生物信息学
-对于双聚类算法的研究

月光不染是非
指导老师：孔祥真老师

2020 年 10 月 8 日

目录

1 前言

本笔记的作用主要在于记录生物信息学双聚类的学习以及实验内容，方便日后自己的学习与回顾，同时希望能够帮助到相同方向的人进行快速学习与认识。

2 Acomparative analysis of biclustering algorithms for gene expression data

2.1 题译

不同的双聚类算法在基因表达数据上对比分析

2.2 个人观点

这些大量的数据集的出现正在推动着更复杂以及性能更为优秀的数据分析类算法进步与发展

2.3 摘要

文章选取了 12 种的双聚类算法进行对比分别为
BBC、BiMax、Cheng and Church、COALESCE、CPB、ISA、OPSM、QUBIC、Spectral、xMOTIFs、Plaid、FABIA

聚类是研究这些大型数据的一种十分有效地方法。聚类算法根据相似度度量最大化组内相似度，或最小化组间相似度寻求将对象划分为聚类。

2.4 介绍

2.4.1 Cheng and Church 算法

该算法从整个数据矩阵开始，去除具有高残差（观察值与估计值（拟合值）之间的差）的行和列。当双聚类的 MSR，即

$$MSR = \frac{1}{|I||J|} \sum_{i \in I, j \in J} (a_{i_j} - a_{i_J} - a_{I_j} + a_{I_J})^2$$

达到给定的阈值参数 d 时，残差比双聚类残差小的行和列被添加回双聚类。如果要恢复多个聚类，则使用随机值掩码找到的聚类，然后对如上过程进行重复。

2.4.2 Order-preserving submatrix problem

保守基因的表达基序

Order-preserving submatrix problem 通常被简写作 OPSM，OPSM 是一种确定性贪婪算法，它寻找行有序的双矩阵。OPSM 模型将一个双聚类定义为一个保持一定顺序的子矩阵，在这个子矩阵中存在一个列的线性顺序，其中该子矩阵的所在行的表达式值呈线性增长。结果表明常数列的移动、缩放和移动-缩放在双聚类模型中都是有序保持的。OPSM 通过迭代地生长部分的双聚类来构建完整的双聚类，并根据每个双聚类生长到某个固定目标大小的概率对其进行评分。在每次迭代中只保留部分最好的聚类。

2.4.3 xMOTIFs

xMOTIFs 是一种不确定性的贪婪向的双聚类算法，它在离散数据集中寻找保存行的双聚类。对于每一行，根据离散状态的间隔与均匀分布，接着通过统计显著性来确定离散状态的间隔。对于每一个随机选择的列（称为种子）和每一个随机选择的列集（称为鉴别集），xMOTIFs 尝试寻找在种子和鉴别集的列上具有相同状态的行。因此，xMOTIFs 可以找到行中带有的聚类的常量值。

表 1: 变量对应内容

Letter	implication
X_{i_j}	数据元素
K	There are K clusters
y	Background effect
m	Clustering effect
a	The sum of the rows is also called effect a
b	Effect of the column
e	stochastic noise

2.4.4 Plaid

Plaid 拟合参数到一个数据生成模型，称为 Plaid 模型：一个数据元素 Xij，假设存在 K 个双聚类，即被生成为一个背景效应 y，聚类效应 m，行的和则为效果 a，列效果为 b 和随机噪声 e

$$X_{i_j} = \theta + \sum_{k=1}^{K} (\mu_k + \alpha_{i_k} + \beta_{j_k}) \rho_{i_k} \kappa_{j_k} + e_{i_j}$$

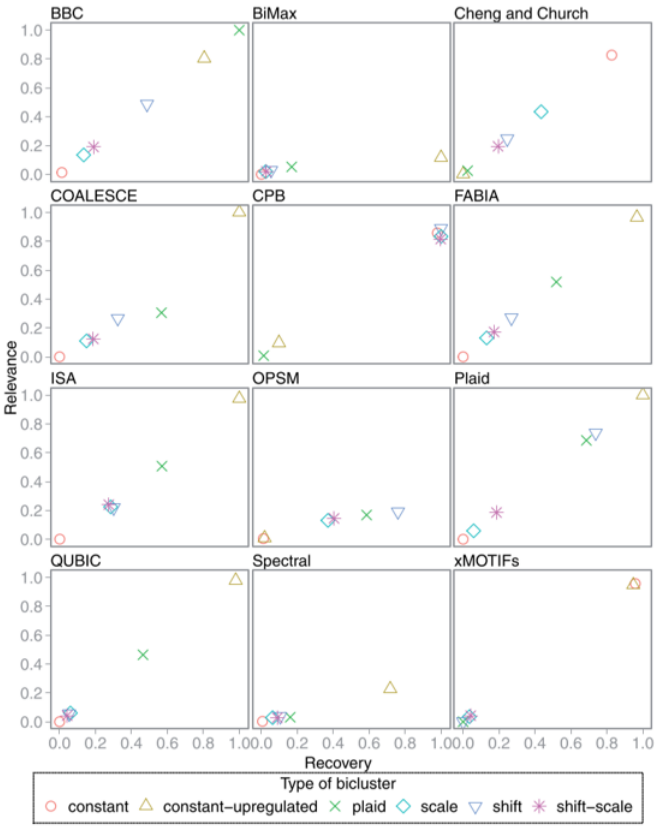
其中，背景是指不属于任何双聚类的任何矩阵元素。Plaid 算法通过迭代更新模型的各个参数来拟合该双聚类模型，使在建模数据和真实数据之间的最小均方误差最小化。

2.5 实验

必须使用两种方法来评估双聚类基因表达数据的结果，因为真正的双聚类是未知的。两种方法分别为：内部方法和外部方法。内部有效性的度量是基于数据和双峰本身的固有属性，而外部度量是将双峰与其他信息来源进行比较。由于我们比较了大量算法，每种算法都适合不同的模型，因此我们选择通过为每个双曲线行计算 GO 富集来使用外部验证。富集分析使用 GOstats 软件包。如果任何基因本体项的调整后的 P 值小于 P□0.05，则认为该值已丰富。这项工作中使用的大多数 GDS 数据集都丢失了价值。这些缺失值使用 pcaMethods 软件包提供的 PCA 插补替换。

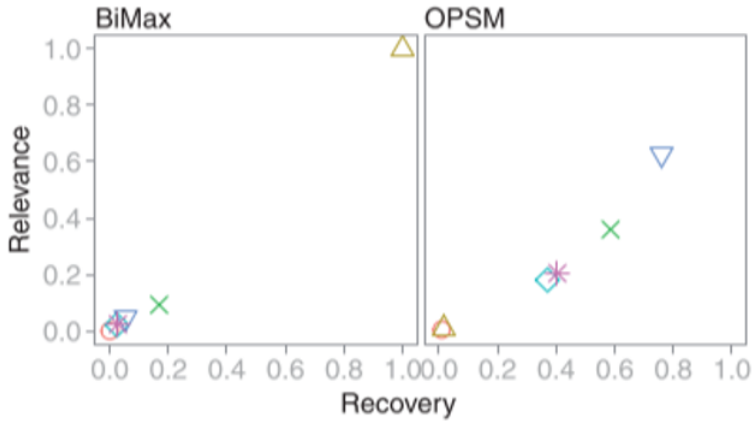
2.5.1 结果对比图-1

纵坐标值表示聚类算法关联程度比
横坐标值表示聚类算法恢复程度比



2.5.2 对比-2

过滤后的双聚类模型实验结果。每个数据点代表 20 个数据集的平均恢复与相关性得分。其中（1,1）得分最好。



2.6 实验结论-1

结果模型实验在先前的比较分析研究中，对从单个模型生成的人工数据进行了算法比较。但是，每种算法都适合于不同的双聚类模型。在单个模型上比较算法通常会得出不完整或误导的结果。取而代之的是，我们在由六个不同模型生成的合成数据集上评估了每个二元组：常量，常量上调，移位，比例，移位比例和格子。生成了六个模型中每个模型的二十个数据集，并且在每个数据集上对每种二类聚类方法进行了评分。结果比对图给出了每个模型的所有 20 个数据集的平均回收率和相关性得分的图。BiMax 和 OPSM 不会过滤其结果，因此这两个算法都会返回许多虚假的二元组，从而损害了他们的相关性得分。但是，我们的实验包 BiBench 提供了双聚类过滤功能，该功能可根据大小并与其他双聚类重叠来删除双聚类。当一对双簇之间的重叠度至少为 25%时，将较小的一个过滤掉后，它们的相关性得分将大大提高。并非所有算法都预期表现良好。

2.7 实验结论-2

在所有数据集上。大多数算法都能够恢复适合其模型的双聚类，但是也有一些例外。BBC 的结果对其使用哪种标准化的程序很敏感。根据选择的程序，它能够在恒定，恒定上调，平移和格子双聚类上获得完美的分数。我们选择使用 IQRN 归一化，以最大化其在格子模型双簇上的性能。期望 Cheng and Church 找到任何具有恒定表达值的双峰，但找不到上调的恒定双峰。我们假设对具有较大表达值的行和列进行了早期修剪，因为它们增加了候选双聚类的 MSR。

3 Discovery of bidirectional contiguous column coherent bicluster in time-series gene expression data

3.1 题译

在时序基因表达数据中发现双向连续列的双聚类

3.2 关键词

Bicluster 、Time series-Gene expression 、Coherent evolution 、Frequent sequential mining 、Bitwise operation

3.3 什么是时序基因表达数据?

时序基因表达数据根据细胞循环过程, 在不同时间点对各基因采集相关数据。

目前, 虽然已有多种算法可以对时序基因表达数据进行聚类分析 (如 k 均值聚类方法、层次聚类方法, 基于统计模型的聚类方法等), 但这些方法通常把时序基因表达数据看作是普通的多维空间向量, 数据中, 时间上的自关联信息完全被忽视, 不能有效影响到聚类的最终结果 (就是聚类结果与时间无关)

3.4 摘要

- □ 本文在 CCC-Bicluster(contiguous column coherent bicluster) 的基础上提出了 BCCC-Bicluster 算法.
- □ 提出了一种基于频繁顺序挖掘的精确的数据挖掘算法，以找到所有最大的 BCCC-Bicluster。
- □ 新定义的频繁不频繁树数组 (FITA) 可以加快遍历过程，其有用的策略源自 Apriori 属性，以避免重复工作.

□ 为了使其更有效，应用了按位运算 XOR 来捕获两行之间相同或相反的连续模式。

□ 实验结论：与 CCC-Bicluster 相比，该算法在恢复合成数据中的双峰方面具有更好的性能，在速度和可扩展性方面均优于无 FITA 的算法。在富集分析中，已证明 BCCC-Bicluster 比其他三种最新的双聚类更能发现与生物过程有关的重要 GO 特征（文中应该表达的是特征的意思）

3.5 实验涉及数据集

模拟（仿真）数据，酵母微阵列数据和人类微阵列数据

Yeast Microarray Data Yeast microarray data are a collection of 2884 genes under 17 contiguous time points

3.6 相关设定

矩阵 d，在图中所示，行表示基因，列表示不同的时间点，不同的环境条件，不同的器官或甚至不同的个体

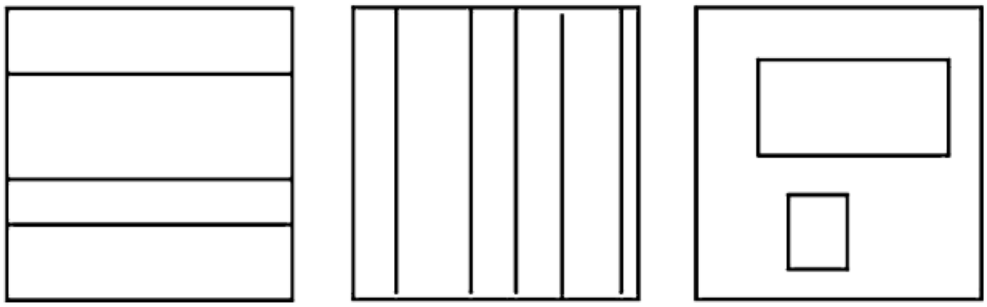
Fig. 1 Gene expression data arranged in a matrix D

$$\begin{pmatrix} d_{11} & d_{12} & \cdots & d_{1m} \\ d_{21} & d_{22} & \cdots & d_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & \cdots & d_{nm} \end{pmatrix}$$

3.7 思路

3.7.1 从最基础的双聚类思路出发，一样参考了 Cheng and Church 最开始的双聚类思路

基因可以参与多种生物学功能，因此可以包含在多个聚类中。提出了群聚类或子空间集群化以克服了传统聚类的问题。Biclustering 对数据矩阵的行和列执行同时聚类，搜索连贯表达的基因和条件的子集。



聚类和双聚类的区别

表 2: D 矩阵的相关信息

Letter	implication
R	表示行
C	表示列

3.7.2 Problem statement

Fig. 1 Gene expression data arranged in a matrix D

$$\begin{pmatrix} d_{11} & d_{12} & \cdots & d_{1m} \\ d_{21} & d_{22} & \cdots & d_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & \cdots & d_{nm} \end{pmatrix}$$

矩阵 d

$$D_{m*n} = (R, C)$$

令 d_{i_j} 表示 D 在第 i 行和第 j 列中的内容，即代表基因 i 在时间 j 的表达水平。
假定给定的双聚类中的所有基因都受到调控（向上或向下）。在这项工作中，我们只考虑基因活动随时间的上调和下调。

由于假设二元簇中的行在连续列的子集中是相关的或反相关的，所以计算每两个连续列之间的列差，以便标识此列子集。
以此将矩阵 D 简化为二进制矩阵

如下：

$$d'_{ij} = \begin{cases} 1, di, j+1-dij \geq 0 \\ 0, di, j+1-dij < 0 \end{cases}$$

由此得到一个新的矩阵 D'
d_{i_j} 则表示 D 中元素

$d'_{ij}\left(d'_{ij}\int\{0,\ 1\}\right)$ 代表基因 i 从时间 j 到时间（j+1）的表达变化快速按位运算的 d_{i_j}' 等于 1 表示上调，而等于 0 表示下调

3.7.3 四个定义

CCC-Bicluster

ACCC-Bicluster

BCCC-Bicluster

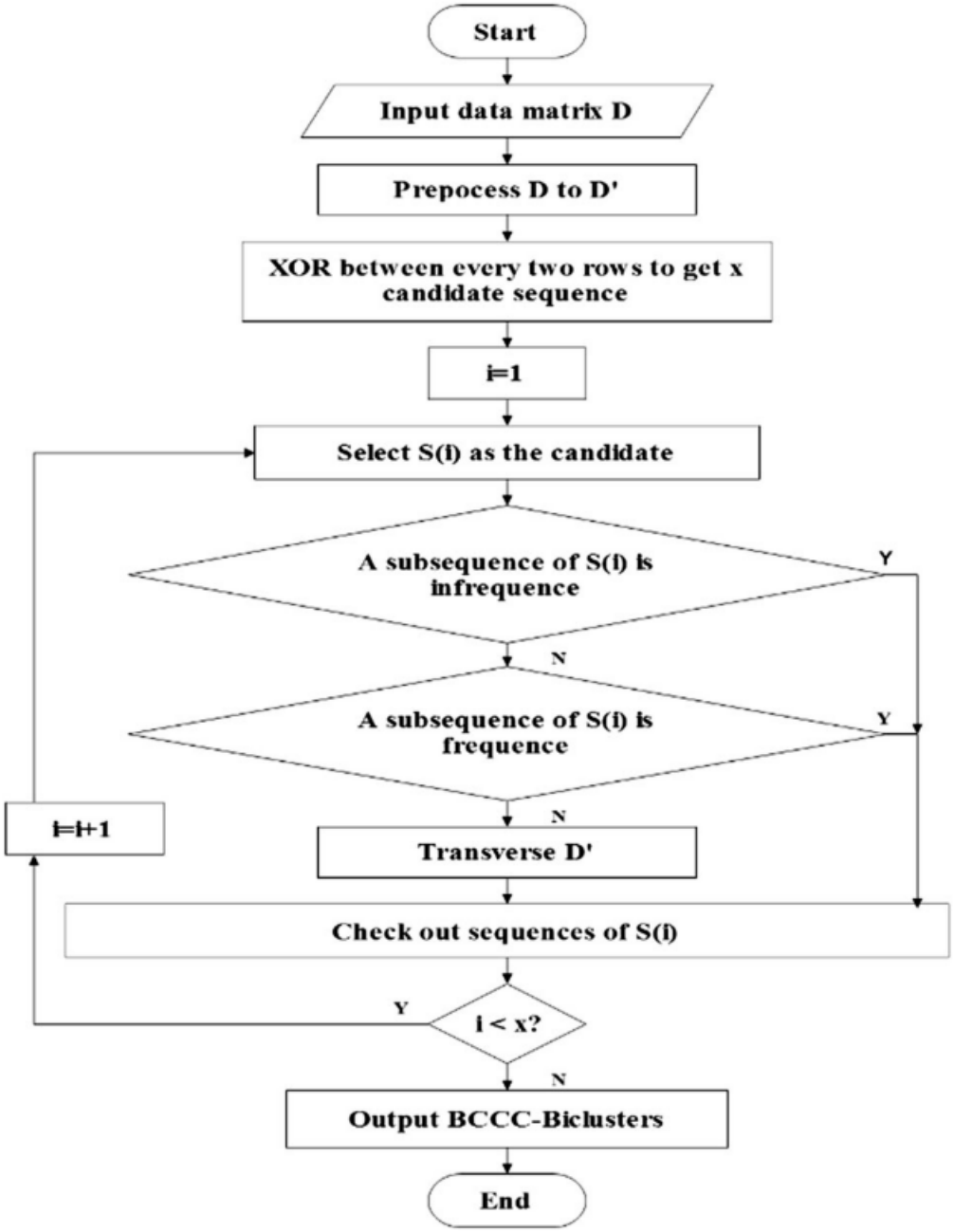
BCCC-Bicluster

3.7.4 算法描述

candidate - 候选者

transverse - 横向的

base on frequent sequential mining (FSM)


Start
Input data matrix D
Prepocess D to D'
XOR between every two rows to get x candidate sequence
i=1
Select S(i) as the candidate
A subsequence of S(i) is infrequence
Y
N
A subsequence of S(i) is frequence
Y
N
i=i+1
Transverse D'
Check out sequences of S(i)
Y
i < x?
N
Output BCCC-Biclusters
End

3.8 涉及到的数据结构

frequent-infrequent tree-array
(似乎是提出的新的数据结构 - 频繁不频繁的树状数组，应该是树状数组的一个扩展) 简写为 (FITA)
新数据结构的作用

(1) 组织候选序列，
(2) 帮助计算出频繁序列的所在行，
(3) 避免使用以前的结果进行冗余工作 (树状数组的存储，需要的时候调用)
(4) 快速遍历的策略 (全局和局部顺序模式的遍历策略)
捕获顺序信息 (capture sequential information)
定义 $S(X1, X2, \theta)$ 去捕获信息，当前的模式必须要确保是最大的 BCCC-bicluster
1) 如果 S 是频繁的，那么它的子序列长度不小于 M_{min} 同时可以包含更多的行 2）如果 S 是不频繁的，那么它的子序列长度不小于 M_{min} 则可能是频繁的。便可以此去依次研究全局模式和局部模式
————————-
给一个数组 H[]，其中每个 H[x] 都包含一个频繁树，每个频繁的连续模式都表示为从根到叶的路径。将根设为为 null
第一个节点称为“tail”，其显示连续序列的最后一列的索引。
除叶节点外，路径上的其他节点由元素等于 1 的列的索引表示，叶子存储支持其同源或反向连续序列模式的行的索引。

FTA 参考: Such a data structure is referred as Frequent-Infrequent Tree-Array (FITA), containing functionally independent but structurally similar Frequent Tree-Array (FTA) and Infrequent Tree-Array (ITA). Actually, FTA is an array whose element consists of a Frequent Tree (FT) storing all frequent contiguous sequences. Similarly, ITA stores infrequent sequences in the form of Infrequent Tree (IT).
Construction-建树

3.9 FITA-example

	t_1	t_2	t_3	t_4	t_5	t_6	t_7
g_1	6	2	8	7	8	6	7
g_2	8	3	12	9	10	11	3
g_3	3	1	2	6	9	6	8
g_4	2	5	1	4	6	3	7
g_5	4	9	8	5	2	7	5

(1)

	t_2-t_1 t'_1	t_3-t_2 t'_2	t_4-t_3 t'_3	t_5-t_4 t'_4	t_6-t_5 t'_5	t_7-t_6 t'_6
g_1	0	1	0	1	0	1
g_2	0	1	0	1	1	0
g_3	0	1	1	1	0	1
g_4	1	0	1	1	0	1
g_5	1	0	0	0	1	0

(2)

	t'_1	t'_2	t'_3	t'_4	t'_5	t'_6
$g_1 \oplus g_2$	***0***	***0***	***0***	***0***	1	1
$g_1 \oplus g_3$	0	0	1	***0***	***0***	***0***
$g_1 \oplus g_4$	1	1	1	***0***	***0***	***0***
$g_1 \oplus g_5$	1	1	0	1	1	1
$g_2 \oplus g_3$	0	0	1	0	1	1
$g_2 \oplus g_4$	1	1	1	0	1	1
$g_2 \oplus g_5$	1	1	0	1	0	0
$g_3 \oplus g_4$	1	1	***0***	***0***	***0***	***0***
$g_3 \oplus g_5$	1	1	1	1	1	1
$g_4 \oplus g_5$	0	0	1	1	1	1

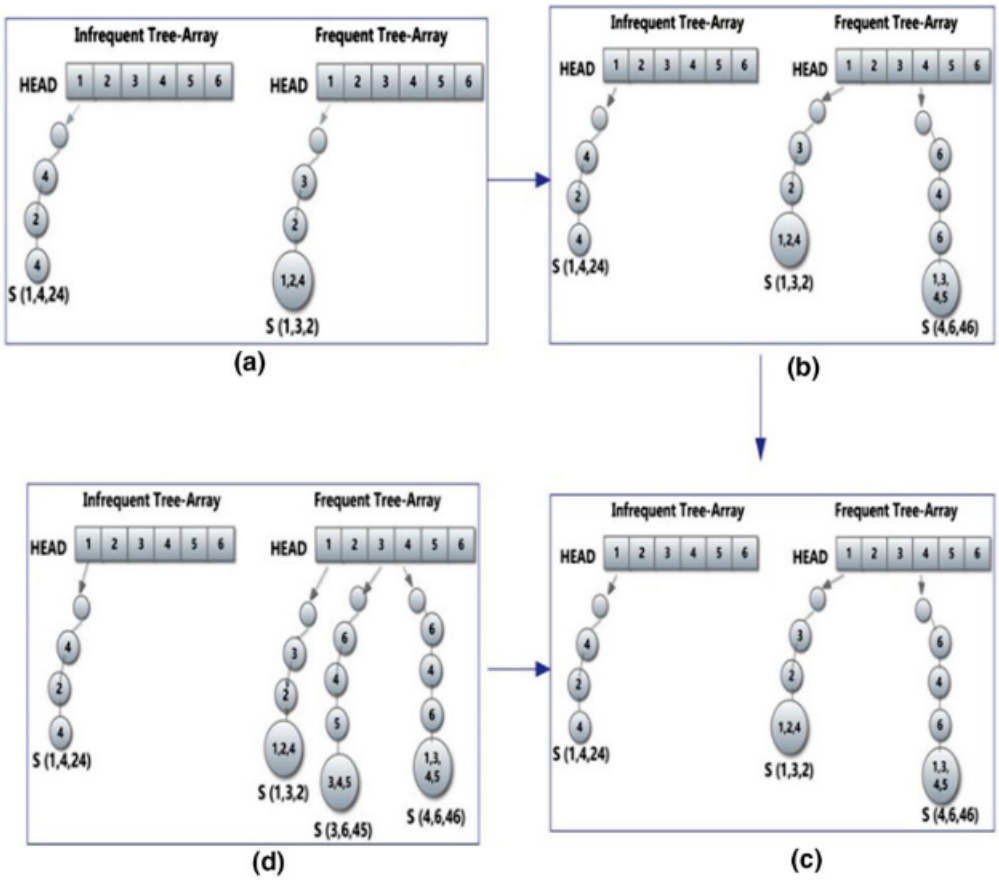
(3)

Fig. 5 *1* Example dataset D, *2* Binary difference dataset D′ after preprocessing. *3* Results after XOR operation

样例结果

表 3: FITA 例子

Letter	implication
(1) 数据矩阵 D	
(2) 处理	
(3) 异或	



3.10 亮点

1. 新颖的数据结构 2.

3.11 总结

4 双聚类效果的各种评价指标

4.1 导言

该篇论文的阅读意义以及总结意义主要在于对于双聚类的评价指标与其对双聚类的分析

4.1.1 问题

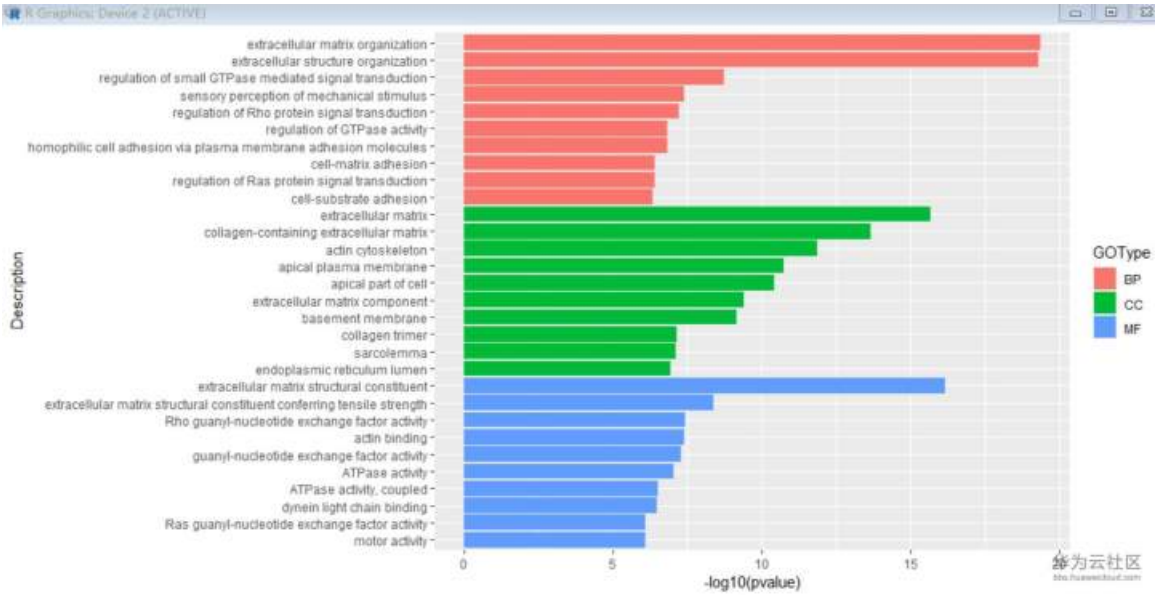
我存在的问题 1. 有哪些指标和分析是可以使用在双聚类上的 2. 这个指标可以比较双聚类的哪个方面 3. 要如何把双聚类的结果拿去分析

4.2 Gene Ontology

4.2.1 参考自知乎

为什么做富集分析？
在我们进行差异表达分析的时候，我们会得到很多的差异表达基因。这些基因如果只是按照基因名放到哪里的话，我们很难找到一个规律说这些有基因之间有什么关系的。

例如下图，我们把这些差异基因平铺来了之后，就是这么一个情况。



高通量的数据的分析，可以让我们得到很多候选的结果。但是如果只是把结果这样的平铺开的话，反正不利于我们去发现事情的本质。所以为了更情况的看清楚这些基因的功能，我们就使用了富集分析。我们可以把富集分析理解为在把很零零碎碎的东西，通过一个整体来反应出来，类似于从微观到宏观的变化。利用富集分析，我们就可以把很多看着杂乱的差异基因总结出一个比较整体反应事件发生的概述性的句子。

如：TP53 信号通路和胃癌的发生有关。而不是说 BAX、BID、ABL1、ATM、BCL2、BOK、CDKN1A 这 7 个基因和胃癌的发生有关系。

2 GO 和 KEGG 是什么？就算没有做过富集分析，但是也肯定见过公司或者一些文章里面写到他们做了 GO 分析和 KEGG pathway 分析。

那么这两个东西到底是什么？对于每个基因而言，其基本的功能基于他们的蛋白结构域以及研究的文献已经可以大致的知道一个基因具有什么样子的功能了。GO 和 KEGG 就是基于不同的分类思想而储存的基因相关功能的数据库。

GO 数据库，全称是 Gene Ontology(基因本体)，他们把基因的功能分

成了三个部分分别是：细胞组分（cellular component, CC）、分子功能（molecular function, MF）、生物过程（biological process, BP）。

利用 GO 数据库，我们就可以得到我们的目标基因在 CC, MF 和 BP 三个层面上，主要和什么有关。
例如： SRSF1 这个基因的在 GO 数据库的注释就有：KEGG 数据库：除了对基因本身功能的注释，我们也知道基因会参与人体的各个通路，基于人体通路而形成的数据库就是通路相关的数据库。而 KEGG 就是通路相关的数据库的一种。其实通路数据库有很多，类似于 wikipathway，reactome 都是相关的通路数据库。只是因为 KEGG 比较被人熟知，所以基本上都做这个分析的。例如： SRSF1 这个基因的在通路数据库的注释就有：

3 GO、KEGG 和富集分析有什么关系呢？通过上面的解释，我们知道，其实 GO 和 KEGG 是两个数据库，里面有每个基因相关的功能信息，而富集分析就是一个把这些功能进行进行整合计算的算法。GO 和 KEGG 是基础，而富集是过程，最后得到的结果就是整合后的宏观的结果。对于 GO 和 KEGG 基本上就是这些。但是对于富集分析，还是有不同的算法的。有时间我们就来简单的介绍一下基本的一些富集分析的算法。