本文的加速技术,即在 MS-ADMM 中让其 $\boldsymbol{x}$-更新的步长因子可调,起到了很好的加速作用. 与文献[18]的 MS-RADMM 相比,也有更快的收敛速率,可在更少的迭代次数内获得相同的目标函数相对偏差.

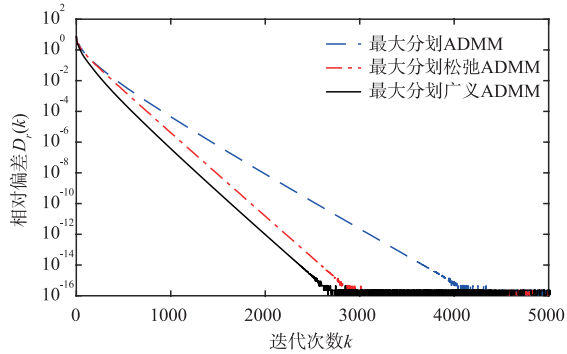


图1 USPS数据集上三种算法的收敛曲线

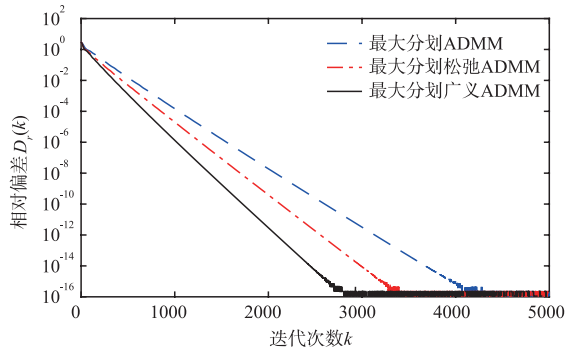


图2 20Newsgroup数据集上三种算法的收敛曲线

4.2 GPU 并行加速实验

用型号为 NVIDIA GeForce RTX 2080Ti 的 GPU,在 MATLAB 环境中用 gpuArray,对 MS-GADMM 及 MS-RADMM 进行并行实现,在所有 5 个数据集上用 GPU 进行并行加速实验. 实验中,正则化参数 $\gamma^2=10^3$;对 Gisette 及 USPS 两个数据集令 $N=20000$,其它数据集令 $N=30000$. 算法的终止条件 $\|\boldsymbol{\Theta}^{k+1}-\boldsymbol{\Theta}^k\|_{\mathrm{F}}<\varepsilon\|\boldsymbol{\Theta}^k\|_{\mathrm{F}}$,其中 $\varepsilon=10^{-4}$ 为容许相对误差. 表 2 列出了两种 ADMM 算法及基于矩阵逆(用 MATLAB 中的左除运算符"\"实现)的方法的迭代次数,无 GPU 时的计算时间,有 GPU 时的计算时间,GPU 加速比及测试集分类正确率等.

① 来自 ASU 特性选择数据集(http://featureselection.asu.edu/datasets.php)

② 来自 UCI 机器学习资源库(http://archive.ics.uci.edu/ml/index.php)

③ 来自 https://cs.nyu.edu/~ylclab/data/norb-v1.0/

④ 来自 http://yann.lecun.com/exdb/mnist/

表 2　基于不同算法的 RELM 训练结果

数据集	算法	迭代次数	计算时间(无 GPU)	计算时间(有 GPU)	GPU 加速比	测试分类正确率
Gisette	逆矩阵方法	–	3.4846s	0.3157s	11.038	97.8000
	MS-RADMM	1126	133.49s	1.4702s	90.797	97.8000
	MS-GADMM	693	81.590s	0.2016s	404.71	97.8000
USPS	逆矩阵方法	–	5.2349s	0.4663s	11.226	94.6687
	MS-RADMM	1123	193.92s	2.5187s	76.992	94.6688
	MS-GADMM	766	133.82s	0.6580s	203.37	94.6687
20Newsgroup	逆矩阵方法	–	18.655s	2.1586s	8.6422	89.5477
	MS-RADMM	1468	568.13s	8.6094s	65.990	89.5344
	MS-GADMM	1000	384.49s	4.4073s	87.239	89.5609
NORB	逆矩阵方法	–	94.329s	12.355s	7.6349	91.6790
	MS-RADMM	1945	1429.9s	17.957s	79.629	91.6790
	MS-GADMM	1356	1007.6s	10.156s	99.212	91.6255
MNIST	逆矩阵方法	–	164.76s	Failure *	NA	97.8686
	MS-RADMM	2812	3651.3s	85.656s	42.628	97.8714
	MS-GADMM	1590	2068.1s	42.383s	48.796	97.8714

* 超出 GPU 的专用内存空间,运行失败

三种算法的测试集分类正确率几乎相等,表示在给定的容许误差下达到了相同的测试性能. 可以看到,与 MS-RADMM 相比,MS-GADMM 的迭代次数少,计算时间短. 没有 GPU 并行加速时,2 种 ADMM 算法都比基于矩阵逆的方法计算时间长. 都采用 GPU 并行加速后,MS-GADMM 与基于矩阵逆的方法计算时间相当,在 Gisette 和 NORB 数据集上,MS-GADMM 的计算时间比基于矩阵逆的方法短. 在 GPU 下 3 种算法都能得到加速,2 种 ADMM 算法的加速比都比矩阵逆的方法大很多,MS-GADMM 的加速比又比 MS-RADMM 大. 即 MS-GADMM 有更好的并行性.

5　结论与未来工作

提出的 MS-ADMM,将优化问题按优化变量分解为标量优化问题,得到的标量化算法具有高度并行的算法结构. 提出的 MS-GADMM 通过增大 $\boldsymbol{x}$-更新步长因子,提高了算法的收敛速度. 应用于 RELM,实验表明 MS-GADMM 比文献[18]的 MS-RADMM 收敛快. GPU 加速实验获得的大加速比,表明基于 MS-GADMM 的 RELM,比基于 MS-RADMM 的 RELM 具有更好的并行性能.

算法的参数选择,非二次型学习的 MS-GADMM 及内存受限时的高效算法,是我们未来的工作方向.

参考文献

[1] HUANG G B, ZHU Q Y, SIEW C K. Extreme learning machine: theory and applications [J]. Neurocomputing, 2006, 70(1–3): 489–501.

[2] HUANG G B, ZHOU H, DING X, et al. Extreme learning machine for regression and multiclass classification [J]. IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics, 2012, 42(2): 513–529.

[3] HUANG Z Y, YU Y L, GU J, et al. An efficient method for traffic sign recognition based on extreme learning machine [J]. IEEE Transactions on Cybernetics, 2017, 47(4): 920–933.

[4] 孙锐,张东东,高隽. 基于分层极限学习机和局部稀疏模型的视觉跟踪算法[J]. 模式识别与人工智能, 2017, 30(4): 302–313.
SUN Y, ZHANG D D, GAO J. Visual tracking via hierarchical extreme learning machine and local sparse model [J]. Pattern Recognition and Artificial Intelligence, 2017, 30(4): 302–313. (in Chinese)

[5] 程玉胜,赵大卫,王一宾,等. 非平衡化标签补全核极限学习机多标签学习[J]. 电子学报, 2019, 47(3): 719–725.
CHENG Y S, ZHAO D W, WANG Y B, et al. Multi-label learning of kernel extreme learning machine with non-equilibrium label completion [J]. Acta Electronica Sinica, 2019, 47(3): 719–725. (in Chinese)

[6] 刘金平,何捷舟,马天雨,等. 基于 KELM 选择性集成的复杂网络环境入侵检测[J]. 电子学报, 2019, 47(5): 1070–1078.
LIU J P, HE J Z, MA T Y, et al. Selective ensemble of

KELM-based complex network intrusion detection[J]. Acta Electronica Sinica,2019,47(5):1070 – 1078. (in Chinese)

[7] 何清,李宁,罗文娟,等. 大数据下的机器学习算法综述[J]. 模式识别与人工智能,2014,27(4): 327 – 336.
HE Q,LI N,LUO W J,et al. A survey of machine learning algorithms for big data[J]. Pattern Recognition and Artificial Intelligence,2014,27(4):327 – 336. (in Chinese)

[8] HE Q,SHANG T F,ZHUANG F Z,et al. Parallel extreme learning machine for regression based on mapreduce[J]. Neurocomputing,2013,102:52 – 58.

[9] WANG Y Q,DOU Y,LIU X W,et al. PR-ELM:Parallel regularized extreme learning machine based on cluster[J]. Neurcomputing,2016,173:1073 – 1087.

[10] 刘鹏,王学奎,黄宜华,等. 基于 Spark 的极限学习机算法并行化研究[J]. 计算机科学,2017,44(12):33 – 37.
LIU P,WANG X K,HUANG Y H,et al. Study of ELM algorithm parallelization based on Spark[J]. Computer Science,2017,44(12):33 – 37. (in Chinese)

[11] CHEN C,LI K L,OUYANG A J,et al. GPU-accelerated parallel hierarchical extreme learning machine on Flink for big data[J]. IEEE Transactions on Systems,Man and Cybernetics:Systems,2017,47(10):2740 – 2753.

[12] BOYD S,PARIKH N,CHU E,et al. Distributed optimization and statistical learning via the alternating direction method of multipliers[J]. Foundations and Trends in Machine Learning,2011,3(1):1 – 122.

[13] GOLDSTEIN T,O'DONOGHUE B,SETZER S. Fast alternating direction optimization methods[J]. SIAM Journal on Imaging Sciences,2014,7(3):1588 – 1623.

[14] YANG J F,ZHANG Y. Alternating direction algorithms for l_1-problems in compressive sensing[J]. SIAM Journal on Scientific Computing,2011,33(1):250 – 278.

[15] WANG H H,GAO Y,SHI Y H,et al. Group-based alternating direction method of multipliers for distributed linear classification[J]. IEEE Transactions on Cybernetics,2017,47(11):3568 – 3582.

[16] WANG H,FENG R B,HAN Z F,et al. ADMM-based algorithm for training fault tolerant RBF networks and selecting centers[J]. IEEE Transactions on Neural Networks and Learning Systems,2018,29(8):3870 – 3878.

[17] 胡园园,罗倩,段中钰,等. 基于部分平行 ADMM 求解谱寻求问题的时频分析[J]. 电子学报,2019,47(11): 2392 – 2398.
HU Y Y,LUO Q,DUAN Z Y,et al. Time-frequency analysis based on partly parallel ADMM solving spectral pursuit problem[J]. Acta Electronica Sinica,2019,47(11): 2392 – 2398. (in Chinese)

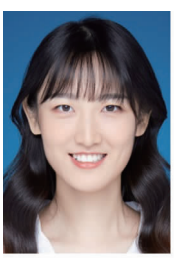
[18] LAI X P,CAO J W,HUANG X F,et al. A maximally split and relaxed ADMM for regularized extreme learning machines[J]. IEEE Transactions on Neural Networks and Learning Systems,2020,31(6):1899 – 1913.

[19] 张立佳,赖晓平,曹九稳. 正则化超限学习机的多分块松弛交替方向乘子法[J]. 模式识别与人工智能,2019,32(12): 1107 – 1115.
ZHANG L J,LAI X P,CAO J W. Multi-partition relaxed alternating direction method of multipliers for regularized extreme learning machine[J]. Pattern Recognition and Artificial Intelligence,2019,32(12):1107 – 1115. (in Chinese)

[20] 马梦瑶,赖晓平,孟海龙. 二维 FIR 滤波器约束最小二乘设计的最大分划松弛 ADMM 算法[J]. 电子学报,2020,48(3): 510 – 517.
MA M Y,LAI X P,MENG H L. A maximally split and relaxed ADMM for constrained least-squares design of two-dimensional FIR filters[J]. Acta Electronica Sinica,2020,48(3):510 – 517. (in Chinese)

[21] ANTSAKLIS P J,MICHEL A N. A Linear Systems Primer[M]. Boston,MA,USA:Birkhäuser,2007.

作者简介



侯秀聪 女,1995 年生于山东济南,现为杭州电子科技大学自动化学院研究生. 研究方向为机器学习.
E-mail:cindy_hxc@163.com

赖晓平(通信作者) 男,1965 年生于江西赣州,现为杭州电子科技大学教授,博士生导师. 主要研究方向为优化方法、数字滤波器设计、机器学习等.
E-mail: laixp@hdu.edu.cn