Research Report - Spectral Clustering

Sourabh Antani
supervisor
Dr. Yunkai Zhou

Mathematics Department,
Southern Methodist University

June 3, 2021

Clustering Algorithms

Below are some examples of clustering algorithms used in practice, aside from spectral clustering.

- Euclidean distance and spatial density based:

Clustering Algorithms

Below are some examples of clustering algorithms used in practice, aside from spectral clustering.

- Euclidean distance and spatial density based:
  - *k*-means [Macqueen, 1967] [Lloyd 1982]

Clustering Algorithms

Below are some examples of clustering algorithms used in practice, aside from spectral clustering.

- Euclidean distance and spatial density based:
    - k-means [Macqueen, 1967] [Lloyd 1982]
    - BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) [Zhang et al. 1996]

Clustering Algorithms

Below are some examples of clustering algorithms used in practice, aside from spectral clustering.

- Euclidean distance and spatial density based:
  - *k*-means [Macqueen, 1967] [Lloyd 1982]
  - BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) [Zhang et al. 1996]
  - Mean-shift [Cheng 1995]

Clustering Algorithms

Below are some examples of clustering algorithms used in practice, aside from spectral clustering.

- Euclidean distance and spatial density based:
  - k-means [Macqueen, 1967] [Lloyd 1982]
  - BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) [Zhang et al. 1996]
  - Mean-shift [Cheng 1995]
  - DBSCAN [Ester et al. 1996]

Clustering Algorithms

Below are some examples of clustering algorithms used in practice, aside from spectral clustering.

- Euclidean distance and spatial density based:
    - *k*-means [Macqueen, 1967] [Lloyd 1982]
    - BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) [Zhang et al. 1996]
    - Mean-shift [Cheng 1995]
    - DBSCAN [Ester et al. 1996]
- Linear Algebra based

Clustering Algorithms

Below are some examples of clustering algorithms used in practice, aside from spectral clustering.

- Euclidean distance and spatial density based:
    - *k*-means [Macqueen, 1967] [Lloyd 1982]
    - BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) [Zhang et al. 1996]
    - Mean-shift [Cheng 1995]
    - DBSCAN [Ester et al. 1996]
- Linear Algebra based
    - Non-Negative Matrix Factorization [Lawton et al. 1971] [Paatero et al. 1991] [Ding et al. 2005]

Clustering Algorithms

Below are some examples of clustering algorithms used in practice, aside from spectral clustering.

- Euclidean distance and spatial density based:
  - *k*-means [Macqueen, 1967] [Lloyd 1982]
  - BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) [Zhang et al. 1996]
  - Mean-shift [Cheng 1995]
  - DBSCAN [Ester et al. 1996]
- Linear Algebra based
  - Non-Negative Matrix Factorization [Lawton et al. 1971] [Paatero et al. 1991] [Ding et al. 2005]
  - PCA Based clustering [Zhang et al. 2018]

Spectral Clustering

Graph Laplacian - Construction

- Given an undirected graph, $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ with set of vertices $\mathbf{V} = \{\mathbf{v_1}, \ldots, \mathbf{v_n}\}$ and $\mathbf{E}$ set of weighted edges. The weight of the edge between v_i and v_j, denoted by w_{ij} is assigned based on the 'similarity' between the data-points v_i and v_j. For example, if the similarity function chosen is a Gaussian kernel and the data that these vertices represent are scalars x_i and x_j, then $w_{ij} = e^{-|x_i - x_j|^2/2}$

- Given an undirected graph, $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ with set of vertices $\mathbf{V} = \{\mathbf{v_1}, \dots, \mathbf{v_n}\}$ and $\mathbf{E}$ set of weighted edges. The weight of the edge between v_i and v_j, denoted by w_{ij} is assigned based on the 'similarity' between the data-points v_i and v_j. For example, if the similarity function chosen is a Gaussian kernel and the data that these vertices represent are scalars x_i and x_j, then $w_{ij} = e^{-|x_i - x_j|^2/2}$
- Construct the Weighted adjacency matrix $\mathbf{W} = (\mathbf{w_{ij}})_{\mathbf{i,j=1,\dots,n}}$ and Degree matrix $\mathbf{D} = diag(d_1, \dots, d_n)$ where $\mathbf{d_i} = \sum_{\mathbf{j=1}}^{\mathbf{n}} \mathbf{w_{ij}}$ is called Degree of a vertex v_i.

- Given an undirected graph, $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ with set of vertices $\mathbf{V} = \{\mathbf{v_1}, \ldots, \mathbf{v_n}\}$ and $\mathbf{E}$ set of weighted edges. The weight of the edge between v_i and v_j, denoted by w_{ij} is assigned based on the 'similarity' between the data-points v_i and v_j. For example, if the similarity function chosen is a Gaussian kernel and the data that these vertices represent are scalars x_i and x_j, then $w_{ij} = e^{-|x_i - x_j|^2/2}$
- Construct the Weighted adjacency matrix $\mathbf{W} = (\mathbf{w_{ij}})_{\mathbf{i,j=1,\ldots,n}}$ and Degree matrix $\mathbf{D} = diag(d_1, \ldots, d_n)$ where $\mathbf{d_i} = \sum_{\mathbf{j=1}}^{\mathbf{n}} \mathbf{w_{ij}}$ is called Degree of a vertex v_i.
- Graph Laplacian is defined as $L = D - W$

Spectral Clustering

Graph Laplacian - Construction

- Given an undirected graph, $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ with set of vertices $\mathbf{V} = \{\mathbf{v_1}, \ldots, \mathbf{v_n}\}$ and $\mathbf{E}$ set of weighted edges. The weight of the edge between v_i and v_j, denoted by w_{ij} is assigned based on the 'similarity' between the data-points v_i and v_j. For example, if the similarity function chosen is a Gaussian kernel and the data that these vertices represent are scalars x_i and x_j, then $w_{ij} = e^{-|x_i - x_j|^2/2}$
- Construct the Weighted adjacency matrix $\mathbf{W} = (\mathbf{w_{ij}})_{\mathbf{i,j=1,\ldots,n}}$ and Degree matrix $\mathbf{D} = diag(d_1, \ldots, d_n)$ where $\mathbf{d_i} = \sum_{\mathbf{j=1}}^{\mathbf{n}} \mathbf{w_{ij}}$ is called Degree of a vertex v_i.
- Graph Laplacian is defined as $L = D - W$
- A fully connected graph can be made sparse by converting it into a k-nearest neighbor or an ϵ-neighborhood graph.

Spectral Clustering

Graph Laplacian - Types and Properties [Luxburg, 2007]

- Unnormalized Graph Laplacian [Mohar, 1991], [Mohar, 1997]
  - $L = D - W$ (Symmetric)

- Unnormalized Graph Laplacian [Mohar, 1991], [Mohar, 1997]
  - $L = D - W$ (Symmetric)
- Normalized Symmetric Graph Laplacian [Chung, 1997]
  - $L_{sym} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}$ (Symmetric)
  - if (λ, u) is an eigenpair of L , then $(\lambda, D^{1/2}u)$ is eigenpair of L_{sym}.

- Unnormalized Graph Laplacian [Mohar, 1991], [Mohar, 1997]
  - $L = D - W$ (Symmetric)
- Normalized Symmetric Graph Laplacian [Chung, 1997]
  - $L_{sym} = D^{-1/2}LD^{-1/2} = I - D^{-1/2}WD^{-1/2}$ (Symmetric)
  - if (λ, u) is an eigenpair of L , then $(\lambda, D^{1/2}u)$ is eigenpair of L_{sym}.
- Normalized Random Walk Graph Laplacian [Chung 1997]
  - $L_{rw} = D^{-1}L = I - D^{-1}W$ (Not-Symmetric)
  - if (λ, u) is an eigenpair of L , then $(\lambda, D^{1/2}u)$ is eigenpair ofL_{rw}.

Spectral Clustering

Graph Laplacian - Types and Properties [Luxburg, 2007]

- Unnormalized Graph Laplacian [Mohar, 1991], [Mohar, 1997]
  - $L = D - W$ (Symmetric)
- Normalized Symmetric Graph Laplacian [Chung, 1997]
  - $L_{sym} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}$ (Symmetric)
  - if (λ, u) is an eigenpair of L , then $(\lambda, D^{1/2}u)$ is eigenpair of L_{sym}.
- Normalized Random Walk Graph Laplacian [Chung 1997]
  - $L_{rw} = D^{-1}L = I - D^{-1}W$ (Not-Symmetric)
  - if (λ, u) is an eigenpair of L , then $(\lambda, D^{1/2}u)$ is eigenpair ofL_{rw}.
- All Laplacians are Positive-semidefinte, hence have non-Negative Real-Valued Eigenvalues

Spectral Clustering

Graph Laplacian - Types and Properties [Luxburg, 2007]

- Unnormalized Graph Laplacian [Mohar, 1991], [Mohar, 1997]
  - $L = D - W$ (Symmetric)
- Normalized Symmetric Graph Laplacian [Chung, 1997]
  - $L_{sym} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}$ (Symmetric)
  - if (λ, u) is an eigenpair of L , then $(\lambda, D^{1/2}u)$ is eigenpair of L_{sym}.
- Normalized Random Walk Graph Laplacian [Chung 1997]
  - $L_{rw} = D^{-1} L = I - D^{-1} W$ (Not-Symmetric)
  - if (λ, u) is an eigenpair of L , then $(\lambda, D^{1/2}u)$ is eigenpair ofL_{rw}.
- All Laplacians are Positive-semidefinte, hence have non-Negative Real-Valued Eigenvalues
- 0 is an eigenvalue with multiplicity equal to number of connected components

Spectral Clustering

Graph Cuts

- The goal of clustering is to maximize the similarity within cluster and minimize the similarity between clusters.

Spectral Clustering

Graph Cuts

- The goal of clustering is to maximize the similarity within cluster and minimize the similarity between clusters.
- Minimizing the between cluster similarity equals to minimizing the objective function

$$\text{cut}(A_1, \ldots, A_k) := \frac{1}{2}\sum_{i-1}^{k} W(A_i, \bar{A}_i), \quad \text{where } W(A, B) = \sum_{i\in A, j\in B} w_{ij}, \bar{A} = V \backslash A$$

Spectral Clustering

Graph Cuts

- The goal of clustering is to maximize the similarity within cluster and minimize the similarity between clusters.
- Minimizing the between cluster similarity equals to minimizing the objective function

$$\text{cut}(A_1,\dots,A_k) := \frac{1}{2}\sum_{i-1}^{k} W(A_i,\bar{A}_i), \quad \text{where } W(A,B) = \sum_{i\in A, j\in B} w_{ij}, \bar{A} = V\backslash A$$

- To maximize the within-cluster similarity, we define two objective functions

$$RatioCut(A_1,\dots,A_k) = \frac{1}{2}\sum_{i=1}^{k}\frac{W(A_i,\bar{A}_i)}{|A_i|} = \sum_{i=1}^{k}\frac{cut(A_i,\bar{A}_i)}{|A_i|}[Hagen, Kahng, 1992]$$

$$Ncut(A_1,\dots,A_k) = \frac{1}{2}\sum_{i=1}^{k}\frac{W(A_i,\bar{A}_i)}{vol(A_i)} = \sum_{i=1}^{k}\frac{cut(A_i,\bar{A}_i)}{vol(A_i)}[Shi, Malik, 2000]$$

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct.
begin

Construct a similarity graph and its weighted adjacency matrix W according to the chosen similarity function
Compute the unnormalized Laplacian L
Compute the first k eigenvectors $u_1, \ldots, u_k$ of L
Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \ldots, u_k$ as columns
For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i^{th} row of U
Cluster the points $(y_i), i = 1, \ldots, n$ in $\mathbb{R}^k$ with the k-means algorithm into clusters $C_1, \ldots, C_k$

end
Output: Clusters $A_1, \ldots, A_k$ with $A_i = \{x_j | y_j \in C_i\}$

Spectral Clustering

Algorithms - Normalized Spectral Clustering - Shi & Malik (2000)

This algorithm uses generalized eigenvectors of L, which are the eigenvectors of normalized random-walk Laplacian $L_{rw} = D^{-1}L$

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct

begin

- Construct a similarity graph and its weighted adjacency matrix W according to the chosen similarity function
- Compute the unnormalized Laplacian L.
- Compute the first k generalized eigenvectors $u_1, \ldots, u_k$ of generalized eigenproblem $Lu = \lambda Du$.
- Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \ldots, u_k$ as columns.
- For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i^{th} row of U.
- Cluster the points $(y_i), i = 1, \ldots, n$ in $\mathbb{R}^k$ with the k-means algorithm into clusters $C_1, \ldots, C_k$.

end

Output: Clusters $A_1, \ldots, A_k$ with $A_i = \{x_j | y_j \in C_i\}$

Spectral Clustering

Algorithms - Normalized Spectral Clustering - Ng, Jordan & Weiss (2002)

This algorithm uses the eigenvectors of normalized symmetric Laplacian $L_{sym} = D^{-1/2}LD^{-1/2}$. Note that if $D^{1/2}u$ is an eigenvector of L_{sym} if u is an eigenvector of L hence an additional normalization step is needed.

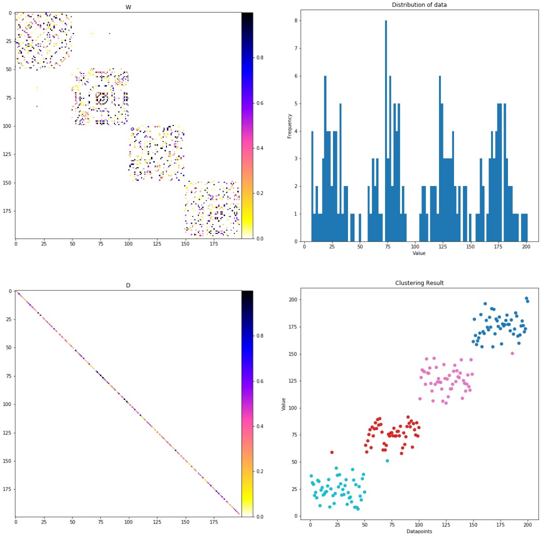
Input: Similarity matrix $S \in \mathbb{R}^{n\times n}$, number k of clusters to construct.
begin

- Construct a similarity graph and its weighted adjacency matrix W according to the chosen similarity function
- Compute the normalized Laplacian L_{sym}
- Compute the first k eigenvectors $u_1, \ldots, u_k$ of L_{sym}
- Let $U \in \mathbb{R}^{n\times k}$ be the matrix containing the vectors $u_1, \ldots, u_k$ as columns
- Form the matrix $T \in \mathbb{R}^{n\times k}$ from U by normalizing the norm to 1, $t_{ij} = u_{ij}/(\sum_k u_{ik}^2)^{1/2}$
- For $i = 1, \ldots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i^{th} row of U
- Cluster the points $(y_i), i = 1, \ldots, n$ in $\mathbb{R}^k$ with the k-means algorithm into clusters $C_1, \ldots, C_k$

end
Output: Clusters $A_1, \ldots, A_k$ with $A_i = \{x_j | y_j \in C_i\}$

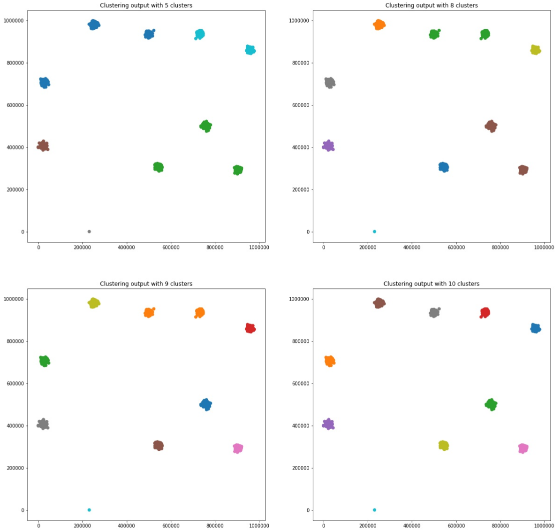
Numerical Experiments

Mixture of 4 Gaussians in $\mathbb{R}^1$. Similarity function: $e^{-|x_i - x_j|^2/2}$



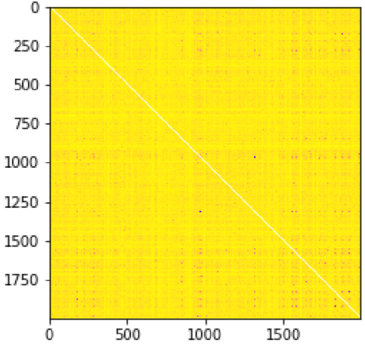
Numerical Experiments

1351 Points in $\mathbb{R}^2$. Similarity function: $||x - y||_2^{-1}$

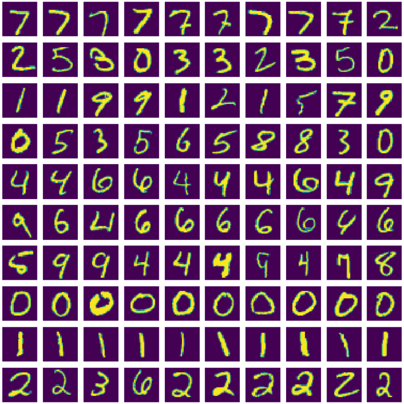


Numerical Experiments

Image Classification. Similarity function: $||\mathbf{v_i} \circ \mathbf{v_j}||_1$



(a) Weighted Adjacency Matrix

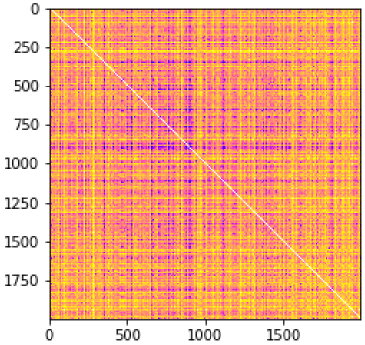


(b) Clustering Result

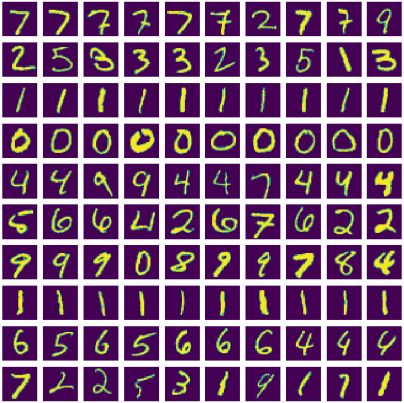
Figure: 2000 images from MNIST data-set were clustered into 10 clusters. (b) shows first 10 images in each cluster with one row per cluster

Numerical Experiments

Image Classification. Similarity function: $||\mathbf{v_i} - \mathbf{v_j}||_2^{-1}$



(a) Weighted Adjacency Matrix

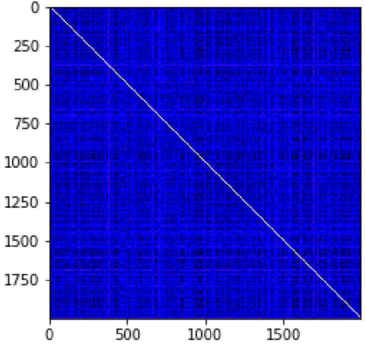


(b) Clustering Result

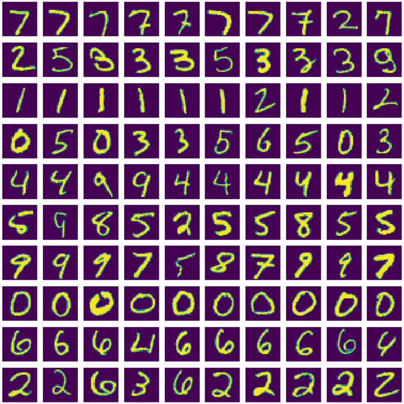
Figure: 2000 images from MNIST data-set were clustered into 10 clusters. (b) shows first 10 images in each cluster with one row per cluster

Numerical Experiments

Image Classification. Similarity function: number of pixels that either 0 or non-zero in both $\mathbf{v_i}$ & $\mathbf{v_j}$



(a) Weighted Adjacency Matrix



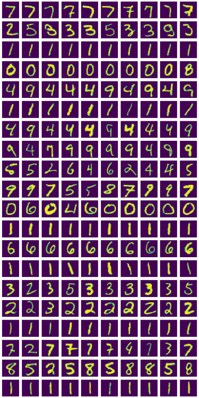
(b) Clustering Result

Figure: 2000 images from MNIST data-set were clustered into 10 clusters. (b) shows first 10 images in each cluster with one row per cluster

Numerical Experiments

Image Classification. Doubling the number of clusters to capture handwriting style differences

(a) Similarity function: $||\mathbf{v_i} \circ \mathbf{v_j}||_1$



(b) Similarity function: $||\mathbf{v_i} - \mathbf{v_j}||_2^{-1}$

Figure: 2000 images from MNIST data-set were clustered into 20 clusters, showing first 10 images in each cluster with one row per cluster.

Compressive Spectral Clustering

[Tremblay et al. 2016]

Challenges with Spectral Clustering: Eigenvector computation, Cold-restart, expensive k-means due to high dimensionality.

Compressive Spectral Clustering

[Tremblay et al. 2016]

Challenges with Spectral Clustering: Eigenvector computation, Cold-restart, expensive k-means due to high dimensionality.

- Sample $\mathcal{O}(\log(k))$ randomly filtered signals on the graph to serve as feature vectors instead of eigenvectors
  - Estimate k^{th} eigenvalue per [Napoli 2013]
  - Approximate the action of a low-pass filter H that selects the first k eigenvectors on the graph Laplacian
  - Generate $\mathcal{O}(\log(k))$ Gaussian signals, filtered by H to generate feature vectors $\mathbf{f_i}$

Compressive Spectral Clustering

[Tremblay et al. 2016]

Challenges with Spectral Clustering: Eigenvector computation, Cold-restart, expensive k-means due to high dimensionality.

- Sample $\mathcal{O}(\log(k))$ randomly filtered signals on the graph to serve as feature vectors instead of eigenvectors
  - Estimate k^{th} eigenvalue per [Napoli 2013]
  - Approximate the action of a low-pass filter H that selects the first k eigenvectors on the graph Laplacian
  - Generate $\mathcal{O}(\log(k))$ Gaussian signals, filtered by H to generate feature vectors $\mathbf{f_i}$
- Clustering random subset of $\mathcal{O}(k \log(k))$ nodes using random feature vectors

Compressive Spectral Clustering

[Tremblay et al. 2016]

Challenges with Spectral Clustering: Eigenvector computation, Cold-restart, expensive k-means due to high dimensionality.

- Sample $\mathcal{O}(\log(k))$ randomly filtered signals on the graph to serve as feature vectors instead of eigenvectors
  - Estimate k^{th} eigenvalue per [Napoli 2013]
  - Approximate the action of a low-pass filter H that selects the first k eigenvectors on the graph Laplacian
  - Generate $\mathcal{O}(\log(k))$ Gaussian signals, filtered by H to generate feature vectors $\mathbf{f_i}$
- Clustering random subset of $\mathcal{O}(k \log(k))$ nodes using random feature vectors
- Infer the cluster label of all N nodes by interpolating.

Future Work

- Accelerated Eigensolvers
  - Accelerating Lanczos
  - Chebychev-Davidson Methods [Zhou & Saad 2010] [Z. Wang, 2015]
  - Taking advantage of structure
- Detecting number of clusters adaptively.
- Hot-restart information from the pass with k clusters to bootstrap clustering process with $k + l$ clusters
- Dimensionality reduction for k-means

Thank You!