Holistic Assessment of Structure Discovery Capabilities of Clustering Algorithms

Frank Höppner ✉ and Maximilian Jahnke

Ostfalia University of Applied Sciences
Dept. of Computer Science, D-38302 Wolfenbüttel, Germany

Abstract. Existing cluster validity indices often possess a similar bias as the clustering algorithm they were introduced for, e.g. to determine the optimal number of clusters. We suggest an efficient and holistic assessment of the structure discovery capabilities of clustering algorithms based on three criteria. We determine the robustness or stability of cluster assignments and interpret it as the confidence of the clustering algorithm in its result. This information is then used to label the data and evaluate the consistency of the stability-assessment with the notion of a cluster as an area of dense and separated data. The resulting criteria of stability, structure and consistency provide interpretable means to judge the capabilities of clustering algorithms without the typical biases of prominent indices, including the judgment of a clustering tendency.

1 Introduction

Clustering algorithms are used in various settings of exploratory data analysis, pattern recognition, etc. They are often used as a tool in a longer preprocessing pipeline to support some other goal than just clustering the data for its own sake (e.g. classification, discretization, compression). The *best clustering algorithm* is then simply the one that supports the original goal best, so we may only be in charge of providing an (external) evaluation of the surrounding task.

We exclude such objectives in this paper, but concentrate on those cases where *clustering* itself is the core objective. We thus understand the *clustering task* in a narrow sense as *structure discovery*: Does the dataset itself suggest a partitioning into multiple, separated groups? This would be a valuable result in an explorative analysis of new data, for instance, it would suggest to explore and compare the partitions individually. In the context of, say, customer relationship management we would *not* ask if it is *possible* to subdivide all customers into groups, which seems always possible in one way or another, but whether the data provides evidence that customers naturally decompose in distinctive groups. This is also reflected by widely used definitions of clustering, where clusters are well-separated groups that define a compact or dense area of data.

From a *structure discovery* perspective, a clustering algorithm claims that it has discovered *structure* in the dataset. So the research question in this paper is how to assess the capabilities of the various existing clustering algorithms in this regard. Although the large number of clustering algorithms is flanked by an impressive number of validity indices, we argue that they are usually not suited for a fair comparison across many different clustering algorithms with respect to their structure discovery capabilities. The main contribution of this paper is a holistic assessment of the structure discovery capabilities of clustering algorithms. We determine the robustness or stability of cluster assignments and interpret it as the confidence of the clustering algorithm in its result. This information is then used to label the data and evaluate its consistency with the notion of a cluster as an area of dense and separated data. This approach allows us to apply methods that are otherwise restricted to supervised learning. The three criteria of stability, discovered structure and consistency provide better means to judge about the capabilities of clustering algorithms without the typical biases of prominent indices, including the judgment of a clustering tendency.

2 Related Work

There is a great variety of clustering algorithms, covered in various textbooks, e.g. [1,8,13]. We do not focus on any particular type of clustering algorithm, but will use a spectrum of well-known algorithms (k-means, hierarchical clustering, mean shift, dbscan) as representatives. We assume the reader is familiar with these popular algorithms. They all share – more or less – the same goal, but vary in the computational approach and bias. The review [12] advises to choose an algorithm based on "the manner in which clusters are formed", which clearly demonstrates the dilemma we face if clustering is intended as an explorative technique and not much is known about the data yet.

With so many algorithms at hand, it seems natural to try them all on new data. This leads to the question, which result we should trust most. Some suggest to use external information (class labels), which might be the right approach if classification is the ultimate goal. With explorative structure discovery in mind we agree "*that it is an inherent flaw in design of clustering algorithms if the researcher designing the algorithm evaluates it only w.r.t. the class labels of classification datasets*" [9], because the class labels do not necessarily respect the typical properties of clusters, such as compactness and separation. But there are also many cluster validity measures that consider *internal information* only (rather than external class labels). Recent extensive studies [2,4] compare 30 such indices and among the best-performing indices were Silhouette [18], Davies-Bouldin (DB) [5], Calinski-Harabasz (CH) [3], and SDBw [11]. The study [19] uses 10 external and 3 internal measures (Silhouette [18], Dunn [7], DB [5]).

Many internal measures were introduced to overcome a parameter selection problem. The k-means algorithm, for instance, requires the number of clusters to be specified in advance, so the algorithm is run for multiple values and the validity index identifies the best choice. This leads to measures particularly tailored to single clustering algorithms (e.g. [17]). For such a purpose a measure works best, if it adopts the bias of the considered clustering algorithm. But if we intend to compare the results of various clustering algorithms *with different biases*, an evaluation based on such a measure would not be impartial (cf. [16]).

While the literature agrees on the objectives of clustering on an abstract level (compactness and separation), the exact interpretation may vary considerably. K-means was designed with spherical clusters in mind, so the mean distance to the cluster center is an appropriate way of measuring compactness, but arbitrary cluster shapes will not be evaluated adequately. Sixteen out of the 30 cluster validity indices covered in [2], however, include the notion of a cluster centroid, which represents a bias on the cluster shape. Other examples for biases on the shape include the use of a cluster diameter or an average within-cluster-distance.

To identify a natural grouping, clusters need to be separated. But how important is the actual distance between clusters? Some measures use a ratio of intra-cluster and inter-cluster distance. While meaningful for small ratios, above some threshold (e.g. > 3) we consider clusters as being well-separated, regardless of the actual ratio. Incorporating the ratio in the measure may overemphasize the separation of clusters. Yet other measures incorporate concepts like the single nearest neighbor (e.g. the Dunn index). They are used, for instance, to measure the gap between two clusters (closest point of a different cluster). As many partitional clustering algorithms exhaustively assign all data points to some cluster, including noise and outliers, such measures are heavily affected by noisy datasets. The measure assumes a noise-free void between the clusters, which also represents a bias. These problems underline that the results of many cluster validity indices for two different algorithms are difficult to interpret, to say the least. A worse validity index cannot be unambiguously attributed to a worse clustering result, it might as well be caused by a bias-mismatch.

Many studies have applied algorithms repeatedly to accumulate evidence of multiple clusterings to find a better partition. In [10] the accumulated evidence was used to compose new similarity data to which yet another clustering algorithm may be applied. Using (only) the stability of the obtained results as a validity measure was proposed in [15]. The stability of k-means clustering was also examined in [14] to pick the correct number of k-means clusters, but there it was observed that the stability correlated well with the accuracy of (ensemble) clustering for some datasets – but poorly with other datasets. In this work we are neither interested in improving partitions nor in parameter selection for a particular clustering algorithm, but to directly compare the performance of *different clustering algorithms*. This includes but is not limited to the stability of the results, as we will demonstrate that stability alone is not sufficient.

3 Threefold Assessment of Structure Discovery

We assume a dataset D of size $n = |D|$ is given. We denote $\mathcal{P} = \{C_1, \ldots, C_c\}$ as a **partition of D of size c** if $\forall 1 \leq i, j \leq c$: $C_i \subseteq D$, $C_i \cap C_j = \emptyset$ and $\bigcup_i C_i \subseteq D$. We use the abbreviation $\bigcup \mathcal{P}$ for $\bigcup_{i=1}^{c} C_i$. Note that we do not require $\bigcup_i C_i = D$: some algorithms (e.g. dbscan) mark data as outliers. We also remove singleton clusters as they also represent outliers.

A clustering algorithm delivers a partition of D of size c, where the c groups are usually called clusters. We use the notation $C_x^{\mathcal{P}}$ for any $x \in D$ to refer to the unique cluster $C_j \in \mathcal{P}$ with $x \in C_j$. For illustrative purposes we use R implementations of hierarchical clustering (single-, complete- and average-linkage), k-means, dbscan and meanshift with varying parameters (e.g. 2-7 clusters). For the hierarchical clustering we obtain the final clusters in a rather naive way by cutting the tree to get the intended number of subtrees (clusters).

3.1 Point Stability

Lacking data from the full population, clustering algorithms are typically executed on a (random) sample, so there is always some uncertainty in the selection of the data involved. We can check an obtained partition to see whether x and y belong to the same cluster, but we actually want to know if x and y would belong to the same cluster *in general*, that is, if the full population was clustered rather than this particular data sample only.

We have only access to a partition $\mathcal{P}$ obtained from our algorithm by applying it to sample D. We would like to know, for any $x \in D$, how likely other objects y, co-clustered with x in $\mathcal{P}$, belong to the same cluster if we had a different sample. Assuming the existence of a *ground truth partition* $\mathcal{T}$ for a moment, we are interested, for a given x, in

$$P(y \in C_x^{\mathcal{T}} | y \in C_x^{\mathcal{P}}) = \frac{|(C_x^{\mathcal{T}} \cap C_x^{\mathcal{P}}) \backslash \{x\}|}{|C_x^{\mathcal{P}} \backslash \{x\}|} = \frac{|C_x^{\mathcal{T}} \cap C_x^{\mathcal{P}}| - 1}{|C_x^{\mathcal{P}}| - 1} \tag{1}$$

This conditional probability characterizes the stability of a single data point as it is perceived by the selected clustering algorithm. A probability of 1 would mean that all *co-grouped data* of x in $\mathcal{P}$ would actually co-group identically in the true partition. As there is no chance of knowing $\mathcal{T}$ and even $\mathcal{P}$ depends on our sample D, we estimate this probability by executing the same algorithm multiple times on different subsets of D: We may then estimate how likely a point y belongs to the same cluster as x (in any other partition), given we observed that x and y co-group in one given partition.

Definition 1 (Point Stability). *Given a dataset D and a clustering algorithm. Let $k, m \in \mathbb{N}$. Just as in k-fold cross-validation, we use a random partition $\mathcal{R} = \{R_1, \dots, R_k\}$ of D with equal-sized groups and define (training) datasets $D_i = D \backslash R_i$, $1 \le i \le k$. This process is repeated m times with shuffled data, such that we obtain a set M of $k \cdot m$ different partitions by applying the clustering algorithm to the resp. D_i. We define the* **(k,m)-point stability** *of $x \in D$ as*

$$PS(x) := \frac{1}{|M_x^2|} \sum_{(\mathcal{P},\mathcal{Q}) \in M_x^2} P(y \in C_x^{\mathcal{Q}} | y \in C_x^{\mathcal{P}}) = \frac{1}{|M_x^2|} \sum_{(\mathcal{P},\mathcal{Q}) \in M_x^2} \frac{|C_x^{\mathcal{Q}} \cap C_x^{\mathcal{P}}| - 1}{|C_x^{\mathcal{P}}| - 1}$$

where M_x^2 is defined as $M_x^2 := \{(\mathcal{P}, \mathcal{Q}) \,|\, \mathcal{P} \in M, \mathcal{Q} \in M, \mathcal{P} \neq \mathcal{Q}, x \in \mathcal{P}, x \in \mathcal{Q}\}$.

Having removed singleton clusters (cf. page 3), we will not face a division by zero. Throughout the paper we use $k = m = 5$, which yields 25 partitions and $25 \cdot 24 = 600$ partition comparisons. The point stability is already an informative

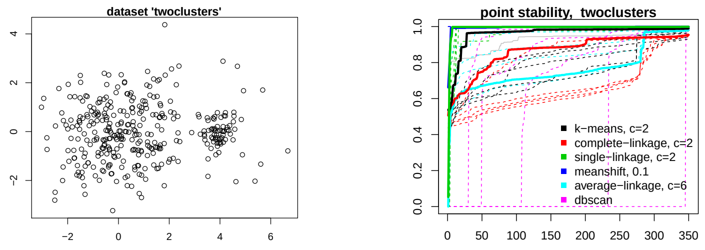

Fig. 1. Simple dataset with two clusters (left) and point stability plot (right) for all considered clustering algorithms (color-coded). Thick lines are discussed in the text.

tool if all values are sorted and plotted as shown in Fig. 1. For the sample data on the left, each line in the point stability plot on the right corresponds to one algorithm/parameter setting. The color indicates the clustering algorithm, the few thick lines are from top to bottom: single-linkage ($c = 2$), almost identical to mean-shift (threshold 0.1), k-means ($c = 2$), complete-linkage ($c = 2$), and average-linkage ($c = 6$). About 50 data objects receive a point stability close to 1 from many algorithms, which means that they get clustered reliably. They correspond to the small cluster on the right. The remaining objects receive quite different stability values. As the true number of clusters is 2, the large cluster is splitted arbitrarily depending on the chosen subsample for $c > 2$. Thus data pairs get grouped differently from run to run and their stability decreases. But even when the clustering algorithms were asked for 2 clusters, there is still considerable variance in the stability curves. The stability of the highlighted meanshift and single-linkage (top curves) is clearly superior over k-means and average-linkage, which respond more sensitive to changes in the sample.

3.2 Stably Discovered Structure

In Fig. 1 two algorithms achieved consistently high stability values for almost all data objects, but for a completely different reason. As already mentioned, the hierarchical single-linkage algorithm was used in a naive way: the hierarchy was cut off to obtain a certain number of clusters. In the particular case of Fig. 1(left), which contains noisy data points, the first few clusters are typically singleton clusters that correspond to outliers. The second cluster then consists of all remaining data and it is not surprising to achieve high stability values for all of them. In contrast, the results of the meanshift clustering consistently discovered both clusters in Fig. 1. Stability alone is thus not sufficient, we have to measure the amount of actually discovered structure.

Definition 2 (Stable Partition). *We define the largest set $D_S \subseteq D$ of data objects that are robustly clustered by the clustering algorithm (i.e. have a point stability of 1) as the* **stably clustered data**. *From all obtained partitions $\mathcal{P}_i \in M$ we can thus identify a single* **stable partition** *$\mathcal{P}_S$ which (1) consists of* *all stably clustered data* ($\bigcup \mathcal{P}_S = D_S$) *and (2) is consistent with all partitions* ($\forall \mathcal{P}_i : \forall C \in \mathcal{P}_i : \exists C_s \in \mathcal{P}_S : C \cap D_S = C_S$)

To identify $\mathcal{P}_S$ we have to keep in mind that, although stability is measured object-wise, an unstable object degrades the stability of *all data objects* in the same cluster. In other words, removing one instable object will increase the stability of all remaining objects in a cluster. Therefore we identify D_S by ordering all objects (increasingly) by their point stability and successively removing objects until all remaining points reach a stability of 1. Each removal increases the stability of all objects in C_x, so D_S is usually much larger than the set of objects that receive values close to 1 in the point stability plots of Fig. 1. We will discuss how to compute D_S efficiently in section 3.5.

So we define the fraction of stable data $|D_S|/|D|$ as the **stability index**, which serves as a first quality indicator. The higher the stability index, the more data was clustered reliably. But we have seen that the number of data objects alone is not sufficient since all stable objects may belong to a single cluster (cf. single-linkage example): Without any (sub)structure being discovered, a high stability is pointless. We may add the number of clusters as a second quality indicator, but this would not take the cluster sizes into account. Instead we use the partition entropy of the stable partition as a **(discovered) structure index**: In an information-theoretic sense, entropy denotes the amount of information in a transmitted message. The message consists of the (reliably) assigned clusters to each data object. The higher the amount of information in the cluster assignments, the more structure has been discovered. Then three clusters receive a higher structure value than two clusters, two equal sized clusters receive a higher value than a 95%:5% cluster constellation (less structured).

Definition 3 (Partition Entropy). *Given a partition* $\mathcal{P} = \{C_1, \ldots, C_n\}$, *by partition entropy we refer to* $PE(\mathcal{P}) = -\sum_{i=1}^{n} p_i \log p_i$ *where* $p_i = \frac{|C_i|}{|\bigcup \mathcal{P}|}$.

Fig. 2(left) shows which data was recognized as stable for a k-means clustering ($c = 2$) and Fig. 2(middle) for average-linkage clustering ($c = 3$). The stably clustered data is shown as black crosses, the instably clustered data as red circles. For k-means we see that all red points lie half way between both cluster centers, the right cluster contains more data than appropriate (due to k-means' bias towards equal-sized clusters). For the average-linkage example the number of clusters was not ideal ($c = 3$ instead of 2): depending on the current subsample, the surplus middle cluster varied in location and size, leading to a large portion of instably clustered data. Compared to k-means, fewer data was stably clustered and the structure index is similar as the stable partition also consists of two clusters only – despite its initialization with $c = 3$.

All algorithms can be compared in the scatterplot of the stability index and the structure index as shown in Fig. 2(right). A single point corresponds to the evaluation of a clustering algorithm (with fixed parameters). The horizontal line indicates the entropy for 2 equal-sized clusters. All results at the bottom of the Fig. correspond to algorithms, where the *fraction of stable data* consisted of

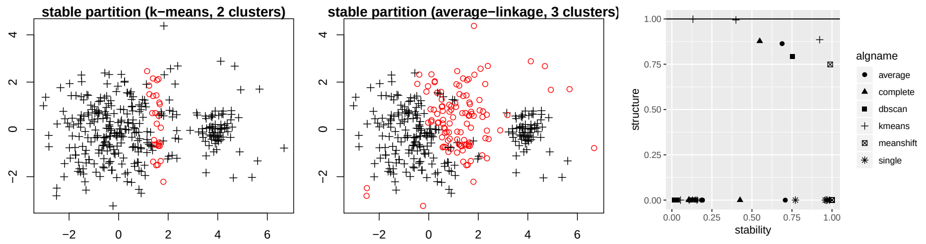

Fig. 2. Stably clustered data (black), instable portion in red for k-means (c=2) and average-linkage (c=3). Right: Entropy and size of stable partition for all algorithms.

a *single cluster only*: the algorithms did not identify any substructure reliably. Another group of results aligns somewhat below the line 'entropy of 1'. From our background information about the dataset (two unequal cluster sizes) we expect the best result to be near this line but not to reach it, because the clusters are not of the same size. Only the stable partition of the meanshift algorithm covers almost 100% of the data **and** reliably detects both clusters. The second best result comes from k-means, which has a higher structure index but less stably clustered data (lower stability index). From the plot we can read immediately, that the meanshift and k-means runs are superior to all other results wrt. to stability and structure index. The inappropriate assignment of data from the “left true cluster” to the “right k-means cluster” is, however, not yet reflected.

3.3 Compactness and Separation

So far we have evaluated the resulting partitions only, but did not use any distance information. The example of Fig. 2(left) shows that information from the partition alone is not sufficient. K-means claims two clusters of roughly the same size, but cuts off some data from the large cluster and disregards the cluster separation. This will be addressed by a third criterion. We will not make assumptions about the shape of the clusters as this would bias our measure towards clustering algorithms with the same assumption.

We have to clarify our notion of compactness and separation first. Clusters correspond to dense groups of data objects, so cluster members should be identifiable by means of some level of data density. As a simple indicator of density we use the following distance $d_k(x)$ to the k^{th} neighbor:

Definition 4 (k^{th} nearest neighbor). *By $d_k(x) = \max_{y \in N_k(x)} \|x-y\|$ we denote the distance to the k^{th} nearest neighbor of x in D, where $N_k(x) \subseteq D$ is the k-neighborhood around x such that $\forall y \in D \backslash N_k(x) : \|x-y\| \geq d_k(x)$ and $|N_k(x)| = k+1$.*

We intend to use $d_k(x)$ as a score that indicates the predisposition of x to belong to a cluster. If a data object has k neighbors within some (small) threshold distance, we may speak of a compact, densely packed area, which

Table 1. Consistency of 'notion of compactness' (density-based, $d_k \leq \varrho$) and 'well-clustered' (partition-based, pure and stable)

	stably clustered and pure	instably clustered or impure
within cluster: $d_k \leq \varrho$	TP	FP
outside cluster: $d_k > \varrho$	FN	TN

$$\text{precision} = \frac{TP}{TP+FP}$$

$$\text{recall} = \frac{TP}{TP+FN}$$

therefore qualifies for cluster membership. Being part of a cluster, we expect objects in the neighborhood to belong to the same cluster. We do *not* expect data from other clusters in the neighborhood (this would violate the separation). Furthermore, we expect from a good clustering algorithm to stably identify a cluster, that is, x as well as its neighbors (but not any instably clustered data).

For any $x \in D$ we now have two sources of information: (i) Based on the distance, we infer a *clustering tendency* from $d_k(x) \leq \varrho$ for some density threshold ϱ. (ii) From our stability analysis we know whether our clustering algorithm perceives x as part of a stable cluster. For a good clustering, both information about x should be consistent: dense data should be clustered stably. So the third criterion measures how well both views match. Are the objects, that likely belong to clusters, stably clustered and well separated? To measure the degree of consistency we consider a contingency table shown in Table 1: If a data object qualifies for cluster membership ($d_k \leq \varrho$), and the algorithm clustered it stably and with pure neighborhood, the compactness (small d_k distance) and the separation (stable, pure neighborhood) are consistent (true positive). If, however, the object was not clustered stably or the neighborhood is not pure, we recognize a false positive.

The consistency depends on ϱ: For a small threshold, only the dense center of a cluster may meet the condition $d_k \leq \varrho$. Those points are likely to get stably clustered (TP, increases precision), but many other stably-clustered but less dense areas will be missed (FN, low recall). Data towards the border of a cluster may require a larger threshold ϱ to accept them as 'dense enough to be part of a cluster'. With ϱ getting larger, the risk of including data from other clusters in the neighborhood increases. For non-separated clusters, an increasing recall may therefore lead to a drop in precision. The overall consistency of the partition with the data density is thus well-captured by means of a precision-recall graph, cf. Fig. 3(left) for the data set from Fig. 1. We expect a curve of a reasonable run to start in the top left corner. If there is a threshold ϱ that separates all clusters from another, we reach 100% recall with 100% precision (line close to the ideal line (0,1)-(1,1)). The earlier the curve drops from the ideal line, the worse the consistency of the clusters (not separated or not stably clustered). We may also observe curves starting at (0,0), which indicates that the clustering algorithm did not even succeed to stably cluster the regions with the highest data density. This may happen if c was chosen too high and multiple prototypes compete for a single true cluster: The true cluster is split into multiple parts and its core data gets assigned to alternate clusters, which renders them instable.

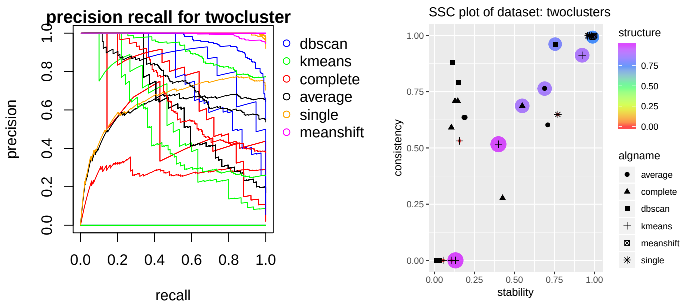

Fig. 3. Precision-Recall graph for data set `twoclusters` and its SSC-plot.

As the optimal curve in such a graph is a constant line of precision 1, the area under the precision-recall graph serves as the third **consistency index**. The final **SSC-plot** (stability, structure, consistency) shows the relative size of the stable dataset on the x-axis, the consistency (area under precision-recall curve) on the y-axis and the structure (entropy of the stable partition) by the dot size and its color, cf. Fig. 3(right). Optimal results lie in the upper right corner (all data stably clustered, highly consistent). Several runs lie in this corner in Fig. 3(right), but only one meanshift result discovered a non-trivial structure. The second best result is still k-means (for 2 clusters), but with lower stability and consistency values, the latter reflects the missing separation in Fig. 2(left).

3.4 Ranking

The SSC-plot offers a holistic view on the algorithms performance. In a particular context there might be a focus on one of the measures stability, structure, or consistency. If considered as a multicriteria problem, there might be no single best solution, so the Pareto front has to be explored. When a unique ranking is needed to select the best algorithm automatically, we suggest to sum how many other runs are dominated by a given run in the three criteria individually. In our running example the two highest scores correspond to the best solutions identified in the discussion: the meanshift run gets a score of 81 and the k-means a score of 77 with an average score of 41.6. (With 34 runs in the experiment the theoretical maximum is 33 for each index, 99 in total for all three indices.)

3.5 Efficient Calculation of a p% Stable Partition

In this section we discuss the identification of a partition with an average stability of p%. Calculating a single point stability $PS(x)$ for data object x is straightforward: For r partitions we carry out $r \cdot (r-1)$ comparisons; for each comparison a contingency table of size $c \times c$ is constructed in $O(n)$, where c is the number of clusters. The contingency tables contain the cluster intersections of definition 1. We arrive at a complexity of $O(r^2 \times n)$. We calculate $PS(x)$ once, order the points increasingly by their point stability, and successively remove objects from the dataset in this order. We stop if the average $PS(x)$ for all remaining x reaches $p\%$. A recalculation of $PS(x)$ after each removal would lead us to $O(r^2 \cdot n^2)$.

A more efficient implementation makes use of the fact that upon removal of the next data object the contingency tables need not be recalculated because they change only in one cell by 1. The partitions associated with the table tell us which cell is affected. Furthermore we want to remove data objects successively until we reach the desired average stability for the remaining data, so we actually do not need to calculate individual stability values $PS(x)$ but only the average stability $PS(x)$ of all remaining x. For the average stability $\overline{PS}$ we have

$$\overline{PS} = \frac{1}{|D|} \sum_{x \in D} PS(x) = \frac{1}{|D|} \sum_{x \in D} \frac{1}{|M_x^2|} \sum_{(\mathcal{P},\mathcal{Q}) \in M_x^2} \frac{|C_x^{\mathcal{Q}} \cap C_x^{\mathcal{P}}| - 1}{|C_x^{\mathcal{P}}| - 1} \tag{2}$$

As the rightmost term in (2) is the same for all data objects in the same cell of the contingency table (and $|M_x^2| = |M|(|M| - 1)$ regardless of x) we arrive at

$$\overline{PS} = \frac{1}{|D||M|(|M|-1)} \sum_{(\mathcal{P},\mathcal{Q}) \in M^2, \mathcal{P} \neq \mathcal{Q}} \; \sum_{P \in \mathcal{P}, Q \in \mathcal{Q}} |Q \cap P| \cdot \frac{|Q \cap P| - 1}{|P| - 1}$$

Thus the calculation of $\overline{PS}$ can be done in $O(r^2 \cdot c^2)$ and is independent of the dataset size n. Recalculating $\overline{PS}$ until it reaches a value of 1 has therefore a complexity of $O(r^2 \cdot c^2 \cdot n)$.

4 Empirical Evaluation

We use standard R implementations for the clustering algorithms as well as the validity indices [6]. We report results for a selection of indices, such as SDbw (identified as best index in [16]) or Silhouette (identified as best index in [2]).

4.1 Artificial Datasets

We consider a set of 6 artificially generated, two-dimensional datasets first, shown in Fig. 4. In this setting, a decision about the correct clustering can be done by visual inspection. For a range of clustering algorithms and parameters the same number of $k \cdot m = 25$ runs were evaluated by the validity indices and then averaged. Among the results, the Tab. 2 reports one run that achieved the best average value. The correctness of the partition was evaluated by visual inspection and is indicated by bold face in the Table.

No structure: We start with a dataset that has no structure at all, the `disc` dataset in Fig. 4(top left). In absence of any clusters or focal points, the clusters in different runs do not have much in common. The SSC plot for this dataset is shown in Fig. 5(top left). The runs that have high consistency and high stability (top right corner) do not discover any structure (no coloured circles visible):

Table 2. Results of popular cluster validity indices for various datasets. Correct result in bold face. If multiple runs achieved the same evaluation, only one is listed.

dataset	CH	Dunn	Silhouette	DB	SDbw
disc	kmeans, 7	single, 2	kmeans, 3	kmeans. 7	kmeans, 7
triangle	**single, 3**	**single, 3**	**single, 3**	**single, 3**	**single, 3**
ring	kmeans, 7	**single, 2**	dbscan, 0.1	kmeans, 7	kmeans, 7
grid	kmeans, 4	average, 7	dbscan 0.1	kmeans, 4	kmeans, 4
block	kmeans, 7	single, 2	**dbscan 0.1**	kmeans, 7	kmeans, 7
ellipse	kmeans, 6	dbscan, 0.075	kmeans, 4	dbscan, 0.075	dbscan, 0.075

The single-linkage and dbscan runs group all data in one cluster (and exclude only a few outliers). All other runs have a very poor stability value, almost all of the data was marked as unstable. The SSC plot shows clearly that none of the algorithms did discover anything of interest – which is perfectly right for the disc dataset. In contrast, most validity measures favour kmeans ($c = 7$) – due to a monotonic behavior in c and $c = 7$ was the highest value in our experiments. Most validity indices are not prepared for this case as they compare at least two clusters.

Compact clusters: Next we consider a clear structure, the `triangle` dataset of Fig. 4(top middle). This is a simple task for many clustering algorithms, but may still cause problems. For instance, even with the correct number of clusters, k-means splits one of the three clusters from time to time (R implementation initializes prototypes with random points). The top right corner of well-performing runs in the SSC plot of Fig. 5(top middle) is populated with various runs (e.g. single linkage, $c = 3$), many of them indicating the stable discovery of 3 equal-sized clusters by means of a structure index of about 1.6. In Fig. 4(top middle) only a single object has been marked as instable by the average-linkage clustering. The validity indices also work well with this dataset, all of the indices prefer the correct number of clusters $c = 3$. But for many indices, the average evaluation of k-means is only slightly worse than that of single-linkage, but the SSC plot tells a different story: Because of the occasional cluster splitting the k-means runs receive lower consistency and stability values, it yields the correct solution less reliable.

Cluster shape: The `ring` dataset in Fig. 4(top right) has been used many times in the literature. It has been chosen because many validity indices have a bias towards compact clusters and would not accept the inner cluster and the ring as two clusters. Only the Dunn validity index selects a correct solution, while the other measures prefer k-means again ($c = 7$). The runs with high values of c, however, do not lead to stable partitions, they subdivide the outer ring arbitrarily, these clusters are not reproducible. In contrast to the validity indices, the SSC-plot in Fig. 5(top right) clearly favors the single-linkage ($c = 2$) solution as an optimal solution. (The icon is somewhat difficult to see, because several meanshift results with zero structure lie at approximately the same position). There is a clear gap to other runs in terms of stability and consistency.

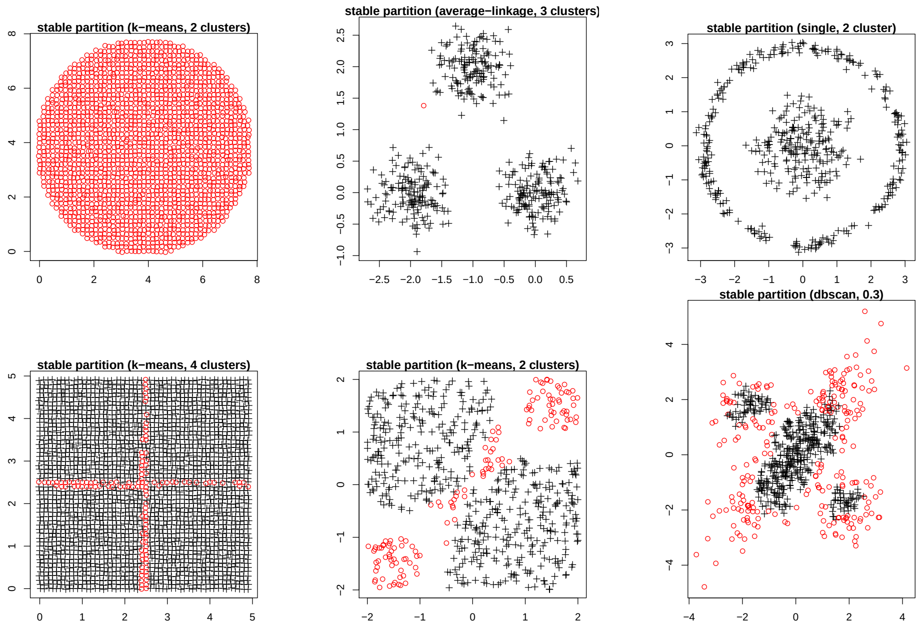

Fig. 4. Datasets from top left to bottom right: disc, triangle, ring, grid, block, ellipse. The red color refers to the stability obtained of some selected clustering algorithm.

A second dataset `ellipse` with varying shapes is shown in Fig. 4(bottom right): two small spherical and one large long-stretched cluster. The SSC plots indicates that only the meanshift algorithm manages to discover these clusters almost perfectly. Other runs in the top-right corner assign all data to a single cluster and thus discover no structure. A k-means run claims to discover more structure, but far less consistent: The visual inspection of the result shows that the long-stretched cluster is broken up into several portions of roughly equal size. Again, the results selected by the cluster validity indices in Tab. 2 do not correspond to the correct solution. Either many clusters are used to split up the long-stretched cluster, a single cluster with all data (meanshift, 0.3), or a few tiny clusters with over 90% of the dataset marked as noise (dbscan, 0.075).

Clustering tendency: The `grid` dataset of Fig. 4(bottom left) is another example for a dataset that carries no internal structure. In contrast to `disc`, most validity indices suggest $c = 4$ instead of $c = 7$ now. This is due to the corners, which were absent in the `disc` dataset. They serve as focal points and stabilize the cluster positions, while at the same time guaranteeing four equal-sized clusters (cf. stable points in Fig. 4). The SSC-plot of Fig. 5(bottom left) looks similar to the `disc` dataset: Most runs have low stability and low consistency (lower left corner) and those runs with higher stability offer no structure. The only exception is the discussed phenomenon with k-means ($c = 4$), where the corner stabilize the cluster positions, which yields a high stability value. How-

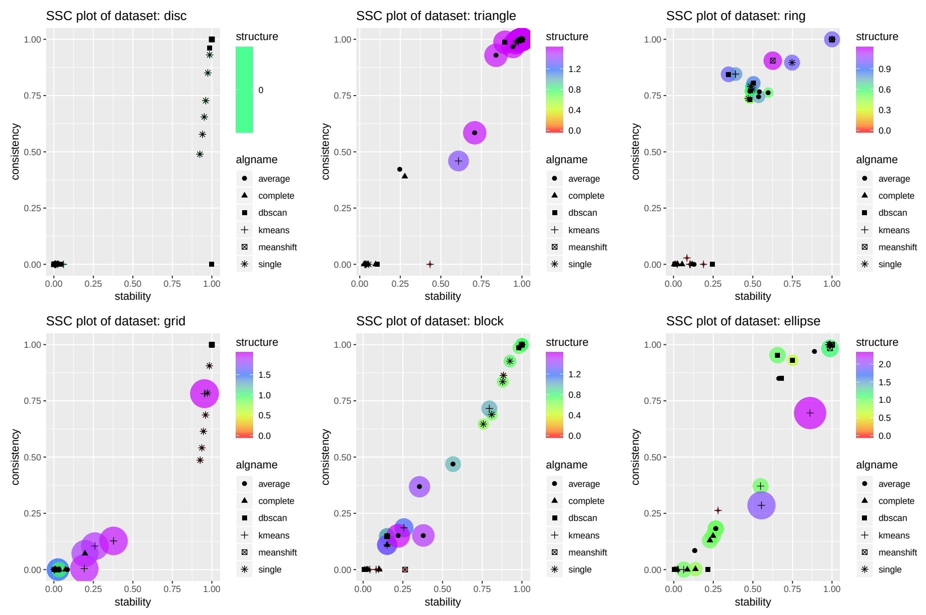

Fig. 5. SSC-plots (top left to bottom right): disc, triangle, ring, block, grid, ellipse.

ever, as we can see from the instable points (marked red) in Fig. 4(bottom left), the clusters and not separated and we achieve a quite low consistency value.

The `grid` dataset is also meant as a counterpart of the `block` dataset, where two of the four corners actually form separated clusters, so we have 3 clusters in total. The SSC plot clearly shows a few runs (dbscan, single-linkage) that deliver optimal results at an entropy close to 0.8 (because we have one very large cluster and two small clusters). On the contrary, the best k-means run suggests two equal-sized clusters that split the large cluster into two (cf. instable portion in Fig. 4(bottom middle)). This leads to a structure index close to 1 (two equal-sized clusters), but the consistency is poor (below 0.75). The SSC-plot reflects the performance again very well, whereas most validity indices favour many clusters (k-means, $c = 7$). None of the measures hints at the correct solution.

In summary, the cluster validity indices were misled by their biases, whereas the SSC-plot (as well as the selection procedure mentioned in Sect. 3.4) managed to identify the most reliable algorithm for every dataset.

4.2 Real Datasets

Next we examine the results on three real dataset from the UCI machine learning repository: `wine` (all attributes except the class), `ecoli` (selected attributes: mcg, gvh, aac, alm1 and alm2), and `pendigits` (attributes #6, #8, #10 and #12). There is no ground truth available with these datasets.

The wine dataset has three classes and it is well-known that a k-means run with 3 clusters matches the classes pretty well. In the SSC-plot in Fig. 6(left) exactly this run stands out by the highest structure index (partition entropy of ≈ 1.6, corresponding to 3 equal-sized clusters). But we also see that only 65% of the data was clustered stably and the consistency is also quite low. So it is safe to conclude that we observe a similar phenomenon as with the grid dataset: The data distribution provides focal points for three clusters, but we do not have three separated clusters. The k-means run with $c = 2$ has much better stability and consistency values.

A 5D-excerpt of the ecoli dataset is shown in Fig. 6(bottom middle). Upon visual inspection some scatterplots offer no structure (aac vs gvh), others seem to suggest two clusters (aac vs alm2), some may even hint at three clusters (gvh vs alm1). The SSC-plot supports this impression we got by visual inspection very well: we see a close-to-optimal meanshift result for two clusters, but also a runner-up with three clusters (86% stability, 92% consistency). Other algorithms reach a similar structure index, but are less stable and less consistent.

Finally we discuss results for a real dataset where the visual inspection is not conclusive: the pendigits dataset (handwritten digit recognition). Although we use only 4 attributes from the original dataset, the scatterplot matrix looks very confusing and nothing hints at the existence of separated clusters. From the SSC-plot in Fig. 6(right) we see a very good k-means result with stability and consistency close to 1 – no cluster splitting or lacking separation with k-means for this run. The precision-recall allows to diagnose the results: For instance, for two k-means runs we observe a steep drop in precision followed by an almost constant segment (curves (b)). For these runs, only a few dense points were stably discovered (TP), but most of the surrounding, less dense data was marked instable as it was assigned to alternating clusters. The k-means curve that drops earliest recovers later (curve (c)): This means that one particularly dense area was not stably clustered (FP), possibly a cluster splitting phenomenon, but for the less dense data the algorithm performed much better (more TPs). Several average-linkage curves (d) achieve good results in medium dense areas, but perform extremely poor in the areas of highest and lowest density.

4.3 Sensitivity

The consistency index depends on the parameter k in $d_k(\cdot)$. We have used $k = 16$ throughout all experiments. The size of k influences the degree of separation that is required for clusters: For $k = 0$ all neighborhoods are pure and none for $k = n$. A cluster that does not even consist of k points will not have any pure neighborhoods. The role of k is that of assuring separation at the border of a cluster, where data from the own cluster may lie on one side and data from another cluster on the opposite side. If the gap is large enough such that the k nearest objects all belong to the same cluster as x, we consider x well separated from other clusters. We have calculated consistency values for $k \in \{8, 12, 16, 20, 24\}$ over 27 datasets and obtained a lowest pairwise Pearson correlation of 0.989

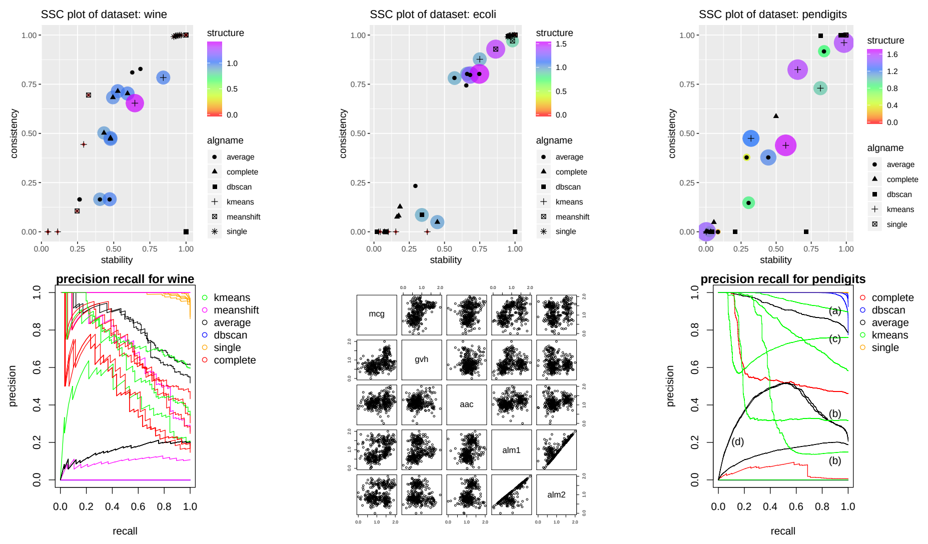

Fig. 6. Real Datasets and SSC-plot from left to right: wine, ecoli, pendigits.

($k = 12$ vs $k = 24$) and a correlation coefficient of 1 for $k = 8$ vs $k = 12$. So the results are robust and not sensitive to the exact choice of k.

5 Summary and Conclusions

We have revisited the problem of evaluating clustering algorithms from a structure discovery perspective. Recent studies list and compare 30 different validity indices to find out the best measure. But all these measures seem to have strong biases, which makes them less suited to directly compare the performance of an arbitrary selection of algorithms on an unknown dataset. This has also been confirmed by our experiments. While it is common to apply a broad range of classifiers to a new dataset to see which method may work best with the unknown data, we seem to have nothing comparable for clustering methods.

For the case of structure discovery capabilities, a collection of three indices has been proposed in this paper that convey a complete picture of the algorithms performance on this dataset: the fraction of stably clustered data, the amount of discovered structure, and the consistency of the resulting partitions with the notion of a cluster being "compact and separated". The experiments demonstrate that the method is less biased towards specific cluster shapes or notions of compactness and separation than existing methods. Addressing the stability of each object and its density separately, allows us to apply methods usually restricted to classification task in the field of unsupervised clustering. The proposal may therefore strike a new path for a systematic and direct comparison of clustering algorithms from different paradigms and with different biases – at least as far as structure discovery capabilities are concerned.

The results are reproducible. The source code, the datasets and all figures are available at `https://public.ostfalia.de/~hoeppnef/validity.html`

References

1. C. C. Aggarwal and C. K. Reddy, editors. *Data Clustering: Algorithms and Applications.* Chapman & Hall, 2013.
2. O. Arbelaitz, I. Gurrutxaga, J. Muguerza, J. M. Perez, and I. Perona. An extensive comparative study of cluster validity indices. *Pattern Recognition*, 46:243–256, 2013.
3. T. Calinski and J. Harabasz. A dendrite method for cluster analysis. *Communications in Statistics – Theory and Methods*, 3(1):1–27, 1974.
4. H. Chouikhi, M. Charrad, and N. Ghazzali. A comparison study of clustering validity indices. In *Global Summit on Comp. & Inform. Techn.*, pages 1–4, 2015.
5. D. Davies and D. Bouldin. A cluster separation measure. *Transactions on Pattern Analysis and Machine Intelligence*, 1, 1979.
6. B. Desgraupes. Clustering indices. R-package 'clusterCrit', 2017.
7. J. Dunn. Well separated clusters and optimal fuzzy partitions. *Journal Cybernetics*, 4(1):95—-104, 1974.
8. B. S. Everitt and S. Landau. *Cluster Analysis.* Wiley, 2011.
9. I. Färber, S. Günnemann, H.-P. Kriegel, P. Kröger, E. Müller, E. Schubert, T. Seidl, and A. Zimek. On using class-labels in evaluation of clustering. In *Proc. MultiClust 2010*, 2010.
10. A. Fred and A. K. Jain. Combining multiple clusterings using evidence accumulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):835–850, 2005.
11. M. Halkidi and V. M. Clustering validity assessment: Finding the optimal partitioning of a data set. In *IEEE Int. Conf. on Data Mining*, pages 187–194, 2001.
12. A. K. Jain, N. M. Murty, and P. J. Flynn. Data Clustering: A Review. *ACM Computing Surveys*, 31(3):264–323, sep 1999.
13. L. Kaufman. *Finding Groups in Data: An Introduction to Cluster Analysis.* Wiley, 2005.
14. L. I. Kuncheva and D. P. Vetrov. Evaluation of stability of k-means cluster ensembles with respect to random initialization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1798–1808, 2006.
15. T. Lange, V. Roth, M. L. Braun, and J. M. Buhmann. Stability-based validation of clustering solutions. *Neural Computation*, 16(6):1299–1323, 2004.
16. Y. Liu, Z. Li, H. Xiong, X. Gao, and J. Wu. Understanding of internal clustering validation measures. In *Proc. Int. Conf. Data Mining*, pages 911–916, 2010.
17. N. R. Pal and J. C. Bezdek. On cluster validity for the fuzzy c-means model. *IEEE Trans. Fuzzy Systems*, 3(3):379–379, 1995.
18. P. J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of comput. and applied mathematics*, 20:53–65, 1987.
19. C. Wiwie, J. Baumbach, and R. Röttger. Comparing the performance of biomedical clustering methods. *Nature Methods*, 12(11):1033–1040, 2015.