Non-exhaustive, Overlapping k-means

J. J. Whang, I. S. Dhilon, and D. F. Gleich

Teresa Lebair

University of Maryland, Baltimore County

October 29th, 2015

Outline

- Introduction
- NEO-K-Means Objective Functional
- NEO-K-Means Algorithm
- Graph Clustering using NEO-K-Means
- Experimental Results
- Conclusions and Future Work

Introduction

- The traditional k-means clustering algorithm is
  - exhaustive (every point is put into a cluster), and
  - non-overlapping (no point belongs to more than one cluster).
- There are many applications in which
  - there are outliers (points do not belong to any cluster) and / or
  - some points belong to multiple clusters.
- Application examples:
  - Outlier detection in datasets.
  - Popular *EachMovie* dataset: some movies belong to multiple genres.
  - Biology: clustering genes by function results in overlapping clusters.

Introduction

- Many papers on clustering consider either cluster outliers, or overlapping clusters, but not both.
- This paper is one of the first to consider both cluster outliers and overlapping clusters in a unified manner.
- One popular application is community detection (e.g., detecting clusters or communities of users in social networks.)
- In this paper, the objective functional for traditional clustering is modified to produce the NEO-K-Means objective functional.

Introduction

- A NEO-K-Means Algorithm is proposed to minimize the NEO-K-Means objective functional.
- NEO-K-Means is also used to solve graph clustering problems.
- NEO-K-Means is tested experimentally on both vector and graph data

Traditional k-means

- $\mathcal{X} := \{x_1, x_2, \ldots, x_n\}$ is a set of data points.
- We want to partition $\mathcal{X}$ into k clusters $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k$, i.e., $\cup_j \mathcal{C}_j = \mathcal{X}$ and $i \neq j \implies \mathcal{C}_i \cap \mathcal{C}_j = \emptyset$.
- The goal of k-means is to pick the clusters to minimize the sum of the distances of the clusters from the cluster centroids, i.e., solve the minimization problem

$$\min_{\{\mathcal{C}_j\}_{j=1}^k} \sum_{j=1}^{k} \sum_{x_i \in \mathcal{C}_j} \|x_i - m_j\|^2, \text{ where } m_j := \frac{\sum_{x_i \in \mathcal{C}_j} x_i}{|\mathcal{C}_j|}$$

- This is an NP-hard problem.
- However, the traditional k-means algorithm monotonically decreases the objective functional.

k-means Extension

- Define the assignment matrix $U = (u_{ij}) \in \mathbb{R}^{n \times k}$ such that

$$u_{ij} = \begin{cases} 1 & \text{if } x_i \in \mathcal{C}_j \\ 0 & \text{otherwise.} \end{cases}$$

- In the case of traditional disjoint exhaustive clustering, each column of U contains exactly one entry of 1; all other entries are zeros. Hence the trace of $U^T U$ is equal to the number of cluster assignments, n.

- To control the number of cluster assignments, we introduce the constraint that the trace of $U^T U$ is equal to $n(1 + \alpha)$, where $0 \leq \alpha \leq (k - 1)$.

k-means Extension

- First extend the k-means objective as follows to minimize over assignment matrices U:

$$\min_{U} \sum_{j=1}^{k} \sum_{i=1}^{n} u_{ij} \|x_i - m_j\|^2, \text{ where } m_j = \frac{\sum_{i=1}^{n} u_{ij} x_i}{\sum_{i=1}^{n} u_{ij}}.$$
$$\text{s.t. } \text{trace}(U^T U) = (1+\alpha)n$$

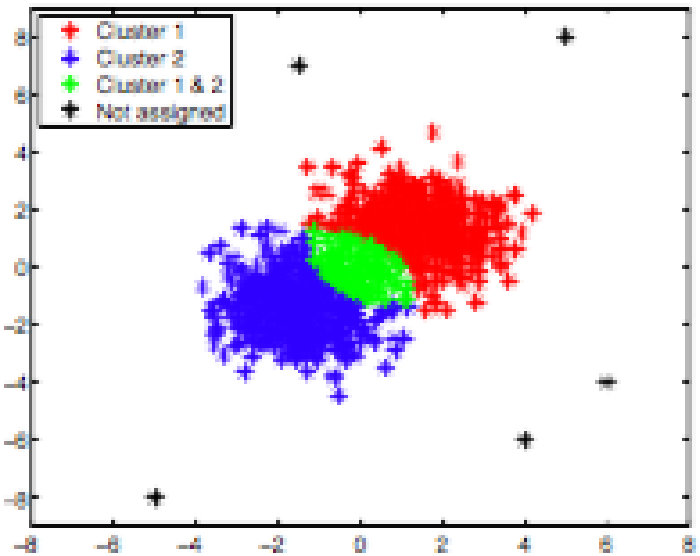
- Take $\alpha << (k-1)$ to avoid assigning each point as its own cluster.
- New objective functional allows for both outliers and overlapping clusters.

Testing k-means extension

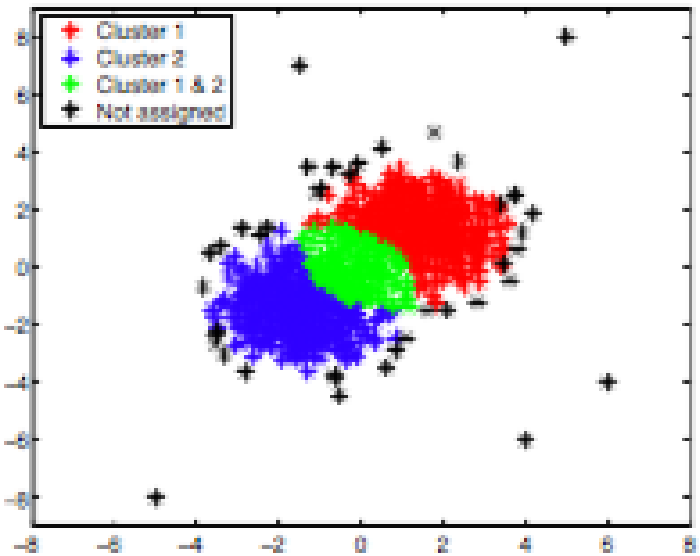
- Tested first extension of the k-means objectve on synthetic data set:
  - Two over-lapping clusters of ordered pairs, with several additional points as outliers.
  - Each cluster generated from a Gaussian distribution.
- Algorithm similar to k-means is used.
- α is set to 0.1, the ground-truth value of α.

Testing k-means extension

- First extension of k-means fails to recover ground-truth clusters.
- Too many points are labeled as outliers.



(a) Ground-truth clusters



(b) First extension of k-means

NEO-K-Means Objective

- Necessary to control the degree of non-exhaustiveness.
- Let $\mathbb{I}$ denote the indicator function:

$$\mathbb{I}(\text{expression}) = \begin{cases} 1 & \text{if the expression is true} \\ 0 & \text{otherwise.} \end{cases}$$

- Let $\mathbf{1} \in \mathbb{R}^k$ denote the vector of all ones.
- Note that $(U\mathbf{1})_i$ denotes the number of clusters to which x_i belongs.
- We update the objective functional by adding a non-exhaustiveness constraint.

NEO-K-Means Objective

- The NEO-K-Means Objective is defined as follows:

$$\min_{U} \sum_{j=1}^{k} \sum_{i=1}^{n} u_{ij} \|x_i - m_j\|^2, \text{ where } m_j = \frac{\sum_{i=1}^{n} u_{ij} x_i}{\sum_{i=1}^{n} u_{ij}}.$$

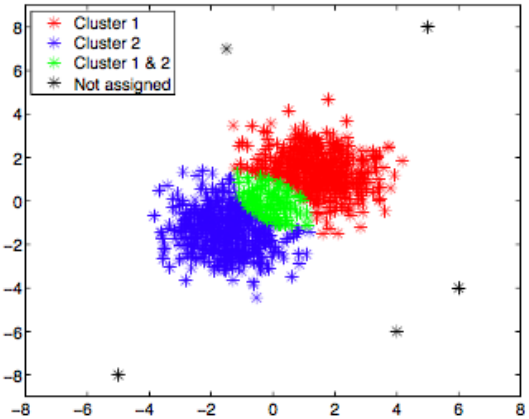
$$\text{s.t. } \text{trace}(U^T U) = (1+\alpha)n, \sum_{i=1}^{n} \mathbb{I}((U\mathbf{1})_i = 0) \leq \beta n$$

- $0 \leq \beta << 1$ controls the amount of points can be labeled as outliers.

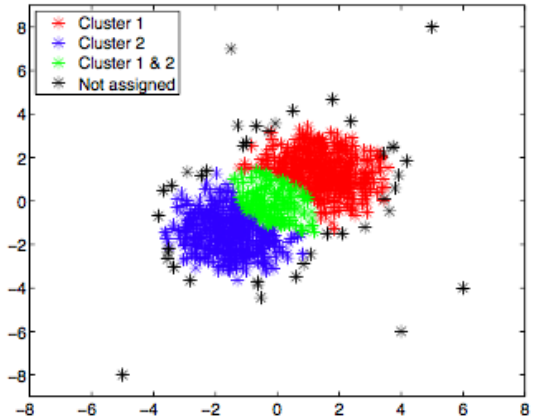
- The choice of $\alpha = \beta = 0$ recovers the traditional k-means objective.

Testing NEO-K-Means Objective

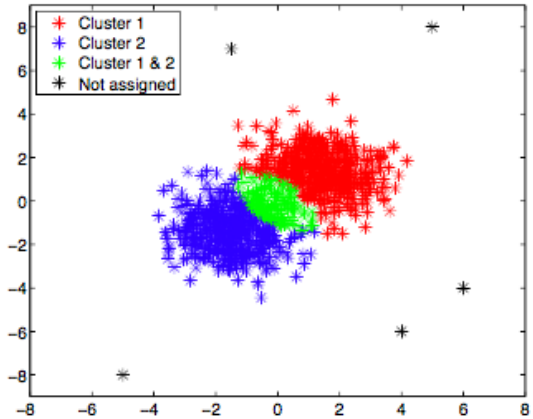
- Tested the NEO-K-Means Objective using the previous synthetic data set.
- Outcome is much better than that of the previous k-means extension.



(a) Ground-truth clusters



(b) First extension of k-means



(c) NEO-K-Means

NEO-K-Means Algorithm

- At most βn data points have no cluster membership $\Longrightarrow$ at least $n - \beta n$ data points belong to a cluster.
- Sketch of NEO-K-Means Algorithm
  - Initialize cluster centroids (use any traditional k-means initialization strategy.)
  - Compute d_{ij}, the distance between each point x_i and each cluster $\mathcal{C}_j$, for all $i = 1, \ldots, n$ and $j = 1, \ldots, k$.
  - Sort data points in ascending order by distance to closest cluster.
  - Assign the first $n - \beta n$ data points in the sorted list to their closest clusters. Let $\bar{\mathcal{C}}_j$ denote the assignments made by this step.

NEO-K-Means Algorithm

- Sketch of NEO-K-Means Algorithm (Continued)
  - Note that $\sum_{j=1}^{k} |\bar{\mathcal{C}}_j| = n - \beta n$.
  - Make an additional $\alpha n + \beta n$ assignments based off of the $\alpha n + \beta n$ smallest entries of $D = (d_{ij})$ not already associated with assignments to $\bar{\mathcal{C}}_j$'s. Let $\hat{\mathcal{C}}_j$ denote the assignments made by this step.
  - Each cluster $\mathcal{C}_j$ is then updated to be $\mathcal{C}_j := \bar{\mathcal{C}}_j \cup \hat{\mathcal{C}}_j$.
  - Repeat process until objective function is sufficiently small, or the maximum number of iterations is reached.

NEO-K-Means Algorithm

Algorithm 1 NEO-K-Means

Input: $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_n\}$, the number of clusters k, the maximum number of iterations t_{max}, α, β

Output: $\mathcal{C}_1, \mathcal{C}_2, \cdots, \mathcal{C}_k$

Initialize cluster means $\{\mathbf{m}_j\}_{j=1}^{k}$, $t = 0$.
while not converged and $t < t_{max}$ **do**
  Compute cluster means, and then compute distances between every data point and clusters $[d_{ij}]_{n \times k}$.
  Initialize $\mathcal{T} = \emptyset$, $\mathcal{S} = \emptyset$, $p = 0$, and $\bar{\mathcal{C}}_j = \emptyset$, $\hat{\mathcal{C}}_j = \emptyset$ $\forall j$.
  while $p < (n + \alpha n)$ **do**
    if $p < (n - \beta n)$ **then**
      Assign $\mathbf{x}_{i^*}$ to $\bar{\mathcal{C}}_{j^*}$ such that $(i^*, j^*) = \underset{i,j}{\operatorname{argmin}}\, d_{ij}$ where $\{(i,j)\} \notin \mathcal{T}, i \notin \mathcal{S}$.
      $\mathcal{S} = \mathcal{S} \cup \{i^*\}$.
    else
      Assign $\mathbf{x}_{i^*}$ to $\hat{\mathcal{C}}_{j^*}$ such that $(i^*, j^*) = \underset{i,j}{\operatorname{argmin}}\, d_{ij}$ where $\{(i,j)\} \notin \mathcal{T}$.
    end if
    $\mathcal{T} = \mathcal{T} \cup \{(i^*, j^*)\}$.
    $p = p + 1$.
  end while
  $\forall j$, update clusters $\mathcal{C}_j = \bar{\mathcal{C}}_j \cup \hat{\mathcal{C}}_j$.
  $t = t + 1$.
end while

NEO-K-Means Algorithm

Theorem

The NEO-K-Means objective functional monotonically decreases through the application of the NEO-K-Means Algorithm while satisfying the constraints for fixed α and β.

Proof.

Let $J^{(t)}$ denote the objective at the t-th iterations. Then

$$J^{(t)} = \sum_{j=1}^{k} \sum_{x_i \in \mathcal{C}_j^{(t)}} \|x_i - m_j^{(t)}\|^2 \geq \sum_{j=1}^{k} \sum_{x_i \in \bar{\mathcal{C}}_j^{(t+1)}} \|x_i - m_j^{(t)}\|^2 + \sum_{j=1}^{k} \sum_{x_i \in \hat{\mathcal{C}}_j^{(t+1)}} \|x_i - m_j^{(t)}\|^2$$

$$= \sum_{j=1}^{k} \sum_{x_i \in \mathcal{C}_j^{(t+1)}} \|x_i - m_j^{(t)}\|^2 \geq \sum_{j=1}^{k} \sum_{x_i \in \mathcal{C}_j^{(t+1)}} \|x_i - m_j^{(t+1)}\|^2 = J^{(t+1)}.$$

□

NEO-K-Means Algorithm: Choosing α and β

- Choosing β:
  - Run traditional k-means. Let d_i be the distance between a data point x_i and its closest cluster.
  - Compute the mean μ and standard deviation σ for these set of distances.
  - Consider d_i and outlier if $d_i \notin [\mu - \delta\sigma, \mu + \delta\sigma]$ for some fixed $\delta > 0$. ($\delta = 6$ usually leads to reasonable β.)
  - Set β equal to the proportion of d_i's that are outliers.
- Choosing α:
  - Two different strategies can be used to choose α.

NEO-K-Means Algorithm: Choosing α

- First strategy for choosing α (for small overlap):
  - For each cluster $\mathcal{C}_j$, consider the distances between each $x_i \in \mathcal{C}_j$ and $\mathcal{C}_j$. Compute the mean μ_j and standard deviation σ_j of the distances.
  - For each $x_\ell \notin \mathcal{C}_j$, compute the distance between x_ℓ and $\mathcal{C}_j$. If $d_{\ell j}$ is less than $\mu_j + \delta\sigma_j$, consider x_j to be in the overlapped region.
  - Count the points in overlapped regions to estimate α.

NEO-K-Means Algorithm: Choosing α

- Second strategy for choosing α (for large overlap):
  - Let d_{ij} denote the distance between the point x_i and the cluster $\mathcal{C}_j$.
  - Compute the normalized distance between each point x_i and cluster $\mathcal{C}_j$: $\bar{d}_{ij} := \frac{d_{ij}}{\sum_{\ell=1}^{k} d_{i\ell}}$
  - Count the number of $\bar{d}_{ij}$'s whose value is less than $\frac{1}{k+1}$. Divide by n to obtain α.
  - Note that if a point x_i is equidistant from all clusters $\mathcal{C}_\ell$, we have $\bar{d}_{i\ell} = \frac{1}{k}$ for all ℓ. Using a threshold of $\frac{1}{k+1}$ gives us a stronger bound.

Weighted Kernel K-Means

- In kernel k-means, each point is mapped into a higher dimensional feature space via the mapping ϕ.
- Additionally, weights $\omega_i \geq 0$ can be introduced to differentiate each point's contribution to the objective functional.
- The weighted kernel k-means objective functional is

$$\min_{\{\mathcal{C}_j\}_{j=1}^{k}} \sum_{j=1}^{k} \sum_{x_i \in \mathcal{C}_j} \omega_i \|\phi(x_i) - m_j\|^2, \text{ where } m_j = \frac{\sum_{x_i \in \mathcal{C}_j} \omega_i \phi(x_i)}{\sum_{x_i \in \mathcal{C}_j} \omega_i}.$$

Weighted Kernel NEO-K-Means

- Can define an analogous weighted kernel NEO-K-Means objective functional:

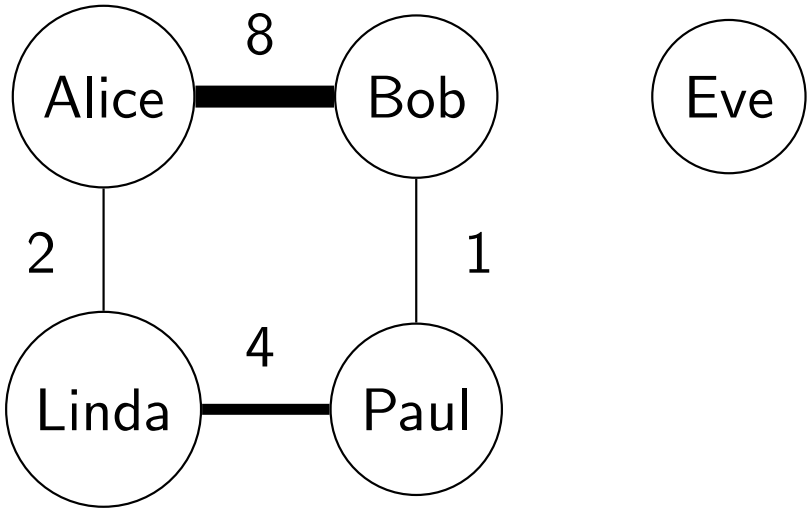
$$\min_{U} \sum_{c=1}^{k} \sum_{i=1}^{n} u_{ic}\omega_i \|\phi(x_i) - m_c\|^2, \text{ where } m_c = \frac{\sum_{i=1}^{n} u_{ic}\omega_i \phi(x_i)}{\sum_{i=1}^{n} u_{ic}\omega_i}.$$

$$\text{s.t. } \text{trace}(U^T U) = (1+\alpha)n, \sum_{i=1}^{n} \mathbb{I}((U\mathbf{1})_i = 0) \leq \beta n$$

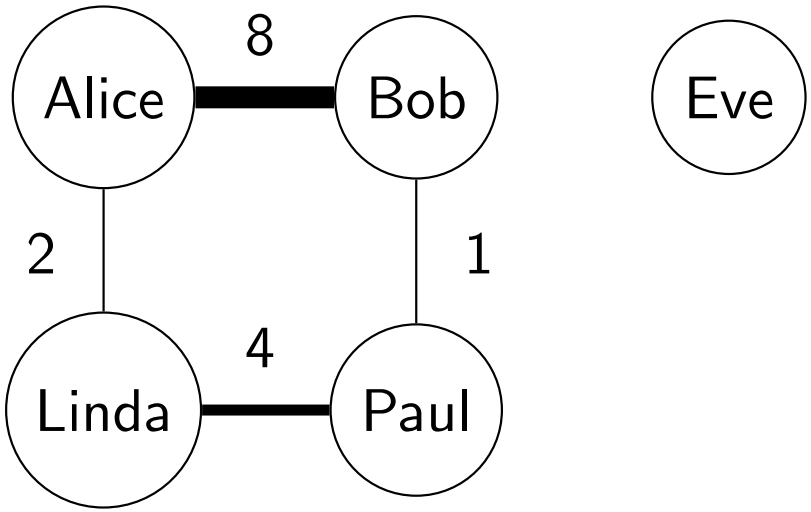
- This NEO-K-Means extension allows us to consider non-exhaustive overlapping graph clustering / overlapping community detection.

Graph Clustering Using NEO-K-Means

- A graph $G = (\mathcal{V}, \mathcal{E})$ is a collection of vertices and edges.
- We can define an adjacency matrix $A = (a_{ij})$, such that a_{ij} is equal to the weight of the edge between vertices i and j.
- Example:



Graph Clustering Using NEO-K-Means



The adjacency matrix is

$$A = \begin{array}{c} \\ A \\ B \\ L \\ P \\ E \end{array} \begin{array}{c} \begin{array}{ccccc} A & B & L & P & E \end{array} \\ \begin{pmatrix} 0 & 8 & 2 & 0 & 0 \\ 8 & 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 4 & 0 \\ 0 & 1 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}.$$

Graph Clustering Using NEO-K-Means

- We assume that there are no connections between any vertex i and itself $\implies a_{ii} = 0$ for all $i = 1, \ldots, n$.
- Also assume that graph is unidirected (so A is symmetric).
- Traditional graph partitioning problem groups vertices into k pairwise disjoint clusters $\mathcal{C}_1 \cup \ldots \mathcal{C}_k = \mathcal{V}$.
- Define the 'links' function between two clusters as the sum of the edge weights between the clusters:
  $\text{links}(\mathcal{C}_j, \mathcal{C}_\ell) := \sum_{x_{i_1} \in \mathcal{C}_j} \sum_{x_{i_2} \in \mathcal{C}_\ell} a_{i_1 i_2}$
- Example: If $\mathcal{C}_1 := \{\text{Alice, Bob}\}$ and $\mathcal{C}_2 = \{\text{ Linda, Paul}\}$, then $\text{links}(\mathcal{C}_1, \mathcal{C}_2) = a_{AL} + a_{BL} + a_{AP} + a_{BP} = 2 + 0 + 0 + 1 = 3$.

Graph Clustering Using NEO-K-Means

- The cut (popular measure of evaluating graph partitioning) of a graph G is defined as

$$\text{Cut}(G) = \sum_{j=1}^{k} \frac{\text{links}(\mathcal{C}_j, \mathcal{V} \setminus \mathcal{C}_j)}{\text{links}(\mathcal{C}_j, \mathcal{V})}.$$

- The normalized cut of a graph partition is the partition of $\mathcal{V}$ that minimizes $\text{Cut}(G)$ over all possible partitions, i.e.

$$\text{NCut}(G) = \min_{\mathcal{C}_1,\ldots,\mathcal{C}_k} \text{Cut}(G).$$

- Let D be the diagonal matrix such that $d_{ii} = \sum_{j=1}^{n} a_{ij}$, i.e., the matrix of vertex degrees.

Graph Clustering Using NEO-K-Means

- We can rewrite $\text{NCut}(G) = \min_{\mathcal{C}_1,\ldots,\mathcal{C}_k} \sum_{j=1}^{k} \frac{\text{links}(\mathcal{C}_j, \mathcal{V} \backslash \mathcal{C}_j)}{\text{links}(\mathcal{C}_j, \mathcal{V})}$ as

$$\text{NCut}(G) = \min_{y_1,\ldots,y_k} \sum_{j=1}^{k} \frac{y_j^T (D - A) y_j}{y_j^T D y_j} = \max_{y_1,\ldots,y_k} \sum_{j=1}^{k} \frac{y_j^T A y_j}{y_j^T D y_j},$$

  where y_j denotes the indicator vector for the cluster $\mathcal{C}_j$, i.e., $y_j(i) = 1$ if $v_i \in \mathcal{C}_j$, and zero otherwise.

- This traditional normalized cut objective is for disjoint, exhaustive graph clustering.

Graph Clustering Using NEO-K-Means

- For NEO-K-Means graph clustering, consider the maximization problem

$$\max_{Y} \sum_{j=1}^{k} \frac{y_j^T A y_j}{y_j^T D y_j}$$

$$\text{s.t. } \text{trace}(Y^T Y) = (1+\alpha)n, \sum_{i=1}^{n} \mathbb{I}((Y\mathbf{1})_i = 0) \leq \beta n,$$

  where Y is an assignment matrix, with the jth column of Y given by y_j.
- Just as with the vector data, we may adjust α and β to control the degree of non-exhaustiveness.
- This optimization problem for graph partitioning can be reformulated as weighted kernel NEO-K-Means problem.

Graph Clustering Using NEO-K-Means

- Let $W \in \mathbb{R}^{n \times n}$ be the diagonal matrix such that w_{ii} is equal to the vertex degree/weight for each $i = 1, \ldots, n$.
- Let K be a kernel matrix given by $K_{ij} = \phi(x_i)\phi(x_j)$.
- Finally, let u_c be the column of the assignment matrix U.
- The weighted kernel NEO-K-Means objective can be rewritten as

$$\min_U \sum_{c=1}^{k} \sum_{i=1}^{n} u_{ic} \omega_i \|\phi(x_i) - m_c\|^2 =$$

$$\min_U \sum_{c=1}^{k} \left(\sum_{i=1}^{n} u_{ic} \omega_i K_{ii} - \frac{u_c^T WKWu_c}{u_c^T Wu_c} \right).$$

Graph Clustering Using NEO-K-Means

- Define the kernel as $K = \gamma W^{-1} + W^{-1} A W^{-1}$, where $\gamma > 0$ is chosen so that K is positive definite.
- Then

$$\begin{aligned}
&\min_U \sum_{c=1}^{k} \left(\sum_{i=1}^{n} u_{ic} \omega_i K_{ii} - \frac{u_c^T A u_c}{u_c^T W u_c} \right) \\
&= \min_U \left(\gamma(1+\alpha)n - \sum_{c=1}^{k} \frac{u_c^T A u_c}{u_c^T W u_c} \right) \\
&= \max_U \sum_{c=1}^{k} \frac{u_c^T A u_c}{u_c^T W u_c}.
\end{aligned}$$

- Letting $W = D$ and noting that $U = Y$ demonstrates that the extended normalized cut objective can be formulated as a weighted kernel NEO-K-Means objective.

Graph Clustering Using NEO-K-Means

- Authors use the following distance function between a vertex v_i and a cluster $\mathcal{C}_j$:

$$\text{dist}(v_i, \mathcal{C}_j) = -\frac{2\text{links}(v_i, \mathcal{C}_j)}{\deg(v_i)\deg(\mathcal{C}_j)} + \frac{\text{links}(\mathcal{C}_j, \mathcal{C}_j)}{\deg(\mathcal{C}_j)^2} + \frac{\gamma}{\deg(v_i)} - \frac{\gamma}{\deg(\mathcal{C}_j)}$$

- $\deg(v_i)$ denotes the degree of the vertex v_i, and $\deg(\mathcal{C}_j)$ denotes the sum of the edge weights connecting vertices in $\mathcal{C}_j$.

- The NEO-K-Means Algorithm can then be applied to the graph data using this distance function.

Experimental Metrics

- To measure the effectiveness of the NEO-K-Means algorithm, we use the average F_1 score:
  - Define F_1 score of the the ground-truth cluster $\mathcal{S}_i$ as $F_1(\mathcal{S}_i) = \frac{2r_{\mathcal{S}_i} p_{\mathcal{S}_i}}{r_{\mathcal{S}_i} + p_{\mathcal{S}_i}}$, where $p_{\mathcal{S}_i} = \frac{|\mathcal{C}_{j*} \cap \mathcal{S}_i|}{|\mathcal{C}_{j*}|}$, $r_{\mathcal{S}_i} = \frac{|\mathcal{C}_{j*} \cap \mathcal{S}_i|}{|\mathcal{S}_i|}$ and j^* corresponds to the cluster $\mathcal{C}_{j*}$ which makes $F_1(\mathcal{S}_i)$ as large as possible out of all the clusters.
  - The average F_1 score is then

$$\bar{F_1} = \frac{1}{|\mathcal{S}|} \sum_{\mathcal{S}_i \in \mathcal{S}} F_1(\mathcal{S}_i),$$

    where $\mathcal{S}$ is the set of ground-truth clusters.
  - Higher $\bar{F_1} \in [0, 1]$ score indicates better clustering.

Experimental Results

- Tested NEO-K-Means Algorithm on several vector data sets.

Table 1: Vector datasets.

	n	dim.	$\lvert\bar{C}\rvert$	outliers	k
synth1	5,000	2	2,750	0	2
synth2	1,000	2	550	5	2
synth3	6,000	2	3,600	6	2
yeast	2,417	103	731.5	0	14
music	593	72	184.7	0	6
scene	2,407	294	430.8	0	6

Experimental Results

- Compared effectiveness of NEO-K-Means to that of several other algorithms.

Table 2: F_1 scores on vector datasets. NEO-K-Means (the last column) achieves the highest F_1 score across all the datasets while the performance of other existing algorithms is not consistent across all the datasets.

	moc	*fuzzy*	*esp*	*isp*	*okm*	*rokm*	NEO
synth1	0.833	0.959	0.977	0.985	0.989	0.969	**0.996**
synth2	0.836	0.957	0.952	0.973	0.967	0.975	**0.996**
synth3	0.547	0.919	0.968	0.952	0.970	0.928	**0.996**
yeast	-	0.308	0.289	0.203	0.311	0.203	**0.366**
music	0.534	0.533	0.527	0.508	0.527	0.454	**0.550**
scene	0.467	0.431	0.572	0.586	0.571	0.593	**0.626**

- NEO-K-Means consistently outperforms the other algorithms for this data set.

Experimental Results

- Additionally test NEO-K-Means with large graph data sets:

Table 4: Graph datasets

	No. of vertices	No. of edges
Amazon	334,863	925,872
DBLP	317,080	1,049,866
Flickr	1,994,422	21,445,057
LiveJournal	1,757,326	42,183,338

- Compare the average normalized cut of each algorithm when applied to large real-world datasets.

Table 3: Average normalized cut of each algorithm on large real-world networks. Lower normalized cut indicates better clustering. NEO-K-Means achieves the lowest normalized cut on all the datasets.

	demon	*oslom*	*bigclam*	*sse*	NEO
Amazon	0.555	0.221	0.392	0.116	**0.105**
DBLP	0.606	0.355	0.617	0.204	**0.188**
Flickr	-	-	0.596	0.515	**0.331**
LiveJournal	-	-	0.912	0.643	**0.373**

Experimental Results

- Also consider (i) average F_1 score of different algorithms, and (ii) average normalized cut and F_1 NEO-K-Means for different α's and β's.

Table 5: F_1 score of each algorithm on Amazon and DBLP. NEO-K-Means shows the highest F_1 score on Amazon, and comparable F_1 score with *sse* on DBLP.

	demon	*oslom*	*bigclam*	*sse*	NEO
Amazon	0.165	0.318	0.269	0.467	**0.490**
DBLP	0.137	0.132	0.151	**0.176**	0.174

Table 6: Average normalized cut and F_1 score of NEO-K-Means with different α and β on Amazon dataset.

	α=30, β=0	α=35, β=0	α=45, β=0	α=30, β=0.0001	α=35, β=0.0001	α=45, β=0.0001
ncut	0.107	0.104	0.104	0.106	0.104	0.104
F_1	0.488	0.490	0.490	0.488	0.490	0.490

Conclusions and Future Work

- NEO-K-Means simultaneously considers non-exhaustive and overlapping clusters.
- New method outperforms state-of-the-art methods in terms of finding ground truth clusters.
- Conclude that NEO-K-Means is a useful algorithm.
- Plan to extend this type of clustering to other types of Bregman divergences.

References

J. Shi and J. Malik

Normalized Cuts and Image Segmentation.

IEE TPAMI, 1997.

J. J. Whang, D. F. Gleich, and I. S. Dhilon

Overlapping community detection using seed set expansion.

CIKM, 2013.

J. J. Whang, I. S. Dhilon, and D. F. Gleich

Non-exhaustive, Overlapping k-means.

SIAM International Conference on Data Mining, pg. 936–944, 2015.

`http://mulan.sourceforge.net/datasets.html`