计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.190900144

基于 Huber 损失的非负矩阵分解算法

王丽星[1]　曹付元[1,2]
1 山西大学计算机与信息技术学院　太原 030006
2 山西大学计算智能与中文信息处理教育部重点实验室　太原 030006
(1608689768@qq.com)

摘　要　非负矩阵分解(Nonnegative Matrix Factorization)算法能为原始数据找到非负的、线性的矩阵表示且保留了数据的本质特征,已被成功应用于多个领域。经典的 NMF 算法及其变体算法大部分使用均方误差函数来度量重建误差,在许多任务中已经显示出其有效性,但它在处理含有噪声的数据时仍然面临一些困难。Huber 损失函数对较小的残差执行的惩罚与均方误差损失函数相同,对较大的残差执行的惩罚是线性增长的,因此与均方误差损失函数相比,Huber 损失函数具有更强的鲁棒性;已有研究证明$L_{2,1}$范数稀疏正则项在机器学习的分类和聚类模型中具有特征选择作用。结合两者的优点,文中提出了一种基于 Huber 损失函数且融入$L_{2,1}$范数正则项的非负矩阵分解聚类模型,并给出了基于投影梯度更新规则的优化过程。在多组数据集上将所提算法与经典的多种聚类算法进行对比,实验结果验证了所提算法的有效性。

关键词:非负矩阵分解;Huber 损失函数;$L_{2,1}$范数;投影梯度法
中图法分类号　TP3-05

Huber Loss Based Nonnegative Matrix Factorization Algorithm

WANG Li-xing[1] and CAO Fu-yuan[1,2]
1 School of Computer and Information Technology, Shanxi University, Taiyuan 030006, China
2 Key Laboratory of Computational Intelligence and Chinese Information Processing (Shanxi University), Ministry of Education, Taiyuan 030006, China

Abstract　Non-negative matrix factorization (NMF) algorithm can find a non-negative and linear matrix representation and retains the essential characteristics of the original data, it has been successfully applied to many fields. The classical NMF algorithm and its variant algorithms mostly use the mean square error function to measure the reconstruction error, which has been shown to be effective in many tasks, but it still faces some difficulties in dealing with noise-containing data. The Huber loss function performs the same penalty for the smaller residual as the mean square error loss function, and the penalty for the larger residual is linearly grown, so the Huber loss function is more robust than the mean square error loss function. It has been proved that the $L_{2,1}$ norm sparse regularization term is a feature selection function in the classification and clustering model of machine learning. Therefore, combining the advantages of the two, a non-negative matrix factorization clustering model based on Huber loss function and incorporating $L_{2,1}$ norm regularization term is proposed, and an effective optimization procedure based on projected gradient method to update variables is given. Compared with the classical NMF multi-clustering algorithm on multiple sets of datasets, the experimental results show the effectiveness of the proposed algorithm.

Keywords　Nonnegative matrix factorization, Huber loss function, $L_{2,1}$ norm, Projected gradient method

1　引言

在大数据时代,如何有效地从海量数据中检索、分类和提取有价值的知识已经成为机器学习的重要任务。聚类作为一种无监督的学习机制,通过特定的相似性度量对未标记数据对象进行分组,使得同一组中的数据点尽可能相似,不同组中的数据点尽可能不相似[1]。目前流行的聚类方法有基于划分的聚类算法、基于密度的聚类算法、基于网格的聚类算法、基于层次聚类的算法、基于非负矩阵分解(NMF)的聚类算法等。

到稿日期:2019-09-23　返修日期:2020-02-08　本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家自然科学基金(61573229,61976128);山西省重点研发计划项目(201803D31022);山西省留学基金项目(2016-003);山西省留学基金择优资助项目(2016-001)
This work was supported by the National Natural Science Foundation (61573229, 61976128), Shanxi Provincial Key Research and Development Program (201803D31022), Shanxi Scholarship Fund Project (2016-003) and Shanxi Scholarship Fund Selection Project (2016-001).
通信作者:曹付元(cfy@sxu.edu.cn)

NMF 最初由 Lee 等[2-3]引入机器学习和模式识别领域中并被广泛应用于聚类算法。NMF 已被证明与多种传统聚类算法具有等价性，并且在通常情况下，NMF 的聚类效果比与它等价的传统聚类方法的效果好。这是因为 NMF 能够为原始数据找到非负的、低秩的、基于部件的表达。这种表达以矩阵的形式呈现，蕴含数据点相似性的信息，使用 k-means 或者谱聚类方法可以量化数据点间的相似性并划分类别[4]。

给定一组非负的数据 $\boldsymbol{X}\in\mathbb{R}^{d\times n}$ 和 $r<\min(d,n)$。NMF 将 $\boldsymbol{X}$ 分解为两个非负的矩阵 $\boldsymbol{W}\in\mathbb{R}^{d\times r}$ 和 $H\in\mathbb{R}^{r\times n}$，使得 $\boldsymbol{X}\approx\boldsymbol{WH}$。$\boldsymbol{W}$ 可以理解为降维后的特征与原始数据的特征之间的一个权值矩阵，$\boldsymbol{H}$ 可以理解为降维后的数据。NMF 的求解是 NP-hard 问题，但是可以转化为交替非负最小二乘问题的求解。标准的 NMF 算法使用如下目标函数：

$$\min_{\boldsymbol{W},\boldsymbol{H}}\frac{1}{2}\|\boldsymbol{X}-\boldsymbol{WH}\|_F^2 \quad \text{s.t. } \boldsymbol{W}\geqslant 0,\boldsymbol{H}\geqslant 0 \tag{1}$$

求解式(1)的常见方法有两种：1)由 Lee 等[3]提出的乘性更新法，通过不断迭代优化求解 $\boldsymbol{W}$ 和 $\boldsymbol{H}$，具有简单易于实施的特点；2)Lin[5]将投影梯度法应用于 NMF 的求解，该方法已被证明比乘性更新法收敛速度更快，投影梯度法常用于求解有界约束的优化问题。

目前已有一些根据不同的需求提出的 NMF 的算法，进一步增强了 NMF 的表达能力，可以有效提高聚类的有效性。Hoyer[6]通过引入稀疏约束来控制系数矩阵的稀疏程度，提出了稀疏非负矩阵分解算法(Non-negative Matrix Factorization with Sparseness Constraints，SNMF)，使得表达系数具有了可解释性。Cai 等[7]通过考虑原始数据中的几何结构，并在分解过程中保留数据集的局部几何信息，提出了图正则化非负矩阵分解算法(Graph Regularized Non-negative Matrix Factorization，GNMF)。Jiang 等[8]提出了稀疏约束图正则非负矩阵分解算法，该算法不仅考虑了数据的几何信息，而且对系数矩阵进行了稀疏约束，并将它们整合于单个目标函数中。Liu 等[9]引入了有约束的非负矩阵分解算法(Constrained Nonnegative Matrix Factorization，CNMF)，该方法将一些先验标签信息作为附加约束，并将其集成到 NMF 模型中以提高辨别能力。

虽然 NMF 在实际中已经有许多成功的应用，但是在有噪声的数据集上仍然面临一些困难，这是因为传统的 NMF 算法都使用均方误差度量矩阵分解的重建误差。已有研究证明，均方误差对于零均值高斯噪声是最佳的[10]，但是现实问题几乎没有符合模型假设的数据。均方误差损失函数为了减小目标函数值，强行拟合离群点，这样会降低对原始数据表达的准确性。为了解决传统 NMF 方法不能处理噪声数据的问题，一些方法将额外的信息融合到 NMF 框架。Du 等[11]提出了一种基于熵诱导度量的 NMF 算法(Non-negative Matrix Factorization Method Based on the Correntropy Induced Metric，CIM-NMF)，该方法假设噪声服从非高斯分布，CIM-NMF 在后续分类或聚类时性能显著优于 NMF。Yang 等[12]对数据噪声使用 Lasso 正则化和拉普拉斯正则化，并将这两种正则化结合到 NMF 的系数矩阵中，提出了联合“稀疏”和图正则化的鲁棒 NMF 算法(Robust Non-negative Matrix Factorization via Joint Sparse and Graph Regularization，RSGNMF)。Kong 等[13]提出了利用 $L_{2,1}$ 范数来惩罚重建损失，同时从数据中去除了异常值的 $L_{2,1}$-NMF 算法。

不同于以上这类融合信息的方法，本文从惩罚函数对噪声数据筛选能力的角度考虑，提出了一种基于 Huber 损失的 NMF 算法。该方法根据矩阵分解的残差实施不同的惩罚，以此排除离群点对聚类效果的影响；之后对分解得到的低维数据 $\boldsymbol{H}$ 进行谱聚类；由于 $L_{2,1}$ 范数正则化已被证明具有特征选择作用[14]，因此模型中引进了 $L_{2,1}$ 范数对系数矩阵进行稀疏约束，以便加快学习过程，提高模型的泛化能力，并减轻维度灾难的影响。在合成数据集和真实数据集上的大量实验验证了本文模型的有效性。

2 投影梯度法

NMF 的求解是一个有界约束优化问题，下面简要介绍用投影梯度法求解有界约束优化问题的基本思路。有界约束优化问题的标准形式为：

$$\min_{x\in\mathbb{R}^n} f(\boldsymbol{x}) \quad \text{s.t. } l_i\leqslant x_i\leqslant u_i,i=1,\cdots,n \tag{2}$$

其中，$f:\mathbb{R}^n\to\mathbb{R}$ 是一个连续可导的函数，并且 l_i 和 u_i 分别是变量 $\boldsymbol{x}_i$ 的下界和上界，k 表示迭代的步数，$\boldsymbol{x}^k$ 表示第 k 次迭代的点。根据投影梯度法用式(3)将 $\boldsymbol{x}^k$ 更新到 $\boldsymbol{x}^{k+1}$：

$$\boldsymbol{x}^{k+1}=\Omega(\boldsymbol{x}^k-\alpha^k\nabla f(\boldsymbol{x}^k)) \tag{3}$$

其中，$\Omega:\mathbb{R}\to\mathbb{R}$ 是投影函数，将数据点投影到可行集内，定义为：

$$\Omega(x_i)=\begin{cases}x_i, & \text{当} l_i\leqslant x_i\leqslant u_i\\ u_i, & \text{当} x_i\geqslant u_i\\ l_i, & \text{当} x_i\leqslant l_i\end{cases} \tag{4}$$

其中，α^k 表示在求解式(3)时第 k 次迭代的更新步长。目前已有大量研究成果用于寻找合适的步长，常用的方法是使用 Armijo 规则寻找合适的步长 α^k[5]，通常要求 α^k 能使 $f(\boldsymbol{x}^k)$ 充分下降，即使 $f(\boldsymbol{x}^k)$ 满足以下的充分下降条件：

$$f(\boldsymbol{x}^{k+1})-f(\boldsymbol{x}^k)\leqslant\sigma\nabla f(\boldsymbol{x}^k)^{\mathrm{T}}(\boldsymbol{x}^{k+1}-\boldsymbol{x}^k) \tag{5}$$

其中，σ 控制目标函数下降的速率，数值范围为 0～1，具体实施中通常取为 0.01[4]；$\nabla f(\boldsymbol{x}^k)$ 表示 $f(\boldsymbol{x})$ 在 $\boldsymbol{x}^k$ 点的梯度。投影梯度法求解有界约束优化问题的一般步骤为：1)选取迭代的初始点；2)将负梯度作为搜索方向；3)沿搜索方向寻求适当的步长使目标函数值下降；4)将当前的迭代点投影到可行域内得到下一个迭代点。投影梯度法的收敛性已被 Calamai 等[15]证明。

3 一种正则化的 Huber 损失 NMF 算法

本节介绍了所提模型和基于投影梯度更新规则的优化过程。

3.1 本文模型

Huber 提出并证明了 Huber 罚函数对噪声的鲁棒性[16]。为了改进 NMF 算法，使其对有噪声的数据集进行聚类时更加鲁棒，下面给出了 Huber 损失函数作为罚函数的模型：

$$\min_{\boldsymbol{W},\boldsymbol{H}}\sum_{i,j}\psi(\boldsymbol{X}_{ij}-(\boldsymbol{WH})_{ij}) \quad \text{s.t. } \boldsymbol{W}\geqslant 0,\boldsymbol{H}\geqslant 0 \tag{6}$$

其中，$\psi:\mathbb{R}\to\mathbb{R}$ 是 Huber 函数，定义为：

$$\psi_\delta(u)=\begin{cases}u^2, & |u|\leqslant\delta\\ \delta\cdot(2|u|-\delta), & |u|>\delta\end{cases} \tag{7}$$

图 1 显示了参数 $\delta=1$ 时，Huber 损失函数、均方误差损失函数以及绝对值函数的取值。图 1 中，实线为 Huber 罚函数，○线为均方误差函数，×线为绝对值函数。

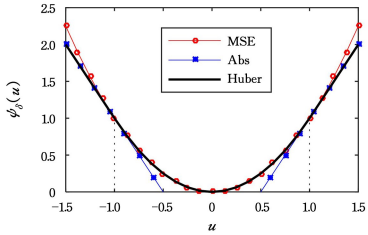


图 1 3 种损失函数的对比

Fig. 1 Comparison of three loss functions

从图 1 可以看出，对于任意小于 δ 的残差，Huber 损失函数的罚函数与均方误差损失函数相吻合，但是对于大于 δ 的残差，本文中的算法采用绝对值函数处理噪声，比均方误差损失函数的惩罚力度小，一定程度上避免了过度拟合。因此，使用 Huber 罚函数的 NMF 算法相比使用均方误差损失函数的标准 NMF 算法对异常值不敏感。

Hoyer[6]将“稀疏表示”的概念引进 NMF，使得模型利用尽可能少的基向量组 $\boldsymbol{W}$ 的线性组合表示数据，且一定能为原始数据找到基于部件的表示，这种“稀疏性”显著提高了 NMF 的表达能力。因此，本文借鉴“稀疏表示”的优点，在提出的模型中使用了正则项对表示矩阵 $\boldsymbol{H}$ 进行稀疏约束。不同算法对稀疏约束的设计有不同的选择，Nie 等[14]已经证明，使用 $L_{2,1}$ 范数的正则化与 L_0 范数的正则化在实际应用中效果近似，但是 $L_{2,1}$ 范数的正则化作为凸优化更容易实施。因此，本文在提出的模型中选择了 $L_{2,1}$ 范数对表示矩阵 $\boldsymbol{H}$ 进行稀疏约束，引进稀疏约束的 HuberNMF 模型定义为：

$$\begin{aligned}&\min_{\boldsymbol{W},\boldsymbol{H}}\sum_{i,j}\psi(\boldsymbol{X}_{ij}-(\boldsymbol{WH})_{ij})+\lambda\|\boldsymbol{H}\|_{2,1}\\&\text{s.t. }\boldsymbol{W}\geqslant 0,\boldsymbol{H}\geqslant 0\end{aligned} \tag{8}$$

其中，$\|\cdot\|_{2,1}$ 表示矩阵的 $L_{2,1}$ 范数；$\lambda>0$ 是一个平衡参数，控制表达矩阵 $\boldsymbol{H}$ 的稀疏程度。

3.2 基于投影梯度更新规则的优化过程

本节使用投影梯度法对式(8)进行优化求解。NMF 求解不是一个凸优化问题，只能保证收敛到局部最优解，下面使用块坐标下降法将 $\boldsymbol{W}$ 和 $\boldsymbol{H}$ 分别作为一个变量进行交替优化。

3.2.1 计算 $\boldsymbol{W}$

固定变量 $\boldsymbol{H}$，优化 $\boldsymbol{W}$ 的子问题目标函数为：

$$\begin{aligned}&\min_{\boldsymbol{W}}\sum_{i,j}\psi(\boldsymbol{X}_{ij}-(\boldsymbol{WH})_{ij})\\&\text{s.t. }\boldsymbol{W}\geqslant 0\end{aligned} \tag{9}$$

这是一个有界约束的凸优化问题。令 $f_1=\sum_{i,j}\psi(\boldsymbol{X}_{ij}-(\boldsymbol{WH})_{ij})$，$\boldsymbol{Y}=\boldsymbol{X}-\boldsymbol{WH}$。由 f_1 对 $\boldsymbol{W}$ 求导，得到：

$$\frac{df_1}{d\boldsymbol{W}}=-\boldsymbol{QH}^{\mathrm{T}} \tag{10}$$

其中，辅助变量 $\boldsymbol{Q}=(q_{ij})_{d\times n}$，

$$q_{ij}=\begin{cases}-2\delta, & Y_{ij}<-\delta\\ 2Y_{ij}, & -\delta\leqslant Y_{ij}<\delta\\ -2\delta, & Y_{ij}>\delta\end{cases} \tag{11}$$

得到 $\boldsymbol{W}$ 的更新规则为：

$$\boldsymbol{W}^{k+1}=\rho(\boldsymbol{W}^k-\alpha^k\cdot\frac{d\boldsymbol{f}_1}{d\boldsymbol{W}}) \tag{12}$$

其中，$\rho:\mathbb{R}\to\mathbb{R}$ 是投影函数，将数据点投影到可行集内，定义为：

$$\rho(v)=\begin{cases}v, & v\geqslant 0\\ 0, & v<0\end{cases} \tag{13}$$

3.2.2 计算 $\boldsymbol{H}$

$\boldsymbol{H}$ 的推导过程与 $\boldsymbol{W}$ 的推导过程类似。固定变量 $\boldsymbol{W}$，优化 $\boldsymbol{H}$ 的子问题目标函数为：

$$\begin{aligned}&\min_{\boldsymbol{H}}\sum_{i,j}\psi(\boldsymbol{X}_{ij}-(\boldsymbol{WH})_{ij})+\lambda\|\boldsymbol{H}\|_{2,1}\\&\text{s.t. }\boldsymbol{H}\geqslant 0\end{aligned} \tag{14}$$

令 $f_2=\sum_{i,j}\psi(\boldsymbol{X}_{ij}-(\boldsymbol{WH})_{ij})+\lambda\|\boldsymbol{H}\|_{2,1}$，$\boldsymbol{V}=\boldsymbol{X}-\boldsymbol{WH}$。由 f_2 对 $\boldsymbol{H}$ 按列求导，得到：

$$\frac{d\boldsymbol{f}_2}{d\boldsymbol{h}_j}=-(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{Q}')_{\cdot j}+\boldsymbol{\lambda}\frac{\boldsymbol{h}_j}{\|\boldsymbol{h}_j\|} \tag{15}$$

其中，$\boldsymbol{h}_j$ 表示矩阵 $\boldsymbol{H}$ 的列向量，$-(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{Q}')_{\cdot j}$ 表示矩阵的第 j 个列向量。辅助变量 $\boldsymbol{Q}'=(q'_{ij})_{d\times n}$，其中：

$$q'_{ij}=\begin{cases}-2\delta, & V_{ij}<-\delta\\ 2V_{ij}, & -\delta\leqslant V_{ij}<\delta\\ -2\delta, & V_{ij}>\delta\end{cases} \tag{16}$$

用 $\frac{df_2}{d\boldsymbol{H}}$ 表示 f_2 对 $\boldsymbol{H}$ 求导的结果，得到 $\boldsymbol{H}$ 的更新规则为：

$$\boldsymbol{H}^{k+1}=\rho(\boldsymbol{H}^k-\alpha^k\cdot\frac{df_2}{d\boldsymbol{H}}) \tag{17}$$

3.2.3 迭代终止条件

在寻找最优解的过程中，需要设定迭代终止条件以便找到优化问题的稳定点。求解 NMF 的一些乘性更新方法使用的充分条件为：相继两次迭代的绝对值误差小于一个预先定义的数时，停止迭代。然而这样的停止条件不能保证迭代到达稳定点。对于有界约束优化问题，检查迭代是否到达稳定点有一个共同的条件，对于提出的 HuberNMF 模型，这个条件的具体表达示为：

$$\|\nabla^{\Omega}L(\boldsymbol{W}^k,\boldsymbol{H}^k)\|_F\leqslant\varepsilon\|L(\boldsymbol{W}^1,\boldsymbol{H}^1)\|_F \tag{18}$$

其中，$L(\boldsymbol{W},\boldsymbol{H})=\sum_{i,j}\psi((\boldsymbol{X}-\boldsymbol{WH})_{i,j})$；$\varepsilon$ 表示对于梯度模长变化的容忍度，通常设置为 10^{-6}[5]。交替优化过程中，设置两个子问题的停止条件分别为：

$$\|\nabla^{\Omega}_{\boldsymbol{W}}L(\boldsymbol{W}^{k+1},\boldsymbol{H}^k)\|_F\leqslant\varepsilon_{\boldsymbol{W}} \tag{19}$$

$$\|\nabla^{\Omega}_{\boldsymbol{H}}L(\boldsymbol{W}^{k+1},\boldsymbol{H}^{k+1})\|_F\leqslant\varepsilon_{\boldsymbol{H}} \tag{20}$$

其中，$\nabla^{\Omega}L(\cdot)$ 表示 L 函数的投影梯度。具体地，设置 $\varepsilon_{\boldsymbol{W}}$ 和 $\varepsilon_{\boldsymbol{H}}$ 为：

$$\varepsilon_{\boldsymbol{W}}=\varepsilon_{\boldsymbol{H}}\equiv\max(10^{-3},\varepsilon)\|\nabla f(\boldsymbol{W}^1,\boldsymbol{H}^1)\|_F \tag{21}$$

算法 1 给出了 HuberNMF 算法的步骤。

算法 1 HuberNMF 的投影梯度

Input: Data $\mathbf{X}$, parameters λ,δ,r, number of clusters c

Output: $\mathbf{W},\mathbf{H}$, Segmentation matrix

1. Initialize $\mathbf{W}^1,\mathbf{H}^1$, $k=0$

Repeat

Fixed $\mathbf{H}^k$, Use(12) to update $\mathbf{W}^{k+1}$

Until

$\mathbf{W}^{k+1}$ satisfy (19)

Repeat

Fixed $\mathbf{W}^k$, Use(17) to update $\mathbf{H}^{k+1}$

Until

$\mathbf{H}^{k+1}$ satisfy (20)

Until

$\| \nabla^{\Omega} L(\mathbf{W}^k, \mathbf{H}^k) \|_F$ satisfy (18)

2. Implement Ncut($\mathbf{H}$) to get segmentation matrix

3.3 时间复杂度分析

因为求解式(8)时交替优化两个变量的求解步骤相似，所以先单独分析求解 $\boldsymbol{W}$ 子问题的时间复杂度，再类似地分析求解 $\boldsymbol{H}$ 子问题的时间复杂度，最后得出总的时间复杂度。

用式(1)求解式(9)时，主要运算分为两步：1)检查α^k是否满足 Armijo 规则的充分下降条件；2)根据式(12)进行更新。充分下降条件为：

$$f_1(\tilde{\boldsymbol{W}}) - f_1(\bar{\boldsymbol{W}}) \leqslant \sigma \cdot \langle \frac{d f_1}{d \boldsymbol{W}}, (\tilde{\boldsymbol{W}} - \bar{\boldsymbol{W}}) \rangle \tag{22}$$

其中，$\tilde{\boldsymbol{W}}$和$\bar{\boldsymbol{W}}$表示相继两次迭代的 $\boldsymbol{W}$ 的变量。$\langle \cdot, \cdot \rangle$表示求两个矩阵的内积。式(22)中的主要运算为 $f_1(\tilde{\boldsymbol{W}}) - f_1(\bar{\boldsymbol{W}})$，该运算的时间复杂度为 $O(dnr)$。更新$\boldsymbol{W}^k$的主要运算为：

$$\alpha^k \cdot \frac{d f_1}{d \boldsymbol{W}} = -\alpha^k \cdot \boldsymbol{Q}\boldsymbol{H}^{\mathrm{T}}$$

其时间复杂度为 O(dnr)。假设求解式(9)的平均循环次数为t_1，则时间复杂为 O($t_1 dnr$)。求解式(14)时，充分下降条件为：

$$f_2(\tilde{\boldsymbol{H}}) - f_2(\bar{\boldsymbol{H}}) \leqslant \sigma \cdot \langle \frac{d f_2}{d H}, (\tilde{\boldsymbol{H}} - \bar{\boldsymbol{H}}) \rangle \tag{23}$$

式(23)的主要运算为 $f_2(\tilde{\boldsymbol{H}}) - f_2(\bar{\boldsymbol{H}})$，时间复杂度为 O($dnr$)。更新$\boldsymbol{H}^k$的主要运算为$\alpha^k \cdot \frac{d f_2}{d H}$，时间复杂度为 O($dnr$)。假设求解式(14)的平均循环次数为$t_2$，则时间复杂度为 O($t_2 dnr$)。假设算法的大循环平均循环次数为 t，那么求解式(8)的总的时间复杂度为 O($t(t_1 + t_2) dnr$)。

4 实验

为了验证提出的 HuberNMF 算法对聚类结果的有效性，在合成数据集和真实数据集上进行了大量的实验，并将其与当前流行的 4 种聚类方法进行了对比。

4.1 评价指标

实验中使用了准确率(Accuracy，ACC)以及标准互信息(Normalized Mutual Information，NMI)[17]作为聚类效果的度量指标，准确率和标准互信息越高，聚类效果就越好。

4.2 对比算法

为了验证模型的有效性，将当前流行的 4 种聚类算法作为基线算法进行对比。

(1)标准的 NMF 算法：标准的 NMF 旨在为原始数据找到低维的、非负的线性表示，根据线性表示矩阵中数据的相似性进行聚类。

(2)标准的谱聚类 Ncut：将数据点视为图中的点，根据数据点间的距离建立相似度矩阵，对相似度矩阵进行无向图切割，让切图后不同的子图间的边权重和尽可能低，而子图内的边权重和尽可能高，从而达到聚类的目的。

(3)k-means：通过最小化数据点和它们的类中心点间的距离得到类标签。

(4)KNSC：新颖的非线性正交 NMF 法，通过将数据点进行非线性映射以及增加图正则项约束，来捕捉数据点的局部几何形状，再进行非负的分解得到聚类结果[18]。

与标准的 *NMF* 的对比实验中，对 $\boldsymbol{W}$ 和 $\boldsymbol{H}$ 使用下式进行初始化。

$$\boldsymbol{W}^1 = abs(randn(d, r)) \tag{24}$$

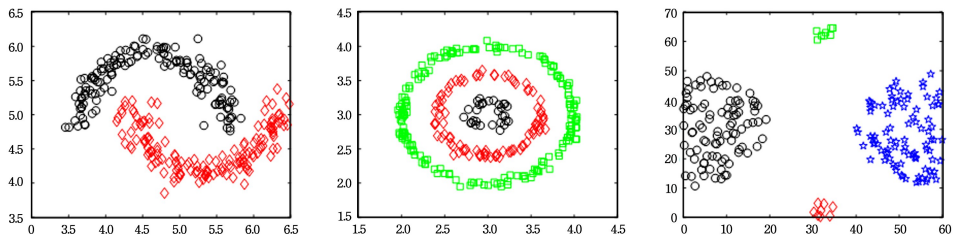
$$\boldsymbol{H}^1 = abs(randn(r, n)) \tag{25}$$

其中，$randn(d, r)$表示生成 $d \times r$ 维的服从正态分布的随机矩阵。与 KNSC 算法的对比实验中，参照文献[18]中设置的参数。提出的 HuberNMF 算法以及对比的 4 种聚类算法，在所有数据集上均进行 50 次独立的实验，并取其平均准确率。

4.3 合成数据集的聚类结果

使用的合成数据集包括双月形数据集、三环形数据集、四高斯堆形数据集，合成方法参见文献[19]。图 2 给出了这 3 种合成数据集的基本形状。例如，图 2(a)中分别用“空心圆”和“菱形”代表不同的数据点，相同形状的数据点表示相同类别。这组实验中使用了式(24)和式(25)初始化 $\boldsymbol{W}$ 和 $\boldsymbol{H}$，其中 r 取 5，残差相关的超参数 δ 设置为 1，ε 容忍度设置为10^{-6}，正则化参数 λ 设置为 0.5。

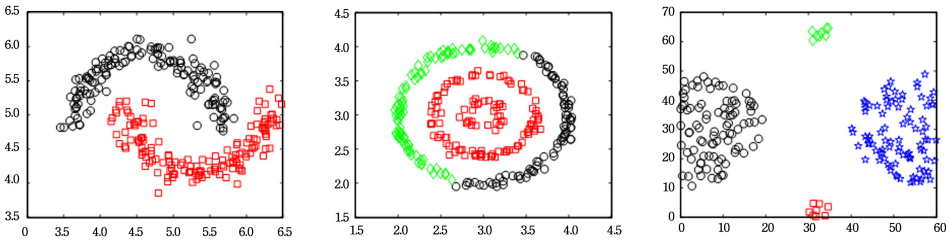
图 3－图 7 给出了 5 种算法在二维合成数据集上的聚类结果。每个图中的 3 个子图表示同一种算法分别在 3 种数据集上的聚类结果。对于图 3－图 7，对比图(a)可以看出，HuberNMF 算法在双月形数据集上能完全正确地分类，明显优于其他 4 种算法；对比图(b)可以看出，HuberNMF 算法对于三环的分类程度更细致一些，因为其他算法将三环沿着直线分割成三个扇形的三环，而 HuberNMF 可以沿着弧线分割；而对比图(c)可以看出，只有 HuberNMF，NMF，Ncut 3 种算法能够完全正确地分类。因此，本文算法优于现有算法。



(a)two moon (b)three ring (c)four cluster

图 2 合成数据集图形

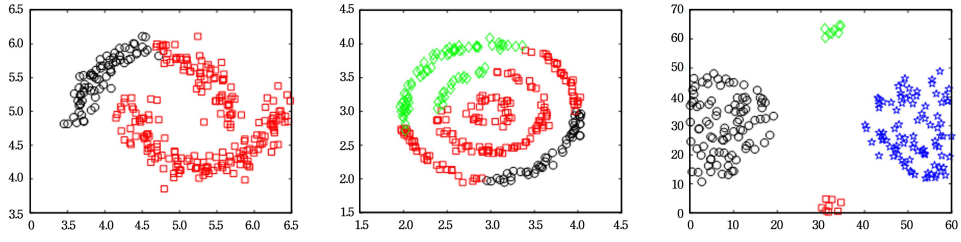
Fig. 2 Compositing data set graphs



(a) HuberNMF two moon result (b) HuberNMF three ring result (c) HuberNMF four cluster result

图 3 HuberNMF 在合成数据集上的聚类结果

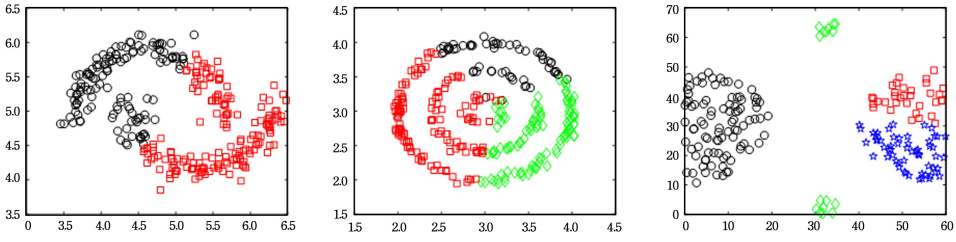
Fig. 3 HuberNMF clustering results in synthetic data sets



(a) NMF two moon result (b) NMF three ring result (c) NMF four cluster result

图 4 NMF 在合成数据集上的聚类结果

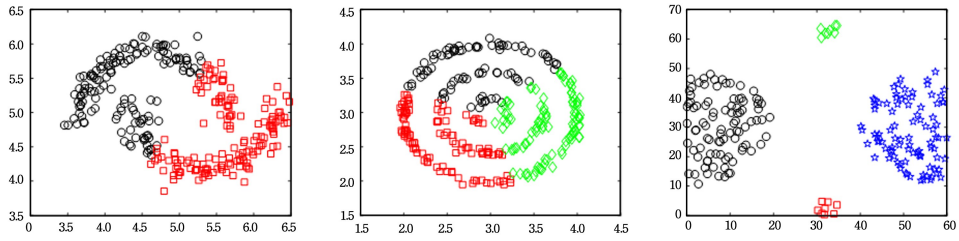
Fig. 4 NMF clustering results in synthetic data sets



(a) KNSC two moon result (b) KNSC three ring result (c) KNSC four cluster result

图 5 KNSC 在合成数据集上的聚类结果

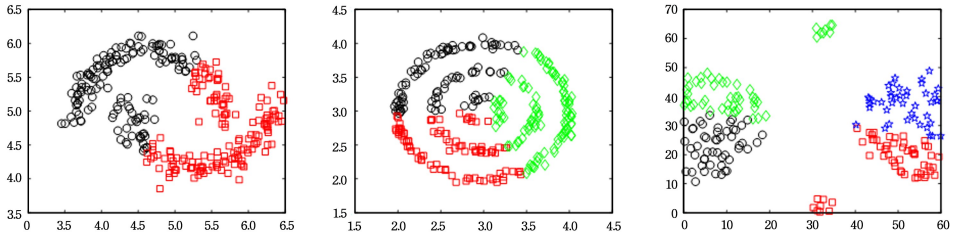
Fig. 5 KNSC clustering results in synthetic data sets



(a) Ncut two moon result (b) Ncut three ring result (c) Ncut four cluster result

图 6 Ncut 在合成数据集上的聚类结果

Fig. 6 Ncut clustering results in synthetic data sets



(a) k-means two moon result (b) k-means three ring result (c) k-means four cluster result

图 7 k-means 在合成数据集上的聚类结果

Fig. 7 k-means clustering results in synthetic data sets

4.4 真实数据集的聚类结果

真实数据集使用了 5 个常用于 NMF 聚类的数据集：UCI 中的 Zoo，Wine，Soybean，Iris，Vehicle[20]。表 1 列出了各个数据集的样本个数、维度和类的数量。为了验证模型对噪声的鲁棒性，给 5 个数据集添加了均值为 0、标准差分别为 0.05，0.1，0.2，0.5 的高斯噪声。

表 2—表 6 列出了 5 种算法在 5 个真实数据集上的聚类结果，其中将最高值加粗显示。由于 Soybean 数据集中负值多，如果使用标准 NMF 算法进行聚类，其乘性更新的过程无法得到有效的表达矩阵，导致算法失效，而 HuberNMF 算法使用了投影梯度法，能有效处理数据中负值的情况。

表 1 数据集信息

Table 1 Information of datasets

Datasets	Samples	Dimension	Clusters
Zoo	101	16	7
Wine	178	12	3
Soybean	47	35	4
Iris	150	4	3
Vehicle	846	18	4

表 2 5 种算法在 Iris 数据集上的聚类结果

Table 2 Clustering results of 5 algorithms on Iris dataset

Dataset	Noise/%	Metric	NMF	KNSC	Ncut	kmeans	HuberNMF
Iris	0	ACC	0.6720	0.7453	0.7880	0.8666	**0.9553**
		NMI	0.6294	0.6014	0.5965	0.7387	**0.8748**
	5	ACC	0.7126	0.7260	0.7933	0.8670	**0.9706**
		NMI	0.6451	0.6128	0.6001	0.7533	**0.9078**
	10	ACC	0.6906	0.7266	0.8000	0.8682	**0.9646**
		NMI	0.6305	0.6048	0.5845	0.7487	**0.8798**
	20	ACC	0.6760	0.7333	0.7333	0.8694	**0.9220**
		NMI	0.6314	0.6071	0.4454	0.7459	**0.8368**
	50	ACC	0.7313	0.7393	0.6573	0.8121	0.8453
		NMI	0.5887	0.5499	0.2795	0.6358	**0.6633**

表 3 5 种算法在 Wine 数据集上的聚类结果

Table 3 Clustering results of 5 algorithms on Wine dataset

Dataset	Noise/%	Metric	NMF	KNSC	Ncut	kmeans	HuberNMF
Wine	0	ACC	0.6646	0.4028	0.7853	0.4943	**0.8134**
		NMI	0.5272	0.0563	0.4189	0.1039	**0.6207**
	5	ACC	0.6904	0.3994	0.7831	0.4961	**0.8005**
		NMI	0.5435	0.0472	0.4091	0.1073	**0.6190**
	10	ACC	0.6387	0.4196	0.7331	0.4991	**0.7719**
		NMI	0.5466	0.0447	0.3874	0.1059	**0.6164**
	20	ACC	0.5955	0.4146	0.7951	0.4905	**0.8106**
		NMI	0.5367	0.0622	0.4185	0.1072	**0.6150**
	50	ACC	0.5814	0.4151	**0.6853**	0.4952	0.6561
		NMI	0.4078	0.0545	0.2531	0.0998	**0.5494**

表 4 5 种算法在 Soybean 数据集上的聚类结果

Table 4 Clustering results of 5 algorithms on Soybean dataset

Dataset	Noise/%	Metric	NMF	KNSC	Ncut	kmeans	HuberNMF
Soybean	0	ACC	NAN	0.7425	0.5285	0.7438	**0.8361**
		NMI	NAN	0.7471	0.3891	0.7453	**0.8084**
	5	ACC	0.7808	0.7191	0.5523	0.7374	**0.8510**
		NMI	0.7622	0.6915	0.4183	0.7575	**0.8299**
	10	ACC	0.7808	0.7361	0.5625	0.7451	**0.8595**
		NMI	0.7621	0.7123	0.4328	0.7691	**0.8308**
	20	ACC	0.8063	0.6744	0.5336	0.7676	**0.8663**
		NMI	0.7753	0.6773	0.3887	0.7522	**0.7799**
	50	ACC	0.8106	0.7170	0.5046	0.7604	**0.8446**
		NMI	0.7585	0.6815	0.4199	0.7507	**0.7726**

表 5 5 种算法在 Zoo 数据集上的聚类结果

Table 5 Clustering results of 5 algorithms on Zoo dataset

Dataset	Noise/%	Metric	NMF	KNSC	Ncut	kmeans	HuberNMF
Zoo	0	ACC	0.5821	0.5821	0.5085	**0.7554**	0.5831
		NMI	0.7017	0.5687	0.5555	**0.7512**	0.7009
	5	ACC	0.5792	0.5871	0.4968	**0.7275**	0.6128
		NMI	0.6948	0.5601	0.5749	**0.7393**	0.7114
	10	ACC	0.5821	0.5811	0.4809	**0.7184**	0.5901
		NMI	0.6947	0.5529	0.5522	**0.7460**	0.7022
	20	ACC	0.6237	0.5544	0.4190	**0.7277**	0.5841
		NMI	0.7084	0.5248	0.4657	**0.7392**	0.6891
	50	ACC	0.6079	0.4881	0.3823	0.6184	**0.6653**
		NMI	0.5921	0.4420	0.3039	0.5958	**0.6179**

表 6 5 种算法在 Vehicle 数据集上的聚类结果

Table 6 Clustering results of 5 algorithms on Vehicle dataset

Dataset	Noise/%	Metric	NMF	KNSC	Ncut	kmeans	HuberNMF
Vehicle	0	ACC	0.4680	0.3574	0.3982	0.4434	**0.4787**
		NMI	**0.2329**	0.0481	0.1134	0.1922	0.2165
	5	ACC	0.4702	0.3468	0.3974	0.4380	**0.4882**
		NMI	0.2249	0.0491	0.1127	0.1962	**0.2355**
	10	ACC	0.4712	0.3457	0.4012	0.4421	**0.4872**
		NMI	0.2214	0.0385	0.1167	0.1947	**0.2344**
	20	ACC	0.4617	0.3478	0.4008	0.4393	**0.4882**
		NMI	0.2082	0.0460	0.1151	0.1972	**0.2258**
	50	ACC	0.4563	0.3468	0.4297	0.4412	**0.4914**
		NMI	0.2153	0.0416	0.1267	0.1980	**0.2284**

实验结果表明，在 Iris，Wine，Soybean，Vehicle 4 个数据集上，HuberNMF 算法的聚类效果明显优于标准 NMF 算法、KNSC 算法和 Ncut 算法。在 Zoo 数据集噪声为 0 以及标准差为 0.05，0.1，0.2 的情况下，虽然文中提出的 HuberNMF 算法的聚类效果次于 k-means 算法，但与标准 NMF 算法相比，HuberNMF 算法仍然优于标准 NMF 算法的聚类效果。

4.5 参数敏感度分析

在现实应用中，要求无监督的学习方法对参数的变化敏感度变弱来增强稳定性。本文以 Iris 和 Soybean 两个真实数据集为例，进行了参数敏感度分析。由于 ε 表示对于梯度模长变化的容忍度，通常根据实际需求给定，这里统一设置为 10^{-6}。

(1)r 取 5，固定参数 δ 为 1，观察 λ 的变化对聚类结果的影响。参数 λ 取 0～1 之间的数，步长设置为 0.1。聚类结果如图 8、图 9 所示。从图 8 可以看出，参数 λ 在 0.1～1 时，算法的聚类准确率在 0.9～1 之间平稳，变化幅度不超过 10%。从图 9 可以看出，虽然聚类准确率出现波动，但是变化幅度不超过 15%。由此，证明了 HuberNMF 算法对于 λ 参数变化的不敏感性。

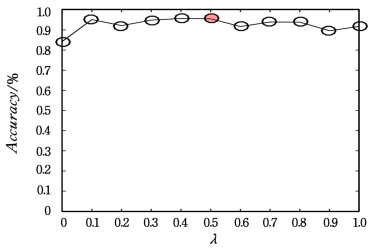


图 8 Iris 数据集上对 λ 的敏感度分析

Fig. 8 Sensitivity analysis of λ on Iris

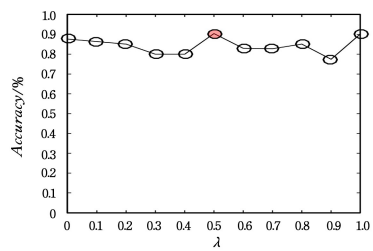


图 9 Soybean 数据集上对 λ 的敏感度分析

Fig. 9 Sensitivity analysis of λ on Soybean

(2)r 取 5，固定 λ 为 0.5，观察 δ 的变化对实验结果的影响，实验结果如图 10、图 11 所示。对于 Iris 数据集，随着 δ 的变化，聚类准确率变化幅度不超过 5%。而对于 Soybean 数据集，随着 δ 的变化，聚类结果出现波动。针对不同的数据集，确定 δ 仍然是一个比较困难的问题。

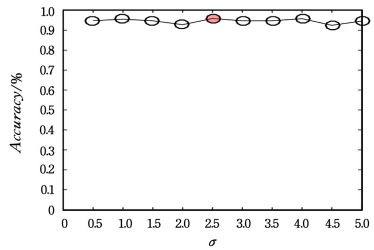


图 10 Iris 数据集上对 δ 的敏感度分析

Fig. 10 Sensitivity analysis of δ on Iris

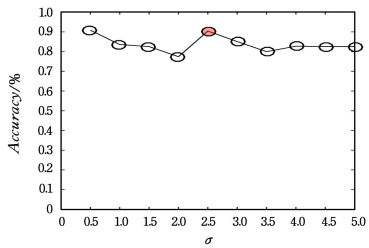


图 11 Soybean 数据集上对 δ 的敏感度分析

Fig. 11 Sensitivity analysis of δ on Soybean

(3)固定 λ 为 0.5，δ 为 1，观察 r 的变化对实验结果的影响，实验结果如图 12、图 13 所示。考虑到 Iris 数据集和 Soybean 数据集的样本维度不同，对 r 有针对性地采用了不同的取值范围进行对比实验。从图 12 中可以看出，随着 r 在 3～10 之间变化，聚类准确率的变化幅度不超过 5%。由于 Iris 数据集的样本维度为 4，r 取较小值 4 时聚类效果更好。而对于 Soybean 数据集，当 r 取较小维度 5 时，聚类准确率明显低于其他取值对应的聚类准确率，这是因为 Soybean 数据集的样本维度为 35，远大于 5，因此 r 取 5 时将导致有效信息丢失。因此，r 的选取应该参照属性的维度大小来决定。

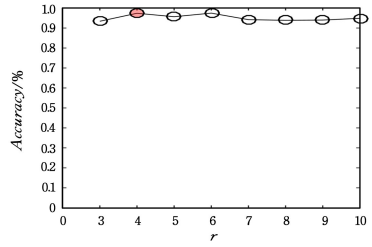


图 12 Iris 数据集上对 r 的敏感度分析

Fig. 12 Sensitivity analysis of r on Iris

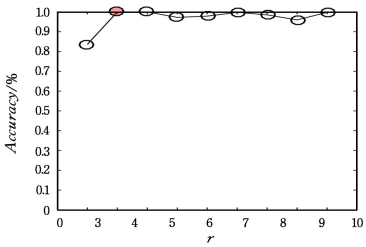


图 13 Soybean 数据集上对 r 的敏感度分析

Fig. 13 Sensitivity analysis of r on Soybean

结束语 本文提出了基于 Huber 损失的 NMF 聚类算法，并且针对提出的模型设计了基于投影梯度法的优化算法。通过在合成的和真实的数据上进行对比实验，证明了 HuberNMF 算法相比于流行的聚类算法，其聚类准确率更高，并且随着数据集中噪声的增加，HuberNMF 算法表现稳定。在合成数据集和 5 个真实数据集上的实验表明，$\lambda=0.5$ 时 HuberNMF 的效果较好。但对于不同的数据集，如何选择参数 δ 的适当值仍然需要进一步研究。

参 考 文 献

[1] LU H T, FU Z Y, SHU X. Non-negative and sparse spectral clustering [J]. Pattern Recognition, 2014, 47(1): 418-426.

[2] LEE D D, SEUNG H S. Learning the parts of objects by non-negative matrix factorization [J]. Nature, 1999, 401: 788-791.

[3] LEE D D, SEUNG H S. Algorithms for non-negative matrix factorization[C]// NIPS. 2000: 535-541.

[4] LI M J, XIE Q, DING Q L. Orthogonal Non-negative Matrix Factorization for K-means Clustering [J]. Computer Science, 2016, 43(5): 204-208.

[5] LIN C J. Projected gradient methods for non-negative matrix factorization [J]. Neural Computation, 2007, 19(10): 2756-2779.

[6] HOYER P O. Non-negative matrix factorization with sparseness constraints [J]. Journal of Machine Learning Research, 2004, 5(9): 1457-1469.

[7] CAI D, HE H, HAN J. Graph regularized nonnegative matrix factorization for data representation [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2011, 33(8): 1548-1560.

[8] JIANG W, LI H, YU X, et al. Graph Regularized Non-negative Matrix Factorization with Sparseness Constraints [J]. Computer Science, 2013, 40(1): 218-220, 256.

[9] LIU H, WU Z, LI X. Constrained nonnegative matrix factorization for image representation [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012, 34(7): 1299-1311.

[10] KONG D, DING C, HUANG H. Robust nonnegative matrix factorization using L21-norm[C]// Proceedings of the 20th ACM CIKM. 2011: 673-682.

[11] DU L, LI X, SHEN Y D. Robust nonnegative matrix factorization via half-quadratic minimization[C]// IEEE. ICDM, 2012: 201-210.

[12] YANG S, HOU C, ZHANG C, et al. Robust non-negative matrix factorization via joint sparse and graph regularization[C]// International Joint Conference on Neural Networks. 2013: 1-5.

[13] KONG D, DING C, HUANG H. Robust nonnegative matrix factorization using l21-norm[C]// The 20th ACM International Conference on Information and Knowledge Management. 2011: 673-682.

[14] NIE F P, HUANG H, CAI X, et al. Efficient and robust feature selection via $L_{2,1}$-norms minimization [C]// Proceedings of International Conference on Neural Information Processing Systems. British, ACM, 2010: 1813-1821.

[15] CALAMAI P H, MORÈ J J. Projected gradient methods for linearly constrained problems [J]. Mathematical Programming, 1987, 39(1): 93-116.

[16] HUBER P J. Robust Statistics (second edition) [M]. New Jersey: John Wiley & Sons, 2009: 1-5.

[17] ANDREW S, TSOCHANTARIDIS I T. HOFMANN. Support vector machines for multiple-instance learning[C]// Advances in Neural Information Processing Systems. USA: The MIT Press, 2003: 577-584.

[18] TOLIC D, ANTULOV F N, KOPRVIA I. A nonlinear orthogonal non-negative matrix factorization approach to subspace clustering [J]. Pattern Recognition, 2018, 82(10): 40-55.

[19] NIE F P, WANG X Q, HUANG H. Clustering and projected clustering with adaptive neighbors[C]// ACM SIGKDD Conference on Knowledge Discovery and Data Mining. New York, ACM, 2014: 977-986.

[20] DUA D, GRAFF C. UCI Machine Learning Repository[OL]. http://archive.ics.uci.edu/ml.

WANG Li-xing, born in 1992, postgraduate. Her main research interests include Subs pace Learning and NMF.

CAO Fu-yuan, born in 1974, professor, is a member of China Computer Federatio. His main research interests include subspace learning and NMF.