Thoughts on clustering*

Avrim Blum
Carnegie Mellon University

Abstract

Clustering is a somewhat confusing topic theoretically. In large part this is because there are many different kinds of clustering problems. In addition, the true goals in clustering are often difficult to measure, making the task seem not well-defined and under-specified. In this note we discuss one approach from [6] to theoretically formulating a certain broad class of clustering problems, and discuss relations between this and other approaches. We also make a few suggestions and state a few open directions.

1 Introduction

Clustering is a somewhat confusing topic – in part because there is not even agreement on whether it is confusing or not! For example, from the "objective-function approach" to clustering everything seems clear: you give me your data (e.g., presented as a weighted graph where nodes are data items and edges are distances or similarities), a desired number of clusters k, and an objective to optimize (say, minimize the sum of all intra-cluster distances) and what is left is just a (potentially NP-hard) problem of optimization. But often the true goals in a clustering problem such as clustering documents by topic do not directly correspond to any specific distance-based objective: the real goal is to organize the data points correctly (e.g., produce a clustering of documents most people would agree with) not to minimize distances. Moreover, how can one even talk about the "best clustering" of a given set of data in a setting where different users might want to cluster the same data for different purposes (e.g., clustering images by who is in them versus clustering images by facial expression, or clustering documents by topic versus clustering documents by genre)?

Here we describe one theoretical framework, from [6], aimed at reconciling a number of these issues, and addressing the question of what kinds of information an algorithm needs in order to cluster well. We discuss types of statements one can show in this model, and also discuss how this relates to other approaches such as those of [23, 17] and approaches based on objective functions. Some key points that appear fairly explicitly in this framework are that a clustering problem is defined not just by the data you have but also what you want to *do* with that data. Also it may help to give the algorithm a bit of additional slack (we will describe what we mean by this shortly) in cases where the given distance or similarity information is not of the highest quality in terms of how well it matches with the clustering goals.

Note that clustering is a broad topic. For any given framework there will be certain issues that it highlights and others that it abstracts away, and some kinds of clustering problems that fit it better than others. For example, the framework presented here is not so aimed at exploratory

*Draft December 9, 2009.

clustering problems (but see discussion in Section 3). In the end, clustering is a sufficiently diverse subject that there is room for multiple theoretical frameworks and approaches, each giving insights and structure appropriate for different settings.

1.1 Goals of a theory

Why even aim for a theoretical framework (or frameworks) for clustering? In general, theoretical models can have a number of distinct benefits, both qualitative and quantitative. Some of these include:

- *To help provide structure in thinking about a given instance of the problem that one cares about, and the various design choices that need to be made.* For example, a theoretical framework can provide guidance in deciding what to use as a measure of similarity or distance between data objects based on one's beliefs or domain knowledge, or in how one's choices might influence the selection of algorithm to use.
- *To bring out and clarify important issues, tradeoffs, or subtle distinctions.* For example, without a theory of NP-completeness it might not be obvious that there is a fundamental difference between the difficulty of problems such as Euler tour versus Hamiltonian cycle (visiting each *edge* exactly once in a graph versus each *vertex* exactly once), or shortest path versus traveling salesman path, or 2-SAT versus 3-SAT. Similarly, a theory for clustering can hope to suggest "if only you change your formulation a bit here" or "if you are able to give your algorithm more flexibility there" then the problem becomes much easier.
- *To provide guarantees for algorithms and understand what kinds of guarantees are achievable.* As with most theoretical work, these may not be tight, and algorithms with worse guarantees may experimentally perform better than algorithms with better guarantees.[1] Nonetheless, these can provide helpful understanding of the nature of the problem and new algorithmic ideas that one might not have considered otherwise.

2 A theoretical framework

2.1 Motivating examples and preliminaries

Before presenting a theoretical framework let us first consider a few motivating examples of clustering problems:

- Given a set of protein sequences, cluster them by function.
- Given a set of images of people, cluster them by who is in them.
- Given a set of images of people, cluster them by facial expression.
- Given a set of documents, cluster them by topic.
- Given a set of products being sold by a large retailer, cluster them to aid in webpage design for online sales.

[1] Many times, the issue here is the old adage "it may work in practice but it will never work in theory". In particular, when designing algorithms to meet theoretical guarantees one tends to strip away heuristics that make perfect sense (such as various local optimizations) but that are not useful in achieving the theorem one is aiming for.

One immediate lesson these examples point out is that a clustering problem is defined not just by the data one is given but also by what one wants to do with it. For example, given a set of images of people, a clustering that is good for clustering by facial expression is likely not good for clustering by who is in the image and vice-versa. Another feature these examples have in common is that they do have a reasonably well-defined notion of a correct or best answer[2] but this is defined in terms of how points are partitioned, *not* in terms of their pairwise distances. In fact, in all these examples, the way that data is represented — the notion of distance or similarity to use when presenting data to a clustering algorithm, is a design decision of the same level of importance as (if not more than) the choice of clustering algorithm to use, and a theory should ideally address this decision as well. In particular, how does (a) the goal of the clustering affect (b) the representation of the data and (c) the choice of algorithm to use and how do (b) and (c) influence each other? Understanding and analyzing this is the aim of the framework below.

2.2 The framework

We now present a framework from [6] and discuss how it addresses some of these issues.

In this framework, a clustering problem is a triple (S, d, C^*) specified as follows. We are given a data set S of n objects, for example documents, images, etc. These objects are presented to us via a pairwise distance metric d. (The paper [6] uses a similarity function between data objects rather than a distance metric, but we discuss in terms of distances here to make relations to other approaches more clear.) Finally, C^* denotes the unknown desired clustering of the data set S into k clusters $C_1^*, \ldots, C_k^*$. For example, C^* might be a correct clustering of images by who is in them or a reasonable clustering of documents by topic. The goal of an algorithm is to produce a clustering $C = \{C_1, \ldots, C_k\}$ that agrees as much as possible with C^*. In particular, we consider the error of C with respect to C^* to be the fraction of points that would have to be reassigned in C to make it equal to C^* up to re-naming of the cluster indices.[3] In other words, we are modeling this as a problem of multiclass classification except (a) the algorithm sees no labeled data at all, and (b) we do not care about the names of the labels (the names of the clusters in the partition), just the partitioning itself.

Since the algorithm has very little to go on (no labeled data at all), the distance metric d had better be related to the desired clustering C^* in some useful way in order for the algorithm to have any hope at all. The main points of this framework are then to analyze:

1. What properties of the relation between d and C^* are sufficient for an algorithm to be able to produce a low-error solution?[4]

2. Can one talk about properties that are sufficient (or both necessary and sufficient) for natural algorithms to succeed? Can one talk about natural properties that require new algorithms?

[2]For the case of clustering documents by topic, one could argue that reasonable people might disagree on the right notion of a topic, especially in degree of specificity. So one could view this as a case where there might be multiple correct answers, and perhaps one would like an algorithm to actually find all of them.

[3]Formally, the error of a clustering $C = \{C_1, \ldots, C_k\}$ is $\min_{\sigma \in S_k} \frac{1}{n} \sum_i |C_i^* - C_{\sigma(i)}|$ where S_k is the set of all permutations on $\{1, \ldots, k\}$. One could alternatively consider measures such as the fraction of *pairs* that are in the same cluster in one clustering and in different clusters in the other, but the point-based measure seems more direct, though this is not crucial to the framework.

[4]As an analogy, even for supervised learning one needs some inductive bias – some kind of relation between the feature information and the target – in order to have any hope for generalization. For example, a typical relation one might consider is that the data points of different classes should be separated by a large margin. Of course, without any labels at all, one should expect to need stronger properties in order to be able to find an accurate partitioning.

3. Can one expand the set of properties that are sufficient for successful clustering by allowing the algorithm a bit of additional slack in some natural way?

Regarding point (3) above, one form of slack that turns out to be incredibly helpful is what we call the *tree relaxation* of the basic framework (or *tree model*), defined as follows.

Tree model: In the *tree relaxation* of the above framework, we allow the algorithm to produce a hierarchical clustering: a tree on subsets such that the root is the entire dataset S, and the children of any node S' in the tree form a partition of S'. We now say that this hierarchy has low error if there exists a *pruning* C of the tree (not necessarily using nodes all at the same level) that has low error. The point here (the sense in which it is giving the algorithm additional slack) is that we do not require the algorithm to identify *which* pruning is the desired one. Note that this formulation is a relaxation because given a low-error clustering C one can always convert it to a hierarchy by arbitrarily merging clusters two at a time. The formulation is also quite natural: for clustering documents, it can be viewed as producing an initial coarse partition (the children of the root), and then saying to the user "I wasn't sure how specific you wanted to be, so if any of these clusters are too broad, just click and I will automatically split it for you (by moving down the hierarchy)".[5] This relaxation allows for making positive statements about a broader class of properties that would simply be informationally insufficient to identify an accurate clustering otherwise.

2.3 A few examples

What kinds of natural properties might one be able to use to cluster? Let us consider a few examples:

Single cutoff: This is a trivial property just to begin with. Suppose the distance metric we are given were so good that for some given cutoff value c we have $d(x, y) < c$ for all pairs x and y that should be in the same cluster, and $d(x, y) > c$ for all pairs x and y that should be in different clusters. Then this is clearly sufficient to recover the clusters by a simple greedy procedure.

Strict separation: Suppose we now weaken the above condition to just require that all points x are closer to all points y from their own cluster than to any points y from any other clusters (but without a common cutoff value). Now, unfortunately, this is no longer sufficient information to uniquely identify the correct answer C^* or even a good approximation to it even if the number of target clusters k is known. For instance, in the example in Figure 1, there are multiple highly distinct clusterings consistent with this property: one with 2 clusters, two with 3 clusters, and one with 4 clusters. Even if one is told the desired clustering has 3 clusters, there is no way for an algorithm to tell which of the two (very different) possible solutions is one we are looking for. On the other hand, the strict separation property *is* sufficient to cluster well in the tree relaxation of the model. In fact, it is quite easy to show that the dendogram of the *single-linkage* algorithm will in fact be a hierarchy satisfying the condition that the desired answer is a pruning of the tree, even if at no single stopping point does the algorithm have the correct clustering as its current state.

Min-stability: a weaker condition than strict-separation is to only ask that for any two target clusters C_i^*, C_j^* and any subsets $A \subset C_i^*, B \subseteq C_j^*$, the minimum distance between A and $C_i^* \setminus A$ be smaller than the minimum distance between A and B. For example, consider n equally-spaced points on a line with one point removed, where the target clustering corresponds to the two "natural" clusters that result. Here, the distance metric satisfies the min-stability property but not

[5] It is also a natural relaxation for settings where one has additional side constraints that an algorithm could then use in a subsequent post-processing step, as in the database deduplication work of [11].

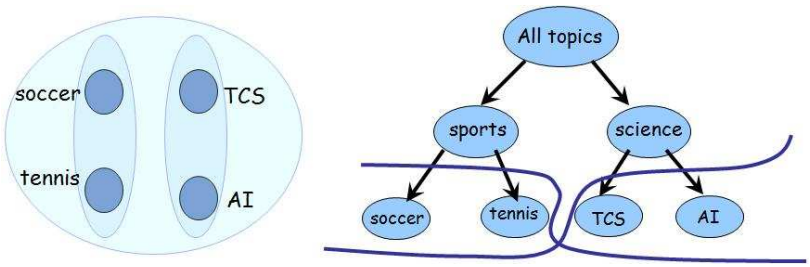

Figure 1: Suppose data lies in four regions (e.g., think of as documents on tennis, soccer, Theoretical Computer Science, and AI). Suppose that $d(x, y) \leq 0.2$ if x and y belong to the same region, $d(x, y) \approx 0.5$ if $x \in$ soccer and $y \in$ tennis or if $x \in$ TCS and $y \in$ AI, and $d(x, y) \approx 1$ otherwise. Even assuming that the target clustering satisfies the strict separation property, there are still multiple consistent solutions, including two consistent 3-clusterings ((soccer $\cup$ tennis, TCS, AI) or (soccer, tennis, TCS $\cup$ AI)). However, there is a single hierarchical decomposition such that any consistent clustering is a pruning of this tree, and moreover this would be produced by the single-linkage algorithm.

the strict-separation property. What is interesting about this property is that (as shown in [6]) this property is both *necessary and sufficient* for the single-linkage algorithm to succeed with zero error in the tree relaxation of the model. That is, this property characterizes exactly what is needed for the dendogram of the single-linkage algorithm to contain the target clustering as some pruning of its tree.

Average-stability: A natural property related to that above is is to ask that for any two target clusters C_i^*, C_j^* and any $A \subset C_i^*, B \subseteq C_j^*$, the *average* distance between A and $C_i^* \setminus A$ be smaller than the *average* distance between A and B. Now, single-linkage may fail. However, this property *is* sufficient for the *average-linkage* algorithm to produce an accurate hierarchy, though it is not a necessary condition. (In fact, it seems quite tricky to come up with a clean characterization of exactly when the average-linkage algorithm will succeed.)

Optimal objective property: This property is that the target is equal or close to the optimal according to some given objective such as k-means, and of course suggests optimizing that objective as an algorithmic goal. We discuss this further in Section 3 below.

Relaxed versions of the above: One can also consider relaxed versions of these properties where we ask only that *most* of the points satisfy the given condition. (For some of these we know of efficient algorithms and for some we do not.)

Note that if a property is too weak (i.e., the distance measure is too loosely related to the target clustering), then there might not even exist a hierarchy that guarantees the desired clustering is some pruning of the tree. In that case, one perhaps might want to consider some alternative navigational structure, or some kind of interactive model where a user can provide additional feedback to help guide the clustering algorithm in such cases where not enough information about the desired partitioning is contained in the distance measure.

3 Discussion

The high-level point of this framework is to think of clustering in terms of what one wants to do with it, and to view the representation of data as a first-class design decision. Rather than thinking of clustering as "given a distance matrix (or set of points in R^n), find me the partitioning

I am looking for" the framework focuses on the relation between how data is represented and the clustering goals, and what does the strength or properties of this relation suggest about the choice of algorithm to use and type of output to ask for.

We now discuss connections between this framework and some other well-studied theoretical approaches to clustering to see how they relate and what insights each can bring.

The objective function approach: In the objective approach to clustering, one poses some directly measureable objective to maximize or minimize, and then designs algorithms or heuristics aimed at optimizing this function. For example, the *k-means* objective asks to find cluster centers $c_1, \ldots, c_k$ such that if each datapoint is assigned to its nearest center—let $c(x)$ denote the center c_i closest to point x—then the quantity $\sum_{x \in S} d^2(x, c(x))$ is minimized. (If data is being represented as points in R^n under the Euclidean distance metric, then c_i will be the mean value of all points assigned to it, hence the name.) This is an NP-hard optimization problem, so a number of heuristics have been developed, including the "k-means algorithm" (also known as Lloyd's method) which finds local optima [18, 19], as well as algorithms achieving constant-factor approximation guarantees [4, 16]. Related objectives are k-median (minimize sum of distances rather than distances squared), min-sum (minimize sum of all pairwise intra-cluster distances), minimizing the sum of cluster radii, and others [20, 10].

In the framework considered here, the objective function approach corresponds to a belief that the distance metric has the property that the optimum clustering according to the objective indeed is equal to, or nearly equal to, the target clustering. Viewing approximations to the objective as a good thing corresponds to the belief that all *approximations* to the optimum—or at least the approximations an algorithm might reasonably find—have low error as well. This in turn has a number of interesting implications, explored more fully in [5, 7]. For example, for a number of standard objectives, one can use the assumption that all c-approximations are ϵ-close to the target to find a clustering that is $O(\epsilon)$-close to the target even in cases where finding a c-approximation in general is NP-hard.

Generative mixture models: In mixture models, one assumes that data is generated by a mixture of simple probability distributions (e.g., Gaussians), one per cluster, and aims to recover these component distributions [1, 3, 14, 13, 15, 22, 12]. In multilevel models, one assumes that the number of clusters itself along with their component weights in the mixture are probabilistic quantities as well [21].

These generative models can be roughly viewed as instances of the framework presented here (there is a well-defined notion of a target, for instance), except the "platonic" property of the relation between distance measure and target is replaced with a more specific generative process. An analog in supervised learning, for instance, would be the difference between assuming the target has a large-margin linear separator (but not making further assumptions about what each class looks like) and assuming each class itself is a Gaussian or has some specific distributional form. Advantages of generative models are that to the extent they accurately model the data and data/target relation they can provide much more specific guidance; on the other hand, they often imply substantial intracluster uniformity which may or may not indeed hold. One further point is that often one solves a generative model by reducing the problem to optimizing a distance-based objective, bringing us to the objective-based clustering setting discussed above.

Axioms/properties of clustering functions: The axiomatic approach of [17, 23] considers properties of *algorithms* rather than properties of the distance-measure/target-function relation. For example, one might ask: "does there exist an algorithm whose output is invariant to multiplicative

scaling of the distance metric, whose output does not change if one reduces intracluster distances or expands intercluster distances arbitrarily ('intra/intercluster' now referring to the output of the algorithm rather than the target clustering), and which is not a-priori restricted in terms of which k-clusterings it can produce?" (answer: yes, but no if you remove the "k").

This approach and the framework presented here are in some sense complementary—one considers properties of the relation between distance measure and target and one considers properties of the algorithm—though they are clearly related. For example, to take a simple case, if the strict separation property holds for some distance function d, then it holds for any scaling of d, or any d' that brings some points together inside target clusters and moves points apart between clusters. Thus, for an algorithm to be successful under this property, it should be invariant to these as well. On the other hand, from the perspective of the framework here, properties of an algorithm are more "means to an end" rather than an end in itself. For example, the k-means optimal clustering can change by bringing intracluster points closer together in a nonuniform manner, so if instead one's belief about d is that that desired clustering is the k-means optimal then one may not need or even want that type of invariance.

Clustering quality measures: The work of [2] considers properties of *quality measures* used to evaluate clusterings. In essense, the types of properties considered in the framework here (relations between distance metric and target clustering) can be viewed as boolean versions of quality measures, and more generally a quality measure is much like a "soft" property in our framework. [2] then consider properties of these properties: and what kinds of properties reasonable quality measures should satisfy and what kinds of properties can be used to distinguish different quality measures.

Exploratory clustering: At some level, exploratory clustering, where one is given a dataset and just wants to find out what kind of interesting groupings it might have, seems the least well-suited to the framework here since there is no real notion of a target clustering. On the other hand, at a different level, the fit is quite strong. In particular, if the notion of "interesting" can be described in some objective way (e.g., given the data, find all clusterings satisfying a certain condition) then this could be used as a property. In that case, an algorithm for clustering under this condition in the tree model would correspond to a method for outputting all interesting clusterings in an easily navigable form (they would all be prunings of the tree produced). In fact, this view corresponds to a line of research in mathematical biology where researchers have considered the construction of structures containing all *clusters* satisfying various properties with respect to the given distance or similarity information [9, 8].

4 Additional thoughts

From the perspective of this framework, it would be interesting to have a repository of clustering problems where in some cases we have the same data represented in different ways, and in some cases we have the same data with different target clusterings. One could then experimentally explore the relation between how data is represented, the clustering goal, and the algorithm used. From a theoretical (and practical) perspective, another interesting direction is to consider other possible relaxations of the basic clustering goal, for cases where there is simply not enough information contained in the distance function to determine a unique best answer or even a unique best tree. This might even involve some sort of interactive model. Finally, the type of loss function considered in the framework presented here is essentially a point-based notion (how many datapoints would

have to be moved to match the target) inherited from supervised learning. It would be interesting to consider the effect of folding in more application specific-kinds of loss functions ("how good was this clustering of products on my webpage with respect to number of sales made"), especially in cases where there might be multiple quite different but equally good solutions.

5 Concluding remarks

Suppose someone came to you saying "I have a computational problem, what algorithm should I use?" Your answer wouldn't be a blanket "use greedy local search" or "well, some algorithms work better some times and others work better other times"; instead, the answer would be "what kind of computational problem is it?" Does it look like a network flow problem? Solving a linear system? Does it have an optimal-substructure property? To the extent that they can describe what they know or believe in a clear way there is an opportunity to use the large and highly-developed field of algorithm design to determine a good approach. Clustering I think similarly encompasses wide range of different types of problems, with different kinds of goals and different kinds of prior knowledge or beliefs about the data. For many of these, I believe we do have interesting theoretical frameworks and results, and we have here discussed a few of these. In the end, for a fairly unstructured problem like clustering, theoretical frameworks can help clarify design decisions (what would I ideally *want* from a distance measure and clustering algorithm), bring out subtle issues, and hopefully produce insights into the nature of the problem itself.

References

[1] D. Achlioptas and F. McSherry. On spectral learning of mixtures of distributions. In *18th Annual Conference on Learning Theory*, 2005.

[2] M. Ackerman and S. Ben-David. Measures of clustering quality: A working set of axioms for clustering. In *Proceedings of Neural Information Processing Systems (NIPS)*, 2008.

[3] S. Arora and R. Kannan. Learning mixtures of arbitrary gaussians. In *Proceedings of the 37th ACM Symposium on Theory of Computing*, 2001.

[4] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local search heuristics for k-median and facility location problems. *SIAM J. Comput.*, 33(3):544–562, 2004.

[5] M.-F. Balcan, A. Blum, and A. Gupta. Approximate clustering without the approximation. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2009.

[6] M.-F. Balcan, A. Blum, and S. Vempala. A discriminative framework for clustering via similarity functions. In *Proceedings of the 40th annual ACM symposium on Theory of Computing (STOC)*, 2008.

[7] M.-F. Balcan and M. Braverman. Finding low error clusterings. In *Proceedings of the 22nd Annual Conference on Learning Theory*, 2009.

[8] H.-J. Bandelt and A.W.M. Dress. Weak hierarchies associated with similarity measures: an additive clustering technique. *Bulletin of mathematical biology*, 51(1):133–166, 1989.

[9] D. Bryant and V. Berry. A structured family of clustering and tree construction methods. *Advances in Applied Mathematics*, 27(4):705 – 732, 2001.

[10] M. Charikar, S. Guha, E. Tardos, and D. B. Shmoy. A constant-factor approximation algorithm for the k-median problem. In *In Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, 1999.

[11] S. Chaudhuri, A. Das Sarma, V. Ganti, and R. Kaushik. Leveraging aggregate constraints for deduplication. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD)*, June 2007.

[12] S. Dasgupta. Learning mixtures of gaussians. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, 1999.

[13] L. Devroye, L. Gyorfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, 1996.

[14] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley, 2001.

[15] R. Kannan, H. Salmasian, and S. Vempala. The spectral method for general mixture models. In *Proc. COLT*, 2005.

[16] T. Kanungo, D. M. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Y. Wu. A local search approximation algorithm for k-means clustering. *Computational Geometry: Theory and Applications*, 28:89–112, 2004.

[17] J. Kleinberg. An impossibility theorem for clustering. In *NIPS*, 2002.

[18] S.P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[19] R. Ostrovsky, Y. Rabani, L. Schulman, and C. Swamy. The effectiveness of lloyd-type methods for the k-means problem. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, 2006.

[20] L.J. Schulman. Clustering for edge-cost minimization (extended abstract). In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, pages 547–555 (electronic), New York, 2000. ACM.

[21] Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101:1566–1581, 2006.

[22] S. Vempala and G. Wang. A spectral algorithm for learning mixture models. *Journal of Computer and System Sciences*, 68(2):841–860, 2004.

[23] R. Bosagh Zadeh and S. Ben-David. A uniqueness theorem for clustering. In *The 25th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, 2009.