CoClust: A Python Package for Co-clustering

François Role, Stanislas Morbieu, Mohamed Nadif
firstname.lastname@parisdescartes.fr

LIPADE, Université Paris Descartes

EGC 2017 - TextMine Workshop
Plan

1 Introduction

2 Methods included in the Coclust Package

3 Conclusion
Document clustering techniques are widely used in text mining applications.

When using a document-term matrix, the goal is to group rows (documents) into different clusters so that documents assigned to a given cluster are more similar to each other than to those in other clusters.
... to Document Co-clustering

- Document co-clustering is a natural extension of standard clustering where the rows (documents) and columns (words) of a document-term matrix are simultaneously grouped into meaningful blocks called co-clusters.

- Co-clustering makes large document sets easier to handle and interpret!

Figure –
Poor Availability of Text Co-clustering Tools in the Python Community

- In contrast to clustering, not so many packaged Python implementations are available for document co-clustering.
  - A few co-clustering implementations are available in scikit-learn, but they are mostly limited to spectral methods.
- The goal of the package presented here is to give access to a larger range of methods for co-clustering textual documents.
1 Introduction

2 Methods included in the Coclust Package

3 Conclusion
### Notation

**Data**
- matrix $\mathbf{X} = (x_{ij})$
- $i \in I$ set of $n$ documents, $j \in J$ set of $d$ terms

**Partition of $I$ in $g$ clusters**
- $\mathbf{Z} = (z_{ik})$ where $z_{ik} = 1$ if $i \in k$th cluster and $z_{ik} = 0$ otherwise
- $\mathbf{Z} = \begin{pmatrix} 3 & 0 & 0 & 1 \\ 2 & 0 & 1 & 0 \\ 3 & 0 & 0 & 1 \\ 2 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$

**Partition of $J$ in $s$ clusters**
- $\mathbf{W} = (w_{ij})$ where $w_{j\ell} = 1$ if $j \in \ell$th cluster and $w_{j\ell} = 0$ otherwise

**From $\mathbf{Z}$ and $\mathbf{W}$**
- Block $(k, \ell)$ is defined by the $x_{ij}$'s with $z_{ik}w_{j\ell} = 1$
High-level View of Available Methods

Labiod & Nadif 2011; Ailem, Role, Nadif, 2015
- Modularity-based methods
- CoclustSpecMod and CoclustMod

Govaert & Nadif, 2013
- Information theoretic based methods
- CoclustInfo

- Model-based, probabilistic methods soon to be included
Bipartite, term-document graph
Diagonal co-clustering
Find a partition of the nodes that maximizes:

\[
\frac{1}{2|E|} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{g} (a_{ij} - \frac{a_{i.}a_{.j}}{2|E|})z_{ik}z_{jk}
\]  

(1)

where \(|E|\) is the total number of edges, \(a_{i.} = \sum_{i'} a_{ii'}\) is the degree of node \(i\) and \(z_{ik} = 1\) if node \(i\) belongs to cluster \(k\) and 0 else.
Bipartite Modularity

Find a partition that maximizes:

$$\frac{1}{a_{..}} \sum_{i=1}^{n} \sum_{j=1}^{d} \sum_{k=1}^{g} (a_{ij} - \frac{a_i a_j}{a_{..}}) z_{ik} w_{jk}$$

where $a_{..}$ is the total number of edges, $a_i = \sum_{i'} a_{ii'}$ is the degree of node $i$ and $Z = (z_{ik})$ and $W = (w_{jk})$ are the binary indicator matrices for the rows and columns resp.
**CoclustSpecMod** uses an approach close to the spectral co-clustering algorithm proposed by Dhillon (Dhillon 2001), except it relies on a spectral approximation of the modularity matrix. The main steps are as follows:

- form a low-dimensional embedding of the data into an Euclidean space.
- perform an hard-assignment (e.g. k-means) on this new space to obtain a simultaneous clustering of the row and columns.
CoclustMod: direct, alternated maximization of modularity

Given current $Z$ and $W$

1) for each row $i$, since:

$$\frac{1}{a..} \sum_{i=1}^{n} \sum_{j=1}^{d} \sum_{k=1}^{g} (a_{ij} - \frac{a_i.a_j}{a..}) z_{ik} w_{jk} = \frac{1}{a..} \sum_{i=1}^{n} \sum_{k=1}^{g} \left( a_{ik}^{W} - \frac{a_i.a_{ik}^{W}}{a..} \right) z_{ik}$$

assign the row to the cluster $k$ maximizing $\left( a_{ik}^{W} - \frac{a_i.a_{ik}^{W}}{a..} \right)$.

2) Then, for each column $j$, since:

$$\frac{1}{a..} \sum_{i=1}^{n} \sum_{j=1}^{d} \sum_{k=1}^{g} (a_{ij} - \frac{a_i.a_j}{a..}) z_{ik} w_{jk} = \frac{1}{a..} \sum_{i=1}^{d} \sum_{k=1}^{g} \left( a_{kj}^{Z} - \frac{a_j.a_{kj}^{Z}}{a..} \right) w_{jk}$$

assign the column to the cluster $k$ maximizing $\left( a_{kj}^{Z} - \frac{a_j.a_{kj}^{Z}}{a..} \right)$

where $A^{W} := \{ a_{ik} = \sum_{j=1}^{d} w_{jk} a_{ij}; i = 1, \ldots, n; k = 1, \ldots, g \}$ and $A^{Z} := \{ a_{kj} = \sum_{i=1}^{n} z_{ik} a_{ij}; j = 1, \ldots, d; k = 1, \ldots, g \}$
Non-diagonal Co-clustering

- Sometimes, you don’t seek a diagonal structure.
- **CoclustInfo** is a valuable method for addressing this situation.
From an Initial Joint Distribution...

Given two variables $I$ and $J$ taking values in the sets $I = \{1, \ldots, i, \ldots, n\}$ of rows and $J = \{1, \ldots, j, \ldots, d\}$ of columns respectively, compute their associated joint distribution $P_{IJ}$.

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.05</td>
<td>0.04</td>
<td>0.06</td>
<td>0.01</td>
<td>0.00</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.06</td>
<td>0.05</td>
<td>0.04</td>
<td>0.00</td>
<td>0.01</td>
</tr>
<tr>
<td>3</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.01</td>
<td>0.00</td>
<td>0.01</td>
<td>0.07</td>
<td>0.05</td>
</tr>
<tr>
<td>4</td>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.01</td>
<td>0.01</td>
<td>0.00</td>
<td>0.06</td>
<td>0.05</td>
</tr>
<tr>
<td>5</td>
<td>21</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.04</td>
<td>0.05</td>
<td>0.03</td>
<td>0.04</td>
<td>0.05</td>
</tr>
<tr>
<td>6</td>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.05</td>
<td>0.04</td>
<td>0.04</td>
<td>0.03</td>
<td>0.04</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>20</td>
<td>21</td>
<td>20</td>
<td>100</td>
<td>0.22</td>
<td>0.19</td>
<td>0.18</td>
<td>0.21</td>
<td>0.20</td>
</tr>
</tbody>
</table>

16/21
...Define a new "aggregated" Joint Distribution

- Let $z$ and $w$ be partitions into $g$ clusters and $m$ clusters of the set $I$ of the rows and the set $J$ of columns of $X = (x_{ij})$
- Define two new random variables $K$ and $L$ taking values in the sets $K = \{1, \ldots, g\}$ and $L = \{1, \ldots, m\}$
- Define a new table $X^{zw} = (x^{zw}_{k\ell})$ according to the partitions $z$ and $w$:
  \[
  x_{k\ell}^{zw} = \sum_{i,j} z_{ik} w_{\ell j} x_{ij} \quad \forall k \in K \quad \text{and} \quad \forall \ell \in L.
  \]
  \[
  p_{k\ell}^{zw} = \frac{x_{k\ell}^{zw}}{N} = \sum_{i,j} z_{ik} w_{\ell j} p_{ij}
  \]
Seek partitions $z$ and $w$ that minimize the loss in mutual information between the two distributions

\[
\mathcal{I}(P_{IJ}) = \sum_{i,j} p_{i,j} \log \frac{p_{i,j}}{p_i p_j}
\]

\[
\mathcal{I}(P_{KL}^{zw}) = \sum_{k,\ell} p_{kl}^{zw} \log \frac{p_{kl}^{zw}}{p_z p_\ell^w}
\]

\[
\mathcal{I}(P_{IJ}) - \mathcal{I}(P_{KL}^{zw})
\]
Plan

1. Introduction

2. Methods included in the Coclust Package

3. Conclusion
Coclust offers

- several effective algorithms for co-clustering based on different approaches
- easy-to-use tools for the interpretation of co-clusters
- easy-to-use tools for comparing the obtained results

Work in progress

- Integration of different algorithms based on the latent block models
- New tools for interpretation
Bibliography