Algorithm for Non-negative Matrix Factorization

论文汇报

2018.10.29

信息内容安全技术国家工程实验室

研究背景

信息时代，人们面临**分析或处理**各种**大规模数据信息的要求**，如卫星传回的大量图像、机器人接受到的实时视频流、数据库中的大规模文本、Web上的海量信息等。

研究动因

对于这些信息，矩阵是人们最常用的数学表达方式，如一幅图像就恰好与一个矩阵对应，矩阵中的每个位置存放着图像中一个像素的位置和色彩信息。由于实际问题中这样的矩阵很庞大，其中存放的信息分布往往不均匀，因此直接处理这样的矩阵效率低下，这对很多实际问题而言就失去了实用意义。

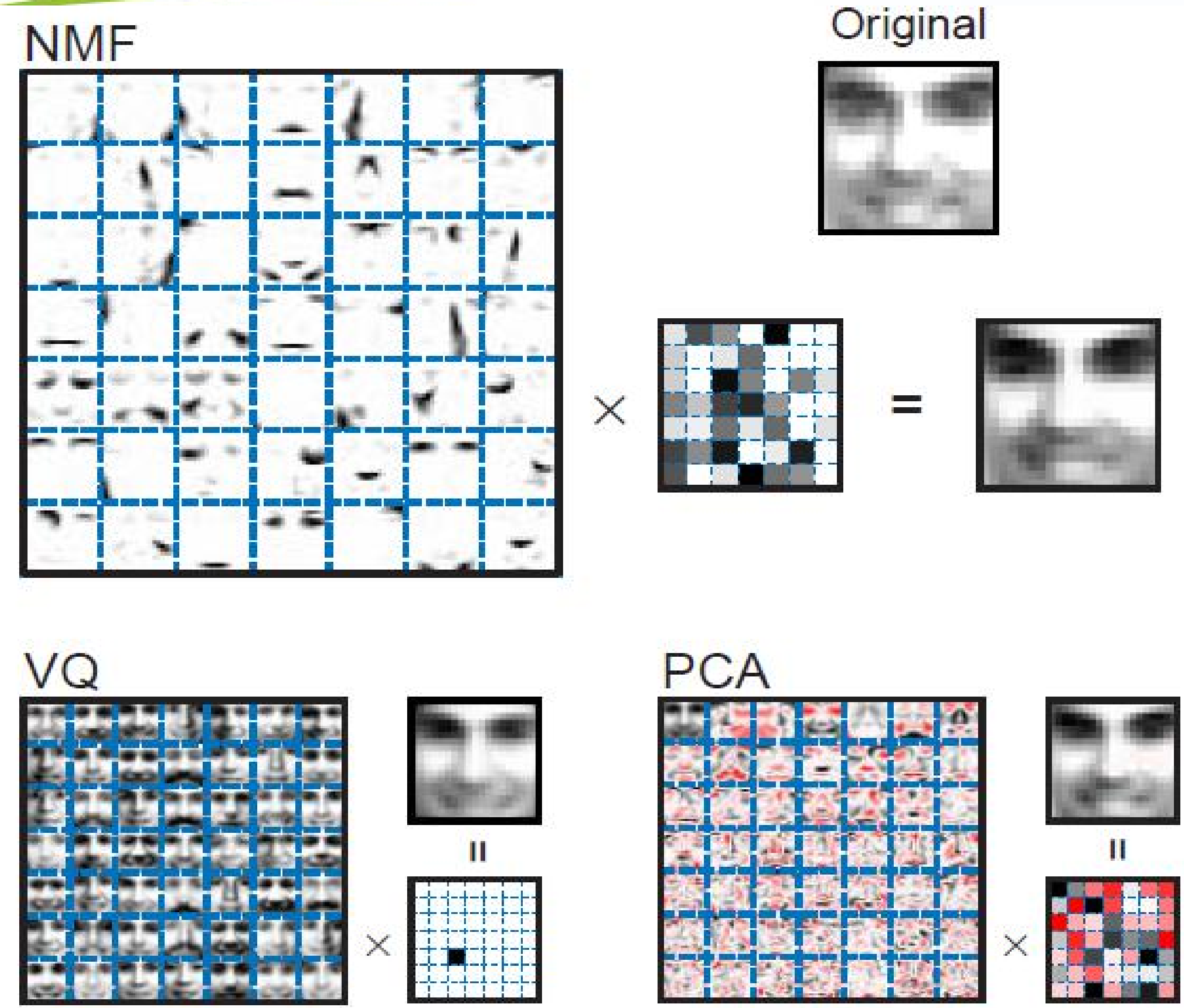
为高效处理这些通过矩阵存放的数据，一个关键的必要步骤便是对矩阵进行分解操作。通过矩阵分解，一方面将描述问题的矩阵的维数进行削减，另一方面也可以对大量的数据进行压缩和概括。

相关工作

- PCA(主成分分析)
- ICA(独立成分分析)
- SVD(奇异值分解)
- VQ(矢量量化)等。

在所有这些矩阵分解方法中，原始的大矩阵V被近似分解为低秩的 V = W*H 形式。

相关工作



相关工作

传统的秩削减算法即使输入的初始矩阵元素是全正的，也不能保证 W 和 H 的非负性。在数学上，分解结果中存在负值是正确的，但负值元素在实际问题中往往是没有意义的。

如图像数据中不可能有负值的像素点；在文档统计中，负值也是无法解释的。

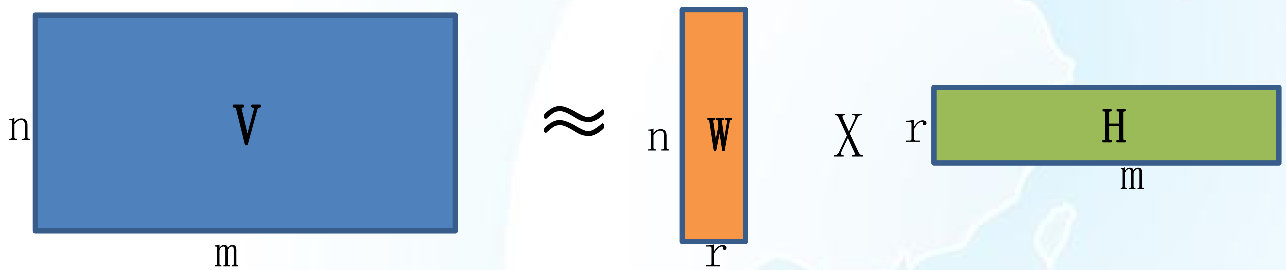
D. D. Lee和H. S. Seung对矩阵的非负分解提出了一个很好的解决方案，NMF算法。

算法介绍

目标

　　给定非负矩阵 V，找到**非负矩阵 W 和 H**，使得下式成立：

$$V \approx W\ H$$



其中$V(n\times m)$，$W(n\times r)$，$V(r\times m)$

经常的r被选定比n或m都要小，以至于W和H比原始V要小。

算法介绍

为了找一个近似的分解V≈WH，首先需要定义**损失函数J**来**量化近似的质量**。

1、矩阵A，B间欧氏距离平方差

$$J = \|A - B\|^2 = \sum_{ij}(A_{ij} - B_{ij})^2 \quad ①$$

2、矩阵A，B间散度

$$J = D(A\|B) = \sum_{ij}\left(A_{ij}\log\frac{A_{ij}}{B_{ij}} - A_{ij} + B_{ij}\right) \quad ②$$

算法介绍

尽管函数 $\|V-WH\|^2$ 和 $D(V\|WH)$ 对于两个变量(W, H)不是同时凸的，但对W或者H是凸问题，我们使用梯度下降法找损失函数的最小值。

目标函数①分别对W和H求导：

$$\frac{\partial J}{\partial W}=\frac{\partial\|V-WH\|^2}{\partial W}=-(V-WH)H^T=-VH^T+WHH^T$$

$$\frac{\partial J}{\partial H}=\frac{\partial\|V-WH\|^2}{\partial H}=-W^T(V-WH)=-W^TV+W^TWH$$

算法介绍

使用梯度更新：

$$
\begin{aligned}
W_{iu} &\leftarrow W_{iu} - \eta_{iu}\frac{\partial J}{\partial W_{iu}} \\
&\leftarrow W_{iu} - \eta_{iu}\left[-(VH^T)_{iu} + (WHH^T)_{iu}\right] \\
&\leftarrow W_{iu} + \eta_{iu}\left[(VH^T)_{iu} - (WHH^T)_{iu}\right]
\end{aligned}
$$

$$
\begin{aligned}
H_{au} &\leftarrow H_{au} - \eta_{au}\frac{\partial J}{\partial H_{au}} \\
&\leftarrow H_{au} - \eta_{au}\left[-(W^TV)_{au} + (W^TWH)_{au}\right] \\
&\leftarrow H_{au} + \eta_{au}\left[(W^TV)_{au} - (W^TWH)_{au}\right]
\end{aligned}
$$

算法介绍

加法更新策略算法流程：

输入：$V_{n\times m}$ ，隐空间维度 r，迭代次数 $iter$

输出：$W_{n\times r}$ $H_{r\times m}$

1. $W \leftarrow random\ \ nonnegative\ \ values$
2. $H \leftarrow random\ \ nonnegative\ \ values$
3. $repeat\ \ 1:iter$
4. $\quad foreach\ \ W_{iu} \in W$
5. $\qquad W_{iu} \leftarrow W_{iu} + \eta_{iu}\left[(VH^T)_{iu} - (WHH^T)_{iu}\right]$
6. $\quad end\ \ for$
7. $\quad foreach\ \ H_{au} \in H$
8. $\qquad H_{au} \leftarrow H_{au} + \eta_{au}\left[(W^TV)_{au} - (W^TWH)_{au}\right]$
9. $\quad end\ \ for$
10. $end\ \ repeat$
11. $return\ W, H$

算法介绍

下图中，左图是原图，右图是重构后的图片。
如果秩（r)和迭代次数(iter)越大，那么重构后的图越接近原图：

算法介绍

当 η 取特殊值时，梯度下降法的加法更新策略可变成如下的乘法更新策略：

$$\eta_{iu} = \frac{W_{iu}}{(WHH^T)_{iu}}$$

$$\begin{aligned}
W_{iu} &\leftarrow W_{iu} + \eta_{iu}\left[(VH^T)_{iu} - (WHH^T)_{iu}\right] \\
&\leftarrow W_{iu} + \frac{W_{iu}}{(WHH^T)_{iu}}\left[(VH^T)_{iu} - (WHH^T)_{iu}\right] \\
&\leftarrow W_{iu}\frac{(VH^T)_{iu}}{(WHH^T)_{iu}}
\end{aligned}$$

$$\eta_{\alpha\mu} = \frac{H_{\alpha\mu}}{(W^TWH)_{\alpha\mu}}$$

$$\begin{aligned}
H_{au} &\leftarrow H_{au} + \eta_{au}\left[(W^TV)_{au} - (W^TWH)_{au}\right] \\
&\leftarrow H_{au} + \frac{H_{\alpha\mu}}{(W^TWH)_{\alpha\mu}}\left[(W^TV)_{au} - (W^TWH)_{au}\right] \\
&\leftarrow H_{au}\frac{(W^TV)_{au}}{(W^TWH)_{au}}
\end{aligned}$$

算法介绍

乘法更新策略算法流程：

输入：$V_{n\times m}$ ，隐空间维度 r，迭代次数 $iter$

输出：$W_{n\times r}$ $H_{r\times m}$

1. $W \leftarrow random\ \ nonnegative\ \ values$
2. $H \leftarrow random\ \ nonnegative\ \ values$
3. $repeat\ \ 1 : iter$
4. $\quad foreach\ \ W_{iu} \in W$
5. $\qquad W_{iu} \leftarrow W_{iu} \frac{(VH^T)_{iu}}{(WHH^T)_{iu}}$
6. $\quad end\ for$
7. $\quad foreach\ \ H_{au} \in H$
8. $\qquad H_{au} \leftarrow H_{au} \frac{(W^T V)_{au}}{(W^T WH)_{au}}$
9. $\quad end\ for$
10. $end\ \ repeat$
11. $return\ W, H$

乘法规则主要是为了在计算的过程中保证非负，而基于梯度下降的加法更新方法中，加减运算无法保证W，H非负。

经严格的数学推导，此**乘法更新策略对损失函数 J 是非增的**，这意味着更新规则的重复迭代保证了**收敛到一个局部最优的矩阵分解**。

算法介绍

如果使用损失函数②来度量矩阵分解近似的质量，经过**同样的推导**可得：

$$H_{au} \leftarrow H_{au} \frac{\sum_i W_{ia} V_{iu}/(WH)_{iu}}{\sum_k W_{ka}}$$

$$W_{ia} \leftarrow W_{ia} \frac{\sum_u H_{au} V_{iu}/(WH)_{iu}}{\sum_v H_{av}}$$

例如：

$$H_{au} \leftarrow H_{au} + \eta_{au} \left[\sum_i W_{ia} \frac{V_{iu}}{(WH)_{iu}} - \sum_i W_{ia} \right]$$

令 $\eta_{au} = \frac{H_{au}}{\sum_i W_{ia}}$

可得：

$$H_{au} \leftarrow H_{au} \frac{\sum_i W_{ia} V_{iu}/(WH)_{iu}}{\sum_k W_{ka}}$$

算法介绍

损失函数②乘法更新策略算法流程：

输入：$V_{n\times m}$ ，隐空间维度 r

输出：$W_{n\times r}$ $H_{r\times m}$

1\. $W \leftarrow random\ \ nonnegative\ \ values$

2\. $H \leftarrow random\ \ nonnegative\ \ values$

3\. $repeat$

4\. $\quad foreach\ \ W_{ia} \in W$

5\. $$W_{ia} \leftarrow W_{ia} \frac{\sum_u H_{au} V_{iu}/(WH)_{iu}}{\sum_v H_{av}}$$

6\. $\quad end\ for$

7\. $\quad foreach\ \ H_{au} \in H$

8\. $$H_{au} \leftarrow H_{au} \frac{\sum_i W_{ia} V_{iu}/(WH)_{iu}}{\sum_k W_{ka}}$$

9\. $\quad end\ for$

10\. $until\ converge$

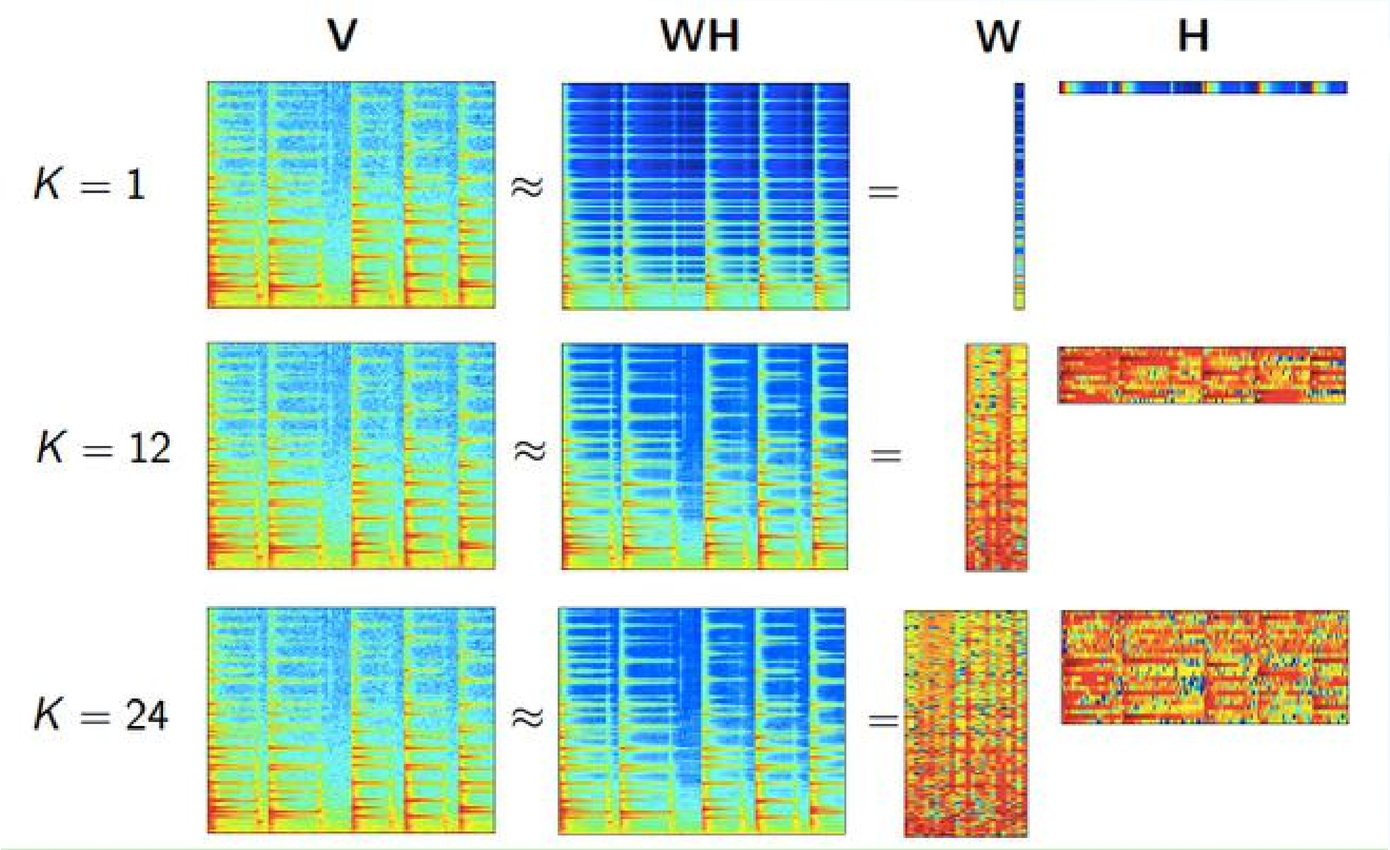
11\. $return\ W, H$

相关应用

1. 图像分析
2. 文本聚类/数据挖掘
3. 语音处理
4. 机器人控制
5. 生物医学工程和化学工程等

应用举例

下图展示NMF在做语音处理时的情形：



算法思考

思考，为啥 **损失函数** 那样定义？有什么现实意义吗？可以定义成其他形式吗？

算法思考

(1) 噪声服从高斯分布

假设噪声服从高斯分布，那么得到最大似然函数为

$$L(W,H)=\prod_{i,j}\frac{1}{\sqrt{2\pi}\sigma_{ij}}\exp\left(-\frac{E_{ij}^2}{2\sigma_{ij}}\right)$$
$$=\prod_{i,j}\frac{1}{\sqrt{2\pi}\sigma_{ij}}\exp\left(-\frac{(V_{ij}-(WH)_{ij})^2}{2\sigma_{ij}}\right)$$

取对数后，得到对数似然函数为

$$\ln L(W,H)=\sum_{i,j}\ln\frac{1}{\sqrt{2\pi}\sigma_{ij}}-\frac{1}{\sigma_{ij}}\cdot\frac{1}{2}\sum_{i,j}[V_{ij}-(WH)_{ij}]^2$$

假设各数据点噪声的方差一样，那么接下来要使得对数似然函数取值最大，只需要下面目标函数值最小。

$$J(W,H)=\frac{1}{2}\sum_{i,j}[V_{ij}-(WH)_{ij}]^2$$

(2) 噪声服从泊松分布

若噪声为泊松噪声，那么得到损失函数为

$$J(W,H)=\sum_{i,j}\left[V_{ij}\ln\frac{V_{ij}}{(WH)_{ij}}-V_{ij}+(WH)_{ij}\right]$$

Thanks