Meta Clustering

Rich Caruana CARUANA@CS.CMU.EDU
Pedro Artigas ARTIGAS@CS.CMU.EDU
Anna Goldenberg ANYA@CS.CMU.EDU
Anton Likhodedov LIKH@CS.CMU.EDU
School of Computer Science, Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15213

Abstract

Most clustering methods search for a single, optimal partitioning of the data. Clustering, however, is not well defined: often it is difficult to devise a clustering criterion and similarity metric such that the optimal clustering with respect to the criterion/metric is the most useful one for the user. In this paper we introduce an alternate approach to the clustering problem: search for many different clusterings of the same dataset and present the user with a means of efficiently navigating between them. We demonstrate two algorithms for generating families of clusterings: Sample-and-Merge and Component Reweighting. We then apply clustering at the meta level to organize the different solutions to the base-level clustering problem. This *MetaClustering* allows us to find qualitatively different solutions and present them to the user organized in a way that increases the chances that they will find a satisfactory solution. We demonstrate MetaClustering on a synthetic data set, and on a real protein data set. The results show that the methods are effective at generating different clusterings, and at organizing them into groups of similar clusterings.

1. Introduction

Most clustering algorithms try to find one correct, optimal partitioning of the data. Clustering however is ill-defined [3]. Unlike supervised learning, which has reasonably well defined performance criteria, clustering criteria are often fairly ad hoc and do not match well the intended uses of the clusters. Also, because clustering sometimes is used as a means of trying to understand the data, the "correct" criterion often cannot be devised before the user is more familiar with the dataset. This creates a "chicken and the egg" situation: users need clustering to better understand the data, but can't define an appropriate clustering criterion until *after* they understand the data.

Because clustering problems often are not defined well enough to make it practical to devise a correct formal clustering criterion, we developed new methods that find many good, qualitatively different clusterings of the same dataset. One of these clusterings is the one that would be found with the standard unmodified clustering algorithm. Many of the others are alternate solutions that are "good" according to the user's clustering criterion and similarity metric.

Examining even one clustering to determine its usefulness can take considerable effort. Presenting many different clusterings to a user is impractical. To prevent users from being swamped by so many alternate solutions, we organize them at a meta level. We define a measure of similarity (distance) between solutions (clusterings of the dataset) and cluster these *clusterings* on the basis of the defined measure). Our goal is to present the user with a navigable space of solutions to the clustering problem where similar solutions are close to each other, and qualitatively different ones are distant.

We call this process of generating multiple, qualitatively different clusterings of the same data set, and then presenting the user with a navigable space of clustering models, *MetaClustering*. This increases the chances that the user will find a useful clustering without having to try to modify the clustering criterion or similarity metric. If clustering is being used to learn about the data, MetaClustering provides additional information that would be difficult to acquire from a single base-level clustering: it allows the user to see alternate clusterings that emphasize different relationships in the data.

1.1 Motivation

Consider the following scenarios:

Scenario 1: You want to organize 100,000 documents into a hiearachy to support efficient browsing or retrieval. The hierachy that is right for you may not be right for me. Do each of us have to re-partition the data using our own criterion? What happens if we don't like the clustering that results from our criterion, will we know how to alter it? Can all the things that are important to us be captured in a mathematical formula? If not, what do we do? How do we elicit the aspects of the problem that are important to us and codify them, given that some of those aspects may be subconscious and thus difficult to elicit and codify.

Scenario 2: You want to cluster 641 proteins into structurally similar groups. There are several different measures of protein 3-D structural similarity, and you are not sure which to use. Also, the data is noisy, so you are not sure that a single clustering of the data will be "correct." You are clustering the proteins to look for clusters from which to learn rules relating sequence to structure. The rule learning system is computationally expensive. If you used it as a criterion during clustering, clustering would take 1000 years on a supercomputer. So the clustering criterion you are using is not the one you want to use, but is one that is computationally tractable and hopefully good enough for your purposes.

Scenario 3: You're given a large data set collected from a space telescope that measures properties for millions of galaxies. No one has ever collected a data set of galaxies this large before. You assume there are discoveries to be found in the data, but searching this much data is hard. You are using clustering as one of the tools to help you search for patterns. You don't know ahead of time what patterns you are looking for. This makes it difficult to define a clustering criterion that captures all the potentially interesting groupings that might be in the data. If you pick one criterion, the results may be self-fulfilling: clustering will return a groupings that reflect the criterion you defined. You would prefer that the data itself could "suggest" interesting ways of grouping galaxies.

We believe that on problems like these it is better to find many good, different, clustering solutions than to attempt to define the *right* criterion and use an algorithm that returns one clustering.

1.2 The Concept of MetaClustering

MetaClustering adopts a new approach to the task of partitioning a dataset into homogeneous groups. It can be motivated in the following way: when using some well defined criterion and similarity metric (e.g. by applying a specific clustering algorithm to distances measured between points one particular way), we always obtain a single solution to the problem: some particular clustering model. If this solution is not satisfactory, the user is forced to modify the clustering criterion or the similarity metric and recluster the data.

Instead of forcing users to repeatedly re-cluster the data until they find a criterion and distance metric that yields a satisfactory solution, we search for an appropriate clustering using their original clustering criterion and similarity metric. We assume that in the space of all possible clusterings of the data, there exists a clustering(s) that is most useful to the user. We further assume that the most useful clustering is a reasonably good clustering, i.e., that it performs well on the user supplied clustering criterion and metric (even though it probably is not optimal with respect to them). For example, the most useful clustering might be one that would result if the data were resampled, or if the user made small changes to the clustering distance metric, or if the user clustered the data using a different clustering method or criterion. We do this by modifying existing clustering algorithms so that they have an additional degree of freedom that allows them to produce different solutions. The modifications are designed to allow the algorithms to generate different, near optimal solutions instead of generating one, optimal solution. (The algorithms are described in the next section.)

Having constructed many different clusterings of the data, how do we organize them before presenting them to the user? We use clustering itself, at the meta level, to organize the base-level clustering. A MetaCluster is a clustering of clusterings—each *point* in a MetaCluster is a complex object: a complete clustering of the entire data set.[1] The MetaClustering organizes clusterings so that similar ones are grouped together, and qualitatively different ones are further away. This makes it easier for a user to find clusterings of the data that are qualitatively different without having to examine all (or even many) of the different clusterings.

The process of clustering solutions involves defining a "distance" between clusterings. We want the distance metric to capture how qualitatively different the

[1] Bradley and Fayyad [5] cluster *clusters* (not clusterings), and use the centroids of these clusters to find good initial parameters for methods such as EM that are sensitive to initialization. This is very different from what we are doing. We are clustering different entire *clusterings* of the data in order to organize them into groups of similar clusterings.

clusterings are to each other. In the next section we present two examples of such distance measures, Hop Distance and Cluster Difference.

1.3 Paper Outline

The paper is developed as follows: Section 2 presents several implementations of metaclustering. Section 3 describes the data sets we used to evaluate the algorithms. Section 4 presents experimental results. Section 5 discusses a number of issues, including computational cost, how to evaluate clustering models, and opportunities for further research.

2. Implementation

MetaClustering is an *approach* to clustering, not a specific algorithm. We decompose this approach into three steps:

1. generating multiple, qualitatively different clusterings of the dataset
2. measuring the similarity/differences between different clusterings of the data (i.e., how to compare clusterings)
3. organizing the different clusterings and presenting them to users so that they can efficiently navigate the space of solutions

We consider two different implementations of each of these steps. Section 2.2 presents two algorithms for generating multiple clusterings of the data: Sample and Merge, and Component Reweighting. Section 2.3 presents two methods for measuring the similarity/difference between clusterings: Hop Distance and Cluster Difference. Section 2.4 presents our approach for organizing clusterings at the meta level so that users may browse them more efficiently (MetaClustering).

2.1 Hierarchical Agglomerative Clustering

Before jumping into MetaClustering, we briefly review Hierarchical Agglomerative Clustering (HAC). (We use agglomerative clustering in this paper because it is preferred for the protein dataset that motivated our work. The MetaClustering concept is equally applicable to non-agglomerative methods. See the discussion of MetaClustering with EM clustering in Section 5.1.)

Hierarchical agglomerative clustering is a sequential algorithm based on pairwise distances between points in the dataset. It starts by placing each point into a singleton cluster, so at the beginning the number of clusters is equal to the number of points in the dataset. The algorithm proceeds by merging the two clusters that have the smallest mean distance between them pointwise. (Many different merging criteria [1] can be used; mean pointwise cluster distance was preferred by the physicists who created the protein data because its physical interpretation best matched their intended use of the clusters.) HAC has the property that if two points are merged together at some stage they remain clustered together for the rest of the process. HAC is *greedy* clustering.

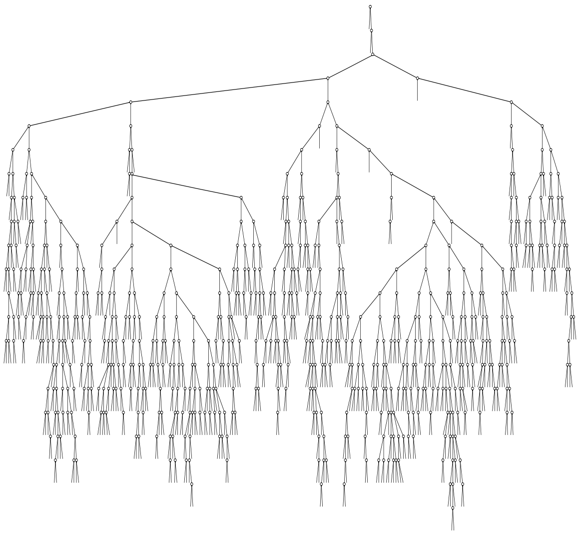

Figure 1. Hierarchical Agglomerative Clustering of the dataset with 641 Proteins

HAC is charachterized by a dendrogram, a clustering tree that shows how the points were merged. Figure 1 shows one clustering dendrogram that results from clustering the 641 proteins in our protein dataset. The tree has 641 leaves corresponding to the singleton clusters HAC starts with. As merges occur between "closest" clusters, one moves up the tree from the leaves towards the root. The root is the single, final cluster containing all 641 points, and is not very interesting. The clusters that are most interesting are the large, tight clusters that typically appear 5-20 levels below the root of the tree. See Section 3 for more detail about the protein dataset.

2.2 Generating Multiple Clusterings

Agglomerative clustering is deterministic. One way to make HAC produce different results is Sample and

Merge, an algorithm inspired by bagging [6]. The following modification to HAC introduces bagging-like sampling. As usual, the algorithm starts with singleton clusters:

1. choose a *random* subset of the set of clusters (e.g. 50% of the total number).
2. applying standard HAC to the selected subset, choose a pair of clusters to merge
3. perform the merge

The step is repeated until the root is reached. Each time the step is repeated, a new random sample of the available clusters is generated.

The randomness in selecting points for the *bag* assures generation of different clustering models. Despite this randomness, however, if the subset of clusters is large enough (e.g., 50% or more), much of the information about patterns in the data is preserved and this ensures that the resulting clustering will be high quality.

One problem with Sample and Merge is that it only works with agglomerative clusterers. We developed a second method for generating different solutions called Component Reweighting that can be used with any clustering method. It is based on the random reweighting of the principal components of the dataset. We use multidimensional scaling to find the principal components of the space from the pairwise distances between points, though any method of creating a low-dimensional projection of the data should work.

The hope is that the principal components correspond to the main features that are responsible for the current set of distances. Reweighting the principal components (e.g. stretching or shrinking the dimensions of the vector space) increases or decreases the relative importance of each of those factors. Recalculating the distances in the dataset from the reweighted components induces a new distance topology that is related to the original topology yet different from it in interesting ways. These new distances are then used as the basis for a new clustering of the data (using an unmodified clustering algorithm).

2.3 Measuring Clustering Similarity

In the previous section we saw two ways of generating different solutions to the particular clustering problem. Presenting all of these solutions to the user is impractical because evaluating each clustering may take considerable effort. Many of the clusterings generated by these algorithms will be similar to each other. Rather than swamp the user with all clusterings, we cluster the solutions into groups according to their pairwise similarity so that the user need only examine a few qualitatively different clusterings from different groups.

To cluster a set of points one needs a measure of how similar the points are to each other. The points we want to cluster are *clusterings*, so we need to measure the similarity between clusterings.

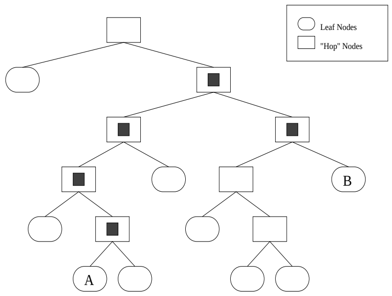

Figure 2. Enlarged portion of a dendrogram

One way of measuring similarity[2] between different clusterings of the same dataset is to use the cluster tree that results from HAC algorithm. Consider the dendrogram shown in Figure 1. Figure 2 shows an enlargement of part of such a dendrogram. The dendrogram shows the sequence of merges made by HAC, represented as a binary tree. The set of points contained in a cluster at some level of the tree is the set of descendants of that cluster node.

We define the Hop Distance between two points in a binary tree as the number of hops required to traverse the tree from one point to the other. As an example, the hop distance between points A and B in Figure 2 is five.

Hop Distance can be used to compare different hierarchical agglomerative clusterings. Intuitively, if in one clustering two points are merged early, and in another clustering they remain in different clusters until the last merge, those clusterings are different with respect

[2]The distances we define between clusterings are not necessarily true metrics because the triangle inequality may not hold. Therefore we avoid using the terms "distance" and "metric" and use "similarity measure" instead. Precisely, similarity is a real-valued function f, defined on a $C \times C$ set, where C is a set of all clusterings of the same dataset. For any two solutions to the original clustering problem f numerically defines how similar they are. Note that the possibilities for similarity definitions are more limited when the original data is available as a set of pairwise distances, rather than points in a vector space. In this paper we consider (without loss of generality) the more difficult case of dealing with pairwise distances.

to those two points. Let H^1_{ij} be the hop distance between points i and j in clustering H^1. Consider the hop distances between *all pairs of points* in the dataset in two different dendrograms: H^1_{ij} and H^2_{ij}.

For any pair i and j $H^1_{ij} - H^2_{ij}$ shows the *difference* in the lengths of sequences of merges that join i and j into one cluster. Calculating the sum of those differences over all i and j yields a measure of similarity of dendrograms (i.e., similarity of clusterings):

$$H(C1, C2) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} (H^1_{ij} - H^2_{ij})^{\alpha}$$

$\frac{1}{n^2}$ makes the measure independent of the size of the dataset. α, controls the relative importance of small and large differences. Large differences suggest that the two clusterings are very dissimilar, so it makes sense to set α greater than one. Note, however, that H does not satisfy the triangle inequality (and therefore is not "true" distance metric) for α greater than one. In practice this does not cause problems, and we usually use $\alpha = 2$.

Hop Distance is not the only possible way to define similarity. Another more straightforward method is as follows: for any two points i and j define d_{ij} as 0, if two points are in the same cluster in both models, or in different clusters in both models. Define d_{ij} as 1 if they are in different clusters in one model and in the same cluster in another. $\sum_{i,j} d_{ij}$ would be the similarity of two clustering models. We call this measure Cluster Difference. Unlike Hop Distance, this measure does not require that the clustering be hierarchical.

2.4 Navigation Between Clusterings

Once we determine the distances between all possible pairs of solutions to clustering problems we can use them to organize the clusterings for the user. Our main approach to this organization is to generate the MetaClustering—a clustering of all the base-level clusterings of the data. We can generate the MetaClustering using any algorithm that can cluster similarity data (it would be more difficult to use clustering methods such as EM at the meta-level because we do not have a vector representation of the different clusterings).

We use agglomerative clustering at the meta level, organizing the solutions into a dendrogram. Thus our Meta Clustering is a hierarchical clustering of clusterings, a tree of trees. In the Meta Clustering dendrogram, clusterings that are qualitatively similar (as measured with Hop Distance or Cluster Difference) will be grouped near each other, while clusterings that are very different will be distal. Users need only to examine a few clusterings from different parts of the tree to find the region that may be useful. Once they find a region that contains promising clusterings, they can then examine points in that neighborhood to find the clustering that is best for their purposes.

An alternate approach for presenting the clusterings to the user is to use a low dimensional projection of the pairwise cluster distances. For example, multidimensional scaling is able to find the principle components of the meta clustering space from the clustering similarity data. The clusterings can then be presented to the user in a low dimensional graph where similar clusterings should be positioned close to each other. One problem we have found with this approach is that sometimes the dimensionality of the meta-level clustering space is not low in dimension, so any low dimensional projection of the data has significant distortion. This makes the 2-dimensional projections that are easy to graph less useful than the Meta Clustering Tree.

3. Data Sets

Two datasets were used in order to evaluate the performance of the algorithms. One is an easy to visualize 2-D synthetic dataset obtained by sampling 40 pairs of coordinates from six gaussian distributions for which mean and variance were specified. This dataset was designed as an illustration to the presented concept.

The second dataset is a set of pairwise similarities between 641 proteins. It was created by crystallographers interested in developing automated techniques to learn relationships between protein sequence and structure. In an attempt to understand the data and prepare it for subsequent use, the developers used a number of techniques including clustering, multidimensional scaling, and univariate and multivariate statistical analysis. Figure 1 shows a cluster tree for this data generated using agglomerative clustering on the average Euclidean distance between matched atoms in pairs of proteins.

4. Results

The Sample and Merge algorithm was evaluated using both real and synthetic datasets. Sample-and-Merge was applied to a synthetic dataset to generate 100 different clusterings of the 240 synthetic data points. After 100 clusterings were generated, the Hop Distance algorithm was used to calculate the similarity measure between all $100^2/2 = 5000$ pairs of clusterings. Then, the agglomerative clustering was applied at this meta level to these 5000 distances.

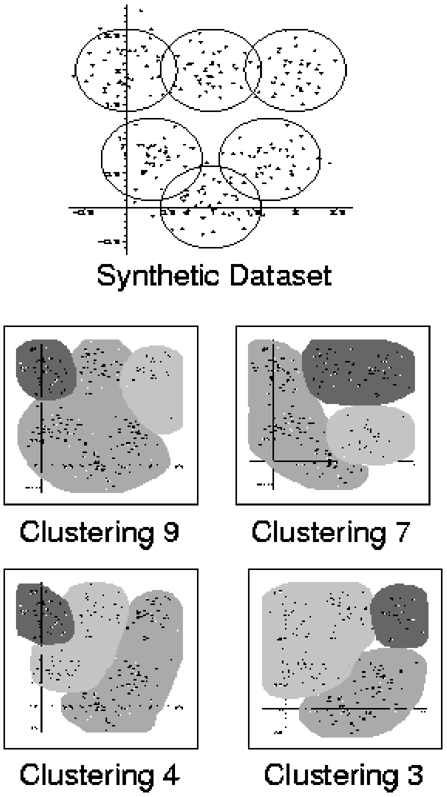

Figure 3. Meta Clustering of 100 clusterings generated for the synthetic data using the Sample-and-Merge algorithm

Figure 3 shows the Meta Clustering for the synthetic data. Each leaf node in this Meta Clustering represents an entire clustering of the dataset. This is a small Meta Clustering containing only 100 different base-level clusterings. (Usually we would generate more base-level clusterings.)

Four different base-level clusterings from different regions of the Meta Clustering tree are examined in detail. Each graph is a plot of the base level data set when it is grouped into three clusters. It can be seen from the picture that the clusterings displayed are quite different. While none of these four is optimal with respect to the clustering criterion, each one represents a good, yet qualitatively different, clustering of the data that might be useful.

The Sample-and-Merge algorithm was also applied to the 641 protein dataset. Figure 4 shows the corresponding hierarchy of solutions to this clustering problem. To keep the figure small, we only generated 100 base-level clusterings of the data. Once again it important to note that this figure is a *meta clustering*: each leaf in this hierarchy is itself an entire clustering of the original 641 proteins (like that shown in Figure 1).

It is difficult to visualize the clustering of proteins due to the high dimensionality of the data. To make the task easier, the four graphs in Figure 4 project the 641 proteins onto the first two principle components of the data, and the clusters of points in the base-level clustering are shaded.

The four graphs in Figure 4 show four different clusterings of the data from different branches of the meta clustering dendrogram. These results confirm that the generated clustering models are qualitatively different. For example a large cluster of proteins in model 37 is split into two or more clusters in the rest of the presented models. Any one of these clusterings might be the solution that is most valuable for use in the protein project.

The results presented in Figures 3 and 4 are obtained with the Sample-and-Merge algorithms using Hop Distance as similarity function. Qualitatively similar results were obtained on these data sets using Component Reweighting and/or Cluster Difference. We do not yet know which methods for generating multiple clusterings and for measuring cluster similarity will perform better on different datasets.

5. Discussion

5.1 Non-Agglomerative Meta Clustering

The Sample-and-Merge algorithm for generating alternate clusterings is only applicable to agglomerative clustering methods. It will not work with EM or probabilistic clustering methods. Similarly, Hop Distance applies only to hierarchical clusters such as those created by agglomerative clustering.

MetaClustering, however, is not restricted to agglomerative methods. Our first research in MetaClustering uses agglomerative methods for two reasons:

1. the protein dataset is pairwise distances, not vectors, so it is easier to apply agglomerative methods
2. we wanted to cluster the proteins into a hierarchy

Thus we chose agglomerative clustering because it fits the particular problem we worked on.

Applying MetaClustering to non-agglomerative or non-hierarchical clustering methods such as EM re-

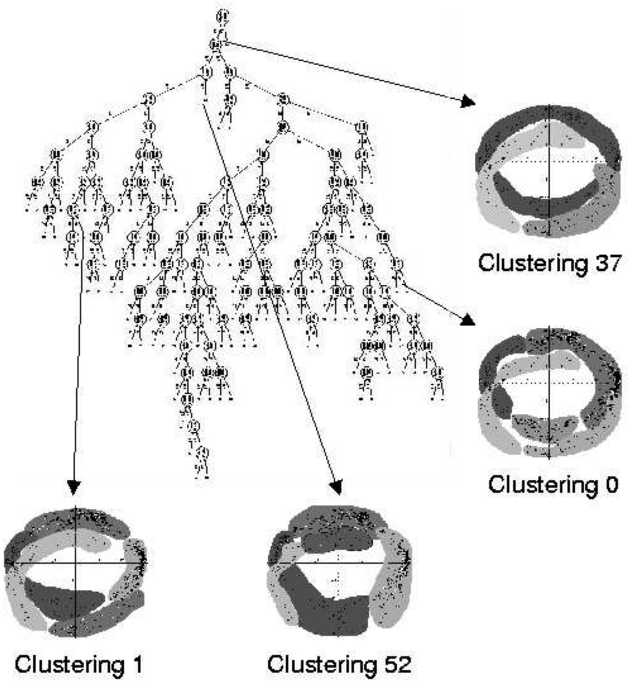

Figure 4. Dendrogram of clusterings of protein data with four sample clusterings

quires other approaches for generating multiple clusterings models and measuring the distances between them. The EM method itself suggests one interesting approach. EM converges to local minima rather than global optima. Because of this, one typically runs EM methods many times starting from different initial conditions [2][5]. The goal of these restarts is to try to find initial conditions that lead EM to the better (preferably optimal) final result [5].

One way to generate many different clusterings of the data is to exploit EM's propensity for getting stuck in local minima, and to use initialization to promote EM to stuck in *different* local minima. Thus a key drawback of the EM method can be used to advantage for MetaClustering—we can obtain a set of different solutions corresponding to different initalization conditions. We suspect that different local minima will represent good, but interestingly different, clusterings of the data.

5.2 Computational Cost

Straightforward implementation of agglomerative clustering is $O(n^3)$, which is prohibitive for data sets larger than 10^3 data points. Advanced data structures ([2]) can reduce this cost to $O(n \log^2 n)$ if the distances between points satisfy the triangle inequality. Effective implementations of agglomerative methods are discussed in [4] and [7]. These make agglomerative clustering practical for data sets as large as 10^7 points. Clustering at the meta level needs to be performed only once. The cost of repeating the base-level clustering usually limits the number of clusters to be clustered at the meta level. Typically we will not generate more than $10^2 - 10^5$ base-level clusterings. Usually it is practical to use agglomerative clustering at the meta level, even if a more efficient clustering method such as EM clustering is being used at the base level. The hierarchical cluster tree that results from agglomerative clustering provides more information than a simple N-class clustering; it let's users navigate the Meta Clustering more easily since it shows the similarity between clusterings at all scales.

Although our implementations were designed to make testing new ideas easier, not for efficiency, the time needed to MetaCluster the protein dataset may be informative. The protein data contains 641 points. It takes about 10 seconds to cluster the proteins once with Sample and Merge Agglomerative Clustering. The Meta Cluster Tree in Figure 4 contains 100 base-level clusterings. It took about 15 minutes to generate these. Clustering the 100 clusterings to form the Meta

Clustering takes about a second. The main expense is calculating the Hop Distance for the 5000 pairs of clusterings. If Hop Distance is calculated using all pairs of points in each base-level clustering, it takes about one second per pair of clusterings, or about an hour to compute distances between all pairs of clusterings. This cost can be reduced by sampling from the points used for Hop Distance, or by using a simpler distance metric such as Cluster Difference.

5.3 Characteristics of the Meta Clustering

The Meta Clustering of the solutions to a particular clustering problem may have properties quite unlike those of the base data set. For example, a principal components analysis of the protein data suggests that it is intrinsically 6 dimensional. (This corresponds well with our physical understanding of the process that generated the data.) A similar analysis applied to the distances between clusterings indicates that the Meta-Clustering space is at least 20 dimensional. We believe that the dimensionality of the Meta Clustering space may be indicative of the number of qualitatively different clusterings of the data.

5.4 Evaluating Clusterings

The MetaClustering organizes the different solutions to the particular clustering problem into a hierarchy. Solutions in the same cluster (at the meta level), or nearby in the MetaClustering Hierarchy, are similar to each other. The hierarchy greatly reduces the number of base-level clusterings the user has to examine to find one that is best suited to their purposes. If a clustering looks inappropriate to the user, they should examine clusterings far away in the MetaClustering Tree. This reduction in the number of clusterings that must be examined is important because usually it is not easy to visualize or evaluate an entire clustering of the data to determine if it is useful or not. Research into how to visualize and evaluate clusterings will be important to future work in Meta Clustering.

6. Summary

Most clustering methods assume that there is one optimal solution to the clustering problem and that a clustering criterion can be defined that will guide the clusterer to it. This paper presents an alternate approach. We assume that it is difficult to capture all important criteria in a precise, operational clustering formula. Instead of trying to devise one ultimate clustering criterion, we generate many alternative solutions for the clustering problem, and provide the user with a framework simplifying the selection of a useful solution. To make the selection easier, we organize the solutions by MetaClustering. MetaClustering groups similar clusterings together, and places qualitatively different clusterings in separate groups. This approach required us to solve two technical problems: 1) how to force a base-level clustering algorithm to generate many qualitatively different clusterings of the data set; and 2) how to measure the similarity between clusterings so that the clusterings themselves may be clustered. Experiments with our current solutions to these problems on a synthetic and real data set indicate that MetaClustering is a promising new approach to clustering problems.

Acknowledgements

We thank Paul Hodor, Bruce Buchanan, and John Rosenberg for their invaluable help with the protein dataset.

7. Bibliography

[1] Duda R. Hart P. (1973). Pattern classification and scene analysis. John Wiley & Sons.

[2] Pelleg D. Moore A. (2000). X-means: Extending K-means with Efficient Estimation of the Number of Clusters

[3] Tishby N. (2000) Personal Communication during *Machines that Learn, the 2000 Snowbird Workshop.*

[4] W.H.E. Day and H. Edelsbrunner. Efficient algorithms for agglomerative hierarchical clustering methods. Journal of Classification, 1(1):7–24, 1984.

[5] Bradley P. S. Fayyad U.M. (1998) Refining initial points for K-Means clustering. Proceedings of the 15-th International Conference on Machine Learning.

[6] Breiman L. (September 1994) Bagging Predictors, Berkeley Statistics Technical Report #421.

[7] F. Murtagh. A survey of recent advances in hierarchical clustering algorithms. Computer Journal, 26:354–359, 1983