不完全非负矩阵分解的加速算法

史加荣,焦李成,尚凡华
(西安电子科技大学智能感知与图像理解教育部重点实验室和智能信息处理研究所,陕西西安 710071)

摘 要: 非负矩阵分解(NMF)已成为数据分析与处理的一种日益流行的方法.当数据矩阵不完全时,可用加权非负矩阵分解(WNMF)来分解矩阵.但是在 WNMF 算法中,对于给定的搜索方向,步长的选取一般来说不是最优的.本文研究了不完全非负矩阵分解(INMF)问题,提出了加速算法(AINMF).首先,将 INMF 问题转化为交替地求解两个非负最小二乘(NNLS)问题.对于每个 NNLS 问题,在搜索方向上采用精确的步长.接着,分析了 NNLS 问题的算法复杂度.最后,试验结果证实了 AINMF 优于 WNMF.

关键词: 非负矩阵分解; 不完全非负矩阵分解; 数据丢失问题; 加权非负矩阵分解; 非负最小二乘
中图分类号: TP391 **文献标识码:** A **文章编号:** 0372-2112 (2011) 02-0291-05

Accelerated Algorithm to Incomplete Nonnegative Matrix Factorization

SHI Jia-rong, JIAO Li-cheng, SHANG Fan-hua
(*Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education of China, Institute of Intelligent Information Processing, Xidian University, Xi'an, Shaanxi* 710071, *China*)

Abstract: Nonnegative matrix factorization (NMF) is an increasingly popular technique for data processing and analysis. For an incomplete data matrix, the weighted nonnegative matrix factorization (WNMF) is employed to decompose it. But the searching step size in WNMF is not optimal along the given searching direction. This paper studies the incomplete nonnegative matrix factorization (INMF) and proposes an accelerated algorithm. First, INMF is transformed into solving alternatively two nonnegative least squares (NNLS) problems. For each NNLS problem, the exact step size is chosen along the searching direction. Then, the complexity of NNLS problems is analyzed. Finally, experimental results show that the proposed method outperforms WNMF.

Key words: nonnegative matrix factorization; incomplete nonnegative matrix factorization; missing data problem; weighted nonnegative matrix factorization; nonnegative least squares

1 引言

数据分析广泛地存在于科学、工程和商业等领域中.许多实际数据具有非负的特点,比如灰度图像、物质成分含量和文章中单词出现的频数等[1].当使用线性方法处理这样的非负数据时,往往期望得到非负的结果.此时,若采用传统的线性分析方法(如主成分分析、奇异值分解等),因其结果中含有负数而失去了物理意义,但采用非负矩阵分解(NMF, Nonnegative matrix factorization)就可以避免这一点[2].

NMF 是将一个给定的非负数据矩阵分解成两个低秩非负矩阵之积,它已成为数据处理与分析的一种日益流行的工具.NMF 是一种基于局部的表示方法,即它仅允许非负基的加性组合,而不允许减性组合[2].NMF 已经被广泛地应用在模式识别、信号与图像处理、聚类、文本挖掘和生物信息学等领域中[1~8].

在实际问题中,许多数据矩阵是不完全的,即它们的一些元素丢失或不能被观测到.根据矩阵的部分元素来恢复整个矩阵这一问题被称为矩阵补全(Matrix completion)问题,它已成为继压缩感知(CS, Compressed sensing)理论之后的又一种重要的信号获取方式[9,10].本文研究了非负矩阵的补全问题,即基于不完全非负矩阵分解(INMF)来补全矩阵.作为 NMF 的乘性迭代更新法的推广,加权非负矩阵分解(WNMF)算法是一种常用的非负矩阵补全方法,但是它的搜索步长的选取一般来说不是最优的.本文将求解 INMF 问题转化为交替地求解两个非负最小二乘(NNLS)问题,对于每个 NNLS 问题,在搜索方向上选取精确的步长.

收稿日期:2009-10-13;修回日期:2010-02-05
基金项目:国家 973 重点基础研究发展计划(No.2006CB705707);国家 863 高技术研究发展计划(No.2007AA12Z223, No.2007AA12Z136);国家自然科学基金(No.60603019, No.60602064, No.60702062);长江学者和创新团队发展计划(No.IRT0645)

2 非负矩阵分解

正式地,非负矩阵分解可以表示为:

$$\boldsymbol{X}\approx\boldsymbol{UV} \tag{1}$$

其中 $\boldsymbol{X}\in\mathbb{R}^{m\times n}$ 是 m 个变量 n 个目标的观测矩阵,$\boldsymbol{U}\in\mathbb{R}^{m\times r}$ 是基向量组成的矩阵,$\boldsymbol{V}\in\mathbb{R}^{r\times n}$ 为系数矩阵.基的数目 r 要小于 m 或 n,当 $r\ll\min(m,n)$ 时,原始的数据矩阵得到了有效的压缩.

矩阵间的误差测量的选择与噪声的类型有关.本文假设噪声为等方高斯噪声,则 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 的极大似然估计是下列优化问题的最优解:

$$\begin{aligned}&\min\quad g(\boldsymbol{U},\boldsymbol{V})=\|\boldsymbol{X}-\boldsymbol{UV}\|_F^2\\&\text{s.t.}\quad \boldsymbol{U}\geqslant 0,\boldsymbol{V}\geqslant 0\end{aligned} \tag{2}$$

其中 $\|\cdot\|_F$ 表示 Frobenius 范数,目标函数 $g(\boldsymbol{U},\boldsymbol{V})$ 度量了 $\boldsymbol{X}$ 和 $\boldsymbol{UV}$ 的误差.优化问题(2)不是凸的,但是当固定 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 中的一个变量矩阵时,它却是凸的.基于此,许多交替式的求解方法被提出,例如:交替式最小二乘法[11]、乘性迭代更新法[2]和投影梯度法[12]等.关于 NMF 算法的综述可参见文献[13].在这些算法中,最流行的是 Lee 和 Seung 的乘性迭代更新法,下面给出其推导过程[14].

目标函数 $g(\boldsymbol{U},\boldsymbol{V})$ 关于矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 的梯度分别为:

$$\begin{aligned}\nabla\boldsymbol{g}_U(\boldsymbol{U},\boldsymbol{V})&=2\boldsymbol{UVV}^{\mathrm{T}}-2\boldsymbol{XV}^{\mathrm{T}}\\\nabla g_V(\boldsymbol{U},\boldsymbol{V})&=2\boldsymbol{U}^{\mathrm{T}}\boldsymbol{UV}-2\boldsymbol{U}^{\mathrm{T}}\boldsymbol{X}\end{aligned} \tag{3}$$

所以,矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 的更新规则为:

$$\begin{aligned}\boldsymbol{U}&\leftarrow\boldsymbol{U}-\boldsymbol{A}.*(2\boldsymbol{UVV}^{\mathrm{T}}-2\boldsymbol{XV}^{\mathrm{T}})\\\boldsymbol{V}&\leftarrow\boldsymbol{V}-\boldsymbol{B}.*(2\boldsymbol{U}^{\mathrm{T}}\boldsymbol{UV}-2\boldsymbol{U}^{\mathrm{T}}\boldsymbol{X})\end{aligned} \tag{4}$$

其中,矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 为步长矩阵,'.*'表示矩阵的 Hadamard 积.选取步长矩阵 $\boldsymbol{A}=\boldsymbol{U}./(2\boldsymbol{UVV}^{\mathrm{T}})$,$\boldsymbol{B}=\boldsymbol{V}./(2\boldsymbol{U}^{\mathrm{T}}\boldsymbol{UV})$,则有如下的迭代规则:

$$\begin{aligned}\boldsymbol{U}&\leftarrow(\boldsymbol{U}.*(\boldsymbol{XV}^{\mathrm{T}}))./(\boldsymbol{UVV}^{\mathrm{T}})\\\boldsymbol{V}&\leftarrow(\boldsymbol{V}.*(\boldsymbol{U}^{\mathrm{T}}X))./(\boldsymbol{U}^{\mathrm{T}}\boldsymbol{UV})\end{aligned} \tag{5}$$

其中,'./'表示矩阵的 Hadamard 商.上述乘性迭代算法收敛到平稳点[12,14].由于此算法简单易行,并且具有良好的性能,所以它受到了广泛的青睐.

3 不完全非负矩阵分解

当给定的非负数据矩阵不完全时,称它的非负分解为不完全非负矩阵分解(INMF).为了表示非负矩阵 $\boldsymbol{X}$ 的丢失元素,$m\times n$ 的指示矩阵 $\boldsymbol{H}$ 定义如下:如果 x_{ij} 存在,则 $h_{ij}=1$;否则,$h_{ij}=0$.本文假设矩阵 $\boldsymbol{H}$ 是已知的,于是度量误差可重新表示为:

$$f(\boldsymbol{U},\boldsymbol{V})=\|\boldsymbol{H}.*(\boldsymbol{X}-\boldsymbol{UV})\|_F^2 \tag{6}$$

3.1 加权非负矩阵分解

作为 NMF 的乘性迭代算法的推广,加权非负矩阵分解(WNMF)可用来求解 INMF.目标函数 $f(\boldsymbol{U},\boldsymbol{V})$ 关于矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 的梯度分别为:

$$\begin{aligned}\nabla f_U(\boldsymbol{U},\boldsymbol{V})&=2(\boldsymbol{H}.*(\boldsymbol{UV}))\boldsymbol{V}^{\mathrm{T}}-2(\boldsymbol{H}.*\boldsymbol{X})\boldsymbol{V}^{\mathrm{T}}\\\nabla f_V(\boldsymbol{U},\boldsymbol{V})&=2\boldsymbol{U}^{\mathrm{T}}(\boldsymbol{H}.*(\boldsymbol{UV}))-2\boldsymbol{U}^{\mathrm{T}}(\boldsymbol{H}.*\boldsymbol{X})\end{aligned} \tag{7}$$

类似于 NMF 步长矩阵的选取,WNMF 算法更新过程如下[15,16]:

$$\begin{aligned}\boldsymbol{U}&\leftarrow[\boldsymbol{U}.*((\boldsymbol{H}.*\boldsymbol{X})\boldsymbol{V}^{\mathrm{T}})]./[(\boldsymbol{H}.*(\boldsymbol{UV}))\boldsymbol{V}^{\mathrm{T}}]\\\boldsymbol{V}&\leftarrow[\boldsymbol{V}.*(\boldsymbol{U}^{\mathrm{T}}(\boldsymbol{H}.*\boldsymbol{X}))]./[\boldsymbol{U}^{\mathrm{T}}(\boldsymbol{H}.*(\boldsymbol{UV}))]\end{aligned} \tag{8}$$

WNMF 算法基于梯度下降法,但在给定的搜索方向上,搜索步长的选取一般来说不是最优的.

3.2 INMF 的等价模型

为更直观地描述 INMF 优化问题,下面给出其等价模型.由于误差函数 $f(\boldsymbol{U},\boldsymbol{V})$ 中存在 Hadamard 积,分析上不易于操作.为此引入几个符号来重新把它表示为简单形式.记 $\boldsymbol{U}=(u_1,\cdots,u_m)^{\mathrm{T}}$,$\boldsymbol{V}=(v_1,\cdots,v_r)^{\mathrm{T}}$.再令 $\boldsymbol{u}=(u_1^{\mathrm{T}},\cdots,u_m^{\mathrm{T}})^{\mathrm{T}}$,$\boldsymbol{v}=(v_1^{\mathrm{T}},\cdots,v_r^{\mathrm{T}})^{\mathrm{T}}$.于是,矩阵 $\boldsymbol{U}$ 和 mr 维向量 $\boldsymbol{u}$ 可视为可以相互交换,矩阵 $\boldsymbol{V}$ 和 nr 维向量 $\boldsymbol{v}$ 亦是如此.因此,$f(\boldsymbol{U},\boldsymbol{V})=f(\boldsymbol{u},\boldsymbol{v})$.

令 k 为观测元素的数目,$\boldsymbol{x}$ 为观测元素 x_{ij} 按 i 和 j 的字典序排列成的 k 维列向量.记 $\boldsymbol{F}=h(\boldsymbol{I}_m\otimes\boldsymbol{V}^{\mathrm{T}})$,$\boldsymbol{G}=h(\boldsymbol{U}\otimes\boldsymbol{I}_n)$,其中 $\boldsymbol{I}_m$ 和 $\boldsymbol{I}_n$ 分别为 m 和 n 阶单位矩阵,'$\otimes$'表示矩阵的 Kronecker 积,$h(\cdot)$ 为矩阵的选择算子.$h(\cdot)$ 定义如下:如果 $h_{ij}=1$,则保留矩阵的第 $i+j$ 行;否则删除此行.$\boldsymbol{X}$ 中丢失与观测元素的分布(即指示矩阵 $\boldsymbol{H}$)确定 $\boldsymbol{F}$ 和 $\boldsymbol{G}$ 维数的大小.显然,$\boldsymbol{F}\in\mathbb{R}^{k\times mr}$,$\boldsymbol{G}\in\mathbb{R}^{k\times nr}$.误差函数 $f(\boldsymbol{u},\boldsymbol{v})$ 可表示为 $f(\boldsymbol{u},\boldsymbol{v})=\|\boldsymbol{Fu}-\boldsymbol{x}\|_2^2$ 或 $f(\boldsymbol{u},\boldsymbol{v})=\|\boldsymbol{Gv}-\boldsymbol{x}\|_2^2$.对于固定的 $\boldsymbol{H}$,$\boldsymbol{F}$ 可看作 $\boldsymbol{v}$ 的函数.因此,可记 $\boldsymbol{F}=\boldsymbol{F}(\boldsymbol{v})$.类似地,记 $\boldsymbol{G}=\boldsymbol{G}(\boldsymbol{u})$.

为了求得最优的向量 $\boldsymbol{u}$ 和 $\boldsymbol{v}$,需求解下列优化问题:

$$\begin{aligned}&\min\quad f(\boldsymbol{u},\boldsymbol{v})\\&\text{s.t.}\quad \boldsymbol{u}\geqslant 0,\boldsymbol{v}\geqslant 0\end{aligned} \tag{9}$$

同样,优化问题(9)关于向量 $\boldsymbol{u}$ 和 $\boldsymbol{v}$ 不是凸的,但是关于每个向量却是凸的.如果固定 $\boldsymbol{u}$ 和 $\boldsymbol{v}$ 中的一个向量,则优化问题(9)等价于非负最小二乘(NNLS)问题.这启发我们交替求解 NNLS 问题.

3.3 INMF 的加速算法(AINMF)

作为 NMF 的特例,NNLS 问题一直都是研究的热点.最近,Merritt 等人提出了求解 NNLS 问题的内点梯度法(IPG),它是 Lee 和 Seung 的乘性迭代算法的加速版本[17].本文提出 INMF 的加速算法(AINMF),即将求解 INMF 问题转化为交替地求解两个 NNLS 问题,并且采用 IPG 求解每个 NNLS 问题.

在求解 NNLS 问题时,为了使用 IPG,先将非负向量 $\boldsymbol{x}$ 变为正向量,即 $\boldsymbol{x}$ 的每个零元素均由一个足够小的正数来代替.现在考虑向量 $\boldsymbol{v}$ 固定的情形.在此情形下,$\boldsymbol{F}$ 也是固定的,取目标函数 $f(\boldsymbol{u},\boldsymbol{v})=\|\boldsymbol{Fu}-\boldsymbol{x}\|_2^2$.

下面给出 $\boldsymbol{u}$ 的迭代过程.目标函数 $f(\boldsymbol{u},\boldsymbol{v})$关于 $\boldsymbol{u}$ 的梯度为$f_u(\boldsymbol{u},\boldsymbol{v})=2(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{u}-\boldsymbol{F}^{\mathrm{T}}\boldsymbol{x})$.根据 Lee 和 Seung 的算法,搜索方向 $\boldsymbol{d}=(-\boldsymbol{u}.*f_u(\boldsymbol{u},\boldsymbol{v}))./(2\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{u})$.希望采取迭代规则:$\boldsymbol{u}\leftarrow\boldsymbol{u}+\alpha\boldsymbol{d}$,其中步长 α 为目标函数沿搜索方向 $\boldsymbol{d}$ 的满足非负约束的精确最小步长.同时,为了保证更新后的向量 $\boldsymbol{u}$ 仍是正的,取

$$\alpha=\min\left(\frac{-\boldsymbol{d}^{\mathrm{T}}f_u(\boldsymbol{u},\boldsymbol{v})}{2\boldsymbol{d}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{d}},\tau\max\{\bar{\alpha}:\boldsymbol{u}+\bar{\alpha}\boldsymbol{d}\geqslant 0\}\right)\quad(10)$$

其中 $\tau\in(0,1)$且充分接近于 1.

类似地,当 $\boldsymbol{u}$ 固定时,可以得到相应的搜索方向 $\boldsymbol{q}$ 和步长 β.AINMF 算法如下.

算法 1　AINMF 算法

Step1　初始化正向量 $\boldsymbol{u}$ 和 $\boldsymbol{v}$,令 $\tau\in(0,1)$.

Step2　依据指示矩阵 $\boldsymbol{H}$,由 $\boldsymbol{u},\boldsymbol{v}$ 和 $\boldsymbol{X}$ 得到 $\boldsymbol{G},\boldsymbol{F},\boldsymbol{x}$.

Step3　计算 $f_u(\boldsymbol{u},\boldsymbol{v})=2(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{u}-\boldsymbol{F}^{\mathrm{T}}\boldsymbol{x})$,令 $\boldsymbol{d}=(-\boldsymbol{u}.*f_u(\boldsymbol{u},\boldsymbol{v}))./(2\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{u})$.

Step4　根据式(10)计算 α,令 $\boldsymbol{u}\leftarrow\boldsymbol{u}+\alpha\boldsymbol{d}$.

Step5　依据指示矩阵 $\boldsymbol{H}$,由 $\boldsymbol{u}$ 得到 $\boldsymbol{G}$.

Step6　计算 $f_v(\boldsymbol{u},\boldsymbol{v})=2(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{v}-\boldsymbol{G}^{\mathrm{T}}x)$,令 $\boldsymbol{q}=(-\boldsymbol{v}.*f_v(\boldsymbol{u},\boldsymbol{v}))./(2\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{v})$.

Step7　计算 $\beta=\min\left(\frac{-\boldsymbol{q}^{\mathrm{T}}f_v(\boldsymbol{u},\boldsymbol{v})}{2\boldsymbol{q}^{\mathrm{T}}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{q}},\tau\max\{\bar{\beta}:\boldsymbol{v}+\bar{\beta}\boldsymbol{q}\geqslant 0\}\right)$,令 $\boldsymbol{v}\leftarrow\boldsymbol{v}+\beta\boldsymbol{q}$.

Step8　依据指示矩阵 $\boldsymbol{H}$,由 $\boldsymbol{v}$ 得到 $\boldsymbol{F}$.

Step9　如果终止条件满足,则停止算法,输出向量 $\boldsymbol{u}$ 和 $\boldsymbol{v}$;否则转 Step3.

在算法 1 中,可以根据(0,1)区间上的均匀分布来随机初始化正向量 $\boldsymbol{u}$ 和 $\boldsymbol{v}$.终止条件可以设置为运行时间或最大迭代次数.

3.4　AINMF 的复杂性分析

一般来说,矩阵 $\boldsymbol{F}$ 和 $\boldsymbol{G}$ 的维数非常大,这会导致存储和计算的困难.考虑到 $\boldsymbol{F}$ 和 $\boldsymbol{G}$ 的特殊结构,可以不必存储矩阵 $\boldsymbol{F}$ 和 $\boldsymbol{G}$,同时降低计算复杂度.在实施算法的过程中,关键的任务是计算搜索方向 $\boldsymbol{d}$、$\boldsymbol{q}$ 和相应的步长 α、β.下面推导出它们的简洁形式.

为了获得 $\boldsymbol{d}$ 和 α 的简洁形式,必须计算下列三项:$\boldsymbol{F}^{\mathrm{T}}\boldsymbol{x}$,$\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{u}$ 和 $\boldsymbol{d}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{d}$.前两项计算如下:

$$\boldsymbol{F}^{\mathrm{T}}\boldsymbol{x}=\begin{pmatrix}\sum_{i=1}^{n}h_{1i}x_{1i}\boldsymbol{v}_i\\ \vdots\\ \sum_{i=1}^{n}h_{mi}x_{mi}\boldsymbol{v}_i\end{pmatrix}=ver(\boldsymbol{V}(\boldsymbol{H}.*\boldsymbol{X})^{\mathrm{T}})\quad(11)$$

$$\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{u}=\mathrm{diag}\Big(\sum_{i=1}^{n}h_{1i}\boldsymbol{v}_i\boldsymbol{v}_i^{\mathrm{T}},\cdots,\sum_{i=1}^{n}h_{mi}\boldsymbol{v}_i\boldsymbol{v}_i^{\mathrm{T}}\Big)\boldsymbol{u}$$

$$=\begin{pmatrix}\sum_{i=1}^{n}h_{1i}\boldsymbol{v}_i\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{u}_1\\ \vdots\\ \sum_{i=1}^{n}h_{mi}\boldsymbol{v}_i\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{u}_m\end{pmatrix}$$

$$=ver(\boldsymbol{V}(\boldsymbol{H}.*(\boldsymbol{U}\boldsymbol{V}))^{\mathrm{T}})\quad(12)$$

其中 $ver(\cdot)$是矩阵按列顺序的向量化算子,$\mathrm{diag}(\cdot)$表示对角矩阵算子.根据上述两项,梯度 $f_u(\boldsymbol{u},\boldsymbol{v})$和搜索方向 $\boldsymbol{d}$ 可立即得到:

$$f_u(\boldsymbol{u},\boldsymbol{v})=2ver(\boldsymbol{V}(\boldsymbol{H}.*(\boldsymbol{U}\boldsymbol{V}-\boldsymbol{X}))^{\mathrm{T}})\quad(13)$$

$$\boldsymbol{d}=[\boldsymbol{u}.*ver(\boldsymbol{V}(\boldsymbol{H}.*(\boldsymbol{X}-\boldsymbol{U}\boldsymbol{V}))^{\mathrm{T}})]./ver(\boldsymbol{V}(\boldsymbol{H}.*(\boldsymbol{U}\boldsymbol{V}))^{\mathrm{T}})\quad(14)$$

因此,最后一项

$$\begin{aligned}\boldsymbol{d}^{\mathrm{T}}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{d}&=\boldsymbol{d}^{\mathrm{T}}\mathrm{diag}\Big(\sum_{i=1}^{n}h_{1i}\boldsymbol{v}_i\boldsymbol{v}_i^{\mathrm{T}},\cdots,\sum_{i=1}^{n}h_{mi}\boldsymbol{v}_i\boldsymbol{v}_i^{\mathrm{T}}\Big)\boldsymbol{d}\\&=\sum_{j=1}^{m}\sum_{i=1}^{n}h_{ji}\boldsymbol{d}_j^{\mathrm{T}}\boldsymbol{v}_i\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{d}_j\\&=\sum_{j=1}^{m}\sum_{i=1}^{n}h_{ji}(\boldsymbol{d}_j^{\mathrm{T}}\boldsymbol{v}_i)^2\\&=\|\boldsymbol{H}.*(\boldsymbol{D}\boldsymbol{V})\|_F^2\end{aligned}\quad(15)$$

其中矩阵 $\boldsymbol{D}=(\boldsymbol{d}_1^{\mathrm{T}},\cdots,\boldsymbol{d}_m^{\mathrm{T}})^{\mathrm{T}}$ 满足 $\boldsymbol{d}=ver(\boldsymbol{D})$.

类似地,为了获得 $\boldsymbol{q}$ 和 β 的简洁形式,必须计算下列三项:$\boldsymbol{G}^{\mathrm{T}}\boldsymbol{x}$,$\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{v}$ 和 $\boldsymbol{q}^{\mathrm{T}}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{q}$.前两项计算如下:

$$\boldsymbol{G}^{\mathrm{T}}\boldsymbol{x}=ver((\boldsymbol{H}.*\boldsymbol{X})^{\mathrm{T}}\boldsymbol{U})\quad(16)$$

$$\begin{aligned}&\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{v}\\&=ver\begin{pmatrix}\sum_{i=1}^{m}h_{i1}u_{i1}(\sum_{j=1}^{r}u_{ij}v_{j1}),\cdots,\sum_{i=1}^{m}h_{i1}u_{ir}(\sum_{j=1}^{r}u_{ij}v_{j1})\\ \cdots\cdots\\ \sum_{i=1}^{m}h_{in}u_{i1}(\sum_{j=1}^{r}u_{ij}v_{jn}),\cdots,\sum_{i=1}^{m}h_{in}u_{ir}(\sum_{j=1}^{r}u_{ij}v_{jn})\end{pmatrix}\\&=ver((\boldsymbol{H}.*(\boldsymbol{U}\boldsymbol{V}))^{\mathrm{T}}\boldsymbol{U})\end{aligned}\quad(17)$$

根据上述两项,梯度 $f_v(\boldsymbol{u},\boldsymbol{v})$和搜索方向 $\boldsymbol{q}$ 可立即得到:

$$f_v(\boldsymbol{u},\boldsymbol{v})=2ver((\boldsymbol{H}.*(\boldsymbol{U}\boldsymbol{V}-\boldsymbol{X}))^{\mathrm{T}}\boldsymbol{U})\quad(18)$$

$$\boldsymbol{q}=[\boldsymbol{v}.*ver((\boldsymbol{H}.*(\boldsymbol{X}-\boldsymbol{U}\boldsymbol{V}))^{\mathrm{T}}\boldsymbol{U})]./ver((\boldsymbol{H}.*(\boldsymbol{U}\boldsymbol{V}))^{\mathrm{T}}\boldsymbol{U})\quad(19)$$

因此,最后一项

$$\begin{aligned}\boldsymbol{q}^{\mathrm{T}}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{q}&=(\boldsymbol{q}_1^{\mathrm{T}},\cdots,\boldsymbol{q}_r^{\mathrm{T}})ver((\boldsymbol{H}.*(\boldsymbol{U}\boldsymbol{Q}^{\mathrm{T}}))^{\mathrm{T}}\boldsymbol{U})\\&=\|\boldsymbol{H}.*(\boldsymbol{U}\boldsymbol{Q}^{\mathrm{T}})\|_F^2\end{aligned}\quad(20)$$

其中矩阵 $\boldsymbol{Q}=(q_1,\cdots,q_r)^{\mathrm{T}}$ 满足 $\boldsymbol{q}=ver(\boldsymbol{Q}^{\mathrm{T}})$.

根据上面的简洁形式,可知算法 1 的空间复杂度为 $O(mn)$,且在每次迭代过程中的计算复杂度为 $O(mnr)$.总之,算法 1 的时空复杂度与 NMF 的乘性迭代算法和 WNMF 算法的复杂度相当.

4　实验分析

4.1　数据描述

在 ORL 和 Yale 人脸数据集上进行实验.ORL 人脸数据库包括 40 个人的人脸图像,其中每个人有 10 幅不同的图像.这些图像在不同时间获取,具有不同的表情和面部细节,而且稍许倾斜(不超过 20°).Yale 人脸数据

库包括 15 个人的人脸图像,其中每个人有 11 幅不同的图像.它们是具有不同表情、遮掩和光照条件的正面图像.为计算方便,这两个数据集的每幅图像均经下采样成为 64×64 的图像.将每幅人脸图像按列排列形成一个列向量,因此,ORL 和 Yale 人脸数据集可分别用 4096×400 和 4096×165 的矩阵来表示.数据矩阵中的零元素用 10^{-6} 代替.

4.2 实验设置

对于每个数据矩阵,令其元素均以相同的概率 p 随机丢失.在 ORL 数据集中取 $r=80$;在 Yale 数据集中取 $r=30$.对于给定的正矩阵 $\boldsymbol{X}$ 和丢失指示矩阵 $\boldsymbol{H}$,若相应的非负矩阵分解为 $\boldsymbol{UV}$,则它们的相对误差定义为:

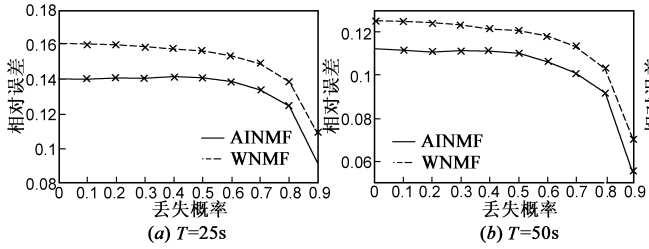
$$\frac{\|\boldsymbol{H}.*\boldsymbol{X}-\boldsymbol{H}.*(\boldsymbol{UV})\|_F}{\|\boldsymbol{X}.*\boldsymbol{H}\|_F} \tag{21}$$

在算法 1 中取 $\tau=0.999$.对于每个丢失概率 p,将算法 1 执行 20 次,最终用平均相对误差来评价算法的性能.

为了将 AINMF 和 WNMF 进行比较,设计两组实验.在第一组实验中,对于相同的运行时间 T,比较两种方法在不同丢失概率下的相对误差.在第二组实验中,对于相同的丢失概率 p,比较两种方法在不同迭代次数下的相对误差.

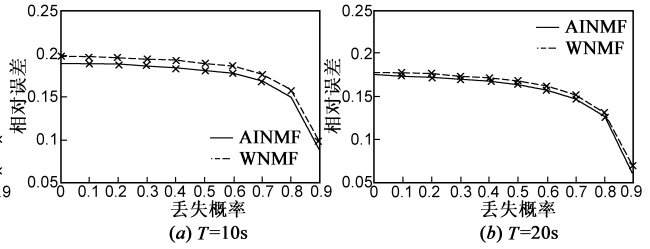
4.3 实验结果

第一组实验的结果如图 1 和图 2 所示.从这两个图上可以看出,在相同的运行时间内,对于各个丢失概率 p,AINMF 的相对误差都要优于 WNMF.同时,两种算法的相对误差的标准差都比较小,这说明这它们都比较稳定.丢失概率为 0 意味着数据矩阵没有丢失元素.第二组实验的结果如图 3 和图 4 所示.可以看出:当迭代次数相同时,AINMF 的相对误差要明显优于 WNMF;两种算法的相对误差的标准差都比较小,这同样说明它们都比较稳定.总之,AINMF 在收敛速度上要优于 WNMF.



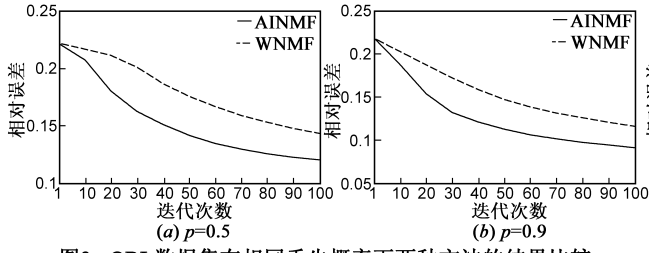
(a) T=25s (b) T=50s

图1 ORL数据集在相同运行时间下两种方法的结果比较



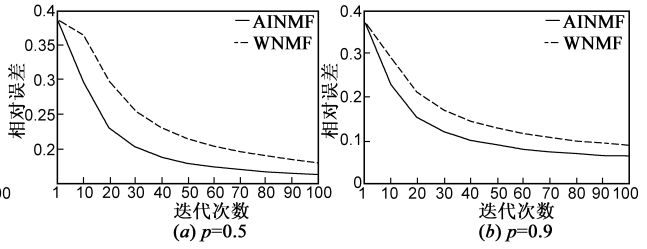
(a) T=10s (b) T=20s

图2 Yale数据集在相同运行时间下两种方法的结果比较



(a) p=0.5 (b) p=0.9

图3 ORL数据集在相同丢失概率下两种方法的结果比较



(a) p=0.5 (b) p=0.9

图4 Yale数据集在相同丢失概率下两种方法的结果比较

5 结论

本文研究了不完全非负矩阵分解(INMF),提出了 AINMF 算法.它是将求解 INMF 问题转化为交替地求解两个非负最小二乘问题.通过复杂度分析可知,AINMF 算法与 WNMF 算法有相同的时空复杂度.由于 AINMF 算法采用了精确的步长,所以 AINMF 算法的收敛速度比 WNMF 算法快.试验结果表明 AINMF 算法优于 WNMF 算法.

参考文献:

[1] W X Liu, N N Zheng, Q B You. Nonnegative matrix factorization and its applications in pattern recognition[J]. Chinese Science Bulletin, 2006, 51(1): 7 – 18.

[2] D D Lee, H S Seung. Learning the parts of objects by nonnegative matrix factorization[J]. Nature, 1999, 401(6755): 788 – 791.

[3] A Pascual-Montano, J M Carazo, K Kochik, et al. Nonsmooth nonnegative matrix factorization (nsNMF)[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2006, 28(3): 403 – 415.

[4] A Cichocki, R Zdunek, S Amari. Nonnegative matrix and tensor Factorization[J]. IEEE Signal Processing Magazine, 2008, 25(1): 142 – 145.

[5] C Fevotte, N Bertin, J L Durrieu. Nonnegative matrix factorization with the Itakura-Saito divergence with application to music analysis[J]. Neural Computation, 2009, 21(3): 793 – 830.

[6] C Ding, X He, H D Simon. On the equivalence of nonnegative matrix factorization and spectral clustering[A]. Proc. SIAM In-

ternational Conference on Data Mining[C]. Newport Beach, California, 2005. 606 – 610.

[7] W Xu, X Liu, Y Gong. Document-clustering based on non-negative matrix factorization[A]. Proc ACM SIGIR[C]. Toronto, Canada, 2003. 267 – 273.

[8] H Kim, H Park. Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis[J]. Bioinformatics, 2007, 23(12): 1495 – 1502.

[9] E J Candes, B Recht. Exact matrix completion via conve optimization[J]. Foundations of Computational Mathematics, 2009, 9(6): 717 – 772.

[10] R H Keshavan, Oh Sewoong. A Montanari. Matrix completion from a few entries[A]. Proc IEEE International Symposium on Information Theory[C]. Seoul, Korea, 2009. 324 – 328.

[11] P Paatero, U Tapper. (1994). Positive matrix factorization: a non-negative factor model with optimal utilization of error estimates of data values[J]. Environmetrics, 1994, 5(2): 111 – 126.

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

[13] 李乐, 章毓晋. 非负矩阵分解算法综述[J]. 电子学报, 2008, 36(4): 737 – 743.
Li Y, Zhang Y J. A survey on algorithms of non-negative matrix factorization[J]. Acta Electronica Sinica, 2008, 36(4): 737 – 743. (in Chinese)

[14] D D Lee, H S Seung. Algorithms for non-negative matrix factorization[A]. Advances in Neural Information Processing Systems 13[C]. Massachusetts: MIT Press, 2001. 556 – 562.

[15] Y D Kim, S Choi. Weighted nonnegative matrix factorization[A]. Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing[C]. Taipei, Taiwan, 2009. 1541 – 1544.

[16] H Lee, J Yoo, S Choi. Semi-supervised nonnegative matrix factorization[J]. IEEE Signal Processing Letters, 2010, 17(1): 4 – 7.

[17] M Merritt, Y Zhang. An interior-point gradient method for large-scale totally nonnegative least squares problems[J]. Journal of Optimization Theory and Applications, 2005, 126(1): 191 – 202.

作者简介:

史加荣　男, 1979 年生于山东东阿, 西安电子科技大学博士研究生, 主要研究方向为机器学习与模式识别.
E-mail: jiarongs3@yahoo.cn

焦李成　男, 1959 年生于陕西白水, 西安电子科技大学教授, 博士生导师, 主要研究方向为智能计算、机器学习和智能信息处理等.
E-mail: lchjiao@mail.xidian.edu.cn