一文讓你通俗理解奇異值分解

智能算法 昨天



> **導讀：**今天，小編和大家分享一道關於推薦系統相關的面試題，如何通俗理解奇異值分解？讓我們一起來看看如何解析這道題吧。

特徵值和奇異值在大部分人的印像中，往往是停留在純粹的數學計算中。而且線性代數或者矩陣論裡面，也很少講任何跟特徵值與奇異值有關的應用背景。

奇異值分解是一個有著很明顯的物理意義的一種方法，它可以將一個比較複雜的矩陣用更小更簡單的幾個子矩陣的相乘來表示，這些小矩陣描述的是矩陣的重要的特性。就像是描述一個人一樣，給別人描述說這個人長得濃眉大眼，方臉，絡腮鬍，而且帶個黑框的眼鏡，這樣寥寥的幾個特徵，就讓別人腦海裡面就有一個較為清楚的認識，實際上，人臉上的特徵是有著無數種的，之所以能這麼描述，是因為人天生就有著非常好的抽取重要特徵的能力，讓機器學會抽取重要的特徵，SVD是一個重要的方法。

在機器學習領域，有相當多的應用與奇異值都可以扯上關係，比如做feature reduction的PCA，做數據壓縮（以圖像壓縮為代表）的算法，還有做搜索引擎語義層次檢索的LSI（Latent Semantic Indexing）

一、特徵值與奇異值

特徵值分解和奇異值分解在機器學習領域都是屬於滿地可見的方法。兩者有著很緊密的關係，接下來會談到特徵值分解和奇異值分解的目的都是一樣，就是提取出一個矩陣最重要的特徵。先談特徵值分解。

1.1 特徵值

如果說一個向量v是方陣A的特徵向量，將一定可以表示成下面的形式：$Av = \lambda v$

這時候λ就被稱為特徵向量v對應的特徵值，一個矩陣的一組特徵向量是一組正交向量。特徵值分解是將一個矩陣分解成下面的形式：

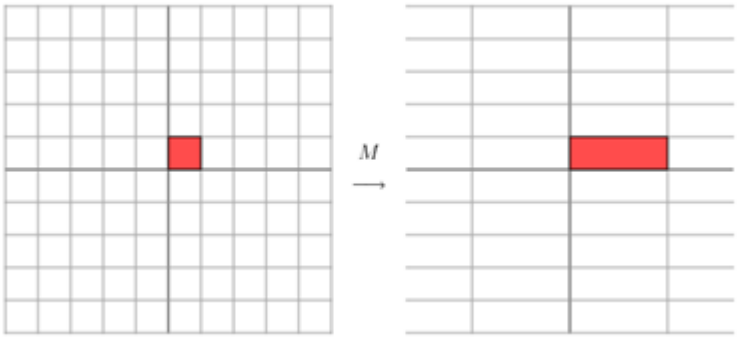
$$A = Q\Sigma Q^{-1}$$

其中Q是这个矩阵A的特征向量组成的矩阵，Σ是一个对角阵，每一个对角线上的元素就是一个特征值。我这里引用了一些参考文献中的内容来说明一下。

首先，要明确的是，一个矩阵其实就是一个线性变换，因为一个矩阵乘以一个向量后得到的向量，其实就相当于将这个向量进行了线性变换。比如说下面的一个矩阵：

$$M = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}$$

它其实对应的线性变换是下面的形式：



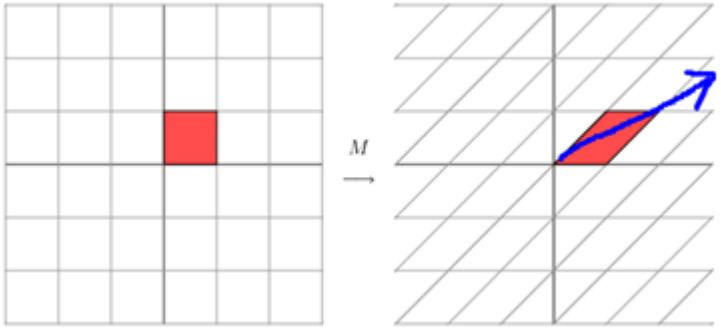
因为这个矩阵M乘以一个向量(x,y)的结果是：

$$\begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 3x \\ y \end{bmatrix}$$

上面的矩阵是对称的，所以这个变换是一个对x，y轴的方向一个拉伸变换（每一个对角线上的元素将会对一个维度进行拉伸变换，当值>1时，是拉长，当值<1时时缩短），当矩阵不是对称的时候，假如说矩阵是下面的样子：

$$M = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

它所描述的变换是下面的样子：

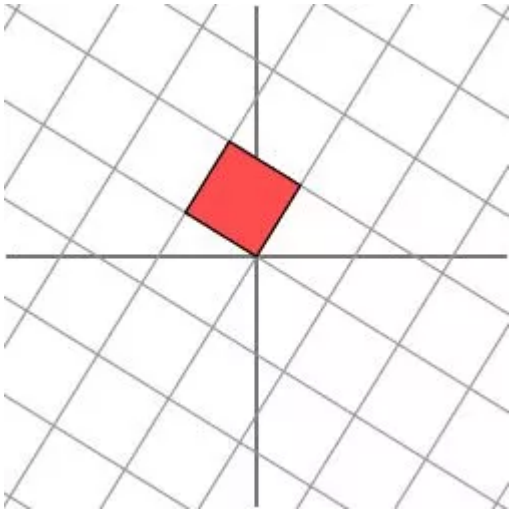


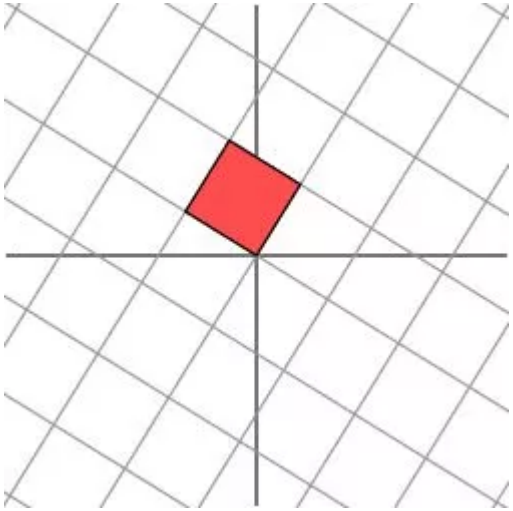
这其实是在平面上对一个轴进行的拉伸变换（如蓝色的箭头所示），在图中，蓝色的箭头是一个最主要的变化方向（变化方向可能有不止一个），如果我们想要描述好一个变换，那我们就描述好这个变换主要的变化方向就好了。反过头来看看之前特征值分解的式子，分解得到的Σ矩阵是一个对角阵，里面的特征值是由大到小排列的，这些特征值所对应的特征向量就是描述这个矩阵变化方向（从主要的变化到次要的变化排列）。

考虑更一般的非对称矩阵 $M = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$

很遗憾，此时我们再也找不到一组网格，使得矩阵作用在该网格上之后只有拉伸变换（找不到背后的数学原因是对一般非对称矩阵无法保证在实数域上可对角化，不明白也不要在意）。

我们退而求其次，找一组网格，使得矩阵作用在该网格上之后允许有拉伸变换和旋转变换，但要保证变换后的网格依旧互相垂直，这是可以做到的，如下图所示。





简言之，当矩阵是高维的情况下，那么这个矩阵就是高维空间下的一个线性变换，这个变换也同样有很多的变换方向，我们通过特征值分解得到的前N个特征向量，那么就对应了这个矩阵最主要的N个变化方向。我们利用这前N个变化方向，就可以近似这个矩阵（变换）。

也就是之前说的：提取这个矩阵最重要的特征。总结一下，特征值分解可以得到特征值与特征向量，特征值表示的是这个特征到底有多重要，而特征向量表示这个特征是什么，可以将每一个特征向量理解为一个线性的子空间，我们可以利用这些线性的子空间干很多的事情。不过，特征值分解也有很多的局限，比如说变换的矩阵必须是方阵。

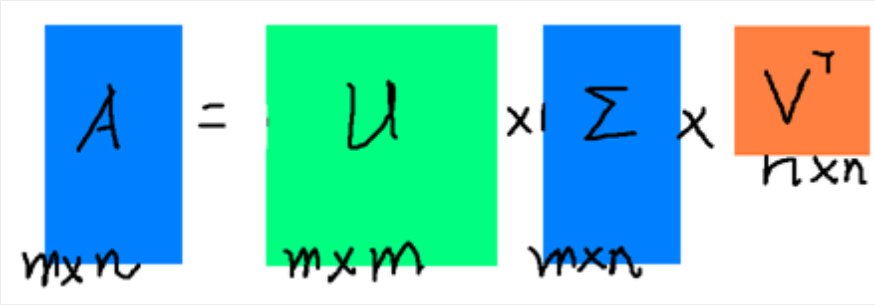
下面我们就可以自然过渡到奇异值分解的引入。

1.2 奇异值

下面谈谈奇异值分解。特征值分解是一个提取矩阵特征很不错的方法，但是它只是对方阵而言的，在现实的世界中，我们看到的大部分矩阵都不是方阵，比如说有N个学生，每个学生有M科成绩，这样形成的一个N * M的矩阵就不可能是方阵，我们怎样才能描述这样普通的矩阵呢的重要特征呢？奇异值分解可以用来干这个事情，奇异值分解是一个能适用于任意的矩阵的一种分解的方法：

$$A = U\Sigma V^T$$

假设A是一个N * M的矩阵，那么得到的U是一个N * N的方阵（里面的向量是正交的，U里面的向量称为左奇异向量），Σ是一个N * M的矩阵（除了对角线的元素都是0，对角线上的元素称为奇异值），V'(V的转置)是一个N * N的矩阵，里面的向量也是正交的，V里面的向量称为右奇异向量），从图片来反映几个相乘的矩阵的大小可得下面的图片



那么奇异值和特征值是怎么对应起来的呢？首先，我们将一个矩阵A的转置 * A，将会得到一个方阵，我们用这个方阵求特征值可以得到：

$$(A^T A)v_i = \lambda_i v_i$$

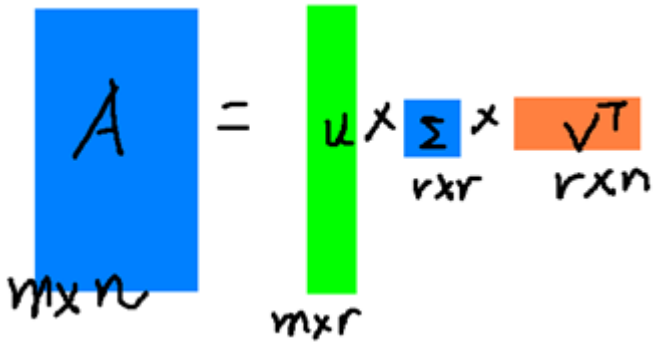
这里得到的v，就是我们上面的右奇异向量。此外我们还可以得到：

$$\sigma_i = \sqrt{\lambda_i}$$
$$u_i = \frac{1}{\sigma_i} A v_i$$

这里的σ就是上面说的奇异值，u就是上面说的左奇异向量。奇异值σ跟特征值类似，在矩阵Σ中也是从大到小排列，而且σ的减少特别的快，在很多情况下，前10%甚至1%的奇异值的和就占了全部的奇异值之和的99%以上了。也就是说，我们也可以用前r大的奇异值来近似描述矩阵，这里定义一下部分奇异值分解：

$$A_{m\times n} \approx U_{m\times r}\Sigma_{r\times r}V^T_{r\times n}$$

r是一个远小于m、n的数，这样矩阵的乘法看起来像是下面的样子：



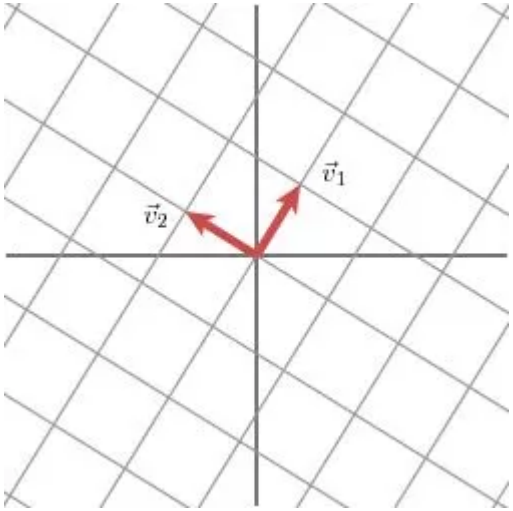
右边的三个矩阵相乘的结果将会是一个接近于A的矩阵，在这儿，r越接近于n，则相乘的结果越接近于A。而这三个矩阵的面积之和（在存储观点来说，矩阵面积越小，存储量就越小）要远远

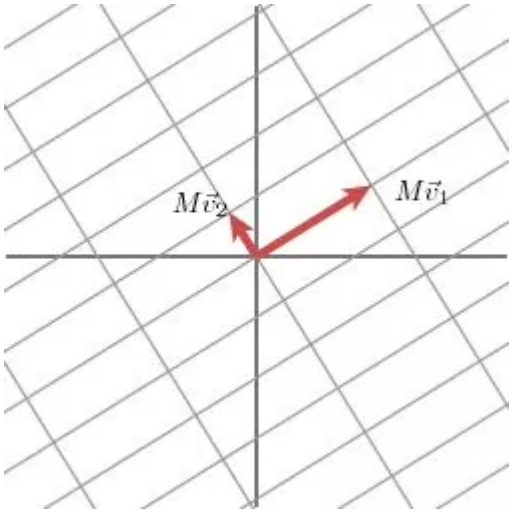
小于原始的矩阵A，我们如果想要压缩空间来表示原矩阵A，我们存下这里的三个矩阵：U、Σ、V就好了。

说句大白话，称作「奇异值」可能无法顾名思义迅速理解其本质，那咱们换个说法，称作「主特征值」，你可能就迅速了然了。

而奇异值分解的几何含义为：对于任何的一个矩阵，我们要找到一组两两正交单位向量序列，使得矩阵作用在此向量序列上后得到新的向量序列保持两两正交。

继续拿1.1节的例子进一步阐述，奇异值的几何含义为：这组变换后的新的向量序列的长度。




$M\vec{v}_2$
$M\vec{v}_1$

当矩阵M作用在正交单位向量v_2和v_2上之后，得到Mv_1和Mv_2也是正交的。令u_1和u_2分别是Mv_1和Mv_2方向上的单位向量，即$Mv_1 = \sigma_1 u_1$，$Mv_2 = \sigma_2 u_2$，写在一起就是$M\begin{bmatrix} v_1 & v_2 \end{bmatrix} = \begin{bmatrix} \sigma_1 u_1 & \sigma_2 u_2 \end{bmatrix}$，整理得：

$$M = M\begin{bmatrix} v_1 & v_2 \end{bmatrix}\begin{bmatrix} v_1^{\mathrm{T}} \\ v_2^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} \sigma_1 u_1 & \sigma_2 u_2 \end{bmatrix}\begin{bmatrix} v_1^{\mathrm{T}} \\ v_2^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} u_1 & u_2 \end{bmatrix}\begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix}\begin{bmatrix} v_1^{\mathrm{T}} \\ v_2^{\mathrm{T}} \end{bmatrix}$$

这样就得到矩阵M的奇异值分解。奇异值σ_1和σ_2分别是Mv_1和Mv_2的长度。很容易可以把结论推广到一般n维情形。

现在咱们给出一个更简洁更直观的奇异值的几何意义。先来一段线性代数的推导。

假设矩阵A的奇异值分解为

$$A = \begin{bmatrix} u_1 & u_2 \end{bmatrix}\begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} v_1^{\mathrm{T}} \\ v_2^{\mathrm{T}} \end{bmatrix}$$

其中u_1，u_2，v_1，v_2是二维平面的向量。根据奇异值分解的性质，u_1，u_2线性无关，v_1，v_2线性无关。那么对二维平面上任意的向量x，都可以表示为：$x = \xi_1 v_1 + \xi_2 v_2$。

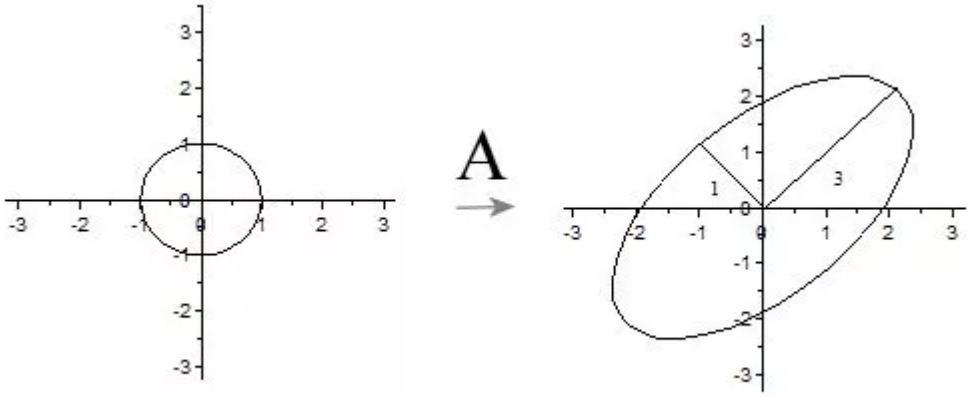
当A作用在x上时，

$$y = Ax = A\begin{bmatrix} v_1 & v_2 \end{bmatrix}\begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix} = \begin{bmatrix} u_1 & u_2 \end{bmatrix}\begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} v_1^{\mathrm{T}} \\ v_2^{\mathrm{T}} \end{bmatrix}\begin{bmatrix} v_1 & v_2 \end{bmatrix}\begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix} = 3\xi_1 u_1 + \xi_2 u_2$$

令$\eta_1 = 3\xi_1$，$\eta_2 = \xi_2$，我们可以得出结论：如果x是在单位圆$\xi_1^2 + \xi_2^2 = 1$上，那么y正好在椭圆$\eta_1^2/3^2 + \eta_2^2/1^2 = 1$上。这表明：矩阵A将二维平面中单位圆变换成椭圆，而两个奇异值正好是椭圆的两个半轴长，长轴所在的直线是$\mathrm{span}\{u_1\}$，短轴所在的直线是$\mathrm{span}\{u_2\}$。

推广到一般情形：一般矩阵A将单位球$\|x\|_2 = 1$变换为超椭球面$E_m = \{y \in \mathbf{C}^m : y = Ax,\ x \in \mathbf{C}^n,\ \|x\|_2 = 1\}$

那么矩阵A的每个奇异值恰好就是超椭球的每条半轴长度。



奇异值的计算是一个难题，是一个O(N^3)的算法。在单机的情况下当然是没问题的，matlab在一秒钟内就可以算出1000 * 1000的矩阵的所有奇异值，但是当矩阵的规模增长的时候，计算的复杂度呈3次方增长，就需要并行计算参与了。Google的吴军老师在数学之美系列谈到SVD的时候，说起Google实现了SVD的并行化算法，说这是对人类的一个贡献，但是也没有给出具体的计算规模，也没有给出太多有价值的信息。

其实SVD还是可以用并行的方式去实现的，在解大规模的矩阵的时候，一般使用迭代的方法，当矩阵的规模很大（比如说上亿）的时候，迭代的次数也可能会上亿次，如果使用Map-Reduce框架去解，则每次Map-Reduce完成的时候，都会涉及到写文件、读文件的操作。个人猜测Google云计算体系中除了Map-Reduce以外应该还有类似于MPI的计算模型，也就是节点之间是保持通信，数据是常驻在内存中的，这种计算模型比Map-Reduce在解决迭代次数非常多的时候，要快了很多倍。

Lanczos迭代就是一种解对称方阵部分特征值的方法（之前谈到了，解A'* A得到的对称方阵的特征值就是解A的右奇异向量），是将一个对称的方程化为一个三对角矩阵再进行求解。按网上的一些文献来看，Google应该是用这种方法去做的奇异值分解的。请见Wikipedia上面的一些引用的论文，如果理解了那些论文，也"几乎"可以做出一个SVD了。

二、奇异值的直观应用

2.1 女神图片压缩

下面，咱们从女神上野树里（Ueno Juri）的一张像素为高度450*宽度333的照片，来直观理解奇异值在物理上到底代表什么意义（请屏幕前的痴汉暂停舔屏）。

我们都知道，图片实际上对应着一个矩阵，矩阵的大小就是像素大小，比如这张图对应的矩阵阶数就是450*333，矩阵上每个元素的数值对应着像素值。我们记这个像素矩阵为A 现在我们对矩阵A进行奇异值分解。直观上，奇异值分解将矩阵分解成若干个秩一矩阵之和，用公式表示就是：

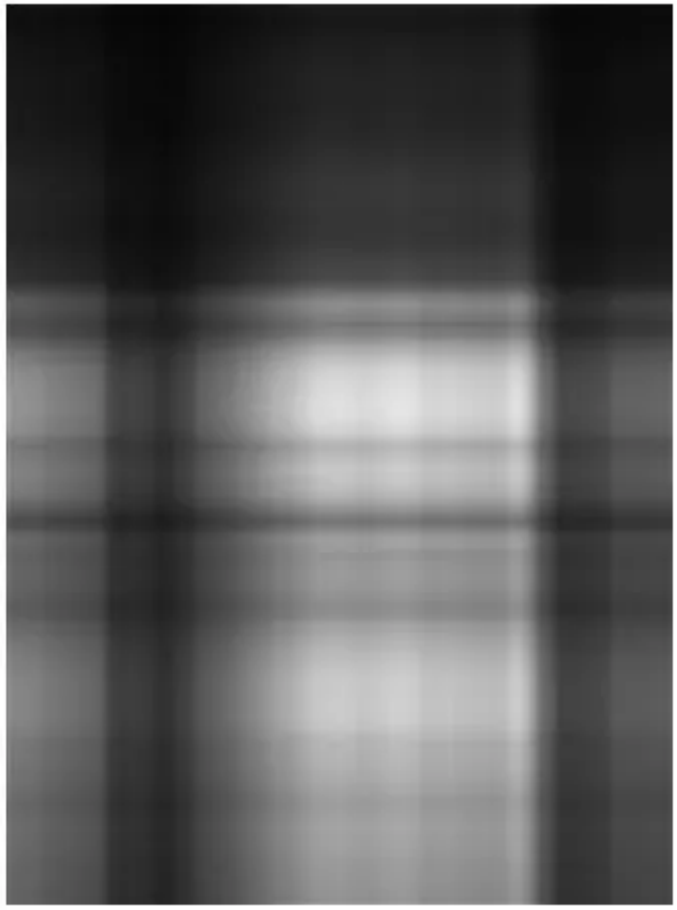
$$A = \sigma_1 u_1 v_1^{\mathrm{T}} + \sigma_2 u_2 v_2^{\mathrm{T}} + \ldots + \sigma_r u_r v_r^{\mathrm{T}} \tag{1}$$

其中等式右边每一项前的系数σ就是奇异值，u和v分别表示列向量，秩一矩阵的意思是矩阵秩为1。注意到每一项uv^{T}都是秩为1的矩阵。我们假定奇异值满足（奇异值大于0是个重要的性质，但这里先别在意）

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$$

如果不满足的话重新排列顺序即可，这无非是编号顺序的问题。既然奇异值有从大到小排列的顺序，我们自然要问，如果只保留大的奇异值，舍去较小的奇异值，这样(1)式里的等式自然不再成立，那会得到怎样的矩阵——也就是图像？

令$A_1 = \sigma_1 u_1 v_1^{\mathrm{T}}$，这只保留(1)中等式右边第一项，然后作图：



结果就是完全看不清是啥……我们试着多增加几项进来：

$$A_5 = \sigma_1 u_1 v_1^{\mathrm{T}} + \sigma_2 u_2 v_2^{\mathrm{T}} + \sigma_3 u_3 v_3^{\mathrm{T}} + \sigma_4 u_4 v_4^{\mathrm{T}} + \sigma_5 u_5 v_5^{\mathrm{T}}$$

再作图

隐约可以辨别这是短发伽椰子的脸……但还是很模糊，毕竟我们只取了5个奇异值而已。下面我们取20个奇异值试试，也就是(1)式等式右边取前20项构成A_{20}

虽然还有些马赛克般的模糊，但我们总算能辨别出这是Juri酱的脸。当我们取到(1)式等式右边前50项时：

我们得到和原图差别不大的图像。也就是说当k从1不断增大时，A_k不断的逼近A。让我们回到公式

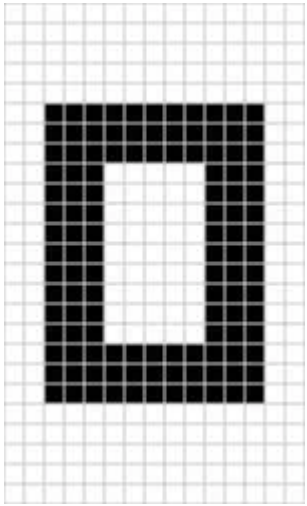
$$(1) \qquad A = \sigma_1 u_1 v_1^{\mathrm{T}} + \sigma_2 u_2 v_2^{\mathrm{T}} + \ldots + \sigma_r u_r v_r^{\mathrm{T}}$$

矩阵A表示一个450*333的矩阵，需要保存$450 \times 333 = 149850$个元素的值。等式右边u和v分别是450*1和333*1的向量，每一项有1+450+333=784个元素。如果我们要存储很多高清的图片，而又受限于存储空间的限制，在尽可能保证图像可被识别的精度的前提下，我们可以保留奇异值较大的若干项，舍去奇异值较小的项即可。例如在上面的例子中，如果我们只保留奇异值分解的前50项，则需要存储的元素为$784 \times 50 = 39200$，和存储原始矩阵A相比，存储量仅为后者的26%。

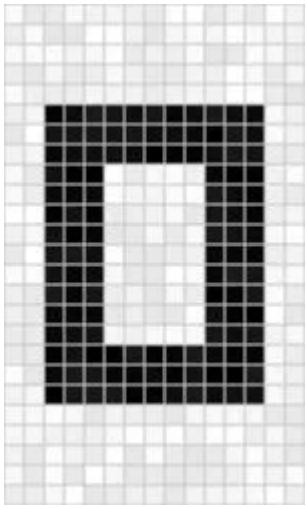
奇异值往往对应着矩阵中隐含的重要信息，且重要性和奇异值大小正相关。每个矩阵A都可以表示为一系列秩为1的“小矩阵”之和，而奇异值则衡量了这些“小矩阵”对于A的权重。

2.2 图像去噪

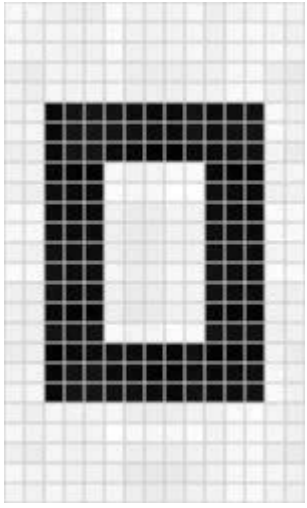
在图像处理领域，奇异值不仅可以应用在数据压缩上，还可以对图像去噪。如果一副图像包含噪声，我们有理由相信那些较小的奇异值就是由于噪声引起的。当我们强行令这些较小的奇异值为0时，就可以去除图片中的噪声。如下是一张25*15的图像



但往往我们只能得到如下带有噪声的图像（和无噪声图像相比，下图的部分白格子中带有灰色）：



通过奇异值分解，我们发现矩阵的奇异值从大到小分别为：14.15，4.67，3.00，0.21，……，0.05。除了前3个奇异值较大以外，其余奇异值相比之下都很小。强行令这些小奇异值为0，然后只用前3个奇异值构造新的矩阵，得到



可以明显看出噪声减少了（白格子上灰白相间的图案减少了）。奇异值分解还广泛的用于主成分分析（Principle Component Analysis，简称PCA）和推荐系统（如Netflex的电影推荐系统）等。在这些应用领域，奇异值也有相应的意义。

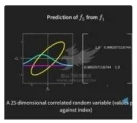
参考文献

1 https://www.cnblogs.com/LeftNotEasy/archive/2011/01/19/svd-and-applications.html

2 https://www.zhihu.com/question/22237507 3 We Recommend a Singular Value Decomposition (Feature Column from the AMS)

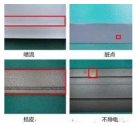
编辑 ∑Pluto

来源：七月算法



缺陷检测比赛Top3方案分享

AI约读社



形象易懂的傅里叶变换、短时傅里叶变换和小波变换

極市平台