Generating a Diverse Set of High-Quality Clusterings[*]

Jeff M. Phillips, Parasaran Raman, and Suresh Venkatasubramanian

School of Computing, University of Utah
{jeffp,praman,suresh}@cs.utah.edu

Abstract. We provide a new framework for generating multiple good quality partitions (clusterings) of a single data set. Our approach decomposes this problem into two components, generating many high-quality partitions, and then grouping these partitions to obtain k representatives. The decomposition makes the approach extremely modular and allows us to optimize various criteria that control the choice of representative partitions.

1 Introduction

Clustering is a critical tool used to understand the structure of a data set. There are many ways in which one might partition a data set into representative clusters, and this is demonstrated by the huge variety of different algorithms for clustering [9], [35], [8], [27], [45], [31], [43], [14], [28].

Each clustering method identifies different kinds of structure in data, reflecting different desires of the end user. Thus, a key exploratory tool is identifying a diverse and meaningful collection of partitions of a data set, in the hope that these distinct partitions will yield different insights about the underlying data.

Problem specification. The input to our problem is a single data set X. The output is a set of k partitions of X. A *partition* of X is a set of subsets $\mathcal{X}_i = \{X_{i,1}, X_{i,2}, \ldots, X_{i,s}\}$ where $X = \bigcup_{j=1}^{s} X_{i,j}$ and for all j, j' $X_{i,j} \cap X_{i,j'} = \emptyset$. Let $\mathcal{P}_X$ be the space of all partitions of X; since X is fixed throughout this paper, we just refer to this space as $\mathcal{P}$.

There are two quantities that control the nature of the partitions generated. The *quality* of a partition, represented by a function $Q : \mathcal{P} \to \mathbb{R}^+$, measures the degree to which a particular partition captures intrinsic structure in data; in general, most clustering algorithms that identify a single clustering attempt to optimize some notion of quality. The *distance* between partitions, represented by the function $d : \mathcal{P} \times \mathcal{P} \to \mathbb{R}$, is a quantity measuring how dissimilar two partitions are. The partitions $\mathcal{X}_i \in \mathcal{P}$ that do a better job of capturing the structure of the data set X will have a larger quality value $Q(\mathcal{X}_i)$. And the partitions $\mathcal{X}_i, \mathcal{X}_{i'} \in \mathcal{P}$ that are more similar to each other will have a smaller distance value $d(\mathcal{X}_i, \mathcal{X}_{i'})$.

A good set of diverse partitions all have large distances from each other and all have high quality scores.

Thus, the goal is this paper is to *generate a set of k partitions that best represent all high-quality partitions as accurately as possible.*

Related Work. There are two main approaches in the literature for computing many high-quality, diverse partitions. However, both approaches focus only on a specific subproblem. *Alternate clustering* focuses on generating one additional partition of high-quality that should be far from a given set (typically of size one) of existing partitions. *k-consensus clustering* assumes an input set of many partitions, and then seeks to return k representative partitions.

Most algorithms for generating alternate partitions [38, 16, 6, 5, 13, 21, 12] operate as follows. Generate a single partition using a clustering algorithm of choice. Next, find another partition that is both far from the first partition and of high quality. Most methods stop here, but a few methods try to discover more alternate partitions; they repeatedly find new, still high-quality, partitions that are far from all existing partitions. This effectively produces a variety of partitions, but the quality of each successive partition degrades quickly.

Although there are a few other methods that try to discover alternate partitions simultaneously [10, 29, 37], they are usually limited to discovering two partitions of the data. Other methods that generate more than just two partitions either randomly weigh the features or project the data onto different subspaces, but use the same clustering technique to get the alternate partitions in each round. Using the same clustering technique tends to generate partitions with clusters of similar shapes and might not be able to exploit all the structure in the data.

The problem of k-consensus, which takes as input a set of $m \gg k$ partitions of a single data set to produce k distinct partitions, has not been studied as extensively. To obtain the input for this approach, either the output of several distinct clustering algorithms, or the output of multiple runs of the same randomized algorithm with different initial seeds are considered [46, 47]. This problem can then be viewed as a clustering problem; that is, finding k clusters of partitions from the set of input partitions. Therefore, there are many possible optimization criteria or algorithms that could be explored for this problem as there are for clustering in general. Most formal optimization problems are intractable to solve exactly, making heuristics the only option. Furthermore, no matter the technique, the solution is only as good as the input set of partitions, independent of the optimization objective. In most k-consensus approaches, the set of input partitions is usually not diverse enough to give a good solution.

In both cases, these subproblems avoid the full objective of constructing a diverse set of partitions that represent the *landscape of all high-quality partitions.* The alternate clustering approach is often too reliant on the initial partition, has had only limited success in generalizing the initial step to generate k partitions. The k-consensus partitioning approach does not verify that its input represents the space of all high-quality partitions, so a representative set of those input partitions is not necessarily a representative set of all high-quality partitions.

Our approach. To generate multiple good partitions, we present a new paradigm which decouples the notion of distance between partitions and the quality of partitions. Prior methods that generate multiple diverse partitions cannot explore the space of partitions entirely since the distance component in their objective functions biases against partitions close to the previously generated ones. These could be interesting partitions that might now be left out. To avoid this, we will first look at the space of all partitions more thoroughly and then pick non-redundant partitions from this set. Let k be the number of diverse partitions that we seek. Our approach works in two steps. In the first step called the *generation step*, we first sample from the space of all partitions proportional to their quality. Stirling numbers of the second kind, $S(n, s)$ is the number of ways of partitioning a set of n elements into s nonempty subsets. Therefore, this is the size of the space that we sample from. We illustrate the sampling in figure 1. This generates a set of size $m \gg k$ to ensure we get a diverse sample that represents the space of all partitions well, since generating only k partitions in this phase may "accidentally" miss some high quality region of $\mathcal{P}$. Next, in the *grouping step*, we cluster this set of m partitions into k sets, resulting in k clusters of partitions. We then return one representative from each of these k clusters as our output alternate partitions.
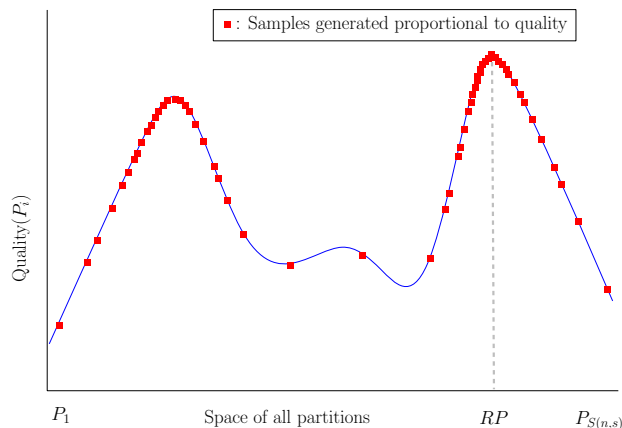

Fig. 1. Sampling partitions proportional to its quality from the space of all partitions with s clusters.

Note that because the *generation step* is decoupled from the *grouping step*, we treat all partitions fairly, independent of how far they are from the existing partitions. This allows us to explore the true density of high quality partitions in $\mathcal{P}$ without interference from the choice of initial partition. Thus, if there is a dense set of close interesting partitions our approach will recognize that. Also because the *grouping step* is run separate from the *generation step*, we can abstract this problem to a generic clustering problem, and we can choose one of many approaches. This allows us to capture different properties of the diversity

of partitions, for instance, either guided just by the spatial distance between partitions, or also by a density-based distance which only takes into account the number of high-quality partitions assigned to a cluster.

From our experimental evaluation, we note that decoupling the generation step from the grouping step helps as we are able to generate a lot of very high quality partitions. In fact, the quality of some of the generated partitions is better than the quality of the partition obtained by a consensus clustering technique called LiftSSD [39]. The relative quality w.r.t. the reference partition of a few generated partitions even reach close to 1. To our best knowledge, such partitions have not been uncovered by other previous meta-clustering techniques. The grouping step also picks out representative partitions far-away from each other. We observe this by computing the closest-pair distance between representatives and comparing it against the distance values of the partitions to their closest representative.

Outline. In Section 2, we discuss a sampling-based approach for generating many partitions proportional to their quality; i.e. the higher the quality of a partition, the more likely it is to be sampled. In Section 3, we describe how to choose k representative partitions from the large collection partitions already generated. We will present the results of our approach in Section 4. We have tested our algorithms on a synthetic dataset, a standard clustering dataset from the UCI repository and a subset of images from the Yale Face database B.

2 Generating Many High Quality Partitions

In this section we describe how to generate many high quality partitions. This requires (1) a measure of quality, and (2) an algorithm that generates a partition with probability proportional to its quality.

2.1 Quality of Partitions

Most work on clustering validity criteria look at a combination of how compact clusters are and how separated two clusters are. Some of the popular measures that follow this theme are S_Dbw, $CDbw$, SD validity index, maximum likelihood and Dunn index $[23, 24, 36, 44, 15, 4, 32, 17]$. Ackerman et. al. also discuss similar notions of quality, namely VR (variance ratio) and WPR (worst pair ratio) in their study of clusterability $[1, 2]$. We briefly describe a few specific notions of quality below.

k-Means quality. If the elements $x \in X$ belong to a metric space with an underlying distance $\delta : X \times X \to \mathbb{R}$ and each cluster $X_{i,j}$ in a partition $\mathfrak{X}_i$ is represented by a single element $\bar{x}_j$, then we can measure the inverse quality of a cluster by $\bar{q}(X_{i,j}) = \sum_{x \in X_{i,j}} \delta(x, \bar{x}_j)^2$. Then the quality of the entire partition is then the inverse of the sum of the inverse qualities of the individual clusters: $\bar{Q}(\mathfrak{X}_i) = 1/(\sum_{j=1}^{s} \bar{q}(X_{i,j}))$.

This corresponds to the quality optimized by s-mean clustering [1], and is quite popular, but is susceptible to outliers. If all but one element of X fit neatly in s clusters, but the one remaining point is far away, then this one point dominates the cost of the clustering, even if it is effectively noise. Specifically, the quality score of this measure is dominated by the points which fit least well in the clusters, as opposed to the points which are best representative of the true data. Hence, this quality measure may not paint an accurate picture about the partition.

Kernel distance quality. We introduce a method to compute quality of a partition, based on the kernel distance [30]. Here we start with a similarity function between two elements of X, typically in the form of a (positive definite) kernel: $K : X \times X \to \mathbb{R}^+$. If $x_1, x_2 \in X$ are more similar, then $K(x_1, x_2)$ is smaller than if they are less similar. Then the overall similarity score between two clusters $X_{i,j}, X_{i,j'} \in \mathfrak{X}_i$ is defined $\kappa(X_{i,j}, X_{i,j'}) = \sum_{x \in X_{i,j}} \sum_{x' \in X_{i,j'}} K(x, x')$, and a single clusters self-similarity for $X_{i,j} \in \mathfrak{X}_i$ is defined $\kappa(X_{i,j}, X_{i,j})$. Finally, the overall quality of a partition is defined $Q_K(\mathfrak{X}_i) = \sum_{j=1}^{s} \kappa(X_{i,j}, X_{i,j})$.

If X is a metric space, the highest quality partitions divide X into s Voronoi cells around s points – similar to s-means clustering. However, its score is dominated by the points which are a good fit to a cluster, rather than outlier points which do not fit well in any cluster. This is a consequence of how kernels like the Gaussian kernel taper off with distance, and is the reason we recommend this measure of cluster quality in our experiments.

2.2 Generation of Partitions Proportional to Quality

We now discuss how to generate a sample of partitions proportional to their quality. This procedure will be independent of the measure of quality used, so we will generically let $Q(\mathfrak{X}_i)$ denote the quality of a partition. Now the problem becomes to generate a set $Y \subset \mathcal{P}$ of partitions where each $\mathfrak{X}_i \in Y$ is drawn randomly proportionally to $Q(\mathfrak{X}_i)$.

The standard tool for this problem framework is a Metropolis-Hastings random-walk sampling procedure [34, 25, 26]. Given a domain X to be sampled and an energy function $Q : X \to \mathbb{R}$, we start with a point $x \in X$, and suggest a new point x_1 that is typically "near" x. The point x_1 is accepted unconditionally if $Q(x_1) \geq Q(x)$, and is accepted with probability $Q(x_1)/Q(x)$ if not. Otherwise, we say that x_1 was *rejected* and instead set $x_1 = x$ as the current state. After some sufficiently large number of such steps t, the expected state of x_t is a random draw from $\mathcal{P}$ with probability proportional to Q. To generate many random samples from $\mathcal{P}$ this procedure is repeated many times.

In general, Metropolis-Hastings sampling suffers from high autocorrelation, where consecutive samples are too close to each other. This can happen when

---

[1] it is commonplace to use k in place of s, but we reserve k for other notions in this paper

far away samples are rejected with high probability. To counteract this problem, often Gibbs sampling is used [41]. Here, each proposed step is decomposed into several orthogonal suggested steps and each is individually accepted or rejected in order. This effectively constructs one longer step with a much higher probability of acceptance since each individual step is accepted or rejected independently. Furthermore, if each step is randomly made proportional to Q, then we can always accept the suggested step, which reduces the rejection rate.

Metropolis-Hastings-Gibbs sampling for partitions. The Metropolis-Hastings procedure for partitions works as follows. Given a partition $\mathcal{X}_i$, we wish to select a random subset $Y \subset X$ and randomly reassign the elements of Y to different clusters. If the size of Y is large, this will have a high probability of rejection, but if Y is small, then the consecutive clusters will be very similar. Thus, we use a Gibbs-sampling approach. At each step we choose a random ordering σ of the elements of X. Now, we start with the current partition $\mathcal{X}_i$ and choose the first element $x_{\sigma(1)} \in X$. We assign $x_{\sigma(1)}$ to each of the s clusters generating s suggested partitions $\mathcal{X}_i^j$ and calculate s quality scores $q_j = Q(\mathcal{X}_i^j)$. Finally, we select index j with probability q_j, and assign $x_{\sigma(1)}$ to cluster j. Rename the new partition as $\mathcal{X}_i$. We repeat this for all points in order. Finally, after all elements have been reassigned, we set $\mathcal{X}_{i+1}$ to be the resulting partition.

Note that auto-correlation effects may still occur since we tend to have partitions with high quality, but this effect will be much reduced. Note that we do not have to run this entire procedure each time we need a new random sample. It is common in practice to run this procedure for some number t_0 (typically $t_0 = 1000$) of *burn-in* steps, and then use the next m steps as m random samples from $\mathcal{P}$. The rationale is that after the burn-in period, the induced Markov chain is expected to have mixed, and so each new step would yield a random sample from the stationary distribution.

3 Grouping the Partitions

Having generated a large collection Z of $m \gg k$ high-quality partitions from $\mathcal{P}$ by random sampling, we now describe a grouping procedure that returns k representative partitions from this collection. We will start by placing a metric structure on $\mathcal{P}$. This allows us to view the problem of grouping as a metric clustering problem. Our approach is independent of any particular choice of metric; obviously, the specific choice of distance metric and clustering algorithm will affect the properties of the output set we generate. There are many different approaches to comparing partitions. While our approach is independent of the particular choice of distance measure used, we review the main classes.

Membership-based distances. The most commonly used class of distances is membership-based. These distances compute statistics about the number of *pairs* of points which are placed in the same or different cluster in both partitions, and return a distance based on these statistics. Common examples include the Rand distance, the variation of information, and the normalized mutual information [33, 40, 42, 7]. While these distances are quite popular, they ignore information

about the spatial distribution of points within clusters, and so are unable to differentiate between partitions that might be significantly different.

Spatially-sensitive distances. In order to rectify this problem, a number of *spatially-aware* measures have been proposed. In general, they work by computing a concise representation of each cluster and then use the earthmover's distance (EMD) [20] to compare these sets of representatives in a spatially-aware manner. These include CDistance [11], d_{ADCO} [6], CC distance [48], and LiftEMD [39]. As discussed in [39], LiftEMD has the benefit of being both efficient as well as a well-founded metric, and is the method used here.

Density-based distances. The partitions we consider are generated via a sampling process that samples more densely in high-quality regions of the space of partitions. In order to take into account dense samples in a small region, we use a *density-sensitive* distance that intuitively spreads out regions of high density. Consider two partitions $\mathcal{X}_i$ and $\mathcal{X}_{i'}$. Let $d : \mathcal{P} \times \mathcal{P} \to \mathbb{R}^+$ be any of the above natural distances on $\mathcal{P}$. Then let $d_Z : \mathcal{P} \times \mathcal{P} \to \mathbb{R}^+$ be a density-based distance defined as $d_Z(\mathcal{X}_i, \mathcal{X}_{i'}) = |\{\mathcal{X}_l \in Z \mid d(\mathcal{X}_i, \mathcal{X}_l) < d(\mathcal{X}_i, \mathcal{X}_{i'})\}|$.

3.1 Clusters of Partitions

Once we have specified a distance measure to compare partitions, we can cluster them. We will use the notation $\phi(\mathcal{X}_i)$ to denote the representative partition $\mathcal{X}$ is assigned to. We would like to pick k representative partitions, and a simple algorithm by Gonzalez [22] provides a 2-approximation to the best clustering that minimizes the *maximum* distance between a point and its assigned center. The algorithm maintains a set of centers $k' < k$ in C. Let $\phi_C(\mathcal{X}_i)$ represent the partition in C closest to $\mathcal{X}_i$ (when apparent we use just $\phi(\mathcal{X}_i)$ in place of $\phi_C(\mathcal{X}_i)$). The algorithm chooses $\mathcal{X}_i \in Z$ with maximum value $d(\mathcal{X}_i, \phi(\mathcal{X}_i))$. It adds this partition $\mathcal{X}_i$ to C and repeats until C contains k partitions. We run the Gonzalez method to compute k representative partitions using LiftEMD between partitions. We also ran the method using the density based distance derived from using LiftEMD. We got very similar results in both cases and we will only report the results from using LiftEMD in section 4. We note that other clustering methods such as k-means and hierarchical agglomerative clustering yield similar results.

4 Experimental Evaluation

In this section, we show the effectiveness of our technique in generating partitions of good divergence and its power to find partitions with very high quality, well beyond usual consensus techniques.

Data. We created a synthetic dataset *2D5C* with 100 points in 2-dimensions, for which the data is drawn from 5 Gaussians to produce 5 visibly separate clusters. We also test our methods on the Iris dataset containing 150 points in

4 dimensions from UCI machine learning repository [18]. We also use a subset of the Yale Face Database B [19] (90 images corresponding to 10 persons and 9 poses in the same illumination). The images are scaled down to 30x40 pixels.

Methodology. For each dataset, we first run k-means to get the first partition with the same number of clusters specified by the reference partition. Using this as a seed, we generate $m = 4000$ partitions after throwing away the first 1000 of them. We then run the Gonzalez k-center method to find 10 representative partitions. We associate each of the 3990 remaining partitions with the closest representative partition. We compute and report the quality of each of these representative partitions. We also measure the LiftEMD distance to each of these partitions from the reference partition. For comparison, we also plot the quality of consensus partitions generated by LiftSSD [39] using inputs from k-means, single-linkage, average-linkage, complete-linkage and Ward's method.

4.1 Performance Evaluation

Evaluating partition diversity. We can evaluate partition diversity by determining how close partitions are to their chosen representatives using LiftEMD. Low LiftEMD values between partitions will indicate redundancy in the generated partitions and high LiftEMD values will indicate good partition diversity. The resulting distribution of distances is presented in Figures 2(a), 2(b), 2(c), in which we also mark the distance values between a representative and its closest other representative with *red* squares. Since we expect that the representative partitions will be far from each other, those distances provide a baseline for distances considered large. For all datasets, a majority of the partitions generated are generally far from the closest representative partition. For instance, in the Iris data set (2(a)), about three-fourths of the partitions generated are far away from the closest representative with LiftEMD values ranging between 1.3 and 1.4.

Evaluating partition quality. Secondly, we would like to inspect the quality of the partitions generated. Since we intend the generation process to sample from the space of all partitions proportional to the quality, we hope for a majority of the partitions to be of high quality. The ratio between the kernel distance quality Q_K of a partition to that of the reference partition gives us a fair idea of the relative quality of that partition, with values closer to 1 indicating partitions of higher quality. The distribution of quality is plotted in Figures 3(a)3(b)3(c). We observe that for all the datasets, we get a normally distributed quality distribution with a mean value between 0.62 and 0.8. In addition, we compare the quality of our generated partitions against the consensus technique LiftSSD. We mark the quality of the representative partitions with *red* squares and that of the consensus partition with a *blue* circle. For instance, chart 3(a) shows that the relative quality w.r.t. the reference partition of three-fourths of the partitions is better than that of the consensus partition. For the Yale Face data, note that we have two reference partitions namely *by pose* and *by person* and we chose the partition *by person* as the reference partition due to its superior quality.

Visual inspection of partitions. We ran multi-dimensional scaling [3] on the all-pairs distances between the 10 representatives for a visual representation of the space of partitions. We compute the variance of the distances of the partitions associated with each representative and draw Gaussians around them to depict the *size* of each cluster of partitions. For example, for the Iris dataset, as we can see from chart 4(a), the clusters of partitions are well-separated and are far from the original reference partition. In figure 5, we show two interesting representative partitions on the Yale face database. We show the mean image from each of the 10 clusters. Figure 5(a) is a representative partition very similar to the partition *by person* and figure 5(b) resembles the partition *by pose*.

5 Conclusion

In this paper we introduced a new framework to generate multiple non-redundant partitions of good quality. Our approach is a two stage process: in the *generation* step, we focus on sampling a large number of partitions from the space of all partitions proportional to the quality and in the *grouping* step, we identify k representative partitions that best summarizes the space of all partitions.
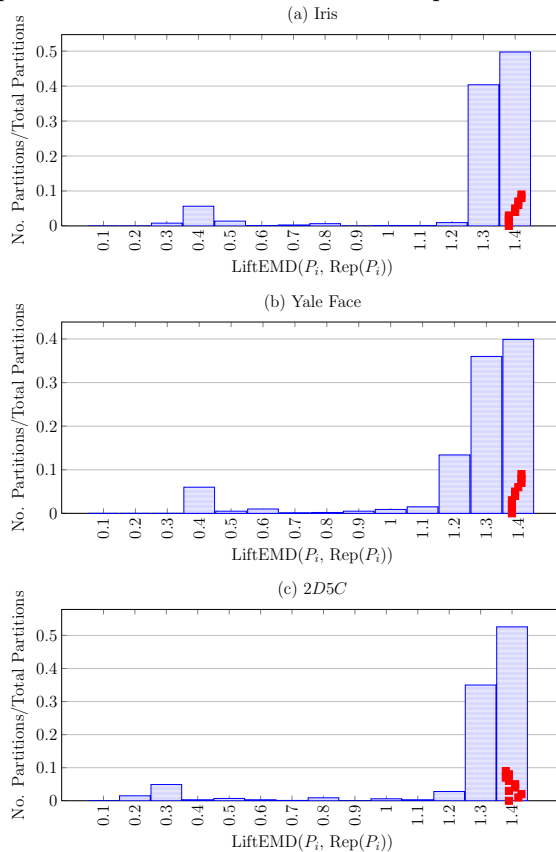

Fig. 2. Distance between partition and its representative.
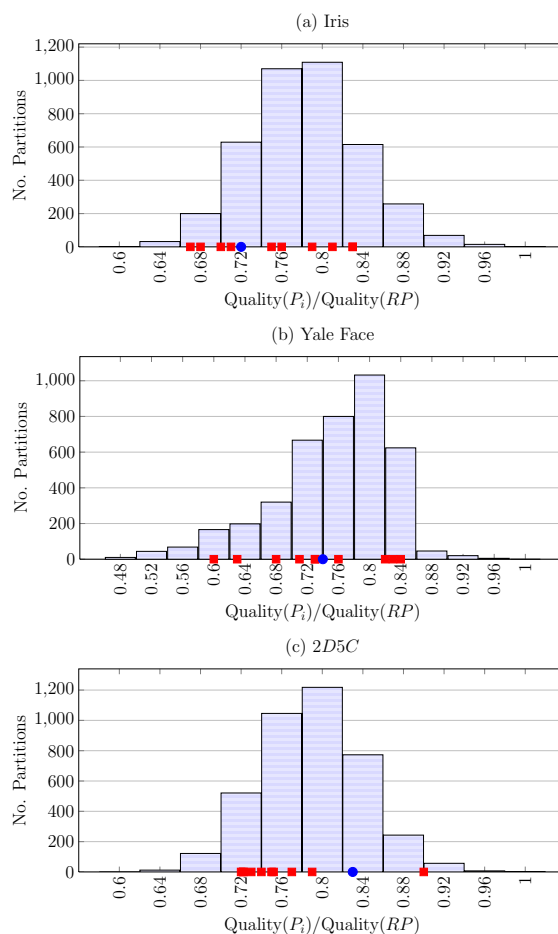
(a) Iris

(b) Yale Face

(c) 2D5C

Fig. 3. Quality of Partitions.

References

1. M. Ackerman and S. Ben-David. Measures of Clustering Quality: A Working Set of Axioms for Clustering. *In proceedings of the 22nd Neural Information Processing Systems*, 2008.
2. M. Ackerman and S. Ben-David. Clusterability: A theoretical study. *Journal of Machine Learning Research - Proceedings Track*, 5:1–8, 2009.
3. A. Agarwal, J. M. Phillips, and S. Venkatasubramanian. Universal multidimensional scaling. *In proceedings of the 16th ACM SIGKDD*, 2010.
4. J. Aldrich. R. A. Fisher and the making of maximum likelihood 1912–1922. *Statist. Sci.*, 12(3):162–176, 1997.
5. E. Bae and J. Bailey. Coala: A novel approach for the extraction of an alternate clustering of high quality and high dissimilarity. *In proceedings of the 6th ICDM*, 2006.
6. E. Bae, J. Bailey, and G. Dong. A clustering comparison measure using density profiles and its application to the discovery of alternate clusterings. *Data Mining and Knowledge Discovery*, 2010.
7. A. Ben-Hur, A. Elisseeff, and I. Guyon. A stability based method for discovering structure in clustered data. In *Pacific Symposium on Biocomputing*, 2002.
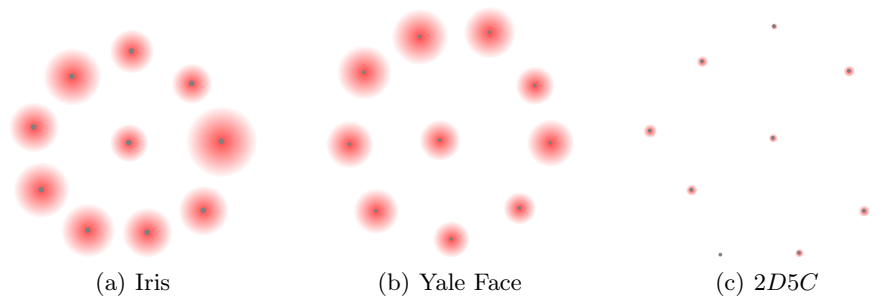
(a) Iris   (b) Yale Face   (c) 2D5C

Fig. 4. MDS rendering of the LiftEMD distances between all representative partitions.

Fig. 5. Visual illustration of two interesting representative partitions on Yale Face.

8. P. Berkhin. A survey of clustering data mining techniques. In J. Kogan, C. Nicholas, and M. Teboulle, editors, *Grouping Multidimensional Data*, pages 25–71. Springer, 2006.
9. R. E. Bonner. On some clustering techniques. *IBM J. Res. Dev.*, 8:22–32, January 1964.
10. R. Caruana, M. Elhaway, N. Nguyen, and C. Smith. Meta clustering. *Data Mining, IEEE International Conference on*, 0:107–118, 2006.
11. M. Coen, H. Ansari, and N. Fillmore. Comparing clusterings in space. In *ICML*, 2010.
12. X. H. Dang and J. Bailey. Generation of alternative clusterings using the CAMI approach. *In proceedings of 10th SDM*, 2010.
13. X. H. Dang and J. Bailey. A hierarchical information theoretic technique for the discovery of non linear alternative clusterings. *In proceedings of the 16th ACM SIGKDD*, 2010.
14. S. Das, A. Abraham, and A. Konar. *Metaheuristic Clustering*. Springer Publishing Company, Incorporated, 1st edition, 2009.
15. R. N. Dave. Validating fuzzy partitions obtained through c-shells clustering. *Pattern Recogn. Lett.*, 17:613–623, May 1996.
16. I. Davidson and Z. Qi. Finding alternative clusterings using constraints. *In proceedings of the 8th ICDM*, 2008.
17. J. C. Dunn. Well separated clusters and optimal fuzzy-partitions. *Journal of Cybernetics*, 1974.
18. A. Frank and A. Asuncion. UCI machine learning repository, 2010.
19. A. Georghiades, P. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 23(6):643–660, 2001.
20. C. R. Givens and R. M. Shortt. A class of wasserstein metrics for probability distributions. *Michigan Math Journal*, 31:231–240, 1984.
21. D. Gondek and T. Hofmann. Non-redundant data clustering. *In proceedings of the 4th ICDM*, 2004.

22. T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.*, 38:293–306, 1985.
23. M. Halkidi and M. Vazirgiannis. Clustering validity assessment: finding the optimal partitioning of a data set. *In proceedings of the 1st ICDM*, 2001.
24. M. Halkidi, M. Vazirgiannis, and Y. Batistakis. Quality scheme assessment in the clustering process. *In proceedings of the 4th PKDD*, 2000.
25. W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57:97–109, 1970.
26. P. D. Hoff. *A First Course in Bayesian Statistical Methods*. Springer, 2009.
27. A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recogn. Lett.*, 2010.
28. A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Comput. Surv.*, 31:264–323, September 1999.
29. P. Jain, R. Meka, and I. S. Dhillon. Simultaneous unsupervised learning of disparate clusterings. *Stat. Anal. Data Min.*, 1:195–210, November 2008.
30. S. Joshi, R. V. Kommaraju, J. M. Phillips, and S. Venkatasubramanian. Comparing distributions and shapes using the kernel distance (to appear). 27th *Annual Symposium on Computational Geometry*, 2011.
31. G. G. C. Ma; and J. Wu. *Data Clustering: Theory, Algorithms, and Applications*. SIAM, Society for Industrial and Applied Mathematics, illustrated edition, May 2007.
32. D. J. C. MacKay. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, New York, NY, USA, 2002.
33. M. Meilă. Comparing clusterings–an information based distance. *J. Multivar. Anal.*, 2007.
34. N. Metropolis, A. W. Rosentbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 1953.
35. P. Michaud. Clustering techniques. *Future Generation Computer Systems*, 1997.
36. G. Milligan and M. Cooper. An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, 50:159–179, 1985. 10.1007/BF02294245.
37. D. Niu, J. G. Dy, and M. I. Jordan. Multiple non-redundant spectral clustering views. In *ICML'10*, pages 831–838, 2010.
38. Z. Qi and I. Davidson. A principled and flexible framework for finding alternative clusterings. *In proceedings of the 15th ACM SIGKDD*, 2009.
39. P. Raman, J. M. Phillips, and S. Venkatasubramanian. Spatially-aware comparison and consensus for clusterings (to appear). *Proceedings of 11th SDM*, 2011.
40. W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.
41. G. O. Roberts, A. Gelman, and W. R. Gilks. Weak convergence and optimal scaling of random walk metropolis algorithms. *Annals of Applied Probability*, 7:110–120, 1997.
42. A. Strehl and J. Ghosh. Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *JMLR*, 3:583–617, 2003.
43. P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.
44. S. Theodoridis and K. Koutroumbas. *Pattern Recognition*. Elsevier, 2006.
45. R. Xu and D. Wunsch. *Clustering*. Wiley-IEEE Press, 2009.
46. Y. Zhang and T. Li. Consensus clustering + meta clustering = multiple consensus clustering. *Florida Artificial Intelligence Research Society Conference*, 2011.
47. Y. Zhang and T. Li. Extending consensus clustering to explore multiple clustering views. *In proceedings of 11th SDM*, 2011.
48. D. Zhou, J. Li, and H. Zha. A new Mallows distance based metric for comparing clusterings. In *ICML*, 2005.