RESEARCH PAPER ON CLUSTER TECHNIQUES OF DATA VARIATIONS

Er. Arpit Gupta 1 ,Er.Ankit Gupta 2,Er. Amit Mishra 3
arpit_jp@yahoo.co.in , ank_mgcgv@yahoo.co.in,amitmishra.mtech@gmail.com
Faculty Of Engineering and Technology(IT department), Mahatma Gandhi Chitrakoot Gramodaya Vishwavidyalaya. Chitrakoot-485780, Satna, Madhya Pradesh, India

Abstract

Cluster analysis divides data into groups (clusters) for the purposes of summarization or improved understanding. For example, cluster analysis has been used to group related documents for browsing, to find genes and proteins that have similar functionality, or as a means of data compression.. In this chapter we provide a short introduction to cluster analysis. We present a brief view of recent techniques which uses a concept-based clustering approach.

Introduction

Cluster analysis divides data into meaningful or useful groups (clusters). If meaningful clusters are our objective, then the resulting clusters should capture the "natural" structure of the data. Cluster analysis is only a useful starting point for other purposes, e.g., data compression or efficiently finding the nearest neighbors of points. Whether for understanding or utility, cluster analysis has long been used in a wide variety of fields: psychology and other social sciences, biology, statistics, pattern recognition, information retrieval, machine learning, and data mining. In this chapter we provide a short introduction to cluster analysis. We present a brief view recent technique, which uses a concept-based approach. In this case, the approach to clustering high dimensional data must deal with the "curse of dimensionality".

Concept of Cluster Analysis

Cluster analysis groups objects (observations, events) based on the information found in the data describing the objects or their relationships. The aim is the objects in a group should be similar (or related) to one another and different from (or unrelated to) the objects in other groups. The greater the similarity (or homogeneity) within a group and the greater the difference between groups, the better the clustering.Cluster analysis is a classification of objects from the data, where by "classification" we mean a labeling of objects with class (group) labels. As such, clustering does not use

previously assigned class labels, except perhaps for verification of how well the clustering worked. Thus, cluster analysis is sometimes referred to as "unsupervised classification" and is distinct from "supervised classification," or more commonly just "classification," which seeks to find rules for classifying objects given a set of pre-classified objects. Classification is an important part of data mining, pattern recognition, machine learning, and statistics (discriminant analysis and decision analysis).

As mentioned above, the term, cluster, does not have a precise definition. However, several working definitions of a cluster are commonly used and are given below. There are two aspects of clustering that should be mentioned in conjunction with these definitions. First, clustering is sometimes viewed as finding only the most "tightly" connected points while discarding "background" or noise points. Second, it is sometimes acceptable to produce a set of clusters where a true cluster is broken into several subclusters (which are often combined later, by another technique). The key requirement in this latter situation is that the subclusters are relatively "pure," i.e., most points in a sub cluster are from the same "true" cluster.

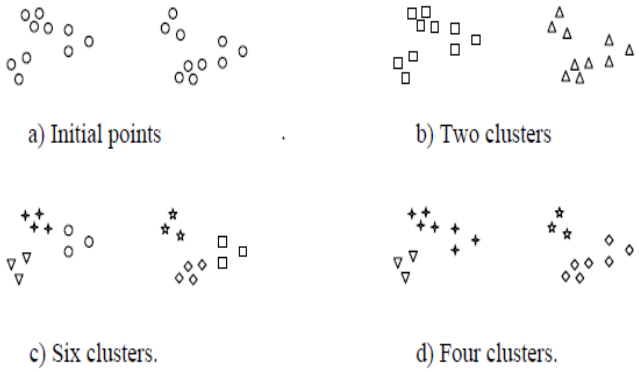
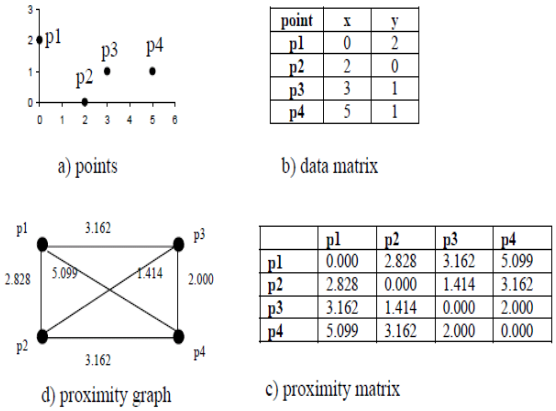


Figure 1: Different clusterings for a set of points.



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

	p1	p2	p3	p4
p1	0.000	2.828	3.162	5.099
p2	2.828	0.000	1.414	3.162
p3	3.162	1.414	0.000	2.000
p4	5.099	3.162	2.000	0.000

Figure 2. Four points, their proximity graph, and their corresponding data and proximity (distance) matrices.

1.Well-Separated Cluster Definition:

A cluster is a set of points such that any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster. Sometimes a threshold is used to specify that all the points in a cluster must be sufficiently close (or similar) to one another. However, in many sets of data, a point on the edge of a cluster may be closer (or more similar) to some objects in another cluster

than to objects in its own cluster. Consequently, many clustering algorithms use the following criterion.

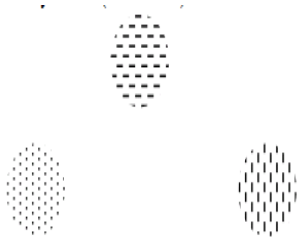


Figure 3: Three well-separated clusters of 2 dimensional points

2.Center-based Cluster Definition:

A cluster is a set of objects such that an object in a cluster is closer (more similar) to the "center" of a cluster, than to the center of any other cluster. The center of a cluster is often a centroid, the average of all the points in the cluster, or a medoid, the "most representative" point of a cluster.

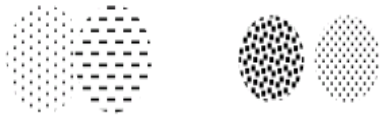


Figure 4: Four center-based clusters of 2 dimensional points.

3.Contiguous Cluster Definition (Nearest Neighbor or Transitive Clustering): A cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster.

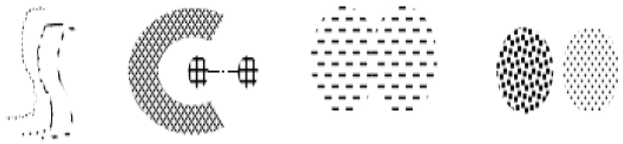


Figure 5: Eight contiguous clusters of 2 dimensional points.

4.Density-based definition: A cluster is a dense region of points, which is separated by low-density regions, from other regions of high density. This definition is more often used when the clusters are irregular or intertwined, and when noise and outliers are present. Notice that the contiguous definition would find only one cluster in Figure . Also note that the three curves don't form clusters since they fade into the noise, as does the bridge between the two small circular clusters.

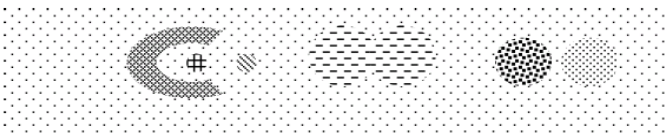


Figure 6: Six dense clusters of 2 dimensional points.

5.Similarity-based Cluster definition: A cluster is a set of objects that are "similar", and objects in other clusters are not "similar." A variation on this is to define a cluster as a set of points that together create a region with a uniform local property, e.g., density or shape. The proximity measure (and the type of clustering used) depends on the attribute type and scale of the data. The three typical types of attributes are shown in Table 1, while the common data scales are shown in Table 2.

Binary	Two values, e.g., true and false
Discrete	A finite number of values, or an integer, e.g., counts
Continuous	An effectively infinite number of real values, e.g., weight.

Table 1: Different attribute types

Qualitative - Nominal - The values are just different names, e.g., colors or zip codes.
Ordinal - The values reflect an ordering, nothing more, e.g., good, better, best
Quantitative - Interval - The difference between values is meaningful, i.e., a unit of Measurement exits. For example, temperature on the Celsius or Fahrenheit scales.
Ratio-The scale has an absolute zero so that ratios are meaningful. Examples are physical quantities such as electrical current, Pressure, or temperature on the Kelvin scale.

Table 2: Different attribute scales

Euclidean Distance and Some Variations

The most commonly used proximity measure, at least for ratio scales (scales with an absolute 0) is the Min kowski metric, which is a generalization of the distance between points in Euclidean space.

$$p_{ij} = \left(\sum_{k=1}^{d} \left| x_{ik} - x_{jk} \right|^{r} \right)^{1/r}$$

where,
r is a parameter, d is the dimensionality of the data object, and x_{ik} and x_{jk} are, respectively, the k^{th} components of the i^{th} and j^{th} objects, x_i and x_j.

For r = 1, this distance is commonly known as the L_1 norm or city block distance. If r = 2, the most

common situation, then we have the familiar L_2 norm or Euclidean distance. Occasionally one might encounter the L_{max} norm (L_∞ norm), which represents the case $r \rightarrow \infty$. Figure 7 gives the proximity matrices for the L1, L2 and L_∞ distances, respectively, using the data matrix from Figure 2.

The r parameter should not be confused with the dimension, d. For example, Euclidean, Manhattan and supremum distances are defined for all values of d, 1, 2, 3, …, and specify different ways of combining the differences in each dimension (attribute) into an overall distance.

point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

L2	p1	p2	p3	p4
p1	0.000	2.828	3.162	5.099
p2	2.828	0.000	1.414	3.162
p3	3.162	1.414	0.000	2.000
p4	5.099	3.162	2.000	0.000

L1	p1	p2	p3	p4
p1	0.000	4.000	4.000	6.000
p2	4.000	0.000	2.000	4.000
p3	4.000	2.000	0.000	2.000
p4	6.000	4.000	2.000	0.000

L_∞	p1	p2	p3	p4
p1	0.000	2.000	3.000	5.000
p2	2.000	0.000	1.000	3.000
p3	3.000	1.000	0.000	2.000
p4	5.000	3.000	2.000	0.000

Figure 7. Data matrix and the corresponding L1, L2, and L_∞ proximity matrices.

Finally, note that various Minkowski distances are metric distances. In other words, given a distance function, dist, and three points **a**, **b**, and **c**, these distances satisfy the following three mathematical properties: reflexivity (dist(**a**, **a**) = 0), symmetry (dist(**a**, **b**) = dist(**b**, **a**)), and the triangle inequality (dist(**a**, **c**) ≤ dist(**a**, **b**) + dist(**b**, **a**)). Not all distances or similarities are metric, e.g., the Jaccard measure of the following section. This introduces potential complications in the clustering process since in such cases, **a** similar (close) to **b** and **b** similar to **c**, does not necessarily imply **a** similar to **c**. The concept based clustering, which we discuss later, provides a way of dealing with such situations.

Hierarchical and Partitional Clustering

The main distinction in clustering approaches is between hierarchical and partitional approaches. Hierarchical techniques produce a nested sequence of partitions, with a single, all-inclusive cluster at the top and singleton clusters of individual points at the bottom. Each intermediate level can be viewed as combining (splitting) two clusters from the next lower (next higher) level. (Hierarchical clustering techniques that start with one large cluster and split it are termed "divisive," while approaches that start with clusters containing a single point, and then merge them are called "agglomerative.") While most hierarchical algorithms involve joining two clusters or splitting a cluster into two sub-clusters, some hierarchical algorithms join more than two clusters in one step or split a cluster into more than two sub-clusters.

Partitional techniques create a one-level (unnested) partitioning of the data points. If K is the desired number of clusters, then partitional approaches typically find all K clusters at once. Contrast this with traditional hierarchical schemes, which bisect a cluster to get two clusters or merge two clusters to get one. Of course, a hierarchical approach can be used to generate a flat partition of K clusters, and likewise, the repeated application of a partitional scheme can provide a hierarchical clustering.

There are also other important distinctions between clustering algorithms: Does a clustering algorithm cluster on all attributes simultaneously (polythetic) or use only one attribute at a time (monothetic)? Does a clustering technique use one object at a time (incremental) or does the algorithm require access to all objects (non-incremental)? Does the clustering method allow a cluster to belong to multiple clusters (overlapping) or does it assign each object to a single cluster (non-overlapping)? Note that overlapping clusters are not the same as fuzzy clusters, but rather reflect the fact that in many real situations, objects belong to multiple classes.

Specific Partitional Clustering Techniques: K-means

The K-means algorithm discovers K (non-overlapping) clusters by finding K centroids ("central" points) and then assigning each point to the cluster associated with its nearest centroid. (A cluster centroid is typically the mean or median of the points in its cluster and "nearness" is defined by a distance or similarity function.) Ideally the centroids are chosen to minimize the total "error," where the error for each point is given by a function that measures the discrepancy between a point and its cluster centroid, e.g., the squared distance. Note that a measure of cluster "goodness" is the error contributed by that cluster. For squared error and Euclidean distance, it can be shown [And73] that a gradient descent approach to minimizing the squared error yields the following basic K-means algorithm. (The previous discussion still holds if we use similarities instead of distances, but our optimization problem becomes a maximization problem.)

Basic K-means Algorithm for finding K clusters.

1. Select K points as the initial centroids.
2. Assign all points to the closest centroid.
3. Recompute the centroid of each cluster.

4. Repeat steps 2 and 3 until the centroids don't change (or change very little).

K-means has a number of variations, depending on the method for selecting the initial centroids, the choice for the measure of similarity, and the way that the centroid is computed. The common practice, at least for Euclidean data, is to use the mean as the centroid and to select the initial centroids randomly.

In the absence of numerical problems, this procedure converges to a solution, although the solution is typically a local minimum. Since only the vectors are stored, the space requirements are O(m*n), where m is the number of points and n is the number of attributes. The time requirements are O(I*K*m*n), where I is the number of iterations required for convergence. I is typically small and can be easily bounded as most changes occur in the first few iterations. Thus, the time required by K-means is efficient, as well as simple, as long as the number of clusters is significantly less than m.

Theoretically, the K-means clustering algorithm can be viewed either as a gradient descent approach which attempts to minimize the sum of the squared error of each point from cluster centroid or as procedure that results from trying to model the data as a mixture of Gaussian distributions with diagonal covariance matrices .

Specific Hierarchical Clustering Techniques: MIN, MAX, Group Average

In hierarchical clustering the goal is to produce a hierarchical series of nested clusters, ranging from clusters of individual points at the bottom to an all-inclusive cluster at the top. A diagram called a dendogram graphically represents this hierarchy and is an inverted tree that describes the order in which points are merged (bottom-up, agglomerative approach) or clusters are split (top-down, divisive approach). One of the attractions of hierarchical techniques is that they correspond to taxonomies that are very common in the biological sciences, e.g., kingdom, phylum, genus, species, … (Some cluster analysis work occurs under the name of "mathematical taxonomy.") Another attractive feature is that hierarchical techniques do not assume any particular number of clusters. Instead, any desired number of clusters can be obtained by "cutting" the dendogram at the proper level. Finally, hierarchical techniques are thought to produce better quality clusters .

In this section we describe three agglomerative hierarchical techniques: MIN, MAX, and group average. For the single link or MIN version of hierarchical clustering, the proximity of two clusters is defined to be minimum of the distance (maximum of the similarity) between any two points in the different clusters. The technique is called single link, because if you start with all points as singleton clusters, and add links between points, strongest links first, these single links combine the points into clusters. Single link is good at handling non-elliptical shapes, but is sensitive to noise and outliers.

For the complete link or MAX version of hierarchical clustering, the proximity of two clusters is defined to be maximum of the distance (minimum of the similarity) between any two points in the different clusters. The technique is called complete link because, if you start with all points as singleton clusters, and add links between points, strongest links first, then a group of points is not a cluster until all the points in it are completely linked, i.e., form a clique. Complete link is less susceptible to noise and outliers, but can break large clusters, and has trouble with convex shapes.

For the group average version of hierarchical clustering, the proximity of two clusters is defined to be the average of the pairwise proximities between all pairs of points in the different clusters. Notice that this is an intermediate approach between MIN and MAX. This is expressed by the following equation:

$$proximity(cluster1, cluster2) = \frac{\sum_{\substack{p_1 \in cluster1 \\ p_2 \in cluster2}} proximity(p_1, p_2)}{size(cluster1) * size(cluster2)}$$

Figure 8 shows a table for a sample similarity matrix and three dendograms, which respectively, show the series of merges that result from using the MIN, MAX, and group average approaches. In this simple case, MIN and group average produce the same clustering.

	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00

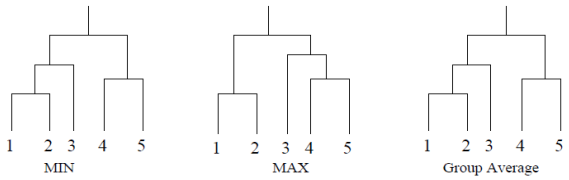


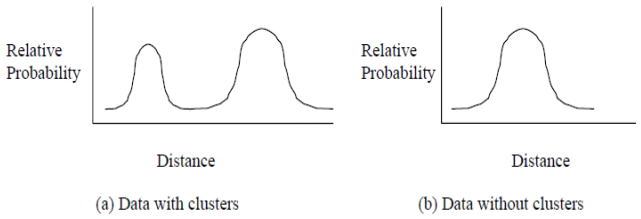
Figure 8: Dendograms produced by MIN, MAX and group average hierarchical clustering technique.

The "Curse of Dimensionality"

It was Richard Bellman who apparently originated the phrase, "the curse of dimensionality," in a book on control theory [Bel61]. The specific quote from [Bel61], page 97, is "In view of all that we have said in the forgoing sections, the many obstacles we appear to have surmounted, what casts the pall over our victory celebration? It is the curse of dimensionality, a malediction that has plagued the scientist from the earliest days." The issue referred to in Bellman's quote is the impossibility of optimizing a function of many variables by a brute force search on a discrete multidimensional grid. (The number of grids points increases exponentially with dimensionality, i.e., with the number of variables.) With the passage of time, the "curse of dimensionality" has come to refer to any problem in data analysis that results from a large number of variables (attributes).

In general terms, problems with high dimensionality result from the fact that a fixed number of data points become increasingly "sparse" as the dimensionality increase. To visualize this, consider 100 points distributed with a uniform random distribution in the interval [0, 1]. If this interval is broken into 10 cells, then it is highly likely that all cells will contain some points. However, consider what happens if we keep the number of points the same, but distribute the points over the unit square. (This corresponds to the situation where each point is two-dimensional.) If we keep the unit of discretization to be 0.1 for each dimension, then we have 100 two-dimensional cells, and it is quite likely that some cells will be empty. For 100 points and three dimensions, most of the 1000 cells will be empty since there are far more points than cells. Conceptually our data is "lost in space" as we go to higher dimensions.

For clustering purposes, the most relevant aspect of the curse of dimensionality concerns the effect of increasing dimensionality on distance or similarity. In particular, most clustering techniques depend critically on the measure of distance or similarity, and require that the objects within clusters are, in general, closer to each other than to objects in other clusters. (Otherwise, clustering algorithms may produce clusters that are not meaningful.) One way of analyzing whether a data set may contain clusters is to plot the histogram (approximate probability density function) of the pairwise distances of all points in a data set (or of a sample of points if this requires too much computation.) If the data contains clusters, then the graph will typically show two peaks: a peak representing the distance between points in clusters, and a peak representing the average distance between points. Figures 9a and 9b, respectively, show idealized versions of the data with and without clusters. Also see [Bri95]. If only one peak is present or if the two peaks are close, then clustering via distance based approaches will likely be difficult. Note that clusters of different densities could cause the leftmost peak of Fig. 9a to actually become several peaks.



(a) Data with clusters (b) Data without clusters

Figure 9: Plot of interpoint distances for data with and without clusters.

There has also been some work on analyzing the behavior of distances for high dimensional data. , it is shown, for certain data distributions, that the relative difference of the distances of the closest and farthest data points of an independently selected point goes to 0 as the dimensionality increases, i.e.,

$$\lim_{d \to \infty} \frac{MaxDist - MinDist}{MinDist} = 0$$

For example, this phenomenon occurs if all attributes are i.i.d. (identically and independently distributed). Thus, it is often said, "in high dimensional spaces, distances between points become relatively uniform." In such cases, the notion of the nearest neighbor of a point is meaningless. To understand this in a more geometrical way, consider a hyper-sphere whose center is the selected point and whose radius is the distance to the nearest data point. Then, if the relative difference between the distance to nearest and farthest neighbors is small, expanding the radius of the sphere "slightly" will include many more points.

In a theoretical analysis of several different types of distributions is presented, as well as some supporting results for real-world high dimensional data sets. This work was oriented towards the problem of finding the nearest neighbors of points, but the results also indicate potential problems for clustering high dimensional data.

The work just discussed was extended to look at the absolute difference, **MaxDist** – **MinDist**, instead of

the relative difference. It was shown that the behavior of the absolute difference between the distance to the closest and farthest neighbors of an independently selected point depends on the distance measure. In particular, for the L_1 metric, **MaxDist** – **MinDist** increases with dimensionality, for the L_2 metric, **MaxDist** – **MinDist** remains relatively constant, and for the L_d metric, $d \geq 3$, **MaxDist** – **MinDist** goes to 0 as dimensionality increase. These theoretical results were also confirmed by experiments on simulated and real datasets. The conclusion is that the L_d metric, $d \geq 3$, is meaningless for high dimensional data.

The previous results indicate the potential problems with clustering high dimensional data sets, at least in cases where the data distribution causes the distances between points to become relatively uniform. However, things are sometimes not as bad as they might seem, for it is often possible to reduce the dimensionality of the data without losing important information. For example, sometimes it is known apriori that only a smaller number of variables are of interest. If so, then these variables can be selected, and the others discarded, thus reducing the dimensionality of the data set. More generally, data analysis (clustering or otherwise) is often preceded by a "feature selection" step that attempts to remove "irrelevant" features. This can be accomplished by discarding
features that show little variation or which are highly correlated with other features. (Feature selection is a complicated subject in its own right.)

Another approach is to project points from a higher dimensional space to a lower dimensional space. The idea here is that that often data can be approximated reasonably well even if only a relatively small number of dimensions are kept, and thus, little "true" information is lost. Indeed, such techniques can, in some cases, enhance the data analysis because they are effective in removing noise. Typically this type of dimensionality reduction is accomplished by applying techniques from linear algebra or statistics such as Principal Component Analysis (PCA) or Singular Value Decomposition (SVD)
To make this more concrete we briefly illustrate with SVD. (Mathematically less inclined readers can skip this paragraph without loss.) A singular value decomposition of an m by n matrix, M, expresses M as the sum of simpler rank 1 matrices as follows:

$M = \sum_{i=1}^{n} s_i u_i v_i^T$, where s_i, a scalar, is the i^{th} singular value of M, u_i is the i^{th} left singular vector, and v_i is the i^{th} right singular vector. All singular values beyond the first r, where r = rank(M) are 0 and all left (right) singular vectors are orthogonal to each other and are of unit length. A matrix can be approximated by omitting some of the terms of the series that correspond to non-zero singular values. (Singular values are non-negative and ordered by decreasing magnitude.) Since the magnitudes of these singular values often decrease rapidly, an approximation based on a relatively small number of singular values, e.g., 50 or 100 out of 1000, is often sufficient for a productive data analysis.

Furthermore, it is not unusual to see data analyses that take only the first few singular values. However, both feature selection and dimensionality reduction approaches based on PCA or SVD may be inappropriate if different clusters lie in different subspaces. Indeed, we emphasize that for many high dimensional data sets it is likely that clusters lie only in subsets of the full space. Thus, many algorithms for clustering high dimensional data automatically find clusters in subspaces of the full space. One example of such a clustering technique is "projected" clustering which also finds the set of dimensions appropriate for each cluster during the clustering process. More techniques that find clusters in subspaces of the full space will be discussed in Section 4.
In summary, high dimensional data is not like low dimensional data and needs different approaches. The next section presents recent work to provide clustering techniques for high dimensional data. While some of this work is represents different developments of a single theme, e.g., grid based clustering, there is considerable diversity, perhaps because of high dimensional data, like low dimensional data is highly varied.

A "Concept-Based" Approach to Clustering High Dimensional Data

A key feature of some high dimensional data is that two objects may be highly similar even though commonly applied distance or similarity measures indicate that they are dissimilar or perhaps only moderately similar [GRS99]. Conversely, and perhaps more surprisingly, it is also possible that an object's nearest or most similar neighbors may not be as highly "related" to the object as other objects which are less similar. To deal with this issue we have extended previous approaches that define the distance or similarity of objects in terms of the number of nearest neighbors that they share. The resulting approach defines similarity not in terms of

shared attributes, but rather in terms of a more general notion of shared concepts. The rest of this section details our work in finding clusters in these "concept spaces," and in doing so, provides a contrast to the approaches of the previous section, which were oriented to finding clusters in more traditional vector spaces.

Concept Spaces

For our purposes, a concept will be a set of attributes. As an example, for documents a concept would be a set of words that characterize a theme or topic such as "Art" or "Finance." The importance of concepts is that, for many data sets, the objects in the data set can be viewed as being generated from one or more sets of concepts in a probabilistic way. Thus, a concept-oriented approach to documents would view each document as consisting of words that come from one or more concepts, i.e., sets of words or vocabularies, with the probability of each word being determined by an underlying statistical model. We refer to data sets with this sort of structure as concept spaces, even though the underlying data may be represented as points in a vector space or in some other format. The practical relevance of concept spaces is that data belonging to concept spaces must be treated differently in terms of how the similarity between points should be calculated and how the objects should be clustered.

To make this more concrete we detail a concept-based model for documents. Figure 14a shows the simplest model, which we call the "pure concepts" model. In this model, the words from a document in the i^{th} class, C_i, of documents come from either the general vocabulary, V_0, or from exactly one of the specialized vocabularies, V_1, V_2, …, V_p. For this model the vocabularies are just sets of words and possess no additional structure. In this case, as in the remaining cases discussed, all vocabularies can overlap. Intuitively, however, a specialized word that is found in a document is more likely to have originated from a specialized vocabulary than from the general vocabulary.

Figure 14b is much like Figure 14a and shows a slightly more complicated (realistic) model, which we call the "multiple concepts" model. The only difference from the previous model is that a word in a document from a particular class may come from more than one specialized vocabulary. More complicated models are also possible.

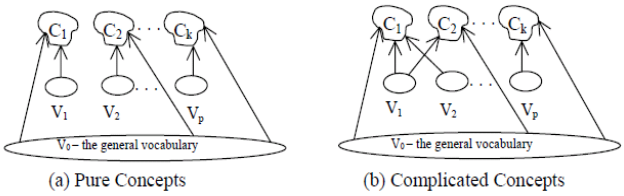


Figure 14: Different concept models.

A statistical model for the concept-based models shown above could be the following. A word, w, in a document, d, from a cluster Ci, comes with one or more vocabularies with a probability given by $P(w \mid Ci) = \Sigma P(w \mid Vj) * P(Vj \mid Ci)$. For the pure concepts model, each word of a document comes only from the general vocabulary and one of the specialized vocabularies. For the multiple concepts model, each word of a document comes from one or more specialized vocabularies.

Our Clustering Approach

We begin by calculating the document similarity matrix, i.e., the matrix which gives the cosine similarity for each pair of documents. Once this similarity matrix is calculated, we find the first n nearest neighbors for each document. (Every object is considered to be its own 0^{th} neighbor.) In the nearest neighbor graph, there is a link from object i to object j, if i and j both have each other in their nearest neighbor list. In the shared nearest neighbor graph, there is a link from i to j if there is a link from i to j in the nearest neighbor graph and the strength of this link is equal to the number of shared nearest neighbors of i and j.

At this point, we could just apply a threshold, and take all the connected components of the shared nearest neighbor graph as our final clusters [JP73]. However, this threshold would need to be set too high since this is a single link approach and

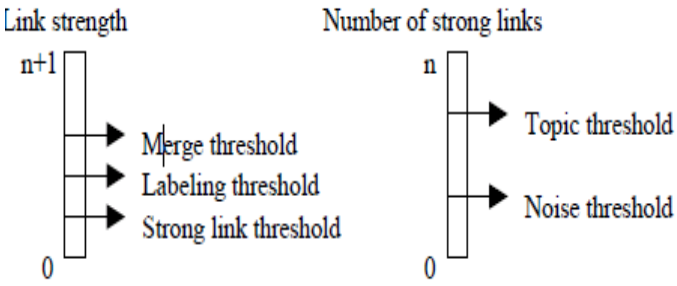


Figure 16: Different types of parameters

will give poor results when patterns in the dataset are not very significant. On the other hand, when a high threshold is applied, a natural cluster will be split into many small clusters due to the variations in the similarity within the cluster. We address these

problems with the clustering algorithm described below.

There are two types of parameters used in this algorithm: one type relates to the strength of the links in the shared nearest neighbor graph, the other type relates to the number of strong links for an object. If the strength of a link is greater than a threshold, then that link is labeled as a strong link.

The details of our shared nearest neighbor clustering algorithm are as follows:

1) For every point i in the dataset, calculate the connectivity, conn[i], the number of strong links the point has.

2) For a point i in the dataset, if conn[i] < noise threshold, then that point is not considered in the clustering since it is similar to only a few of its neighbors. Similarly, if conn[i] > topic threshold, then that point is similar to most of its neighbors and is chosen to represent its neighborhood.

3) For any pair of points (i, j) in the dataset, if i and j share significant numbers of their neighbors, i.e. the strength of the link between i and j is greater than the merge threshold, then they will appear together in the final clustering if either one of them (or both) is chosen to be a representative. Our algorithm will not suffer from the effects of transitivity since every other point on a chain of links has to be chosen to be a representative. In other words, two objects that are not directly related will be put in the same cluster only if there are many other objects between them that are connected with strong links, half of which must represent their own neighborhood.

4) Labeling step: Having defined the representative points and the points strongly related to them, we can bring back some of the points that did not survive the merge threshold. This is done by scanning the shared nearest neighbor list of all the points that are part of a cluster, and checking whether those points have links to points that don't belong to any cluster and have a link strength greater than the labeling threshold.

After applying the algorithm described above, there may be singleton clusters. These singleton clusters are not equivalent to the singleton clusters obtained using the JP method. Note that if only a threshold is applied after converting the nearest neighbor graph to the shared nearest neighbor graph, there will be several clusters (which are the connected components after applying the threshold), and the rest will be singletons. By introducing the topic threshold, we are able to mark the documents that have similar documents around. In the end, if a document that is labeled as a topic remains as a singleton, this does not mean that it is a noise document. For that document to be labeled as a topic, it must have enough number of strong links, which means that it has many similar neighbors but the strength of those links were not strong enough to merge them.

Singleton clusters give us some idea about the less dominant topics in the dataset, and they are far more valuable than the singletons that are left out (labeled as background). To the best of our knowledge, there is no other algorithm that produces valuable singleton (or very small) clusters. Being able to make use of the singleton clusters can be very useful. If we're trying to detect topics in a document set, we don't have to force the parameters of the algorithms to the edge to find oend up getting a singleton cluster, that document will give us an idea about several other documents, whereas noise documents do not give us any idea about any other document.

The method described above finds communities of objects, where an object in a community shares a certain fraction of its neighbors with at least some number of neighbors. While the probability of an object belonging to a class different from its nearest neighbor's class may be relatively high, this probability decreases as the two objects share more and more neighbors. This is the main idea behind the algorithm.

Some Final Comments on Concept Based Clustering

While we have restricted our discussion here to concept based clustering for documents, the shared nearest neighbor approach to similarity on which it is based can be applied to many different sorts of data. In particular, the shared nearest neighbor approach from which concept-based is derived, was originally used for two-dimensional spatial data, and we have also successfully applied our data to such data. A major task ahead of us is to more precisely define those situations in which is it applicable

Conclusions

In this paper we have provided a brief introduction to cluster analysis with an emphasis on the challenge of clustering high dimensional data. The principal challenge in extending cluster analysis to high dimensional data is to overcome the "curse of dimensionality," and we described, in some detail, the way in which high dimensional data is different from low dimensional data, and how these differences might affect the process of cluster analysis. We then described several recent approaches to clustering high dimensional data, including our own work on concept-based clustering. All of these approaches have been

successfully applied in a number of areas, although there is a need for more extensive study to compare these different techniques and better understand their strengths and limitations.

In particular, there is no reason to expect that one type of clustering approach will be suitable for all types of data, even all high dimensional data. Statisticians and other data analysts are very cognizant of the need to apply different tools for different types of data, and clustering is no different.

Finally, high dimensional data is only one issue that needs to be considered when performing cluster analysis. In closing we mention some other, only partially resolved, issues in cluster analysis: scalability to large data sets, independence of the order of input, effective means of evaluating the validity of clusters that are produced**,** easy interpretability of results, an ability to estimate any parameters required by the clustering technique, an ability to function in an incremental manner**,** and robustness in the presence of different underlying data and cluster characteristics.

References

1. Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. "Automatic subspace clustering of high-dimensional data for data mining applications," In ACM SIGMOD Conference on Management of Data (1998).

2. Charu Aggarwal, Cecilia Procopiuc, Joel Wolf, Phillip Yu, and Jong Park. "Fast algorithms for projected clustering," In ACM SIGMOD Conference, (1999).

3. Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest, Introduction to Algorithms, Prentice Hall, 1990.

4. Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim, "ROCK: A Robust Clustering Algorithm for Categorical Attributes," In Proceedings of the 15th International Conference on Data Engineering (ICDE '99), pp. 512-521 (1999).

5. A. Hinneburg, C. Aggarwal, and D.A. Keim, "What is the nearest neighbor in high dimensional spaces?" In Proceedings 26th International Conference on Very Large Data Bases (VLDB-2000), Cairo, Egypt, September 2000, pp. 506-515, Morgan Kaufmann (2000).

6. Anil K. Jain and Richard C. Dubes, Algorithms for Clustering Data, Prentice Hall (1988).

7. Anil K. Jain, M. N. Murty, P. J. Flynn, "Data Clustering: A Review," ACM Computing Surveys, 31(3): 264-323 (1999).

8. R. A. Jarvis and E. A. Patrick, "Clustering Using a Similarity Measure Based on Shared Nearest Neighbors," IEEE Transactions on Computers, Vol. C-22, No. 11, November (1973).

9. George Karypis and Eui-Hong (Sam) Han, "Concept Indexing: A Fast Dimensionality Reduction Algorithm with Applications to Document Retrieval & Categorization", Ninth International Conference on Information and Knowledge Management (CIKM 2000), McLean, VA (2000).