

# Vahadane论文复现report

我的代码在<https://github.com/xindubawukong/Vahadane>

## 方法概述

这个方法的主要流程是：

1 把输入图片（source和target）转换为  $V = \ln \frac{I_0}{I}$  的形式

2 SNMF-Based Stain Separation，将V转换为W\*H的形式，包括两步：

(1) Dictionary learning确定W，调用spams.trainDL

(2) Sparse decomposition确定H，调用spams.lasso

3 SPCN，根据Ws, Hs, Wt, Ht计算出V，进而得到结果

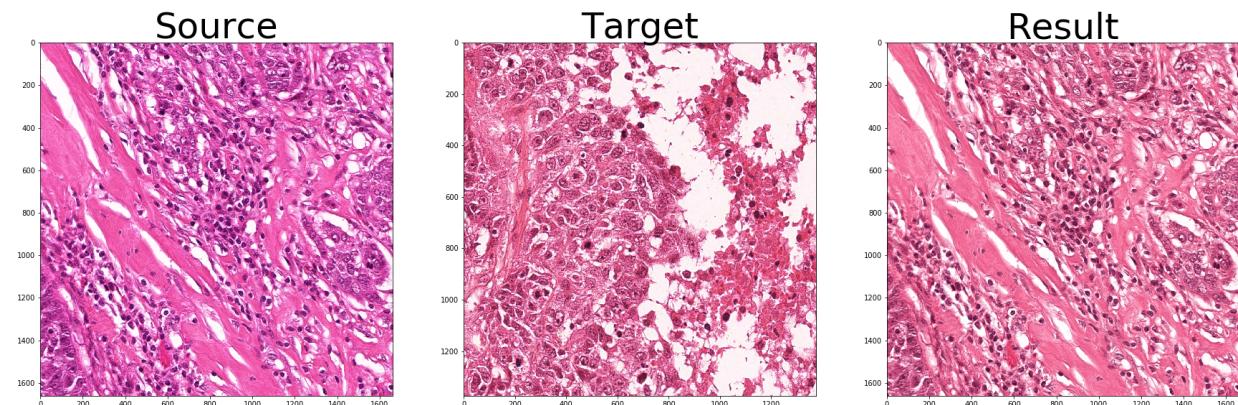
## 效果展示

目前效果最好的参数设置是：

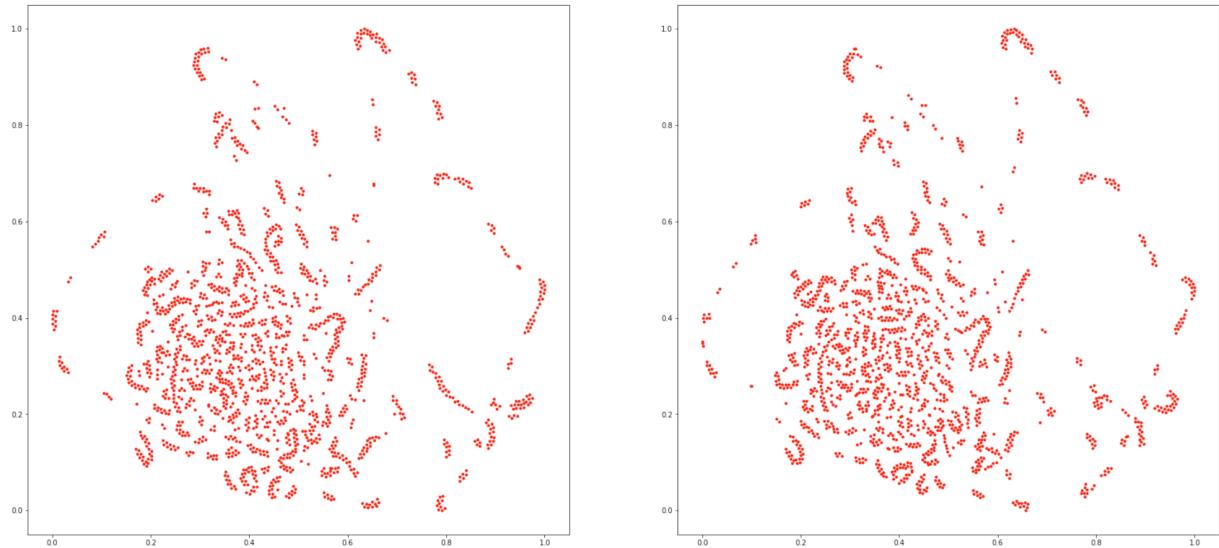
第1步里设图片亮度的 $threshold = 0.9$

SNMF中 $\lambda = 0.01$

目前的效果：

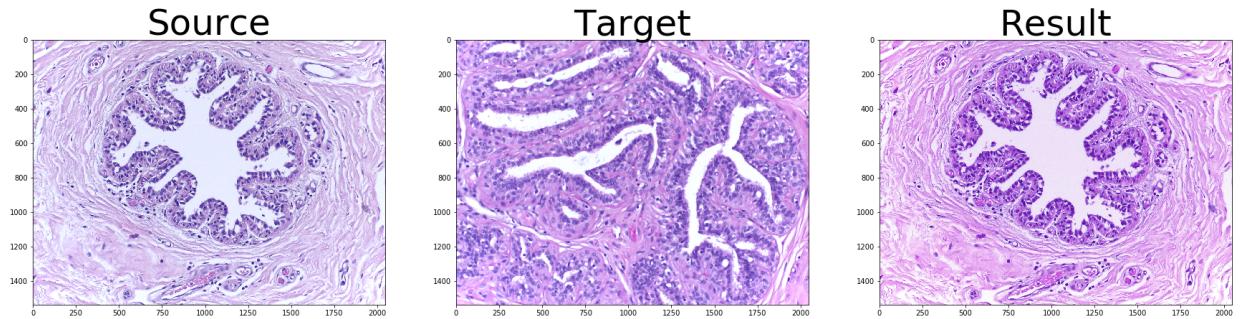


对source和result进行TSNE之后的结果：

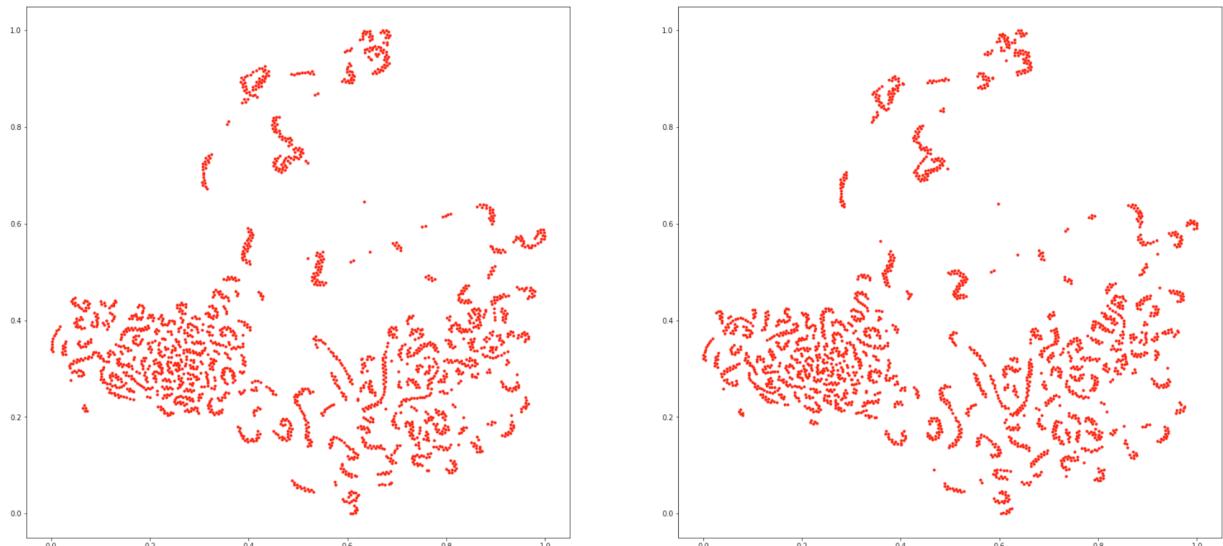


两者的TSNE图差不多，说明组织结构保存的比较完整。

另一个结果：

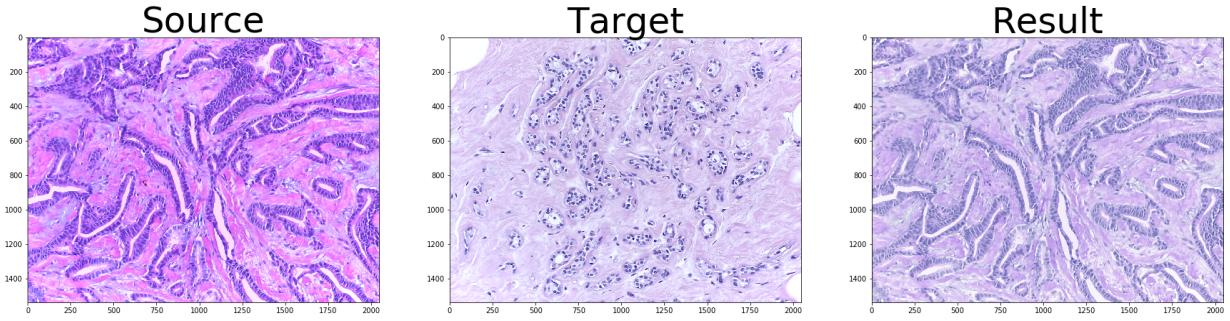


这个source和result的TSNE图：



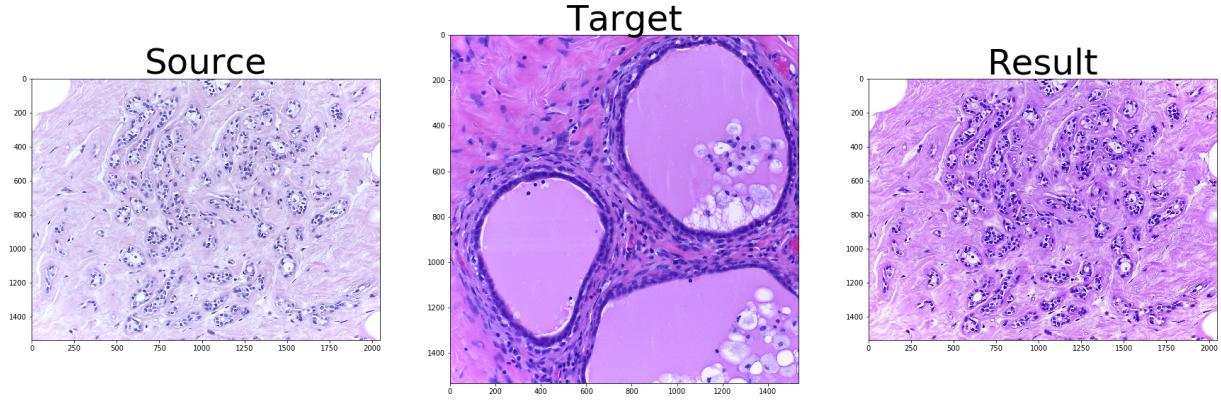
有些result的TSNE结果，先归一化到[0,1]中，需要绕(0.5,0.5)旋转一下才能与source对齐。

上面是结果比较好的，也有一些效果不太好：



这种在normalization之后不同的组织颜色比较接近，不如原来的图清晰。目前认为这跟target本身染色就不太清楚有关，target选颜色较鲜艳的时效果好。

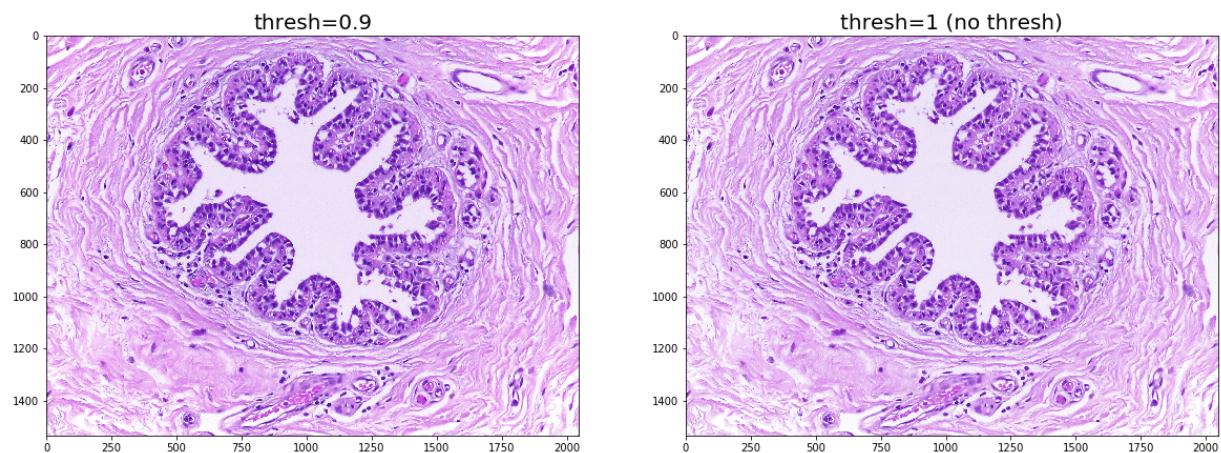
目前手动运行了约100组不同的source和target，还没有批量跑，因为自己电脑性能问题。其他的一些结果：



## 参数对效果的影响

首先，在读入图片时，因为图片可能本身色彩偏弱（RGB值普遍比较小），代码里在读入图片时进行了颜色增强，将颜色值的90%归一化到255。

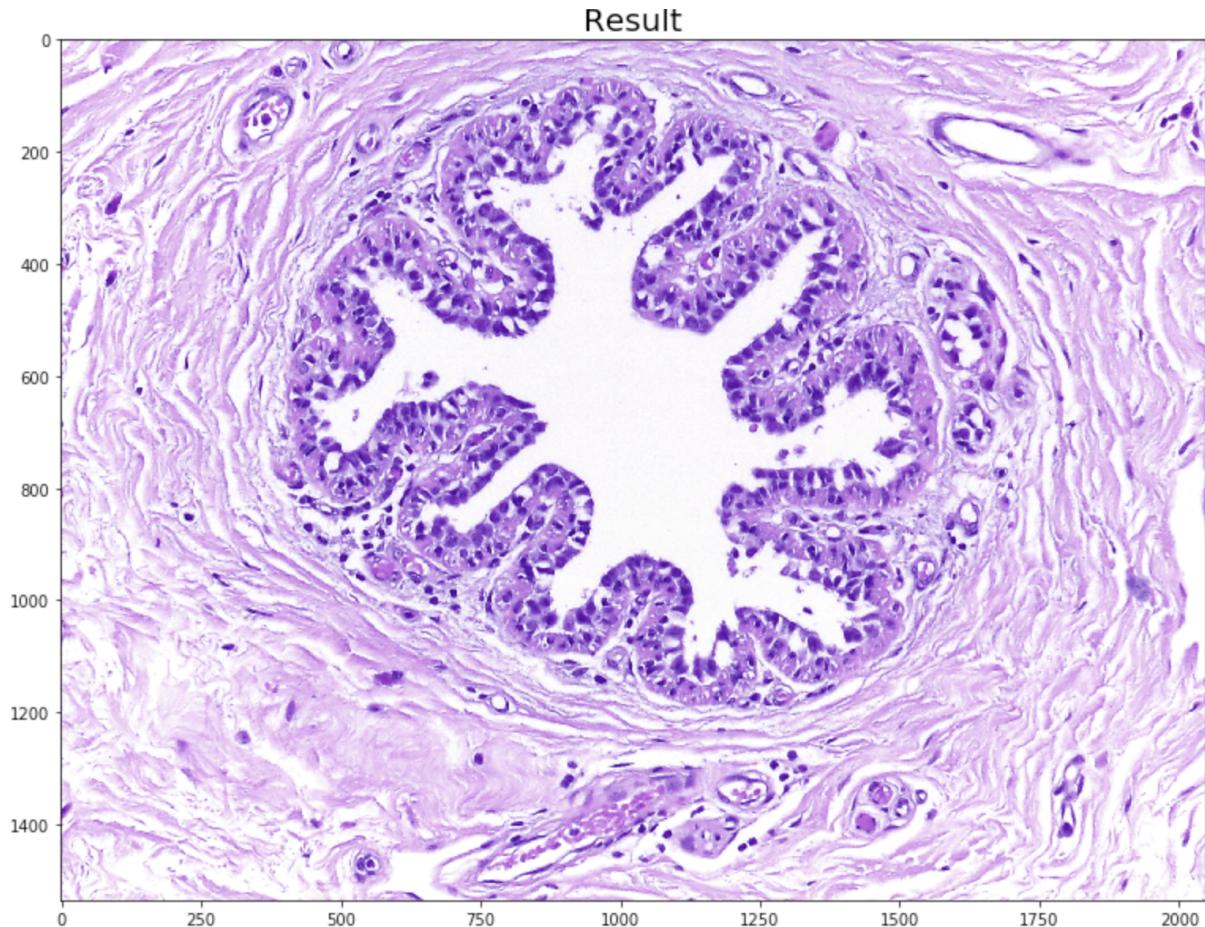
然后根据论文中的做法，计算V时，将Lab颜色空间中的亮度 $L > \text{threshold}$ 的像素点删去，防止气泡等出现。 $\text{threshold}$ 设为0.8-0.9之间效果较好。



Threshold不设置时，得到的图片对比度不好，中间部分细胞核颜色应该向左边一样比周围更深一些。

$\lambda$ 的调整比较玄学。论文中的结果是 $\lambda=0.02$ ，不过我尝试后感觉 $\lambda=0.01$ 更好一些。另外理论上spams.trainDL和spams.lasso两步需要用相同的 $\lambda$ ，但是实际上前者的 $\lambda$ 略大于后者时效果好。

$\lambda$ 主要是为了得到稀疏的H而使用的。 $\lambda$ 太大上面的图会变成这样：



感觉就像只用紫色染色剂染了色。L1范数限制了H的稀疏性，如果 $\lambda$ 太大，就会导致H中太多0，结果失真。

另外，spams.trainDL中有一个iter参数可以控制迭代次数，默认情况下是迭代1秒钟。实际上可以控制iter=50到100之间就可以，再多迭代结果基本不变了。iter=20时效果也不错。iter=10时就不太好看了。

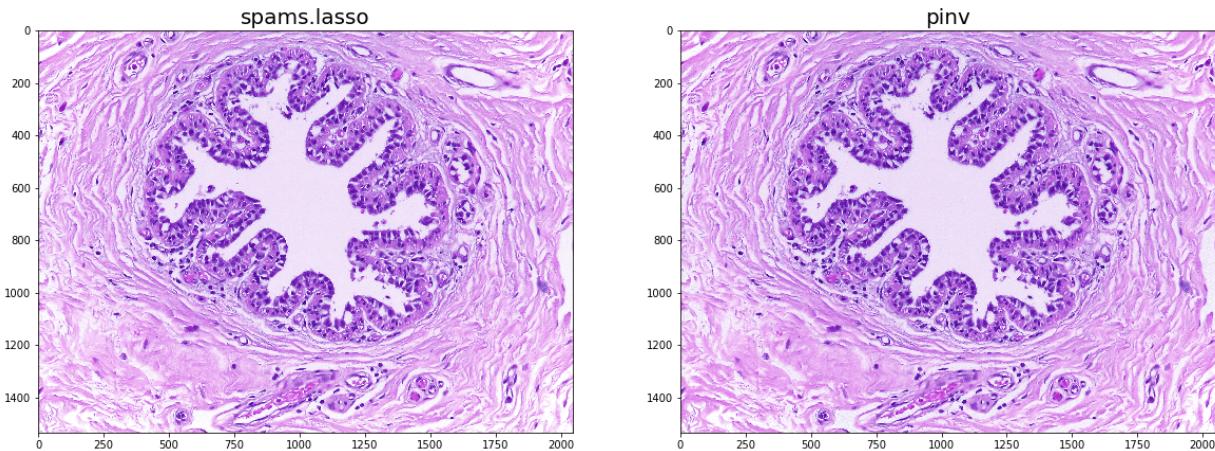
## 运行速度

分解一张1500\*2000的图片在我的机子上需要1s左右。时间主要浪费在求W和H上，SPCN的时间可以忽略。（Demo运行的很慢是因为要用plt展示图片并且保存）

论文中介绍了一种快速算法，方法是在原始图片中取若干个小的切片，对每一个切片分别求出 $W_i$ ，最后对所有 $W_i$ 取平均值得到W，然后直接通过color deconvolution ( $H = (W^T W)^{-1} W^T V, H \geq 0$ ) 求出H。

实验结果是，通过切片得到的W与与普通的算法相差2%左右。但是因为我的电脑的原因，在数据集里的小图片上跑的很慢，图片太小显示不出算法的优越性。我把64张相同的图片拼成了一个大图片（ $12000 * 8000$ ），在这个图片上跑fast算法运行时间是普通算法的 $1/3$ 。论文里展示效果用的图片是 $20000 * 20000$ 的，我的电脑跑不了了已经。并且，fast算法的另一个特点是适合并行计算，不同的切片可以多核运行。

但是用color deconvolution代替spams.lasso，虽然速度提升很大，但是得到的图片质量有问题：



可能看的不是很明显，但是右边的图片中间本来应该是白色的部分有一些灰点，整个图片像加了一层灰色。



就像红圈的这两个地方，本来应该是白的，但是现在有一点灰。不过整体看上去还可以。

我没有发现其他效果更好的回归算法。。可能是我不知道有哪些。。另外spams.lasso中的一个参数 max\_length\_path设的小一点可以加快一些速度，这个参数默认是 $4 * p = 8$ ，设成3或4可以略微加快一些，但是分解精度下降，所以不好用。求H矩阵可以对每个像素点分开做，就是求每个点两种染料的配比，对每个点的式子是这样的：

$$\min_{h1, h2} : \frac{1}{2} * \left\| \begin{bmatrix} R \\ G \\ B \end{bmatrix} - \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix} * \begin{bmatrix} h1 \\ h2 \end{bmatrix} \right\|_{F^2}^2 + \lambda(|h1| + |h2|)$$

RGB，W和lambda都是已知的。我觉得肯定有更好的做法，比如可以训练一个神经网络？目前不知道怎么做。。求伪逆的方法得到的不是最优解。

另外，对target进行分解时最好不要用fast算法，毕竟target只需要分解一次，最好精确一点好。