A Tutorial on Manifold Clustering using Genetic Algorithms

Hector D. Menendez

Department of Computer Science
University College London

2nd September 2015
What’s Clustering?

**Definition**

Group Information (or Data) Blindly.
Blindly?

Clustering: Science or Art?

Ulrike von Luxburg  
Max Planck Institute for Intelligent Systems, Tübingen, Germany

Robert C. Williamson  
Australian National University and NICTA, Canberra ACT 0200, Australia

Isabelle Guyon  
ClopiNet, 955 Creston Road, Berkeley, CA 94708, USA
What do I need to cluster things?

1. Objects.
What do I need to cluster things?

1. Objects.
2. Criterion / Criteria.
What do I need to cluster things?

1 Objects.
2 Criterion / Criteria.
3 Minimization Process.
The objects

The objects are usually data. They can be contained in a specific space or not. The model could have all the objects or receive it in real-time.
The Criterion

It defines the final goal of the grouping process.
If defines similarity/distance among objects.
The Minimization Process

The criterion defines a Cost Function. This Cost Function is used to guide a minimization process. The main goal is to minimize the cost.
Clustering Approaches

**Partitional**: Divides the whole set of objects in Clusters with no overlapping.

**Overlapping**: Divides the whole set of objects in Clusters with overlapping.

**Hierarchical**: Nest the clusters by hierarchical levels.
Clustering Approaches

**Parametric**: The Cost Function is a statistical estimator whose parameters are optimized.

**Non-Parametric**: There is no statistical estimators during the optimization process.

**Hybrid**: Combines the previous methods.
Clustering using GAs

Hector D. Menendez (UCL)
2nd September 2015
Static Data

Numerical or Categorical Data.
It can be discriminated in an Euclidean Search Space.
The most used algorithm is K-means.
K-means
Similarities

It is possible that we have no idea about the space, but we have a metric to compare objects. Using this distance/similarity metric we can cluster the objects. The most frequent approaches are Medoid-based approaches.
PAM

Require: \( k \) number of clusters and \( D = d_{i,j} \) distance matrix between the elements of \( X = x_1, \ldots, x_n \).

Ensure: \( m_1, \ldots, m_k \) best \( k \) medoids and \( C = \{ c_1, \ldots, c_k \} \) clusters

Choose random medoids.

while iteration < MaxIterations and !convCriterion() do

   for all \( x_i \in X \) do
      Insert \( x_i \) in cluster \( c_a \) where \( m_a \) is the closest medoid.

   for all \( c_a \in C \) do
      Calculate medoid \( m_a = \min_j \sum d_{i,j} \) where \( x_i, x_j \in c_a \).
Text

Normally used to detect topics or group documents. It is combined with NLP techniques. The approach usually generates a search space or a Similarity matrix.
Term-Document

Documents → Vector-space representation

We study the complexity of influencing elections through bribery. How computationally complex is it for an external actor to determine whether by a certain amount of bribing voters a specified candidate can be made the election's winner? We study this problem for election systems as varied as scoring ...

<table>
<thead>
<tr>
<th>Term</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>D5</th>
</tr>
</thead>
<tbody>
<tr>
<td>complexity</td>
<td>2</td>
<td></td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>algorithm</td>
<td>3</td>
<td>4</td>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>entropy</td>
<td>1</td>
<td></td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>traffic</td>
<td></td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>network</td>
<td>1</td>
<td></td>
<td></td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>
Images

Process of detect objects in Images: Pixels or Regions considered as static data.
Group images by similarity.
Objects
Graphs

Clustering can be applied to graphs in order to group nodes by similarity: **Community Detection**. Graph Structures can help to create new types of Clustering: **Spectral Clustering**.
Community Detection
Spectral Clustering

Data Example

Heatmap of the Similarity Graph

Spectral Clustering results

1st Eigenvector

2nd Eigenvector

3rd Eigenvector

1st Eigenvector Normalized

2nd Eigenvector Normalized

3rd Eigenvector Normalized
Big Data

The Large Datasets require new strategies to apply these algorithms.

We will focus on two approaches: Large Datasets and Real-Time data.
Large Data

Large data can be processed parallel. We will focus on two approaches: Large Datasets and Real-Time data.
Map Reduce

Input data

Map() → Reduce() → Output data

Map() → Reduce()

Map() → Reduce()

Split [k1, v1]
Sort by k1
Merge [k1, [v1, v2, v3...]]
K-means Map Reduce

Diagram showing the process of K-means clustering using Map Reduce.

Init: Data is split into Data Block 1, Data Block 2, Data Block 3, and Data Block M.

Map: Each Data Block is processed to calculate distances to the centroids. The output is a list of distances from each data point to each centroid.

Reduce: The distances are then reduced to find the closest centroid for each data point. The output is the centroids for each cluster.

The process is repeated until the centroids converge, indicating that the clusters have been formed.

Centroids: The final centroids are C_1, C_2, C_3, ..., C_n.
Real-Time Data

Stream data can be analysed in Real-Time using clustering.

The algorithm needs to update its structure with every instance.

These algorithms are fast and keep as less as possible information about past instances.
Stream
Bio-Inspired Computation

Bio-inspired computation is based on natural behaviours.
Some examples are: Particle Swarm Optimization, Ant Colony Optimization, Artificial Bee Colony. We focus on Evolutionary Algorithms.
Evolutionary Algorithms

Evolutionary Algorithms are based on the evolution of a chromosome population. These algorithms combine an encoding, operations and a fitness function to guide the search.

There are several different approaches based on evolutionary computation.
Types of Evolutionary Algorithms

Simple Genetic Algorithms.
Multi-Objective Genetic Algorithms.
Evolutionary Strategies.
Island Models.
Simple Genetic Algorithms

It is divided in three parts: Encoding, Operations and Fitness.

They are applied for optimization problems.

The idea is to encode a set of possible solutions as the population and evolve them in order to find the best solution according to the fitness function.
Simple Genetic Algorithms
Multi-Objective Genetic Algorithms

Different objectives are combined during the search.
A Pareto Front of solutions is generated.
Evolutionary Strategies

The chromosomes evolves according to the position in the search space.
The evolutionary process is also adaptive.
Island-based Algorithms

Create islands that share information in two steps. Each island performs a micro search. The general model performs a macro search.
GAs in Manifold Identification

Create a Genetic Algorithm for Manifold Identification.
Compare it with Spectral Clustering.
Encodings

(a) Label-based

Chromosome 1

| 1 | 1 | 1 | 2 | 2 | 2 | 3 | 3 | 3 |

Cluster 1: {1, 2, 3}  
Cluster 2: {4, 5, 6}  
Cluster 3: {7, 8, 9}

(b) Medoid-based

Clusters of data instance from 1 to 9

![Graph showing clusters of data instance from 1 to 9]
Operations: Selection y Reproduction

**Selection**: $(\mu + \lambda)$.

**Reproduction**: Tournament selection of individuals.
Operations: Crossover

(a) Crossover Label-based

<table>
<thead>
<tr>
<th>nodes</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chromosome 1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chromosome 2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>New Chromosome 1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>New Chromosome 2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Intersection:

<table>
<thead>
<tr>
<th></th>
<th>Clusters 1</th>
<th>Clusters 2</th>
<th>Clusters 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chromosome 1</td>
<td>{1, 2, 3}</td>
<td>{4, 5, 6}</td>
<td>{7, 8, 9}</td>
</tr>
<tr>
<td>Chromosome 2</td>
<td>{2, 3}</td>
<td>{5, 6, 7}</td>
<td>{1, 4, 8, 9}</td>
</tr>
<tr>
<td>Intersection</td>
<td>{2, 3}</td>
<td>{5, 6}</td>
<td>{8, 9}</td>
</tr>
<tr>
<td>New Chromosome 1</td>
<td>{2, 3, 4}</td>
<td>{5, 6, 1}</td>
<td>{8, 9, 7}</td>
</tr>
<tr>
<td>New Chromosome 2</td>
<td>{2, 3, 1, 7}</td>
<td>{5, 6, 4}</td>
<td>{8, 9}</td>
</tr>
</tbody>
</table>
Operations: Mutation

**Mutation:** It is *adaptive*. The mutation probability depends on their assignation probability to the current cluster. For each encoding there are different mutations:

- Label-based: Randomly modifies the value of an allele.
- Medoid-based: Moves an allele from a cluster to another.
# Operations: Mutation example

<table>
<thead>
<tr>
<th>nodes</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Chromosome</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>2</th>
<th>2</th>
<th>3</th>
<th>3</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chromosome Mutated</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cluster 1</th>
<th>Cluster 2</th>
<th>Cluster 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chromosome</td>
<td>{1, 2, 3}</td>
<td>{4, 5, 6}</td>
</tr>
<tr>
<td>Chromosome Mutated</td>
<td>{1, 2}</td>
<td>{4, 5, 6, 8}</td>
</tr>
</tbody>
</table>
Fitness Function: Weight Clustering Coefficient

\[ C_i^w = \frac{\sum_{j,h} \left( \frac{w_{ij} + w_{ih}}{2} \right) a_{ij} a_{ih} a_{jh}}{S_i (k_i - 1)} \]
La Función Fitness: KNN-Minimal Cut

Combines K-Nearest Neighbourhood (KNN) and Minimal Cut. KNN guarantees the data continuity. MinCut guarantees the clustering separation.
Fitness Function: KNN-Minimal Cut

\[ KNN = \sum_{x \in C} \frac{|\{y|y \in \Gamma(x) \land y \in C_x\}|}{|\Gamma(x)|} \]
\[ MC = \sum_{x \in C} \frac{\sum_{y \notin C_x} w_{xy}}{|\{y|y \notin C_x\}|} \]

\[ \frac{KNN}{|C|} \times \left(1 - \frac{MC}{|C|}\right) \]
The robustness problem

**GCC** is focus on improving the **robustness**.

Given three objects: \( o_1, o_2, o_3 \), and their distances \( d_{12}^{M_1}, d_{23}^{M_1}, d_{13}^{M_1}, d_{12}^{M_2}, d_{23}^{M_2}, d_{13}^{M_2} \) where \( M_1 \) and \( M_2 \) are metrics, when they satisfy:

\[
\begin{align*}
    d_{12}^{M_1} &< d_{23}^{M_1} < d_{13}^{M_1} \\
    d_{12}^{M_2} &< d_{23}^{M_2} < d_{13}^{M_2}
\end{align*}
\]

The clustering algorithm results have to be the same (except for 0 or infinite distance)
Robustness of SC and GGC

Accuracy of the Spirals Dataset

- Spectral Clustering
- GGC

Sigma value
Accuracy achieved

0 1000 2000 3000 4000
0.0 0.2 0.4 0.6 0.8 1.0
0 1000 2000 3000 4000
0.0 0.4 0.6 0.8 1.0

Hector D. Menendez (UCL)
Less Memory, More Accuracy

Use multi-objective search to improve the results. Reduce the memory usage reducing the Similarity graph.
MOGGC Motivation

GGC shows some problems according to the Memory Usage. The convergence of GGC can also be improved through a MOGA algorithm. The algorithm can employ other graph representation to obtain the same results.
MOGGC Algorithm

It uses the **same initial parameters** than GGC and the same encoding.

The MOGA algorithm used to guide the algorithm is **SPEA2**. It used a **K-Similarity Graph** which is a KxN matrix where K is constant and N is the number of elements. It **grows linearly** instead of exponentially.

The fitness functions have been modified to improve the data continuity and **cluster separation**.
MOGGC Fitness Objectives: DC

Data Continuity Degree: It calculates the total edges sum for each minimal spanning tree of each connected component of the K-Graph.
MOGGC Fitness Objectives: CS

Clusters Separation: It calculates the arithmetic average value of the edge weights between the different clusters.
MOGGO Fitness Objectives: Pareto
Find the Best K

Include the Number of Clusters in the Search.
Goal: Find the best number of Manifolds.
CEMOG Motivation

CEMOG was designed in order to include the number of clusters in the genetic search.

The algorithm is based on Co-Evolutionary algorithm. It is mainly divided in two kind of evolutions:

- Macro-evolution: the evolution of the different islands as a whole.
- Micro-evolution: the evolution of each island as an independent population. It is performed through MOGGA.
CEMOG Algorithm: Macro-Evolution

\[ S_{Pop_{k_{max}}} \]

\[ S_{Pop_{k_{max}}-1} \]

\[ S_{Pop_{k_{min}}+1} \]

\[ S_{Pop_{k_{min}}} \]
Pareto

Pareto Front for Aggregation

Pareto Front for Spirals

Pareto Front for R15

Pareto Front for Jain

Data Continuity

Clusters Separation

K=5
K=6
K=7
K=8
K=9

K=2
K=3
K=4
K=5
K=6

K=13
K=14
K=15
K=16
K=17

K=2
K=3
K=4
K=5
K=6
Large Data: Sub-sampling

Choose relevant points to estimate the model.
Goal: improve the performance of the whole analysis.
Nyström Method and GANY

\[ W = \begin{pmatrix} w_{1,1} & \cdots & w_{1,N} \\ \vdots & \ddots & \vdots \\ w_{N,1} & \cdots & w_{N,N} \end{pmatrix} \]

Full-Matrix

\[ W = \begin{pmatrix} w_{1,1} & \cdots & w_{1,N} \\ \vdots & \ddots & \vdots \\ w_{N,1} & \cdots & w_{N,N} \end{pmatrix} \]

Eigenvectors Extraction

\[ (A|B) = \begin{pmatrix} a_{1,1} & \cdots & a_{1,n} & b_{1,1} & \cdots & b_{1,N-n} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & \cdots & a_{n,n} & b_{n,1} & \cdots & b_{n,N-n} \end{pmatrix} \]

Nyström extension

\[ \begin{pmatrix} a_{1,1} & \cdots & a_{1,n} & b_{1,1} & \cdots & b_{1,N-n} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & \cdots & a_{n,n} & b_{n,1} & \cdots & b_{n,N-n} \end{pmatrix} \]

Equivalent Eigenvectors

\[ UU^T \]

\[ UU^T \]

Full-Matrix Eigenvectors Extraction

n samples

(\begin{pmatrix} w_{1,1} & \cdots & w_{1,N} \\ \vdots & \ddots & \vdots \\ w_{N,1} & \cdots & w_{N,N} \end{pmatrix})
**GANY Algorithm**

| 0.1 | 0.5 | 0.1 | -0.7 | -0.1 | -0.6 | -0.8 | 0.6 | 0.6 |

**Initialization:** *Uniform* initialization. The **Nyström method** generates a normalized search space.

**Selection-Reproduction:** *Elitism* selection and *Tournament* reproduction.

**Crossover:** Exchanges strings.

**Mutation:** *Uniform* random mutation of the centroids coordinates.
Large Data: Reduce the Space

We need to zoom out the space.
Goal: analyze the data in a general way.
Reducing the space: Clustering as Manifolds

Voronoi Tessellation for Jain
MOGCLA Motivation

MOGCLA tries to **optimize** the **set of centroids** using a search in two levels:

1. The **micro** search implements a **centroid selection**.
2. The **macro** search **joins the regions** to form the manifolds.
MOGCLA Algorithm

Data

K-means M-R

Centroids

\[ C_1 \quad C_2 \quad C_3 \quad \ldots \quad C_n \]

Parallel MOGGC

\[ \text{Centroids} \]

\[ \text{New Pop.} \]

\[ \text{Best Sol. 1} \quad \text{New Sol. 1} \]

\[ \text{Best Sol. 2} \quad \text{New Sol. 2} \]

\[ \text{Best Sol. 3} \quad \text{New Sol. 3} \]

\[ \text{Best Sol. M} \quad \text{New Sol. M} \]
Online Clustering

Data Real-time analysis.
Goal: construct the manifold with the data online.
The new MOGGOC algorithm tries to analyse the data in an online way in order to generate a **real-time analysis**.

- This algorithm follows the same criteria than the previous one, however, using online clustering.
- It has been compared with online approaches.
MOGGOC Algorithm

Choose closest Centroid $q^*$

Data Stream

$x_1, x_2, \ldots, x_n$

Centroids

$c_1, c_2, c_3, \ldots, c_n$

Choose closest Centroid $q^*$

$\quad c_{q^*}^{(\text{new})} = c_{q^*} - \zeta (x_i - c_{q^*})$

Centroid Cluster Solutions 1

Centroid Cluster Solutions 2

Centroid Cluster Solutions 3

Centroid Cluster Solutions M

Best Solutions 1

Best Solutions 2

Best Solutions 3

Best Solutions M

New Solutions 1

New Solutions 2

New Solutions 3

New Solutions M

Fitness

Operations

New Pop.
Conclusions

Evolutionary Computation shows promising approaches for Manifold Clustering. There are still some problems that need to be solve: like manifold intersection, improve the number of Manifold selection and more stream-based memory.
A Tutorial on Manifold Clustering using Genetic Algorithms

Hector D. Menendez

Department of Computer Science
University College London

2nd September 2015