HSIC Clustering with Multiple Sources

Chieh Wu, Northeastern University

July 2020

1 Formulation

Let $X_1, X_2, ..., X_\tau \in \mathbb{R}^{n\times d}$ be τ separate datasets of n samples with d features and let $U \in \mathbb{R}^{n\times k}$ be the corresponding clustering labels where k denotes the number of classes. Given $\kappa_{X_i}(\cdot,\cdot)$ and $\kappa_U(\cdot,\cdot)$ as two kernel functions that applies respectively to X_i and U to construct kernel matrices $K_{X_i} \in \mathbb{R}^{n\times n}$ and $K_U \in \mathbb{R}^{n\times n}$. Also let H be a centering matrix where $H = I - (1/n)\mathbf{1}_n\mathbf{1}_n^T$ with $H \in \mathbb{R}^{n\times n}$, I as the identity matrix and $\mathbf{1}_n$ as a vector of 1s. Let $K_U = UU^T$ and $\Gamma = HK_UH$, our goal is to find all $W_i \in \mathbb{R}^{d\times q}$ that maximizes

$$\begin{aligned} \max_{W_i,U,\lambda_i} \ & \mathrm{Tr}((\lambda_1 K_{X_1W_1} + \lambda_2 K_{X_2W_2} + ... + \lambda_\tau K_{X_\tau W_\tau})HUU^TH) \\ \text{s. t.} \quad & \lambda_1 + ... + \lambda_\tau = 1, U^TU = I, W_i^TW_i = I. \end{aligned} \tag{1}$$

This objective requires us to solve 3 groups of variables: W_i, λ_i and U. We solve this objective by holding 2 variable groups constant while maximizing just one variable group. In the following 3 sections, we will break up the objective into 3 forms with each solving a group of variables.

Finding U. We start by setting $\lambda_i = 1 \forall i$ and let W_i be identify matrices $\forall i$, implying that the only variable is U. This implies that we can set the following matrix to a constant where

$$K_X = \lambda_1 K_{X_1W_1} + \lambda_2 K_{X_2W_2} + ... + \lambda_\tau K_{X_\tau W_\tau}. \tag{2}$$

Let $\mathcal{L} = HK_XH$, the objective reduces down to

$$\begin{aligned} \max_U \quad & \mathrm{Tr}(U^T\mathcal{L}U^T) \\ \text{s. t.} \quad & U^TU = I. \end{aligned} \tag{3}$$

Traditionally, this is a simple objective with a known solution, i.e, U is simply the set of the most dominant eigenvectors. However, since in this case we wish to automatically discover key hyperparameters, additional complexity must be added.

Specifically, since the spectral embedding U is going to be used for clustering, we need to determine the number of classes. Therefore, instead of using the standard eigen-decomposition, we will employ sparse eigen-decomposition. This is motivated by the idea that an ideal clustering should be block diagonal, with the off-diagonals as 0. By employing sparse eigen-decomposition, we hope to recover the sparse block diagonal structure, an idea proposed by Ogino and Yukawa [1]. By combining this idea with sparse-eigen-decomposition and matrix deflation as proposed by Moghaddam et al. [2] and Mackey [3], we developed a novel approach to identify the number of clusters. The sparse-eigen-decomposition allows us to identify the block structure. Consequently, by counting the number of blocks, we can also identify the number of classes.

To perform sparse eigen-decomposition, we perform the following steps.

1. Given λ and v as the most dominant eigen-value, eigen-vector of $\mathcal{L}$, and let v_i be the ith element in the vector v. We use the following equation to identify the initial submatrix size k_0

$$k_0 = \min_i \quad i \qquad \text{s.t.} \quad \sum_{j=1}^{i} \frac{v_j}{\langle v, \mathbf{1} \rangle} \geq 0.9. \tag{4}$$

2. We uniformly sample a principal submatrix $\hat{\mathcal{L}}$ starting from the size of $\mathbb{R}^{k_0 \times k_0}$. We record the most dominant eigen-value $\hat{v}$ of $\hat{\mathcal{L}}$ and compare it against the original v. We keep sampling $\hat{\mathcal{L}}$ with increment of 1 until its $\hat{v}$ is within a reasonable percentage p of v. At this point, we let this $\hat{\mathcal{L}}$ be the key submatrix and let its domainant eigen-vector be the subvector. More specifically, we solve

$$\begin{aligned} \hat{n} &= \min_i i \\ &\text{s.t.} \quad \hat{\mathcal{L}} \in \mathbb{R}^{i \times i}, \mathrm{Tr}(\hat{v}^T \hat{\mathcal{L}} \hat{v}) > p\lambda. \end{aligned} \tag{5}$$

   For the experiments, we set $p = 0.99$, or 99%. Note that the *Inclusion Principal* guarantees that $\mathrm{Tr}(\hat{v}^T \hat{\mathcal{L}} \hat{v}) \leq \mathrm{Tr}(v^T \mathcal{L} v)$.

3. Given $\hat{v}$ and $\hat{\mathcal{L}}$, the samples associated with $\hat{v}$ will be replaced in v while the rest are set to 0. This is how the sparse vectors are set and we refer to this first sparse eigen-vector as $\mathbf{v}_1$.

4. We remove this eigenvector from $\mathcal{L}$ by performing Matrix Projection Deflation to obtain the next Laplacian where

$$\mathcal{L}_{i+1} = (I - \mathbf{v}_i \mathbf{v}_i^T) \mathcal{L}_i (I - \mathbf{v}_i \mathbf{v}_i^T) \tag{6}$$

5. Go back to Step 1 using $\mathcal{L}_{i+1}$ as the new Laplacian matrix to obtain the next eigen-vector $\mathbf{v}_2$.

6. In general, given $\mathbf{v}_i$, we need to use Gram Schmidt to make sure that $\mathbf{v}_i$ is orthogonal to $\mathbf{v}_{j:j<i \forall j}$.

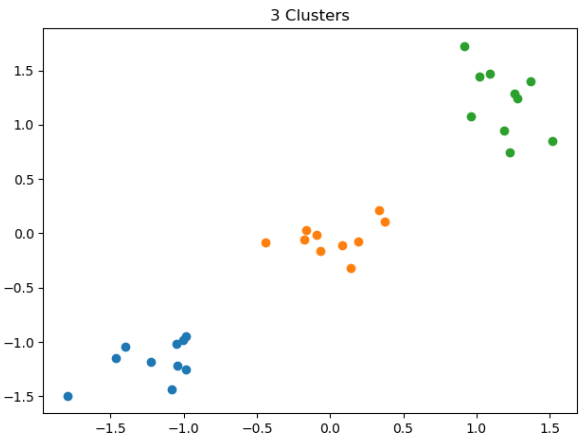
7. Note that each time we obtain a new sparse eigen-vector, some elements of the n size $\mathbf{v}_i$ will be zero while others are non-zero. Given n samples, we stop obtaining new sparse eigen-vectors when every sample has a non-zero weight in one of the eigenvector.

8. The sparse eigen-vectors is then set to U and the number of columns becomes the number of clusters c.

The source code for this sparse eigen-decomposition can be found at

https://github.com/endsley/ml_examples/blob/master/sparse_eigen_decomposition/sparse_pca.py

Here is an example of this algorithm given the following data.



Here is the results sparse eigen-decomposition result.

Cluster 1			Cluster 2			Cluster 3		
0	-0.28	0	-0.399	0	0	0	0	0.34
0	-0.365	0	-0.377	0	0	0	0	0.32
0	-0.327	0	-0.272	0	0	0	0	0.35
0	-0.217	-0	-0.218	0	0	0	0	0.24
0	-0.288	0	-0.344	0	0	0	0	0.23
0	-0.386	0	-0.302	0	0	0	0	0.39
0	-0.313	0	-0.231	0	0	0	0	0.37
0	-0.147	0	-0.291	0	0	0	0	0.23
0	-0.383	0	-0.289	0	0	0	0	0.39
0	-0.368	0	-0.382	0	0	0	0	0.36

Finding λ_i Holding U and W_i as constants, we solve λ_i with the objective

$$\begin{aligned} \max_{\lambda_i} \quad & \mathrm{Tr}((\lambda_1 K_{X_1 W_1} + \lambda_2 K_{X_2 W_2} + ... + \lambda_\tau K_{X_\tau W_\tau})\Gamma) \\ \text{s. t.} \quad & \lambda_i > 0 \forall i, \sum \lambda_i = 1. \end{aligned} \tag{7}$$

Given that τ is relatively small, this objective can be solved simply by randomly sample λ_i values and pick the one that gives us the best HSIC value.

Finding $\mathbf{W_i}$ Since each W_i is completely independent of each other, we can obtain a sparse W_i matrix separately by solving each objective as a separate ISM problem where

$$\begin{aligned} \max_{W_i} \quad & \mathrm{Tr}(K_{X_i W_i}\Gamma) \\ \text{s. t.} \quad & W_i^T W_i = I. \end{aligned} \tag{8}$$

Note that the constant λ_i is unncessary since we are solving each W_i term separately.

References

[1] Yuto Ogino and Masahiro Yukawa. Spectral clustering with automatic cluster-number identification via finding sparse eigenvectors. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pages 1187–1191. IEEE, 2018.

[2] Baback Moghaddam, Yair Weiss, and Shai Avidan. Spectral bounds for sparse pca: Exact and greedy algorithms. In *Advances in neural information processing systems*, pages 915–922, 2006.

[3] Lester W Mackey. Deflation methods for sparse pca. In *Advances in neural information processing systems*, pages 1017–1024, 2009.