上下文分解机的自适应更新策略

姚 杏[1] 朱福喜[1] 阳小兰[2] 郑 麟[1] 刘世超[1]
(武汉大学计算机学院 武汉 430072)[1] (武昌理工学院信息工程学院 武汉 430223)[2]

摘 要 分解机模型已经被成功应用于上下文推荐系统。在分解机模型的学习算法中,交替最小二乘法是一种固定其他参数只求单一参数最优值的学习算法,其参数数目影响计算复杂度。然而当特征数目很大时,参数数目随着特征数目急剧增加,导致计算复杂度很高;即使有些参数已经达到了最优值,每次迭代仍更新所有的参数。因此,主要改进了交替最小二乘法的参数更新策略,为参数引入自适应误差指标,通过权重和参数绝对误差共同决定该参数更新与否,使得每次迭代时重点更新最近两次迭代取值变化较大的参数。这种仅更新自适应误差大于阈值的参数的策略不但减少了需要更新的参数数目,进而加快了算法收敛的速度和缩短了运行时间,而且参数权重由误差决定,又修正了误差。在Yahoo和Movielens数据集上的实验结果证明:改进的参数更新策略运行效率有明显提高。

关键词 分解机模型,交替最小二乘法,推荐系统,自适应误差
中图法分类号 TP181 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.9.013

Adaptive Parameters Updating Strategy of Context-aware Factorization Machines

YAO Xing[1] ZHU Fu-xi[1] YANG Xiao-lan[2] ZHENG Lin[1] LIU Shi-chao[1]
(School of Computer, Wuhan University, Wuhan 430072, China)[1]
(School of Information Engineering, Wuchang University of Technology, Wuhan 430223, China)[2]

Abstract Context-aware factorization machine has been successfully applied in the context-aware recommendation system. In the learning algorithm of factorization machines, alternating least-squares is a learning algorithm that fixes other parameters just to find the optimal value of a single parameter, and the number of parameters and the sample size will affect the computational complexity. However, when the number of features is large, the number of parameters increases along with the increase of the number of features, resulting in high computational complexity. Even though some parameters have achieved the optimal value, all parameters will be updated in each iteration. This paper mainly improved the para-meters updating strategy of alternating least-squares. Adaptive error index was introduced into the parameter. Updating the parameter or not is co-determined by the weights and the absolute error of parameters, so that each iteration focuses on parameters whose last two iterative values change greatly. This strategy only updates parameters whose adaptive errors are greater than the thresholds. It not only reduces the number of parameters that need to be updated, so as to accelerate the algorithm convergence speed and shorten the operation time, but also the weight of parameters is determined by the error, to correct the error. The results of experiments on Yahoo and Movielens data sets show that the effect of the improved parameter updating strategy is better.

Keywords Factorization machines, Alternating least-squares, Recommender systems, Adaptive error

1 前言

推荐系统中传统协同过滤的方法CF[1,2]和矩阵分解MF[3,4]等都仅仅考虑用户和物品,而忽略了它们所处环境的上下文信息。然而在评分预测、社交网络预测等领域中,这些上下文信息(例如时间、地点等)往往对预测结果起着关键的作用。例如,2009年KDD会议中Koren的基于时间上下文信息的推荐系统受到广泛关注,文献[5]系统地总结了各种使用时间信息(包括用户近期行为、时间的周期性等),将这些信息用于推荐系统中获得了很好的预测效果。

随着上下文信息的加入,需要考虑的数据类型也增多。但是并非所有模型都能处理所有的输入数据。例如SVM[6]、SVD模型[7]等无法处理非数值型数据,张量分解[8]无法处理分类集领域的数据和数值数据等。Rendle在2010年[9]提出的FM模型可以利用其规定的3类数据领域的转换规则,将任何数据类型的数据统一转换为数值型数据进行处理,从而解决了其他模型[6-8]不能处理某些数据类型的问题。随着FM模型的发展,出现了基于FM模型的各种变形,例如FM

到稿日期:2015-07-12 返修日期:2015-08-12 本文受国家自然科学基金(61272277),湖北省自然科学基金(2014CFB356)资助。
姚 杏(1990-),女,硕士生,主要研究方向为数据挖掘、推荐系统,E-mail:yaoxing_1990@163.com;朱福喜(1957-),男,教授,博士生导师,主要研究方向为智能计算、Web数据挖掘、自然语言处理;阳小兰(1978-),女,副教授,主要研究方向为Web数据挖掘;郑 麟(1984-),男,博士生,主要研究方向为数据挖掘、推荐系统;刘世超(1989-),男,博士生,主要研究方向为数据挖掘、社会网络。

模型 Boosting[10-12]以及非线性变形：梯度集成分解机 GBFM、高斯过程 GPFM 等。然而这些基于分解机的模型中，并没有对参数的更新与否进行控制，即使有些参数的取值变化很小，其仍然更新所有的参数。

在 FM 模型中加入了上下文相关信息，导致特征数目急剧增加，从而影响学习算法的计算复杂度。虽然很多文献研究了特征的选择方法[13,14]，也取得了一定的效果，但是它们都是建立在训练模型之前的预处理，并没有自适应地选择特征。事实上，影响计算复杂度的本质是模型参数数目和特征矩阵中非 0 的元素数目，所以在学习模型的过程中自适应地选择参数更新将会是一个很大的挑战。

在分解机模型的学习算法中存在梯度下降法和交替最小二乘法，其他的学习算法都是在这两种学习算法上的改进。因交替最小二乘法收敛速度远比梯度下降法收敛速度快，故本文主要改进了交替最小二乘学习算法（Alternating Least-Squares，ALS）[15]的参数更新策略，通过将自适应地选择特征转化为自适应地选择模型参数进行更新，在每次迭代中停滞有限次更新已经接近最优状态的参数，通过权重控制其在未来某一次迭代中是否重新获得更新。这种更新策略对于特征数目很大的模型在缩短运行时间方面尤为重要。

本文提出的自适应参数更新策略交替最小二乘法（Adaptive Parameters Updating Strategy With Alternating Least-Squares，AU-ALS）算法具有如下优点：

1）每次迭代中减少了进行更新的参数数目，从而减少了每次迭代的运行时间。

2）每次迭代中重点关注变化率较大的参数，使它们获得更多的更新机会，从而修正误差。

3）整体上加快收敛速度，提高运行效率。

2 相关工作

2.1 上下文相关的 FM

FM 模型由 Rendle 在 2010 年[9]首次提出，适用于回归和分类问题。提出该模型的初衷是为了弥补经典线性回归模型中没有考虑互异特征分量之间的相互关系的缺陷，针对每个维度的特征分量，引入辅助向量，则辅助向量的内积即为互异特征分量之间相互关系的权重。其不但用理论证明了 FM 仅仅通过特征矩阵就能表示大部分的分解模型，如 MF、SVD++、PITF和 FPMC，而且证明了 FM 模型具有线性的计算复杂度，可以对任意的（即使高度稀疏的）实值向量进行处理，并有很高的预测精度。2011 年[15] Rendle 将 FM 模型应用于上下文推荐系统中，针对上下文信息的评分预测系统，举例说明了将分类领域（Categorical domain）、分类集领域（Categorical set domain）、数值领域（Real valued domains）这 3 种领域的数据转换成用于 FM 模型输入矩阵的数据的方法，并提出了一种基于 FM 模型的新的学习算法——交替最小二乘法。这种学习算法通过固定其他参数，求解使平方损失函数最小的单个参数的最优值。它的优点包括：1）不需要事先确定类似随机梯度下降法 SGD 和 SGDA[16]的学习速率；2）与基于上下文协同过滤的多维张量分解模型（Multiverse Recommendation）[8]相比，它具有线性计算复杂度和更好的预测效果。2011 年[18] Rendle 提出了基于贝叶斯理论的 BFM 模型。该模型是分层贝叶斯模型：即首先根据超参数的先验分布，利用 Gibbs 采样求出超参数的后验分布，再利用 Gibbs 采样求出模型参数的后验分布，最后得到参数值。这种分层贝叶斯模型与张量分解[8]、KNN 模型[19]等相比，具有更快的收敛速度、更好的可扩展性和更高的预测精度。2012 年[16] Rendle 提出了基于上下文分解机的自适应正则化方法。它直接通过验证集在训练模型的过程中自适应确定正则化系数，这样不仅不需要手动控制正则化系数，而且折中了模型过拟合和模型复杂程度的关系，从而提高了模型的泛化能力。2013 年马尔可夫链蒙特卡罗法（Markov Chain Monte Carlo，MCMC）[17]针对特征矩阵很大导致收敛速度慢等问题，通过对特征矩阵进行分块，用 BFM 模型对每一块进行处理，最后利用基于贝叶斯概率的马尔科夫链蒙特卡罗推理进行整合。2012 年[20] Rendle 总结了基于 FM 的 4 种学习算法[15-17]。

2.2 交替最小二乘法

若给定样本集 $D,(x^{(i)},y^{(i)})\in D,i=1,\cdots,m$，其中 m 为样本数，$x^{(i)}\in\mathbb{R}^{1\times n}$ 为特征向量，$y^{(i)}\in\mathbb{R}$ 为目标值。本文只考虑 2-FM 模型且损失函数为误差平方和，那么在 ALS 算法学习过程中，需要解决的是只含单个参数的最优化问题：

$$\arg\min_{\theta}\{\sum_{D}[\hat{y}(x^{(i)}|\Theta)-y^{(i)}]^2+\sum_{\Theta}\lambda_\theta\theta^2\} \tag{1}$$

式(1)对 θ 求导，令导数为 0，可得到交替最小二乘法的单参数最优解：

$$\theta^*=\frac{\sum_{i=1}^{m}[\theta h_\theta(x^{(i)})-e_i]h_\theta(x^{(i)})}{\sum_{i=1}^{m}h_\theta^2(x^{(i)})+\lambda_\theta}=\frac{\theta\sum_{i=1}^{m}h_\theta^2(x^{(i)})-\sum_{i=1}^{m}e_ih_\theta(x^{(i)})}{\sum_{i=1}^{m}h_\theta^2(x^{(i)})+\lambda_\theta} \tag{2}$$

其中：

$$h_\theta(x^{(i)})=\begin{cases}1, & \text{if } \theta \text{ is } \omega_0\\ x_l^{(i)}, & \text{if } \theta \text{ is } \omega_l\\ x_l^{(i)}(q_f^{(i)}-\upsilon_{l,f}x_l^{(i)}), & \text{if } \theta \text{ is } \upsilon_{l,f}\end{cases}$$

$$e_i=y^{(i)}-\hat{y}(x^{(i)}|\Theta) \tag{3}$$

$$q_{i,f}=\sum_{l=1}^{n}\upsilon_{l,f}x_l^{(i)} \tag{4}$$

这样，每个 θ^* 的计算都归结于：$\sum_{i=1}^{m}h_\theta^2(x^{(i)})$，$\sum_{i=1}^{m}e_ih_\theta(x^{(i)})$。

3 自适应参数更新交替最小二乘法（AU-ALS）

本节首先描述了相关的准备工作和基于上下文信息的推荐系统中存在的问题；然后重点阐述了解决这些问题的自适应更新策略的理论基础；最后提出了一种改进更新策略的算法，即 AU-ALS 算法，并给出了它的伪代码描述。

3.1 准备工作

对于评分预测问题，其目标是估计下面的函数：

$\hat{y}:\mathbb{R}^d\to\mathbb{R}$

利用 2-FM 模型将 d 维的特征向量 $x^{(i)}\in\mathbb{R}^d$ 映射到一个实数 $\hat{y}^{(i)}\in\mathbb{R}$ 上。

大量上下文信息的加入虽然能在很大程度上改善评分预测的效果，但也付出了花费更多时间用于训练的惨痛代价。因此如何高效地解决上下文感知的评分预测问题是一个很大的挑战。一般来说，由于无法客观地了解特征之间的内在相关关系，进而无法直观地进行特征的选择，特征选择的预处理

也仅仅局限在凭借主观经验删除一些认为对预测结果没有影响的特征上。同时在利用 ALS 算法学习 FM 模型的过程中，测试样本集的平均绝对误差(MAE)和均方误差(RMSE)一开始下降很快，而后趋于平缓，说明了一定时间之后，每次迭代更新所有的参数的计算复杂度不变，却对减少测试误差作用不大。

3.2 AU-ALS 算法

AU-ALS 算法的更新策略是一种自适应地选择参数进行更新的方法，其焦点主要放在最近两次迭代中取值变化大于阈值的参数上，使得最近两次迭代取值变化较大的参数获得相对更多的更新机会；而对于最近两次迭代中取值变化小于阈值的参数，通过权重来控制其在未来哪次迭代中重新获得更新。自适应参数更新策略的主要思路为：当前迭代中，是否更新该参数取决于该参数的自适应误差。如果其值大于阈值，则按原 ALS 算法的参数近似最优解式(2)更新该参数，并按式(5)更新该参数的自适应误差；如果其值小于阈值，只通过该参数的自适应误差乘以式(7)定义的权重，得到其最新的自适应误差，不更新该参数的值。

3.2.1 参数的自适应误差

ALS 算法中对于某个特定的参数来说，每次迭代其取值总是朝着使损失函数最小的方向近似最优解。所以对于相同的参数，可以认为后一次迭代的取值优于前一次迭代的取值，即第 $t+1$ 次迭代的参数值与最优解的距离小于第 t 次迭代的参数值与最优解的距离：$|\theta^* - \theta^{t+1}| \leqslant |\theta^* - \theta^t|$。由于不同参数近似其最优解的速度不一样，因此重点更新最近两次迭代中取值变化较大的参数，这样在较少的运行时间内达到稳定状态的参数数目较多。

定义在第 $t+1$ 次迭代中参数的自适应误差为：

$$Ade_\theta^{t+1} = |(\theta^t - \theta^{t-1})/\theta^{t-1}| \tag{5}$$

那么当迭代次数足够大，参数的取值无限近似其最优值($\theta^t \approx \theta^{t+1} \approx \theta^*$)时，参数的自适应误差趋于 0，即存在一个 $\varepsilon_\theta > 0$，使得 $\lim\limits_{t\to\infty} p(Ade_\theta^{t+1} < \varepsilon_\theta) = 1$。所以如果每次迭代不更新自适应误差小于阈值 ε_θ 的参数，重点更新自适应误差大于阈值的参数，则不仅可以减少每次迭代的运行时间，而且对模型的整体预测精度没有影响。

3.2.2 参数更新

某次迭代中，依据如下更新策略判断参数的取值是否改变：

$$\theta_t := \begin{cases} \theta_{t-1}, & Ade_\theta^{t-1} < \varepsilon_\theta \\ \theta_t^*, & Ade_\theta^{t-1} \geqslant \varepsilon_\theta \end{cases} \tag{6}$$

对于满足自适应误差大于或等于阈值的参数，利用式(2)更新该参数的取值。而对于满足自适应误差小于阈值的参数，仅保持前一次迭代得到的参数值不变，即不改变该参数的取值。在接下来的迭代中，通过不断增大该类参数的自适应误差，使得其在未来的某次迭代中该参数的自适应误差大于阈值，从而使该类参数重新获得更新。

在第 $T(T>t)$ 次迭代中获得更新的机会由权重 β_t 决定。

$$\beta_t = \exp\{1 + e_{t-1}^2\} \tag{7}$$

式(7)为样本平均误差的指数函数，其中 e_{t-1} 为第 $t-1$ 次迭代样本平均误差。则对于 $\forall t>0$，有 $\beta_t \in (e, \infty)$。随着样本平均误差的降低，认为样本平均误差再减小的能力下降，则通过计算参数最优值使得样本平均误差降低的作用不大，因此可以通过减小参数获得更新机会的权重，使参数停滞更新的时间更长，从而减少每次迭代的运行时间。

第 t 次迭代时参数的自适应误差按如下更新：

$$Ade_\theta^t := \begin{cases} Ade_\theta^{t-1} \times \beta_{t-1}, & Ade_\theta^{t-1} < \varepsilon_\theta \\ |(\theta^t - \theta^{t-1})/\theta^{t-1}|, & Ade_\theta^{t-1} \geqslant \varepsilon_\theta \end{cases}$$

其中，从第 t 次迭代开始，对于自适应误差大于阈值的参数，其自适应误差的计算根据式(5)即可。在第 t 次迭代中，对于满足自适应误差小于阈值的参数，其在任意第 $T>t$ 次迭代中的自适应误差为 $Ade_\theta^T := Ade_\theta^{T-1} \times \beta_{T-1}$。则存在实数 $s>0$，停止 s 次参数 θ' 的更新，则第 $s+t$ 次迭代结束，其自适应误差为：

$$\begin{aligned} Ade_{\theta'}^{t+s} &= Ade_{\theta'}^{t+s-1} \times \beta_{t+s-1} \\ &= Ade_{\theta'}^{t+s-2} \times \beta_{t+s-2} \times \beta_{t+s-1} \\ &= Ade_{\theta'}^{t-1} \times \beta_{t-1} \times \cdots \times \beta_{t+s-2} \times \beta_{t+s-1} \end{aligned}$$

其中自适应误差一旦给定就是正实数定值，且对于 $\beta_i > 1 (i = t-1, \cdots, t+s-1)$，可以得到 $\beta_{t-1} \times \cdots \times \beta_{t+s-2} \times \beta_{t+s-1} > \beta_i > 1$ 成立，即权重序列 $\{\prod_{i=1}^{\infty} \beta_i\} (i = 1, 2, \cdots)$ 是一个递增序列，则 $Ade_{\theta'}^{t+s} > \varepsilon_{\theta'}$。故在第 $t+s+1$ 次迭代参数 θ'_{t+s+1} 重新获得更新机会，使得 $\theta'_{t+s+1} := \theta'^*_{t+s+1}$。所以对于所有满足自适应误差小于阈值的参数，第 t 次迭代结束后，在接下来的迭代中通过每次乘以权重值增加其对应的自适应误差值，直到第 $T(T>t, T=t+s)$ 次迭代其自适应误差大于阈值，该参数又获得了更新机会。

3.2.3 阈值确定

ALS 算法中有 3 类参数：$\omega_0 \in \mathbb{R}$，$w \in \mathbb{R}^n$、$V \in \mathbb{R}^{n \times k}$。其中 ω_0 用于全局控制，只自适应地选择 $w = (\omega_1, \cdots, \omega_n)$ 和 $V = (v_{1,1}, \cdots, v_{1,k}, v_{2,1}, \cdots, v_{2,k}, \cdots, v_{n,1}, \cdots, v_{n,k})$ 更新。具体的阈值确定方案有以下 3 种。

(1)直接给定阈值(Constant Threshold)

事先给定阈值，例如：$\varepsilon_w = 1e-6$，$\varepsilon_V = 1e-6$ 等。这种方法的缺点是阈值一旦给定就不能改变，每次需要人为进行调整，而且对于不同的数据集，阈值的选择不同，所以该方法移植性较差。

(2)分位数(Quartile Threshold)

通过限制每次迭代中不进行更新的参数数目，自适应地调整阈值。例如：每次迭代后，取 w 对应的自适应误差集合 $Ade_w = (Ade_{\omega_1}, \cdots, Ade_{\omega_n})$ 的 1 分位数、V 对应的自适应误差集合 $Ade_V = (Ade_{v_1}, \cdots, Ade_{v_{n \times k}})$ 的 2 分位数等。这种方法的阈值动态变化，移植性较好。

(3)高斯随机(Gauss Random Threshold)

利用标准高斯分布函数随机产生一个(0,1)之间的小数，对该小数乘以同一类参数的个数 m 的积取整，得到该类参数自适应误差阈值所在的位置为 $[0, m-1]$ 之间的值。例如，若 w 中参数个数为 n，V 中参数个数为 $n \times k$，且标准高斯分布随机产生的数为 α，则 w 对应的自适应误差的阈值为 Ade_{ω_i}，V 对应的自适应误差的阈值为 Ade_{v_j}，其中，$i = \lfloor \alpha \times n \rfloor$，$j = \lfloor \alpha \times n \times k \rfloor$。

3.2.4 AU-ALS 算法

AU-ALS 算法通过引入衡量参数变化率(即参数的自适应误差)的指标，停滞有限次变化率大于阈值的参数的更新，而重点关注变化率大于阈值的参数，使得样本误差下降得更

快。同时在每次迭代中，通过减少需要更新的参数数目，降低模型计算复杂度，从而提高了整个模型的运行效率。改进更新策略之后的 ALS 算法的伪代码如算法 1 所示。

算法 1 自适应参数更新策略交替最小二乘法(AU-ALS)

输入：训练集 $D=\{x^{(i)},y^{(i)}\}_{i=1}^{m}$

输出：$\hat{y}(x)=\omega_0+\sum_{i=1}^{n}\omega_i x_i+\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\omega_{i,j}x_i x_j$

1. 初始化所有参数 $\theta\in\Theta$，阈值 $\varepsilon_w=\varepsilon_V=1e-6$
2. 根据初始化参数、式(3)和式(4)分别预计算：误差向量 e、缓存矩阵 q
3. 令 T 为迭代的最大次数，令 t=1
4. 当 t=1 时按照交替最小二乘法更新所有的参数 θ、误差向量 e、缓存矩阵 q；并且计算所有参数(除 ω_0 之外)的自适应误差：$\{Ade_\theta^1 \mid \theta\in\Theta\backslash\omega_0\}$
5. t=t+1
6. FOR t=2 TO T DO
7. 根据式(2)更新 ω_0
8. IF $Ade_\theta^{t-1}<\varepsilon_\theta$ THEN
9. 更新 $Ade_\theta^t:=Ade_\theta^{t-1}\times\beta_{t-1}$
10. CONTINUE
11. ELSE
12. 根据式(2)更新 $\theta\backslash\omega_0$，根据式(3)更新 e，根据式(4)更新 q
13. 更新 $Ade_\theta^t:=|(\theta^t-\theta^{t-1})/\theta^{t-1}|$
14. ENDIF
15. 选择 3.2.3 节中阈值确定方法中的一种分别更新 $\varepsilon_w,\varepsilon_V$
16. t=t+1
17. ENDIF

3.3 算法复杂度分析

AU-ALS 学习算法主要从时间上对原学习算法进行改进，是典型的用空间换取时间的策略。

3.3.1 时间复杂度比较

ALS 学习算法的计算复杂度是关于模型的参数个数 P 和特征矩阵 X 中非 0 元的个数 $N_Z(X)$ 的线性复杂度：$O(P\times N_Z(X))$，其中 $P=(1+n+n\times k)$，k 为选取的特征维度，n 为特征个数。AU-ALS 学习算法的计算复杂度为 $O(p\times N_Z(X))$，其中 p 为每次迭代中需要更新的参数数目。因为 AU-ALS 学习算法每次更新的参数只是 ALS 学习算法中参数的子集，所以 AU-ALS 中更新参数个数 p 远小于 ALS 学习算法的参数个数 P，即 $p\ll P$，因此 AU-ALS 算法的计算复杂度小于 ALS 算法的计算复杂度。例如，阈值采用分位数的方法，取参数向量 w 和参数矩阵 V 的中位数，则 AU-ALS 算法每次迭代需要更新的参数数目为 $p\approx[1+\frac{n}{2}+\frac{(n\times k)}{2}]$。

3.3.2 空间复杂度比较

AU-ALS 学习算法由于要存储所有参数的自适应误差值，因此比 ALS 学习算法多出了 $O(1+n+n\times k)$ 的空间开销。

4 实验分析

4.1 数据集

选取两个被广泛用于评分推荐系统的数据集(Yahoo Webscope 和 Movielens)评价自适应更新策略的交替最小二乘法。按训练集 80%、测试集 10%、验证集 10%的比例随机抽取样本，全部实验都是在相同的操作系统环境和 python 编程环境下进行的。具体的统计数据如表 1 和表 2 所列。

表 1 数据集统计

Dataset	# Users	# Items	# Observed Entries	Size(kB)
Yahoo	7642	106959	221367	31497
Movielens	7342	10681	1004399	65547

表 2 选取的预测变量和 $N_Z(X)$

Dataset	Predictor Variables	Complexity $N_Z(X)$
Yahoo	user, item, genres, directors, actors	4006106
Movielens	user, item, time, title, genres	6650745

4.2 实验一：阈值方法比较

本文介绍了 3 种确定阈值的方法，在 Yahoo 和 Movielens 数据集上这 3 种确定阈值方法的比较指标为平均每次迭代运行的时间。该实验中，当特征维度 $k=22$、迭代 100 次时，采用直接给定阈值(Constant Threshold)方法，取 $\varepsilon_w=1e-3$，$\varepsilon_V=1e-2$；采用分位数确定阈值(Quartile Threshold)方法取参数向量 w 的自适应误差的 1 分位数、参数矩阵 V 的自适应误差的 2 分位数，图 1 所示为 Yahoo 数据集上 w 和 V 两类参数在前 11 次迭代中自适应误差的箱形图。

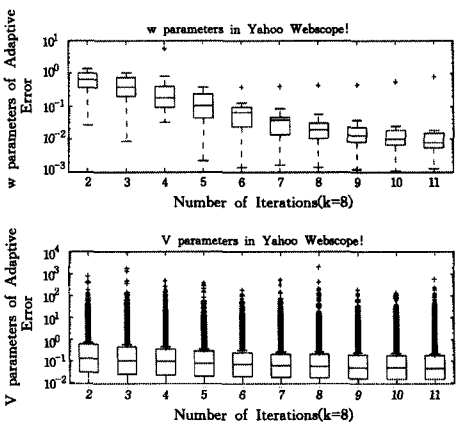


图 1 Yahoo 数据集上 w 和 V 两类参数在前 11 次迭代中的自适应误差

从图 1 可知，在前 11 次迭代中向量 w 对应的自适应误差的阈值在不断减小，然后趋于平缓，矩阵 V 对应的自适应误差的阈值减小得平缓；只有高斯随机(Gauss Random Threshold)确定阈值的方法不需要提前设置任何参数。图 2 所示为在这两种数据集上 3 种确定阈值方法平均每次迭代运行时间的比较。

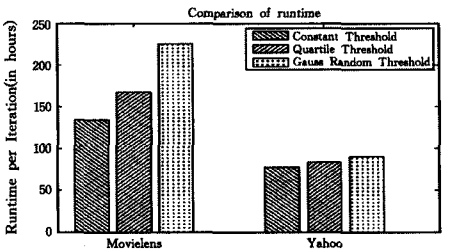


图 2 3 种确定阈值的方法下运行时间的比较

由图 2 可知，直接给定阈值的方法的运行时间最少，这是因为给定的阈值相对较大。这种阈值一旦确定就不改变的方法决定了选取不同的阈值对运行时间有较大影响，两个极端情况：如果选取的阈值很小，将导致所有参数的自适应误差都大于该阈值，所有参数需要更新，即 AU-ALS 算法就变成原

来的 ALS 算法；如果选取的阈值很大，导致所有参数的自适应误差都小于阈值，可能造成预测效果不佳。所以对于未知的数据集很难直接确定合适的阈值，这样就需要大量重复的调参工作。对于高斯随机方法，如果迭代的次数较少，那么运行时间的长短很具偶然性，也可能会影响预测结果。对于分位数确定阈值的方法，每次迭代的运行时间较稳定，能自适应地控制阈值，但也需要提前设置选取的是第几分位数作为阈值。这 3 种方法各有优缺点，虽然本文提供了 3 种阈值确定的方法，但在实际的阈值选取中应该根据具体情况而定。

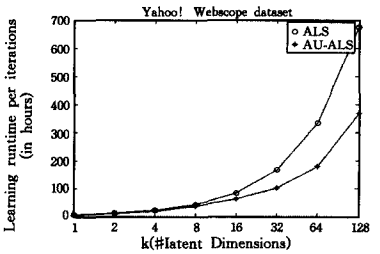
4.3 实验二：运行时间比较

该实验为特征维度 k 取 1，2，4，8，16，32，64，128，阈值确定采用分位数方法（Quartile Threshold）时，迭代 20 次 ALS 和 AU-ALS 分别作用于 Yahoo 和 Movielens 数据集上，平均每次迭代的运行时间（单位：h）的比较。具体实验数据如表 3 所列。

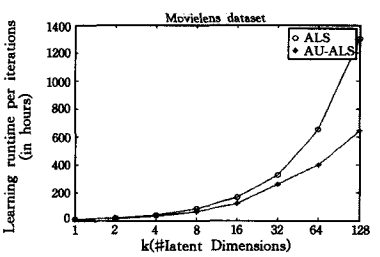
表 3 运行时间比较（h）

Dataset	Method	k=1	k=2	k=4	k=8	k=16	k=32	k=64	k=128
Yahoo	ALS	5.04	10.40	21.12	42.85	83.13	167.08	335.26	675.53
	AU-ALS	**4.34**	**8.68**	**17.03**	**34.90**	**61.56**	**101.21**	**178.54**	**390.09**
Movielens	ALS	10.08	19.49	40.77	82.54	164.1	326.16	651.36	1303.8
	AU-ALS	**8.75**	**14.06**	**30.98**	**62.52**	**120.3**	**240.39**	**399.07**	**643.31**

由图 3 可以得到，随着 k 值的增大和特征数目的增多，在相同的情况下 AU-ALS 的运行时间明显少于 ALS 的运行时间。



(a) Yahoo 数据集上运行时间的比较



(b) Movielens 数据集上运行时间的比较

图 3

4.4 实验三：预测质量比较

该实验的评价指标为均方根误差（RMSE），其表达式为：

$$RMSE=\sqrt{\frac{\sum_i(\hat{y}^{(i)}-y^{(i)})^2}{N}}$$

均方根误差是推荐系统中评分预测问题最常见的反映预测精度的评价指标，测试集的均方根误差越小，则预测质量越好。

该实验中取 $k=22$、阈值确定采用分位数方法（Quartile Threshold），分别在 Yahoo 和 Movielens 两种数据集上对测试集均方根误差进行比较，实验结果如表 4 所列。

表 4 均方根误差的比较

Dataset	Method	RMSE	Runtime(in hours)
Yahoo	ALS	**0.9845**	114.63
	AU-ALS	0.9891	**84.06**
Movielens	ALS	0.8867	209.08
	AU-ALS	**0.8734**	**167.52**

由表 4 可以看出，在 ALS 和 AU-ALS 算法的预测精度基本相同的情况下，AU-ALS 算法的运行时间较少。

综上所述，在不同的两个数据集上 AU-ALS 算法相对 ALS 的运行时间有明显减少，测试集均方根误差也有一定的减小。

结束语 本文主要改进了固定其他参数只求单一参数的最优值的学习算法的更新策略。通过引入参数的自适应误差指标，只更新自适应误差大于设定的阈值的参数。该方法不仅防止了自适应误差比阈值小的参数在其最优值左右震荡，而且使自适应误差大的参数获得更多的更新机会。同时每次迭代只更新原来参数的子集，减少了运行时间。由于改进前的 ALS 算法在减小误差上已经取得了较好的成果，而本文主要致力于提高算法的运行效率，其在算法误差减小上的改进幅度不大，这也是作者后续研究的重点。

参 考 文 献

[1] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms[C]// Proc International Conference on the World Wide Web. ACM, 2001: 285-295

[2] Breese J, S D H, Kadie C. Empirical Analysis of Predictive Algorithms for Collaborative Filtering[C]// Proc Conference on Uncertainty in Artificial Intelligence. 1998: 43-52

[3] Paterek A, Paterek A. Improving regularized singular value decomposition for collaborative filtering[C]// Proceedings of Kdd Cup & Workshop. 2007: 58

[4] Srebro N, Rennie J D M, Jaakola T S. Maximum-Margin Matrix Factorization[J]. Advances in Neural Information Processing Systems, 2004, 37(2): 1329-1336

[5] Koren Y. Collaborative filtering with temporal dynamics [J]. Communications of the ACM, 2010, 53(4): 89-97

[6] Chih-Wei H, Chang C C, Lin C J. A practical guide to support vector classification[D]. National Taiwan University, 2010

[7] Klema V, Laub A J. The singular value decomposition: Its computation and some applications[J]. IEEE Transactions on Automatic Control, 1980, 25(2): 164-176

[8] Karatzoglou A, Amatriain X, Oliver N, et al. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering[C]// Proceedings of the Fourth ACM Conference on Recommender Systems. 2010: 79-86

[9] Steffen R. Factorization machines[C]// Proceedings of the 10th IEEE International Conference on Data Mining. IEEE Computer Society, 2010: 995-1000

[10] Cheng C, Xia F, Zhang T, et al. Gradient boosting factorization machines[C]// Proceedings of the 8th ACM Conference on Re-

commender Systems. ACM, 2014: 265-272

[11] Friedman J H. Greedy Function Approximation: A Gradient Boosting Machine[J]. Annals of Statistics, 2000, 29(5): 1189-1232

[12] Riccardi A, Fernandez-Navarro F, Carloni S. Cost-sensitive AdaBoost algorithm for ordinal regression based on extreme learning machine[J]. Cybernetics IEEE Transactions on, 2014, 44(10): 1898-1909

[13] Yu L, Liu H. Feature selection for high-dimensional data: a fast correlation-based filter solution[C]// Proceedings of International Conferences on Machine Learning. 2003: 856-863

[14] H P, F L, C D. Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy [J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2005, 27(8): 1226-1238

[15] Steffen R. Learning recommender systems with adaptive regularization[C]// Fifth ACM International Conference on Web Search & Data Mining. 2012: 133-142

[16] Steffen R, Zeno G, Christoph F, et al. Fast context-aware recommendations with factorization machines[C]// Proceedings of the 34th ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2011: 635-644

[17] Steffen R. Scaling Factorization Machines to Relational Data[C]// Proceedings of the 39th International Conference on Very Large Data Bases (VLDB 2013). Trento, Italy, 2013: 337-348

[18] Christoph F, Lars S T, Steffen R. Bayesian Factorization Machines[C]// Workshop on Sparse Representation and Low-rank Approximation, Neural Information Processing Systems (NIPS-WS). Granada, Spain, 2011

[19] Cover T, Hart P. Nearest neighbor pattern classification [J]. IEEE Transactions on Information Theory, 1967, 13(1): 21-27

[20] Steffen R. Factorization Machines with libFM[J]. ACM Transactions on Intelligent Systems and Technology, 2012, 3(3): 451-458

(上接第 56 页)

确预测学术关系。以后的研究工作将重点考虑社交关系方向问题，将构建方法应用到有向网络中。

参 考 文 献

[1] Zhang H P, Zhang R Q, Zhao Y P, et al. Big data modeling and analysis of microblog ecosystem[J]. International Journal of Automation and Computing, 2014, 11(2): 119-127

[2] Wang Y, Gao L. Social circle-based algorithm for friend recommendation in online social networks[J]. Chinese Journal of Computer, 2014(4): 801-808(in Chinese)
王珏，高琳. 基于社交圈的在线社交网络朋友推荐算法[J]. 计算机学报，2014(4): 801-808

[3] Lin Y F, Wang T Y, Tang R, et al. An effective model and algorithm for community detection in social networks[J]. Journal of Computer Research and Development, 2012, 49(2): 337-345(in Chinese)
林友芳，王天宇，唐锐，等. 一种有效的社会网络社区发现模型和算法[J]. 计算机研究与发展，2012，49(2)：337-345

[4] McAuley J, Leskovec J. Discovering social circles in ego networks[J]. ACM Transactions on Knowledge Discovery from Data (TKDD), 2014, 8(1): 73-100

[5] Wang J, Lochovsky F H. Data extraction and label assignment for web databases[C]// Proceedings of the 12th International Conference on World Wide Web. New York: ACM Press, 2003: 187-196

[6] Huang A N. Similarity measures for text document clustering [C]// Proceedings of the Sixth New Zealand Computer Science Research Student Conference (NZCSRSC2008). Christchurch, 2008: 49-56

[7] Ahlgren P, Jarneving B, Rousseau R. Requirements for a cocitation similarity measure, with special reference to Pearson's correlation coefficient[J]. Journal of the American Society for Information Science and Technology, 2003, 54(6): 550-560

[8] Deng A, Zhu Y, Shi B L. A collaborative filtering recommendation algorithm based on item rating prediction[J]. Journal of Software, 2003, 14(9): 1621-1628

[9] Leskovec J, Mcauley J J. Learning to discover social circles in ego networks[J]. Advances in Neural Information Processing Systems, 2012: 539-547

[10] Robert W P L. Chinese string searching using the KMP algorithm[C]// Proceedings of the 16th Conference on Computational Linguistics. Stroudsburg: Association for Computational Linguistics, 1996(2): 1111-1114

[11] White T. Hadoop: The definitive guide[M]. O'Reilly, 2009

[12] Lin W Q, Lu F S, Ding Z Y, et al. Parallel computing hierachical community approach based on weighted-graph[J]. Journal of Software, 2012, 23(6): 1517-1530(in Chinese)
林旺群，卢凤顺，丁兆云，等. 基于加权图的层次化社区并行计算方法[J]. 软件学报，2012，23(6)：1517-1530

[13] He L, Wu L D, Cai Yi-zhao. Survey of clustering algorithm in data mining[J]. Application Research of Computer, 2007, 24(1): 10-13(in Chinese)
贺玲，吴玲达，蔡益朝. 数据挖掘中的聚类算法综述[J]. 计算机应用研究，2007，24(1)：10-13

[14] Suo H G, Wang Y W. An improved k-means algorithm for document clustering[J]. Journal of Shandong University (Natural Science), 2008, 43(1): 60-64(in Chinese)
索红光，王玉伟. 一种用于文本聚类的改进 k-means 算法[J]. 山东大学学报(理学版)，2008，43(1)：60-64

[15] Mahdavi M, Abolhassani H. Harmony K-means algorithm for document clustering[J]. Data Mining and Knowledge Discovery, 2009, 18(3): 370-391

[16] Gupta H, Srivastava R. K-means based document clustering with automatic "k" selection and cluster refinement[J]. International Journal of Computer Science and Mobile Applications, 2014, 2(5): 7-13

[17] Mavroeidis D, Marchiori E. Feature selection for k-means clustering stability: theoretical analysis and an algorithm[J]. Data Mining and Knowledge Discovery, 2014, 28(4): 918-960

[18] Gaudani H, Lakhani K, Chhatrala R. Survey of document clustering[J]. International Journal of Computer Science and Mobile Computing, 2014, 3(5): 871-874

[19] Xie J Y, Wang Y E. K-means algorithm based on minimum deviation initialized clustering centers[J]. Computer Engineering, 2014, 40(8): 205-211, 223(in Chinese)
谢娟英，王艳娥. 最小方差优化初始聚类中心的 K-means 算法[J]. 计算机工程，2014，40(8)：205-211，223

[20] Bennett K, Robertson J, Milton P M, et al. MATLABapplications for the practical engineer[M]. InTech, 2014