Hypergraph Clustering based on Game Theory

Ahmed Abdelkader, Nick Fung, Ang Li and Sohil Shah

May 8, 2014

1 Introduction

Data clustering considers the problem of grouping data into clusters based on its similarity measure. It is one of the central problems for data analysis, with a wide applications to a variety of areas such as marketing research, data mining and social behavior analysis.Real world objects can be modeled as points in a high dimensional metric. So clustering these real world data is corresponded to assign each point to a cluster label.

A classic approach to clustering is called K-means algorithm which randomly selects k initial cluster centers and iteratively assigns each data point and updates the cluster centers. A lot of approaches like K-means algorithm considers the pairwise distance as a similarity measurement. In that case, the clustering result is produced by optimizing an objective function which maximizes the distances between pairs of points from different clusters and simultaneously minimizes the distances between pairs of points from the same clusters. Fig.1 shows an example of clustering points into 3 clusters based on their pairwise similarity measure.

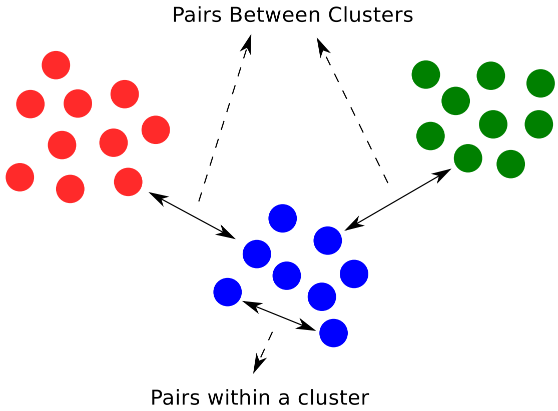


Figure 1: An example of clustering data points into 3 clusters which are annotated using 3 different colors

However, pairwise distances are not enough to measure more complex data relations. For example, if points are called similar when they belong to the same straight line. Since every pair of points can determine a straight line going through them exactly, a higher order similarity measure has to be defined for

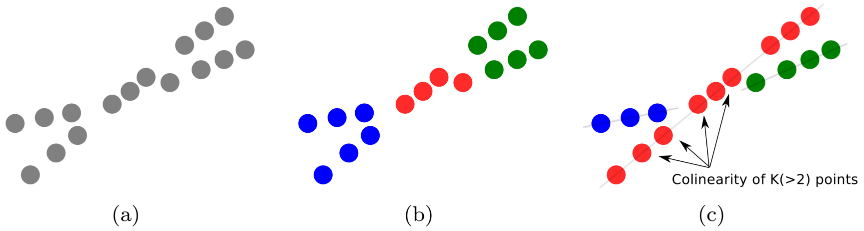


Figure 2: An example of line clustering: (a) original data; (b) a possible clustering result using pairwise distances (c) an ideal line clustering result based on the collinearity measure of more than 2 points

this situation. In this case, more than 2 points needs to be evaluated together. In fig.2, we show an example of the line clustering problem.

1.1 Hypergraph Representation

In order to represent the high order data similarity, we introduce the concept of hypergraph. Hypergraph is a generalization of traditional graph in that each edge can contains more than 2 vertices.

Definition 1.1 (Hypergraph)**.** *A hypergraph H is a pair $H = (V, E)$ where V is a set of elements called nodes or vertices, and E is a set of non-empty subsets of V called hyperedges or edges.*

Definition 1.2 (Weighted Hypergraph)**.** *A weighted hypergraph $H(V, E, \omega)$ is a hypergraph where each hyperedge is associated with a weight defined by ω.*

Definition 1.3 (Weighted k-graph)**.** *A weighted k-graph (aka k-uniform hypergraph) $H(V, E, \omega)$ is a weighted hypergraph such that all its hyperedges have size k.*

The higher order data similarity can be represented by a weighted hypergraph. Therefore, the general clustering problem is reduced to a hypergraph clustering problem:

Given a k-graph $H(V, E, \omega)$ where for each vertex combinations $(v_1, v_2, \ldots, v_k) \in V$, their similarity (the possibility that they come from the same cluster) is defined by $\omega(v_1, v_2, \ldots, v_k) \in [0, 1]$. The Hypergraph Clustering problem is to cluster the vertices from V into multiple clusters $\{C_1, C_2, \ldots\}$ (the total number of clusters is unknown) such that

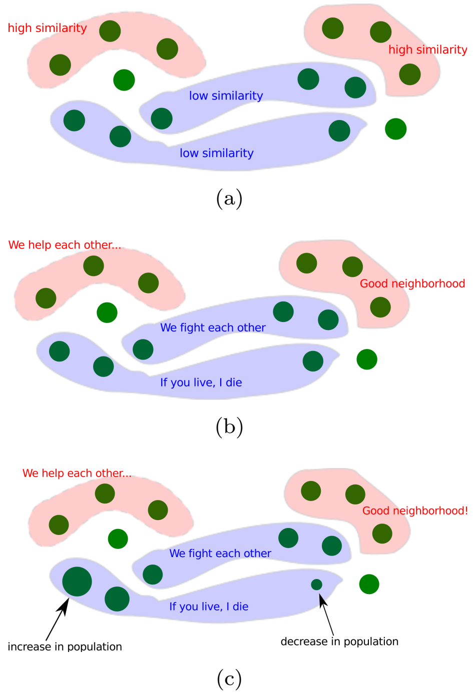
1. each vertex belongs to one and only one cluster;
2. vertices from the same cluster have higher similarities;
3. vertices from different clusters have lower similarities.

One of the main problem of clustering is to define an objective function for globally measuring the "higher" and "lower" terms. A popular way to do that is to sum up all of the weights within a cluster and try to minimize the weight sums for all clusters.

Generally, there are two main streams of relating the cllustering problem to game theory concepts. One is by modeling the problem as population games and looking for the evolutionary stable states [1, 5]. The other is by modeling the problem as a coalition game and analyzing its solution space using game theory approaches [3, 2].

1.2 Clustering based on evolutionary game theory

One of the main approaches is based on evolutionary game theory. The clustering problem is modeled using replicator dynamics concept which is inspired by Darwinian evolution theory. As fig.3 shows, the original data is represented using a hypergraph. Each hyperedge is associated with a weight. If we consider each point as a population of people. The population is uniformly in the beginning (represented by the circle area in fig.3a). A large weight for each hyperedge means these corresponding peoples are easier to exist together. Similarly, a small weight generally means these peoples are fairly difficult to live together in which case the peoples are conflicting and competing with each other. There has to be one side of the peoples getting less and less population, e.g. fig.3c.



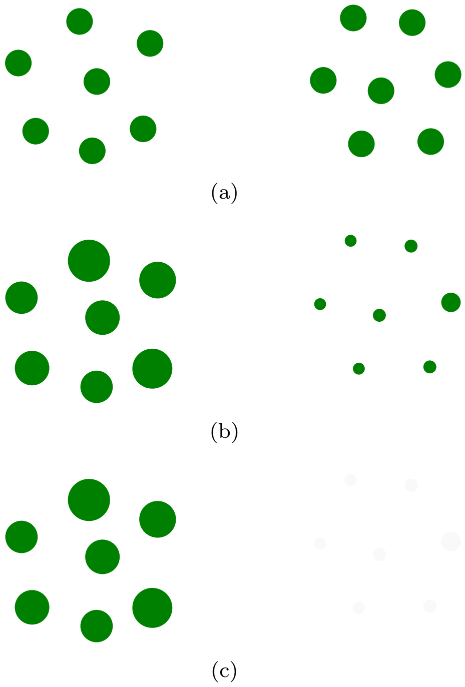
(a)

(b)

(c)

Figure 3: Replicator Dyanmics: (a) the original hypergraph representation of data (b) an analogy to population evolution (c) a possible population profile in the next generation

As is known, the ability for a people to live is related with their population. If some peoples have the trend to getting less population, there are probably some other peoples stronger than them and they become weaker. Gradually, these peoples vanished and no longer exist in the community. Simultaneously, if some peoples get more population during the evolutions. They could become stronger and stronger. One likely evolution states are shown in fig.4. In this case, only one side of the peoples are left at the end and these peoples can be proved to be strongly connected.



(a)

(b)

(c)

Figure 4: Clustering by Replicator Dynamics: (a) the original population (b) population during evolutions (c) the final evolutionary stable states of the population

It is noticed that the population left at the stable state can be corresponded to a cluster of the original data points. Therefore, if the replicator dynamics procedure is repeated, the original data can be partitioned into multiple clusters.

2 Hypergraph Clustering

2.1 General formulation

The problem of clustering a k-graph $H(V, E, \omega)$ formed by a set of N vertices $V = \{1, \ldots, N\}$, set of hyperedge $E \subseteq V^k$ and a real value affnity function $\omega : e \mapsto R$, can be mathematically defined as solving,

$$C^* = \arg\max_{\mathbf{C}} \; S(C) \tag{1}$$

$$\text{s.t.} \quad S(C) = \frac{1}{m^k} \sum_{e \in C : C \subseteq E} \omega(e) \tag{2}$$

where $S(C)$ is the cluster score of cluster $C \subseteq E$ each with m vertices.
This can be reformulated using an assignment vector,

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} \sum_{e \in E} \omega(e) \prod_{v_i \in e} x_{v_i} \tag{3}$$

such that

$$\mathbf{x} \in \left\{0, \frac{1}{m}\right\}^N \quad \text{where} \quad \mathbf{x} = (x_1, x_2, \dots, x_{|V|}) \tag{4}$$

Thus finding a good cluster means finding a subset of vertices with high cluster score. Solving for 3 would be an expensive combinatorial problem and thus NP-Hard. We will now look at the relax version of this problem from the game theoretic point of view.

3 Non-Cooperative Games

Let us consider k players game $\Gamma = (P, S, \pi)$ where $P = \{1, \dots, k\}$ each with N pure strategies $S = \{1, \dots, N\}$ and the payoff function be defines by $\pi : S^k \mapsto R$. Let $\mathbf{x}$ be the n-dimensional vector with each x_i denoting the probability of playing the $i^{th}-$strategy and let it be defined over the simplex set,

$$\Delta = \{x \in R^N : \sum_{j \in S} x_j = 1, x_j \geq 0, \forall j \in S\} \tag{5}$$

Let $\mathbf{x}^{(j)}$ denote the mixed-strategy of the j^{th} player. Then the overall expected payoff of the population is given by,

$$u(x^{(1)}, \dots x^{(k)}) = \sum_{(s_1, \dots s_k) \in S^k} \pi(s_1, \dots s_k) \prod_{i=1}^{k} x_{s_i}^{(i)} \tag{6}$$

3.1 Evolutionary Stable Strategy

As explained in the introduction, the solution $\mathbf{x}$ of the above game is to find equillibrium $x \in \Delta$ such that every player obtains some expected payoff and no strategy can prevails upon others. Noe observe that if we solve for Nash equilibria, then we obtain,

$$u(e^i, x^{[k-1]}) \leq u(x^{[k]}), \forall i \in S \tag{7}$$

where e^i denotes the player only playing $i^{th}-$strategy. Thus at Nash equilibrium every player in the population performs at most as well as the overall population expected payoff. But this notion is too weak as it lacks stability under small perturbations. Instead we need strict inequality in the above equilibrium criteria. Formally, for any $y \in \Delta \setminus \{x\}$ and $w_\delta = (1 - \delta)x + \delta y$ we need,

$$u(y, w_\delta^{[k-1]}) < u(x, w_\delta^{[k-1]}) \tag{8}$$

to hold true for sufficiently small ϵ. Indeed, every ESS is necessarily a Nash equilibrium but not the other way around.

3.2 Non-Cooperative Clustering Games

Before we go on to point out the ways to solve the original problem, we would like to briefly mention the analogy of non-cooperative games to hypergraph clustering problem.

- In order to solve the hypergraph clustering problem $H(V, E, \omega)$ mentioned in (3) using the non-cooperative game $\Gamma = (P, S, \pi)$ in (6) we need strong assumption that π is supersymmetric i.e., $\pi_{\sigma(i)} = \pi \ \forall i$.
- Within the class of supersymmetric functions one such trivial payoff function we can think of is,

$$\pi(s_1, \ldots, s_k) = \frac{1}{k!}\omega(s_1, \ldots, s_k), \quad \forall \{s_1, \ldots, s_k\} \in E \tag{9}$$

- The N input data points of clustering problem forms the N pure strategies of each of the k-players. Further, note that as such k players do not exist and they are all virtual players.
- The support of solution x to ESS problem, correspond to the final points belonging to that cluster.
- Multiple clusters are extracted one after another. After the extraction of the best cluster, data points of that cluster is removed and then the above game is reran on the remaining points.
- In [1], it has been shown that the ESS clusters satisfies the two basic properties of an cluster, *Internal coherency* - elements belonging to that cluster has high mutual similarities and *External incoherency* - the overall cluster internal coherency decreases by introducing external elements.

4 Optimization

As mentioned earlier, we will solve original problem (3) using the relax problem (6) using the solution to ESS problem. Observe that the function in (6) is homogeneous polynomial equation and thus it is a convex optimization problem. Further in [1], the author proves that the Nash equilibria of the game Γ are the critical points of $u(x^{[k]})$ and ESS are the strict local maximizers of $u(x^{[k]})$ over the simplex region. Thus it is enough if we find the maxima points of the (6). One popular method to solve this optimization problem is by using projected gradient ascent algorithm on the region Δ. But this requires large number of iterations. Instead we take a look at two popular algorithm in the literature used to solve the above

4.1 Baum-Eagon Algorithm

Any homogeneous polynomial f($\mathbf{x}$) in variable $x \in \Delta$ with nonnegative coefficients can be approximately solve using the following heuristics,

$$x_j^* = x_j \frac{\frac{\partial f(\mathbf{x})}{\partial x_j}}{\sum_{l=1}^{n} x_l \frac{\partial f(\mathbf{x})}{\partial x_l}} \tag{10}$$

Using this heuristics for solving (6), we obtain,

$$x_j(t+1) = x_j(t)\frac{d_j}{u(x(t)^{[k]})} \quad \forall j = 1, \dots n \tag{11}$$

where $d_j = u(e^j, x(t)^{[k-1]})$ and $u(x(t)^{[k]}) = \sum_l x_l d_l$

4.2 Frank-Wolfe Algorithm

In [4], author discusses the approximation algorithm to efficiently solve the hypergraph clustering problem which we found to be very equivalent to Frank-Wolfe algorithm. Frank -Wolfe algorithm is very much from the class of gradient ascent algorithm i.e., line search algorithm except that we do not require a projection step if we start with initial point from the bounded set.
Define a ϵ-bounded simplex set Δ_ϵ s.t. $x \in [0, \epsilon]^N$. With this we presume that the minimum data points in a cluster is $\frac{1}{\epsilon}$.
Initialize $\mathbf{x}(0) \in \Delta_\epsilon, t \leftarrow 0$. Iterate,

1. Compute $\mathbf{d}$.
2. $\mathbf{y}^* \leftarrow \arg\max \mathbf{d}^T\mathbf{y} \quad s.t. \quad \mathbf{y} \in \Delta_\epsilon$.
3. If $\mathbf{d}^T(\mathbf{y}^* - \mathbf{x}(t)) = 0$, return $\mathbf{x}$(t).
4. $\delta^* \leftarrow \arg\max u(w_\delta^{[k]}) \quad s.t. \quad w_\delta = (1-\delta)\mathbf{x}(t) + \delta\mathbf{y}^*$.
5. $\mathbf{x}(t+1) \leftarrow w_{\delta^*}$

where $\mathbf{d}$ is the vector of d_j elements defined in the previous algorithm. The overall complexity of each iteration of all the algorithm is $\mathcal{O}(N^k)$. Frank-Wolfe algorithm converges the fastest with an average of 10 iterations.

5 Cooperative Clustering Using Shapley Value

5.1 Density-Restricted Agglomerative Clustering Algorithm

Using the tools that we have established, we can implement a clustering algorithm based on the work of Garg[3] and Dhamal[2]. This agglomerative technique takes advantage of the Shapley value to establish cluster centers and adds points to these clusters based on a k-wise similarity function. The clusters can be expanded by points that also have similar Shapley values.

The algorithm selects cluster centers from the unallocated points with maximum Shapley value. The maximum Shapley value can be considered as a good cluster center because we saw in the previous section that it is comprised of the sum of similarities. Having good similarity with other points is a good property of a cluster center.

We then add points to the cluster based on a similarity function. Although we showed that similar points have similar Shapley values, we cannot assume that the reverse is also true. For this reason, we need to consider similarity (e.g. Euclidean distance in k=2 settings) when adding points to a cluster. Finally, we can consider points with close similarity and close Shapley value to the cluster center as good expansion points.

Again, consider that the Shapley value is a sum of similarities across all points. In the example case of $k = 2$, we can say that this is a measurement of density. A point that is similar to the cluster center in distance and density will share expansion properties in clustering tightly spaced points.

We implemented this algorithm for both k=2 and k=3 using Euclidean distance and collinearity respectively as similarity functions. We obtained promising results, even comparing favorably against popular clustering techniques such as k-means under certain data sets.

References

[1] Samuel Rota Bulo and Marcello Pelillo. A game-theoretic approach to hypergraph clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(6):1312–1327, June 2013.

[2] Swapnil Dhamal, Satyanath Bhat, K. R. Anoop, and Varun R. Embar. Pattern clustering using cooperative game theory. *CoRR*, abs/1201.0461, 2012.

[3] Vikas K. Garg, Y. Narahari, and M. Narasimha Murty. Novel biobjective clustering (bigc) based on cooperative game theory. *Knowledge and Data Engineering, IEEE Transactions on*, 25(5):1070–1082, May 2013.

[4] Marius Leordeanu and Cristian Sminchisescu. Efficient hypergraph clustering. In *International Conference on Artificial Intelligence and Statistics*, pages 676–684, 2012.

[5] Hairong Liu, Longin Jan Latecki, and Shuicheng Yan. Revealing cluster structure of graph by path following replicator dynamic. *CoRR*, abs/1303.2643, 2013.