Clustering Theory and Spectral Clustering Lecture 1

Sorin Istrail

Department of Computer Science
Brown University, Providence
sorin@cs.brown.edu

April 7, 2020

Outline

1. Ch. 5 Clustering Theory and Spectral Clustering
   - Clustering spaces and distance measures
   - The 'Curse of Dimensionality"
   - Definition of a Distance Measure
2. A Set of Fundamental Distance Measures
   - Euclidean distances
   - Jaccard Distance
   - Cosine Distance
   - Edit Distance
   - Hamming Distance
3. Hierarchical Clustering
   - Hierarchical Clustering in Euclidean Space
   - Hierarchical Clustering in non-Euclidean Space
4. Distributions of Distances in a High-Dimensional Space

Ch. 5 Clustering Theory and Spectral Clustering

Overview for Ch. 5

- Clustering spaces and distance measures
- The "curse of dimensionality"
- Classification of clustering algorithms
- Hierarchical Clustering Algorithms
- k-means Clustering Algorithms
- EM Clustering Algorithms
- Euclidean vs Non-Eulcidian Spaces for Clustering
- An Introduction to Spectral Graph Theory: eigenvalues and eigenvectors in graph theory
- Dimensionality Reduction: Principal Component Analysis
- Spectral Clustering Algorithms

Clustering space and its distance

- A clustering problem is given by a **space** of elements we refer to as **points** which are objects belonging to that space. Think of the space as being a set from which a set points, the **input data set**, are drawn.
- For the spaces for whcih we define clustering problems, we need to have a **distance measure** between any two points in the space.
- Clustering problems could be formulated for spaces which are **Euclidean spaces** or **Non-Euclidean spaces**.

Euclidean (of low dimesnions) spaces for clustering

- **Euclidean spaces of low dimensions**: points are vectors of real numbers.The components of the points (vectors) are called **coordinates**. The number of coordinates is the dimention of the space.

  - Examples of Euclidean spaces and their distances: R^n the n-dimensional real numbers vector space;
    Distance: The **common Euclidean distance**, which is the square root of the sum of the squares of the difference between the coordinates of the points in each dimension.

  - Other distances: the **Manhattan distance** which is defined as the sum of magnitudes of the differences in each dimension; also the L_∞-**distance** which is defined as the maximum magnitude of the difference in any dimension

Non-Euclidean spaces for clustering

- **Non-Eucliden spaces**: Euclidean spaces of very high dimension of spaces are not Euclidean at all
  - Examples of Non-Euclidean spaces and their distances:
    Clustering documents by their topic based on the occurence of common, unusual words in the documents;
    clustering social media communities by the types of preferences they have, e.g., moviegoers by the type of movies they like;
    clustering biological sequences by their similarity and distance measures.
    Examples of distance metrics for Non-Euclidean spaces are the **Jaccard distance**, the **cosine distance** (dot product), the **Hamming distance**, and the **edit distance**.

The 'Curse of Dimensionality"

- It is quite unintuitive to think about high-dimensional Euclidean spaces. They have a set of surpising properties hard to imagine and visualize and thing about: these non-intuitive structures are called **"The Curse of Dimensionality"**

- Here are two shocking propoerties of such high-dimensional Eulidean spaces
  1. **In high-dimensions Euclidean spaces almost all pairs of points are equally far away from one another**
  2. **in high-dimensions Euclidean spaces almost any two vectors are almost orthogonal**

Non-Euclidean Spaces

- A very important property of Euclidean spaces is that the **average of points in a Euclidean space always exists** and is a point in that space.

- Consider the space of finite sets, and there consider the Jaccard distance for two sets S and T ($=$ 1- ratio of the intersetion of S and T and the uninion of S and T).
  The notion of average in this space of sets makes no sense!

- Consider the space of finite strings where we can use the Edit distance.
  It makes no sense to define the notion of average on sets of strings!

Non-Euclidean Spaces

- Vector spaces for which we can use the Cosine distance, may or may not be Euclidean.
  If the vectors are real numbers vectors then the space is Euclidean.

  If we restrict the vectors to have integer coordinates, the space is no longer Euclidean. We cannot find the concept of average of vectors $[1, 5]$ and $[4, 3]$. If we think of them as being part of the two-dimensional Euclidean space, a concept of average could be $[2.5, 4]$ but this point is not in the space of vectors with integer coordinates!
  Similarly, the space of vectors with all components Boolean (i.e., 0 and 1) is also a non-Euclidean space.

Definition of a Distance Measure

- Suppose we have a set of points called a **Space**.
  A **distance measure** on this space is a function $d(x, y)$ that takes two points in the space as arguments and produces a real number, and satisfies the following axioms;
  1. Axiom 1. No negatie distances: $d(x, y) \geq 0$
  2. Axiom 2. Zero distance: $d(x, y) = 0$ if and only iff $x = y$
  3. Axiom 3. Distance is symmetric: $d(x, y) = d(y, x)$
  4. Axiom 4. The triangle inequality: $d(x, y) \leq d(x, z) + d(z, y)$

Five Important Distance Measures

- Euclidean Distances
- Jaccard Distawnce
- Cosine Distance
- Edit Distance
- Hamming Distance

Euclidean distance

- The most well-known distance measure; our intutive notion of distance
- The Euclidean space is the *n*-**dimensional Euclidean space** and consists of all the points that are vectors of n real numbers.
- The **Euclidean distance** d or the L_2-**norm** is defined by

$$d([x_1, ..., x_n], [y_1, ..., y_n]) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Other Euclidean Distances

- For any constant r we can define the L_r-**norm** to be the distance measure d defined by

$$d([x_1, ..., x_n], [y_1, ..., y_n]) = (\sum_{i=1}^{n} \mid x_i - y_i \mid^r)^{\frac{1}{r}}$$

- The L_1-**norm** is also called the **Manhattan distance**, i.e, the distance between two points is the sum of the magnitudes of the differences in each dimension (distance measured amoung grid points as along the streets in Manhattan).
- The L_∞-**norm**, which is the limit as r goes to infinity of the L_r-norm. L_∞ is defined as the maximum of all $\mid x_i - y_i \mid$ for all i.

Jaccard Distance

- The **Jaccard similarity** is a measure of how close two sets are but it is not formally a distance measure. The closer the sets are, the higher the Jaccard similarity. We can associate an actual distance measure to it, whcih will call the **Jaccard distance**.

- the **Jaccard similarity** of two sets S and T is given by

$$SIM(S,T) = \frac{|\, S \cap T \,|}{|\, S \cup T \,|}$$

- the **Jaccard distance** $d(x,y)$ is given by

$$d(x,y) = 1 - SIM(x,y)$$

- i.e., it equals 1 minus the ratio of the sizes of the intersection and the union of the sets x and y.

Cosine Distance

- The **Cosine distance** is defined in spaces that have dimensions e.g., Euclidean spaces with real numbers as coordinates of the vectores. But also applies to **Discrete spaces**, i.e., spaces of n-dimensional vectors with coordinates that are **integers** as well as spaces with vectors having **Boolean** coordinates , i.e., 0 and 1. Discrete spaces are non-Euclidean spaces.

Cosine Distance

- We think of the points (vectors) as representing directions. We do not distinguish between a vector and a multiple of that vector. Then the cosine distance between two points is the angle that the vectors to the points make.

  This angle will always be between 0 and 180 degrees no matter how many dimensions the space has. we can calculate the cosine distance by first computing the cosine of the angle and then applying the arc-cosine function to translate to an angle of the $0 - 180$ degree range.

Cosine Distance

- Given two vectors x and y, the cosine of the angle between them is the **dot product** $x.y$ divided by the product of the L_2-norms of x and of y. For each x, the L_2-norm of x is just the Euclidean distance from the origin; similarly for y.
- The **dot product** of $x = [x_1, ..., x_n]$ and $y = [y_1, ..., y_n]$ is given by

$$[x_1, ..., x_n].[y_1, ..., y_n] = \sum_{i=1}^{n} x_i y_i$$

Edit Distance

- For the space of **strings** we have the **Edit distance**. The distance between two strings $x = x_1...x_n$ and $y = y_1...y_n$ is the smallest number of insertions and deletions of single characters that will convert x to y

Hamming Distance

- Given a space of vectors, we define the **Hamming Distance** between two vectors to be the number of components in which they differ.
- Usually, the Hamming distance is used when the vectors are Boolean, i.e. consist of 0s and 1s only.

Hierarchical Clustering in Euclidean and non-Euclidean Space

- The **Hierarchical Clustering algorithm** in Euclidean spaces uses the notion of **Centroid** and can be used only on relatively small data sets
- When the space is non-Euclidean there are additional problems associated with hierarchical clustering; we will consider **"clustroids"** and new ways to represent the clusters when there is no centroid point in the cluster

Hierarchical Clustering intuition

- The intuition for the hierachical clustering algorithm is as follows.
- We begin with every point in its own cluster.
- As the algorithm progresses larger clusters are constructed by combining two smaller clusters
- Issues that we need to consider
  - CLUSTER REPRESENTATION: How do we represent the clusters?
  - MERGING RULE: How to determine which clusters to merge?
  - STOPPING RULE: When the algorithm will stop merging clusters?

Hierarchical Clustering in Euclidean spaces: Cluster Representation

- We assume that the clustering space is Euclidean
- This allows us to represent a cluster by its **centroid** or average of the points in the cluster
- In a cluster with just one point, that point is the centroid, so we can initialize the clusters this way. We start the algorithm by placing each point in its own cluster, with the point as its centroid.

Hierarchical Clustering in Euclidean spaces: Merging Rule

- We use as the distance between clusters the Euclidean distance between their centroids.
- We merge then the two clusters at shortest distance of each other; if ties, break ties arbitrarily.
- There are other possible definition of the merging rule, e.g., using different distance measures

Hierarchical Clustering in Euclidean spaces: Stopping Rule

- There are several ways to defining stopping rules for the hierarchical clustering
  1. If the number of clusters k is given as part of input, then we need to stop when there are k clusters
  2. Another stopping option is to stop the algorithm when the next merging produces a cluster that is inadequate. There are number of such tests of adquacy of a cluster, e.g., we can require that the the average distance in a cluster between the centroid and any point in the cluster to be no greater than some constant distance.Such restrictions should be based on the structure of the data domain that we cluster
  3. The most common rule is to stop the algorithm whenre there is only one cluster.

Hierarchical Clustering in Euclidean spaces: The resulting clusterig represented as a tree

- We present the output of the algorithm as a tree representing the way in which all the points were combined through the merging rule.
- This representation is inspired by the philogeny tree construction algorithms where brancing represented evolutionary brancing events and the distance measure captures evolutionary time approximations. The philogeny tree algorithms clusterig space is non-Eulcidean but the principles of hierachical clustering carry over with some modifications to non-Euclidean clustering.

Hierarchical Clustering in Euclidean spaces: Time complexity and speed ups

- The algorithm for hierarchical clustering is not really efficient. At each step it must compute the distances between each pair of clusters in order to find the best merge move. The initial step takes $O(n^2)$ and then the next steps take respectively $(n-1)^2, (n-2)^2, \ldots$ and so the total time (the sum of squares up to ne) is $O(n^3)$. This is a cubic time algorithm. As such it can be applied only to clustering of small number of points

Hierarchical Clustering in Euclidean spaces: Time complexity and speed ups

- Here is how we can speed up the algorithm
  - We start by computing all pairwise distances between the n points, taking time $O(n^2)$
  - Place all the pair of points and their distances in a priority queue (a basic data structure) so that it can always find the smallest distance in one step; this takes time $O(n^2)$
  - When we decide to merge two clusters A and B we remove all entries in the priority queue that involves one of these two clusters; this requires time $O(n \log n)$ as there are at most $2n$ deletions to be performed and priority queue deletions can be done in $O(\log n)$

Hierarchical Clustering in Euclidean spaces: Time complexity and speed ups

- The algorithm then computes all the distances between the new cluster and the remaining clusters; this work takes also $O(n \log n)$ as there are at most n entries to be inserted into the priority queue, and insertions in the priority queue takes $O(\log n)$
- Since the last two steps are executed at most n times and the first two steps are executed only once, the overall running time of this algorithm is $O(n^2 \log n)$. Clearly this is faster time than $O(n^3)$. Still this implies that n cannot be too big.

Hierarchical Clustering in Euclidean spaces: more ideas on algorithm design for Merging rules

- We can use different Merge Rules than MERGE RULE 0: the pair of clusters with the smallest distance between centroids. These new rules will give rise in general to entirely different clustering structures
  - MERGE RULE 1: use the distance between two clusters to be the minimum of the distances between any two points, one chosen from each cluster.
  - MERGE RULE 2: use the distane betweent two clusters to be the average distance of all pairs of points, one from each cluster.

Hierarchical Clustering in Euclidean spaces: more ideas on algorithm design for Merging rules

- MERGE RULE 3: The **radius** of a cluster is is the maximum distance between any all points and the centroid. Use this new merging rule: merge two clusters whose resulting radius is the smallest. Other choices are:
  - use minimal average distance between all points and the centroid
  - use minimal sum of squares between all points and the centroid
- MERGE RULE 4:The **diameter** of a cluster is the maximum distance between any two points fo the cluster. use as a merging rule to merge two clusers whose resulting cluser has the smallest possible diameter. Similar variations could include average and sum of squares diameter computations.

Hierarchical Clustering in Euclidean spaces: more ideas on algorithm design for Stopping rules

- STOPPING RULE 0: is when we have obtained k clusters, wher k is given by the input
- STOPPING RULE 1: stop when the diameter of the cluster that results from the best merger exceeds a threshold, set in advance. Variations on the theme: radius and its variations given above

Hierarchical Clustering in Euclidean spaces: more ideas on algorithm design for Stopping rules

- STOPING RULE 2: stop if the **density** of the cluster that results from the best merger is below a threshold, set in advance. One way to define density is to number of cluster points per "unit volume" of the cluster, defined as the ratio of thenumber of points in the cluster divided by the diameter or radius raised to some power. This power is the number of dimensions of the space, but in practice power is chosen as 1 or 2.

iHerarchical Clustering in Euclidean spaces: more ideas on algorithm design for Stopping rules

- STOPPING RULE 3: stop when there is evidence that the next pair of clusters to be merged would yield a bad cluster. One way to think about this is as follows. Compute all the diameters of the clusters as we advance in the algorithm. Track the average of the diameters as we advance in the algorithm. Averave will go up gradually as we move along, meaning that the clustering brings together points that are close to each other. A "bad" clustering move would be one that make the average diameter jump.

Hierarchical Clustering in non-Euclidean spaces

- When the space is non-Eulcidean we need some distance measures that are computed from points:
  - Jaccard distance
  - Cosine distance
  - Edit distance
  - Hamming distance

Hierarchical Clustering in non-Euclidean spaces: Clustroids

- Our Hierachical Clustering algorithms requires distances between points to be computed, but we cannot base the distances by "location".
- We know how to compute the above distance measures;
- What we do not have in an non-Euclidian space is the concept "centroid" and so we cannot represent the cluster by a centroid as before.
- A new concept: **Clustroid** which is just one of the points in the cluster, and we adjust our algorithm design, in a natural ways, using clustroids.

Hierarchical Clustering in non-Euclidean spaces: Clustroids

- We represent a cluster by its clustroid. We pick one of the points in the cluster itself to represent the cluster.
- Ideally the **clustroid** point is close to all other points of the cluster, so in some sense it is the "center".
- We can choose the clustroid in a number of ways, similar with the centroid based algorithms, to minimize the distances between the clustroid and other points in the cluster.

Hierarchical Clustering in non-Euclidean spaces: Clustroids

- Choices for selecting the clustroid point that minimizes:
  - the sum of distances to the other points in the cluster
  - the maximum distance to any other point in the cluster
  - the sum of squares of the distances to other points in the cluster

Hierarchical Clustering in non-Euclidean spaces: Distance between clusters

- Minimum distance between clustroids
- Minimum distance between all pair of points from the two clusters
- Average distance between all pair of points from the two clusters

Hierarchical Clustering in non-Euclidean spaces: Density of the clusters

- Measuring density of a cluster can be generalized to non-Euclinean spaces. We can measure density based on diameter and radius which can be defined for non-Eulcidean spaces as well.
- The **diameter** is still the largest distance between any two points in the cluster
- The **radius** can be defined using the clustroid instead of the centroid
-

Hierarchical Clustering in non-Euclidean spaces: Stopping rules

- The notions of merge and stopping rules can be generalized with no substantial change from Euclidean to non-Euclidean spaces

Distributions of Distances in a High-Dimensional Space

- Consider a d-dimensional Euclidean space.
- Suppose we choose n random points in the unit cube, i.e., points $[x_1, ..., x_n]$ where each x_i is in tange 0 to 1. Suppose d is very large. The Euclidean distance between two random points $[x_1, ..., x_d]$ and $[y_1, ..., y_d]$ is

$$\sqrt{\sum_{i=1}^{d}(x_i - y_i)^2}$$

- Think each x_i and y_i is a random variable chosen uniformally in the range 0 to 1.

Distributions of Distances in a High-Dimensional Space

- The maximum distance between two points is $\sqrt{d}$
- Almost all points will have a distance close to average distance
- If there are essentially no pairs of points that are close, it is hard to build clusters at all
- There is little justificationfor grouping one pair of points and not another

Distributions of Distances in a High-Dimensional Space

- The above argument is based on random points and for real data sets there could be stucture useful even in the high-dimensional spaces
- However, the argument about random data suggests that it will be hard to find these clusters among so many pairs that are all at approximately the same distance