RESAMPLING FOR FUZZY CLUSTERING

CHRISTIAN BORGELT

European Center for Soft Computing
c/ Gonzalo Gutiérrez Quirós s/n, 33600 Mieres, Spain
christian.borgelt@softcomputing.es

Received (received date)
Revised (revised date)

Resampling methods are among the best approaches to determine the number of clusters in prototype-based clustering. The core idea is that with the right choice for the number of clusters basically the same cluster structures should be obtained from subsamples of the given data set, while a wrong choice should produce considerably varying cluster structures. In this paper I give an overview how such resampling approaches can be transferred to fuzzy and probabilistic clustering. I study several cluster comparison measures, which can be parameterized with \( t \)-norms, and report experiments that provide some guidance which of them may be the best choice.

Keywords: fuzzy clustering, cluster evaluation, number of clusters, resampling

1. Introduction

A core problem of most prototype-based clustering algorithms—like the classical \( c \)-means algorithm \(^1,20,27\), its fuzzy counterpart (fuzzy \( c \)-means) \(^2,4,21\), or the expectation maximization algorithm for estimating a mixture of Gaussians \(^9,12,5\)—is that they require the number of clusters to be known in advance. This is, of course, inconvenient in practice, since in applications we rarely find ourselves in such a favorable position. Rather we would like to have a method to determine the number of clusters (automatically or at least semi-automatically) from the data set.

A common approach to tackle this problem is to cluster the given data set several times, each time with a different number of clusters from a user-specified range. The clustering results are evaluated and then the number of clusters yielding the best evaluation is chosen. The selection criterion may also be a (reasonably clear) local optimum, or a clear change in the behavior of the evaluation (for example, a knee or a maximum or minimum in the first or second derivative). In fuzzy clustering, this approach is very common in connection with so-called \textit{internal cluster evaluation measures}, like, for example, the partition entropy \(^2,32\), the Fukuyama-Sugeno index \(^15\), or the Xie-Beni index \(^33\) (overviews of such measures can be found in \(^4,21,18,19,7\)). However, several of these measures are fairly unreliable and can yield inconclusive results even if the cluster structure is actually fairly clear \(^9\). In part this is due to
implicit assumptions about the cluster shapes (spherical, for example, is the most common assumption), which may not be met by the data set in question.

In this paper I study an alternative approach that has recently attracted a lot of attention in crisp and probabilistic clustering. The core idea is that if we cluster subsamples of the given data set with the “right” number of clusters, we should end up with basically the same cluster structure in each run. With a “wrong” number of clusters, however, the clustering result should be unstable, showing considerable variation between different subsamples. Thus, by measuring the stability of the clustering result w.r.t. subsampling (that is, by measuring the similarity of clustering results from different runs), one may be able to determine the “best” number of clusters: it is the one for which the clustering results are most stable.

Intuitively, one may think of this as follows: if the “true” number of clusters is $c$ and we try to find $c + 1$ clusters, (at least) one cluster has to be split. If we try to find $c - 1$ clusters, some pair of clusters has to be merged (or the data points of some cluster have to be distributed on neighboring ones—which may also be seen as a form of splitting a cluster). As it depends on the particular properties of the subsample which cluster is split or which clusters are merged, we should get somewhat differing structures in each run. By measuring how well the clustering results coincide, we can thus discover such situations and choose the number of clusters based on this information. An concrete example of how clusters may be split or merged differently on different subsamples is discussed in Section 4.

Note that, in contrast to internal cluster evaluation measures, there is no implicit assumption about the cluster shape in a resampling approach. Resampling works with whatever assumption is imposed by the used clustering algorithm, but does not make any assumption itself. As a consequence it is much more versatile and can be applied to a wider range of clustering methods, provided the resulting clustering model allows to assign new data points to a cluster (see below). As a consequence it yields, of course, only the “best” number of clusters for the given algorithm (and its implicit assumptions about the cluster structure).

In addition to a general discussion of the highly promising approach of resampling for fuzzy clustering, I study experimentally how the choice of $t$-norms in the needed relative cluster evaluation measures (to combine membership degrees and coincidence matrix entries, see below) affects the quality and clarity of the results, that is, how well the “best” number of clusters can be determined.

2. Relative Cluster Evaluation Measures

Relative cluster evaluation measures compare two partitions of given data, one being a clustering result and the other either also a clustering result or given by a classification or a user-defined grouping. In the latter case one also speaks of external cluster evaluation measures, although the methods used are usually the same. Two clustering results, however, may also be compared based on the cluster parameters alone, although I do not discuss such methods here.
I assume that a partition of the given data set is described by a $c \times n$ partition matrix $U = (u_{ij})_{1 \leq i \leq c, 1 \leq j \leq n}$, where $c$ is the number of clusters and $n$ the number of data points. An element $u_{ij}$ of such a matrix states, in the crisp case, whether the $j$-th data point belongs to the $i$-th cluster ($u_{ij} = 1$) or not ($u_{ij} = 0$). In the fuzzy case, $u_{ij}$ is the degree of membership to which the $j$-th data point belongs to the $i$-th cluster (usually satisfying the constraint $\forall j; 1 \leq j \leq n: \sum_{i=1}^{c} u_{ij} = 1$).

I also assume that the two partition matrices, which are to be compared, have the same dimensions, that is, that they refer to the same numbers of clusters and the same data points. In principle, it is also imaginable to compare matrices with different numbers of rows (clusters), although some measures may give misleading results in this case, since they are based on the assumption that it is possible to set up a bijective mapping between the clusters of the two partitions.

Regardless of whether the numbers of rows (clusters) coincide or not, we face the problem of relating the clusters of the one partition to the clusters of the other partition. There are basically three solutions to this problem:

1. For each cluster in the one partition we determine the best fitting cluster in the other and thus set up a (uni-directional) cluster mapping.
2. We find the best permutation of the rows of one partition matrix (mapping rows with the same index onto each other), that is, we find the best one-to-one mapping of the clusters from the two models.
3. We compare the partition matrices indirectly by first setting up a coincidence matrix for each of them, which records for each pair of data points whether they are assigned to the same cluster or not, and then compare these matrices.

The first alternative has the advantage of being fairly efficient (time complexity $O(nc^2)$), but the severe disadvantage that we cannot make sure that we obtain a one-to-one relationship between the clusters. Some clusters in the second partition may not be paired with any cluster in the first, which also renders the approach asymmetric. The second alternative has the advantage that it definitely finds the best one-to-one relationship, while its disadvantage is the slightly higher computational cost (time complexity $O(nc^2 + c^3)$, see below). The third alternative has the disadvantages that it does not yield a direct indication of how to relate the clusters to each other and that it can have fairly high computational costs (time complexity $O(n^2c)$), especially for a large number of data points. However, the fact that it does not need an explicit mapping between the clusters can also be seen as an advantage, because it renders this method very flexible. In particular, this method is well suited to compare partitions with different numbers of clusters.

2.1. Comparing Partition Matrices

The first two approaches outlined above directly compare two $c \times n$ partition matrices $U^{(1)}$ and $U^{(2)}$. For both of them we need a measure that compares two rows, one from each matrix. Such measures can be derived from measures comparing binary
classifications, like, for example, the accuracy or the $F_1$-measure $^{30}$.
Formally, we set up a $2 \times 2$ contingency table for each pair of rows, one from each matrix (cf. Table 1). That is, for each pair $(i, k) \in \{1, \ldots, c\}^2$ of rows we compute

$$
\begin{align*}
n_{11}^{(i,k)}(U^{(1)}, U^{(2)}) &= \sum_{j=1}^{n} u_{ij}^{(1)} \cdot u_{kj}^{(2)}, \\
n_{10}^{(i,k)}(U^{(1)}, U^{(2)}) &= \sum_{j=1}^{n} \left(1 - u_{ij}^{(1)}\right) \cdot u_{kj}^{(2)}, \\
n_{01}^{(i,k)}(U^{(1)}, U^{(2)}) &= \sum_{j=1}^{n} u_{ij}^{(1)} \cdot \left(1 - u_{kj}^{(2)}\right), \\
n_{00}^{(i,k)}(U^{(1)}, U^{(2)}) &= \sum_{j=1}^{n} \left(1 - u_{ij}^{(1)}\right) \cdot \left(1 - u_{kj}^{(2)}\right).
\end{align*}
$$

(In the following I generally drop the arguments $U^{(1)}$ and $U^{(2)}$ to make the formulas easier to read.) These numbers may also be computed from fuzzy membership degrees, where they have a fairly natural interpretation: in the crisp case, $n_{11}$ is the number of data points that are assigned to the $i$-th cluster of the first partition and to the $k$-th cluster of the second partition, where the and is formally expressed by a product. Allowing membership degrees from $[0, 1]$ and drawing on the theory of fuzzy logic, we see that this is only a special case of a $t$-norm that combines the two statements. Hence, in the general case, we may replace the product by an arbitrary $t$-norm. Analogously, the expressions $1 - u_{ij}$ can be seen as resulting from an application of the standard fuzzy negation, and indeed: they refer to negated statements “The $j$-th data point does not belong to the $i$-th cluster.” In this way we achieve a straightforward generalization of all following measures to fuzzy clustering results.

From the numbers $n_{ab}^{(i,k)}$, $a, b \in \{0, 1\}$, we may now compute any measure for evaluating a binary classification, summing over the rows, and maximizing the result over all row permutations.\textsuperscript{a} An example is the (averaged) $F_1$ measure $^{30}$

$$
F_1(U^{(1)}, U^{(2)}) = \max_{c \in \Pi(c)} \frac{1}{c} \sum_{i=1}^{c} 2 \frac{\pi_{i,c(i)} \rho_{i,c(i)}^{\alpha(c)}}{\pi_{i,c(i)} + \rho_{i,c(i)}^{\alpha(c)}},
$$

\textsuperscript{a}Note that with the so-called Hungarian method for solving optimum weighted bipartite matching problems $^{28}$ the time complexity of finding the maximum over all permutations for given pairwise column comparison values is $O(c^3)$ and not $O(c^2)$.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|c|c|c|}
\hline
 & $u_{kj}^{(1)} = 1$ & $u_{kj}^{(2)} = 0$ & $\Sigma$ \\
\hline
$u_{ij}^{(1)} = 1$ & $n_{11}^{(i,k)}$ & $n_{10}^{(i,k)}$ & $n_{11}^{(i,k)}$ \\
$u_{ij}^{(1)} = 0$ & $n_{01}^{(i,k)}$ & $n_{00}^{(i,k)}$ & $n_{00}^{(i,k)}$ \\
\hline
$\Sigma$ & $n_{11}^{(i,k)}$ & $n_{10}^{(i,k)}$ & $n$ \\
\hline
\end{tabular}
\caption{Contingency table comparing rows of two (crisp) partition matrices ($i$ and $k$ are the cluster indices).}
\end{table}
where \( \Pi(c) \) is the set of all permutations of the \( c \) numbers 1, \ldots, \( c \), which describes all possible row to row mappings. Cluster-specific precision and recall are

\[
\pi_{i,k} = \frac{n_{11}^{(i,k)}}{n_{01}^{(i,k)} + n_{11}^{(i,k)}} \quad \text{and} \quad \rho_{i,k} = \frac{n_{11}^{(i,k)}}{n_{10}^{(i,k)} + n_{11}^{(i,k)}}.
\]

Another example is \textit{(cross-classification) accuracy}, averaged over all columns:

\[
Q_{acc}(U^{(1)}, U^{(2)}) = \max_{\varsigma \in \Pi(c)} \frac{1}{cn} \sum_{i=1}^{c} \left( \frac{n_{01}^{(i,\varsigma(i))} + n_{11}^{(i,\varsigma(i))}}{n_{11}^{(i,k)}} \right).
\]

Two partition matrices \( U^{(1)} \) and \( U^{(2)} \) are the more similar, the higher the values of the (averaged) \( F_1 \) measure or the (cross-classification) accuracy.

An alternative to these classification-based measures is a simple mean squared difference comparison of the partition matrices (which, at least to my knowledge, has not been used in clustering up to now). That is, we compute

\[
Q_{diff}(U^{(1)}, U^{(2)}) = \min_{\varsigma \in \Pi(c)} \frac{1}{cn} \sum_{i=1}^{c} \sum_{j=1}^{n} \left( u_{ij}^{(1)} - u_{\varsigma(i)j}^{(2)} \right)^2.
\]

The smaller this measure, the more similar are the partitions. Note that for crisp clustering (that is, for \( u_{ij} \in \{0, 1\} \)) this measure may also be written as

\[
Q_{diff}(U^{(1)}, U^{(2)}) = \min_{\varsigma \in \Pi(c)} \frac{1}{cn} \sum_{i=1}^{c} \left( n_{01}^{(i,k)} + n_{10}^{(i,k)} \right).
\]

Hence it is closely related to the (cross-classification) accuracy, since it is obviously \( n_{01}^{(i,k)} + n_{10}^{(i,k)} = n - (n_{00}^{(i,k)} + n_{11}^{(i,k)}) \) for crisp partitions. This measure is actually the most natural for fuzzy clustering and thus it is not surprising that, as we will see in Section 4, it performs very well for fuzzy clustering.

The idea underlying the measure \( Q_{diff} \) actually suggests to generally investigate measures for comparing fuzzy sets, since each fuzzy cluster can be seen as a fuzzy set in the data space, the set of all fuzzy clusters as a fuzzy partition of the data space.\(^b\) Examples include equality measures for fuzzy sets, which may be based, for instance, on other aggregations of the (absolute) differences of the membership degrees. In addition, such measures may also make it possible to compare the (fuzzy) clusters as they are defined by their parameters, rather than relying on the partition matrices, which evaluate the cluster membership functions only at the data points.

\(^b\) I am grateful to an anonymous reviewer for pointing out this possibility.

2.2. Comparing Coincidence Matrices

As an alternative to comparing partition matrices directly, one may first compute from each of them an \( n \times n \) coincidence matrix, also called a \textit{cluster connectivity matrix} \(^2\)\(^6\), which states for each pair of data points whether they are assigned to
the same cluster or not. Formally, a coincidence matrix $\Psi = (\psi_{jl})_{1 \leq j, l \leq n}$ can be computed from a partition matrix $U = (u_{ij})_{1 \leq i \leq c, 1 \leq j \leq n}$ by

$$\psi_{jl} = \sum_{i=1}^{c} u_{ij} u_{il}.$$ 

Note again that these values may also be computed from fuzzy (or probabilistic) membership degrees, possibly replacing the product (which again represents a conjunction of membership statements) by another t-norm.

After coincidence matrices $\Psi^{(1)}$ and $\Psi^{(2)}$ are computed from the two partition matrices $U^{(1)}$ and $U^{(2)}$, the comparison is carried out by computing statistics of the number of data point pairs that are in the same group in both partitions, in the same group in one, but in different groups in the other, or in different groups in both.

The main advantage of this approach is, of course, that we are freed of the need to pair the groups of the two partitions and thus need not consider permutations. We rather exploit that data points that are considered (dis)similar by one partition should also be considered (dis)similar by the other.

Formally, we compute a $2 \times 2$ contingency table (cf. Table 2) containing the numbers (which are basically counts of the different pairs $(\psi_{jl}^{(1)}, \psi_{jl}^{(2)})$)

<table>
<thead>
<tr>
<th>\psi_{jl}^{(1)}</th>
<th>\psi_{jl}^{(2)}</th>
<th>\Sigma</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\psi_{jl}^{(1)} = 1$</td>
<td>$\psi_{jl}^{(2)} = 1$</td>
<td>$N_{SS}$</td>
</tr>
<tr>
<td>$\psi_{jl}^{(1)} = 1$</td>
<td>$\psi_{jl}^{(2)} = 0$</td>
<td>$N_{SD}$</td>
</tr>
<tr>
<td>$\psi_{jl}^{(1)} = 0$</td>
<td>$\psi_{jl}^{(2)} = 1$</td>
<td>$N_{DS}$</td>
</tr>
<tr>
<td>$\psi_{jl}^{(1)} = 0$</td>
<td>$\psi_{jl}^{(2)} = 0$</td>
<td>$N_{DD}$</td>
</tr>
<tr>
<td>$\Sigma$</td>
<td>$N_{S}$</td>
<td>$N_{D}$</td>
</tr>
</tbody>
</table>

where the index $S$ stands for “same group” and the index $D$ stands for “different groups” and the two indices refer to the two partitions. (To make the formulas easier to read, the arguments $\Psi^{(1)}$ and $\Psi^{(2)}$ are dropped in the following.) Again the product may be replaced by any t-norm (note that $\psi_{jl} \in [0, 1]$, since fuzzy clustering satisfies $\forall j; 1 \leq j \leq n : \sum_{i=1}^{c} u_{ij} = 1$).
From these numbers a large variety of measures may be computed, including the Rand statistic or Rand index \(^{29}\)

\[
Q_{\text{Rand}}(\Psi^{(1)}, \Psi^{(2)}) = \frac{N_{SS} + N_{DD}}{N},
\]

which is a simple ratio of the number of data point pairs treated the same in both partitions to all data point pairs, and the Jaccard coefficient or Jaccard index \(^{23}\)

\[
Q_{\text{Jaccard}}(\Psi^{(1)}, \Psi^{(2)}) = \frac{N_{SS}}{N_{SS} + N_{SD} + N_{DS}}.
\]

which ignores negative information, that is, pairs that are assigned to different groups in both partitions. Both measures are to be maximized. Another frequently encountered measure is the Fowlkes–Mallows index \(^{14}\)

\[
Q_{\text{FM}}(\Psi^{(1)}, \Psi^{(2)}) = \frac{N_{SS}}{\sqrt{(N_{SS} + N_{SD})(N_{SS} + N_{DS})}},
\]

which can be interpreted as a cosine similarity measure, because it computes the cosine between two binary vectors, each of which contains all elements of one of the two coincidence matrices \(\Psi^{(1)}\) and \(\Psi^{(2)}\). Consequently, this measure is also to be maximized. A final example is the Hubert index \(^{22}\) (also known as the Hubert-Arabie index or the adapted Rand statistic/index)

\[
Q_{\text{Hubert}}(\Psi^{(1)}, \Psi^{(2)}) = \frac{N \cdot N_{SS} - N_{S} \cdot N_{S}}{\sqrt{N_{S} \cdot N_{S} \cdot N_{D} \cdot N_{D}}},
\]

where

\[
N_{S} = N_{SS} + N_{SD}, \quad N_{D} = N_{DS} + N_{DD},
\]

\[
N_{S} = N_{SS} + N_{DS}, \quad N_{D} = N_{SD} + N_{DD},
\]

\[
N_{..} = N_{SS} + N_{SD} + N_{DS} + N_{DD}
\]

The Hubert index may either be interpreted as a product-moment correlation, computed from the set of pairs \((\psi_{jl}^{(1)}, \psi_{jl}^{(2)})\), \(1 \leq j,l \leq n\), which are seen as points in a two-dimensional space. Alternatively, it may be interpreted as the square root of the (normalized) \(\chi^2\) measure, as it can be computed from the 2 \(\times\) 2 contingency table shown in Table 2.\(^{c}\) Hence this measure is also to be maximized.

It should be clear that this list does not exhaust all possibilities. Basically all of the abundance of measures, by which (binary) vectors and matrices can be compared, are applicable. In this paper, however, I confine myself to those listed above, which are most frequently used for (crisp) clustering.

\(^c\)The \(\chi^2\) measure can be seen as measuring the strength of dependence between two random variables, one for each partition, which indicate for each data point pair whether the data points are in the same group (cluster) or not.
3. Resampling

Resampling can be seen as a special Monte Carlo method, that is, as a method for finding solutions to mathematical and statistical problems by simulation. It has been applied to cluster estimation problems already fairly early and it seems to have gained increased attention in this domain recently. Its main purpose in clustering is the validation of clustering results as well as the selection of an appropriate cluster model—in particular the choice of an appropriate number of clusters—by estimating the variability (or, equivalently, the stability) of the result.

Resampling methods can be found with basically two sampling strategies. In the first place, one may use subsampling, that is, the samples are drawn without replacement from the given data set, so that each data point appears in at most one data subset. This strategy is usually applied in a cross validation style, that is, the given data set is split into a certain number of disjoint subsets (with two subsets being the most common choice). The alternative is bootstrapping, in which samples are drawn with replacement, so that a data point may even appear multiple times in the same data subset. There are good arguments in favor and against both approaches, but the results often do not differ much.

The general idea of applying resampling for cluster validation and model selection was already outlined in the introduction: a cluster model can usually be applied as a classifier with as many classes as there are clusters (i.e. one class per cluster). In this way data points that have not been used to build the cluster model can be assigned to clusters (or the corresponding classes). Thus we can obtain, with the same algorithm, two different groupings of the same set of data points. For example, one grouping may be obtained by clustering the data set, the other by applying a cluster model that was built on another data set. Regardless of their origin, these two groupings can then be compared using, for example, one of the relative evaluation measures discussed in the preceding section. By repeating such comparisons with several samples drawn from the original data set, one can obtain an assessment of the variability of the cluster structure (or, more precisely, an assessment of the variability of the evaluation measure for the similarity of partitions).

Specific algorithms following this general scheme have been proposed in. The approaches in and are basically identical. Both are based on a bootstrapping approach and work as follows: first the full given data set is clustered with the chosen algorithm. Formally, this may be seen as an estimate of the “average” partition. Then a user-defined number of random samples of user-defined size are drawn (with replacement) from the data set and clustered as well. The cluster models obtained from the samples are applied to the full data set, thus obtaining two groupings of this data set. These two groupings are compared by one of the relative evaluation measures based on coincidence matrices that were discussed in Section 2. Finally, the average of the evaluation measure values for each of these comparisons is taken as an assessment of the cluster variability. As an alternative, it has been suggested that one may do without an estimate for the “average”
partition (which is estimated by the cluster model obtained from the full data set) and rather assess the variability of the cluster structures by comparing all pairs of cluster models obtained from the samples on the full data set.

This resampling approach may be applied to select the most appropriate cluster model, in particular, the “best” number of clusters. To do so, the above algorithm is executed for different parameterizations of the clustering algorithm and then the cluster model/parameterization exhibiting the lowest variability is selected. Experimental results reported in the literature indicate that this approach is very robust and a fairly reliable way of choosing the number of crisp clusters.

In contrast to the bootstrapping approaches, one may rely on a (repeated) two-fold cross validation sampling scheme. In each step the given data set is split randomly into two parts of about equal size. Both parts are processed with the same clustering algorithm and the cluster model obtained on the second half of the data is applied to the first half. Thus one obtains two groupings for the first half of the data, which are compared with a risk-based evaluation measure. This (relative) measure is defined on the two partition matrices and thus has to find the best matching of the clusters of the two groupings (see above). However, in principle all relative measures discussed in the preceding section (including those based on coincidence matrices) may be applied (just as measures based on partition matrices may be applied in the bootstrapping approaches). Experimental results on several data sets have been reported in the literature, which show that the number of Gaussian distribution clusters can thus be selected in a fairly reliable way.

When applying these resampling methods it should be noted that all approaches in this direction only assess the variability in the results obtained with some clustering algorithm. Although a low variability is surely a desirable property, it is not sufficient to guarantee a good clustering result. For example, a clustering algorithm that always yields the same partition of the data space, regardless of the data it is provided with, has no variability at all, but surely yields unsatisfactory clustering results. Hence the clustering algorithms that are compared with such schemes should not differ too much in their flexibility, because otherwise the simpler and thus more stable algorithm may be judged superior without actually being.

Furthermore, the power of many such statistical tests, like the estimation of the variability of the clustering structure as it was discussed above, decreases quickly with increasing data dimensionality. This is not surprising, because due to what is usually called the curse of dimensionality, the data space necessarily is less and less densely populated, the more dimensions there are. In addition, the noise in the different dimensions tends to sum, which in combination with the tendency of larger average distances between the data points, makes it more and more difficult for a clustering algorithm to find reasonable groups in the data. This, of course, must lead to a higher variability in the clustering result. For low-dimensional data sets, however, resampling is a very powerful technique and seems to be among the best available approaches to determine the number of clusters.
Fig. 1. Artificial data sets with 3 (equally populated), 4 (differently populated), 6 (equally populated) spherical clusters.

Fig. 2. An artificial data set with 5 (equally populated) ellipsoidal clusters and a view of the wine data set (attributes 7, 10, and 13).

4. Experiments

I carried out several experiments by applying a resampling approach for fuzzy clustering based on the above explanations to five data sets. The first three are artificial two-dimensional data sets of 400 data points each with three, four, and six clusters, respectively. They are shown in Figure 1. They were generated by sampling from normal distributions (variance 1), located at (0, 0), (4, 0), and (2, 3) for the first data set (equal cluster probabilities), at (0, 0), (4, 0), (0, 4), and (4, 4) for the second data set (different cluster probabilities), and at (0, 0), (2, −3), (6, −3), (8, 0), (6, 3), and (2, 3) for the third data set (equal cluster probabilities). The fourth data set is an artificial three-dimensional data set of 400 data points with five equally populated, but ellipsoidal clusters. It is shown on the left in Figure 2. The last data set is the well-known wine data set from the UCI machine learning repository \(^6\), a view of which is shown on the right in Figure 2. It comprises three classes of Italian wines and thus one can expect to find three clusters. I used attributes 7, 10, and 13, which are the most informative w.r.t. the class.
In order to illustrate the rationale of resampling for finding the number of clusters, consider Figures 3 to 5. In Figure 3 two subsamples of the first artificial data set (three equally populated clusters), which are complementary halves of this data set, are clustered with two clusters (standard fuzzy $c$-means algorithm). The results are completely different. On the left, the top and the right cluster are nicely separated, but the cluster on the left is split, because there is no third cluster left to cover it. A full “merging” of clusters (although one may argue that the left cluster is almost merged with the top one as the majority of its data points are assigned—at least to a higher degree—to this cluster) does not occur due to the fact that the fuzzy $c$-means algorithm finds only spherical clusters.

Fig. 3. Two subsamples (complementary halves) of the first artificial data set clustered with two clusters (standard fuzzy $c$-means algorithm). Note how the data is split in a completely different way: the cluster boundaries are almost perpendicular to each other.

Fig. 4. Two subsamples (complementary halves) of the first artificial data set clustered with three clusters (standard fuzzy $c$-means algorithm). Note that the clusters are basically the same: the locations of the cluster boundaries differ only slightly for the subsamples.

Fig. 5. Two subsamples (complementary halves) of the first artificial data set clustered with four clusters (standard fuzzy $c$-means algorithm). Note how the (superfluous) fourth cluster is squeezed between the other three clusters, which cover the classes fairly well, in two different ways.
On the right of Figure 3, the situation is mirrored: the top and left cluster are nicely separated, but the cluster on the right is split, with the majority of the data points assigned (to a higher degree) to the top cluster. Again a perfect merging of two clusters is impossible due to the restriction to spherical clusters.

In Figure 5 the same two subsamples of the first artificial data set are clustered with four clusters. Again the cluster structure is considerably different for the two subsamples. While on the left the cluster on the lower left is fairly well recognized, the other two clusters (top and right bottom) lose a large number of data points to the fourth (superfluous) cluster in the middle. On the right, we have an almost prototypical “split” of a cluster, since the fourth (superfluous) cluster in the middle covers mainly data points from the cluster on the lower left, while the other two clusters (top and right bottom) are much less affected.

On the other hand, if this data set is clustered with the “right” number of clusters (that is, the number of clusters that matches the model used to generate the data, which is three for this data set), the clustering results are almost identical, see Figure 4. For other splits into subsamples a similar behavior can be observed: clustering with three clusters proves to be most stable, while clustering with two and four clusters lead to considerably varying results.

For the actual experiments all datasets were normalized in all dimensions to mean 0 and standard deviation 1 to rule out scaling effects. The experiments were carried out with the following resampling scheme: first the whole data set was clustered. Then 100 random samples (without replacement) were drawn from the data set, each of which comprised about half of the data points. (The data set was split into two equal parts, one of which was used). Each sample was clustered with the same number of clusters as the full data set and then the two cluster structures (one obtained from the full data set and one from the sample) were compared on the full data set using the measures described in Section 2. The evaluation results were averaged over the 100 samples, thus yielding a stability measure. I tried 2 to 8 clusters for all data sets, but the results are similar if one increases the upper bound to 10 or 12. (For the small number of data points considered in these test cases, higher numbers of clusters do not appear to be useful choices.)

Note that this scheme, which has the advantage that the whole data set is used for the evaluation, could be used here due to the fairly small size of the data sets. For larger data sets (tens of thousands of data points) it may be impossible to combine this scheme with a measure that is based on coincidence matrices due to the prohibitive size of these matrices. However, this does not render resampling approaches based on coincidence matrix measures infeasible. As described in Section 3, one may just as well compare clustering results on two subsamples, thus dealing with possible storage problems by choosing an appropriate size for the subsamples. In order to compensate for the smaller size of the data sets, which renders the individual results less reliable, one may have to increase the number of sampling runs, though.

In the relative cluster evaluation measures, as they were described in Section 2, I used four different $t$-norms to combine membership degrees and the coincidence
matrix entries (see Figure 6 for illustrations):

\[
\tau_{\text{prod}}(a, b) = a \cdot b,
\]

\[
\tau_{\text{min}}(a, b) = \min\{a, b\},
\]

\[
\tau_{\text{minnp}}(a, b) = \begin{cases} 
\min\{a, b\}, & \text{if } a + b \geq 1, \\
0, & \text{otherwise,}
\end{cases}
\]

\[
\tau_{\text{Luka}}(a, b) = \max\{0, a + b - 1\},
\]

where \(\tau_{\text{minnp}}\) is the so-called nil-potent minimum.

The rationale underlying this choice is as follows: the product has to be considered, since it represents the standard way in which the formulas are written (even though it is just a convenient choice for the crisp case, where it does not have any real meaning). However, it is also a common and meaningful choice in fuzzy theory. The minimum, as the “standard” \(t\)-norm, is obligatory, since it is the largest (and thus “most possibilistic”) \(t\)-norm and the only one that is idempotent. The nil-potent minimum and the Lukasiewicz \(t\)-norm are fairly similar to the (normal) minimum and the product, respectively, but cancel the influence of pairs of degrees of membership, both of which are low, or one of which is very low. This appears to be plausible for evaluating clustering results, since intuitively one would like to focus on strong assignments and neglect weak ones.

In the measures for comparing partition matrices I tried all \(t\)-norms. Since there are two places where a \(t\)-norm is needed in the measures based on comparing coincidence matrices, I tried all pairs of \(t\)-norms to explore their interactions. As it turns out, they cannot be combined freely: some combinations do not work well. On the other hand, some measures are fairly insensitive to the choice of the \(t\)-norm.

Since it is not possible to show all individual results in this paper (there are simply too many different experiments), I try to give an impression of the performance of the different measures (in combination with different choices of \(t\)-norms) by providing a rough overview and reporting some individual results. The overview is shown in Tables 3 and 4 and uses grades to assess the performance of the different measures, with the following meanings:
Table 3. Results of comparing partition matrices with different measures and $t$-norms on the different data sets. Fuzzy $c$-means clustering was used for the first four rows, Gustafson–Kessel clustering for the last two.

<table>
<thead>
<tr>
<th>data</th>
<th>diff</th>
<th>accuracy</th>
<th>$F_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>prd</td>
<td>min</td>
<td>mnp</td>
</tr>
<tr>
<td>art. 3</td>
<td>4</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>art. 4</td>
<td>3</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>art. 6</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>wine</td>
<td>4</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>art. 5</td>
<td>4</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>wine</td>
<td>5</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>

6: clear global optimum\(^d\) at the correct cluster number, no local optimum at any other cluster number
5: clear global optimum at the correct cluster number, but there is a (weak) local optimum\(^e\) at another cluster number
4: only weak global optimum at the correct cluster number, or a competing local optimum at another cluster number
3: clear local optimum at the correct cluster number, but global optimum is at another cluster number
2: only weak local optimum at the correct cluster number, or global optimum is significantly higher than local optimum
1: only a discernable step at the correct cluster number, but not even a weak local optimum
0: no discernable characteristics at the correct cluster number

With grades 6 and 5, maybe also 4, the measure is usable for fully automatic selection, with grades 4, 3 and 2 for semi-automatic processing (with user interaction). With grades 1 and 0 a measure fails to find the correct cluster number.

These result tables show that one has to be very careful when choosing the measure and the $t$-norm(s), since a lot of combinations fail miserably. However, there are also a lot of combinations that work very nicely. Especially the Hubert index, which appears to be fairly robust w.r.t. the choice of the $t$-norms yields excellent results if either the Łukasiewicz $t$-norm or the nil-potent minimum are chosen to combine the membership degrees. (The $t$-norm used to combine the membership degrees is stated in the second header row.) This behavior is almost independent of the $t$-norm that is used to combine the coincidence matrix entries (first header row), even though the nil-potent minimum and the product appear to be slightly better than the other two measures. All other coincidence matrix based measures seem to have problems with the wine data set (see below for a possible explanation).

\(^d\)A clear global or local optimum is at least 5% higher (or lower, depending on the measure) than the evaluations for the neighboring numbers of clusters (one cluster more, one cluster less).

\(^e\)A local optimum is called weak if it is less than 1.5 times the global optimum. Otherwise it is called a competing local optimum, since it is similarly expressed as the global optimum.
Table 4. Results of comparing coincidence matrices with different measures and $t$-norms on the different data sets. In each table the upper header row shows the $t$-norm for combining coincidence matrix entries, the lower header row the $t$-norm for combining membership degrees. Fuzzy $c$-means was used for the first four rows, Gustafson–Kessel clustering for the last two rows of each table.

<table>
<thead>
<tr>
<th></th>
<th>Rand</th>
<th>min</th>
<th>minnp</th>
<th>prod</th>
<th>Luka</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>prd</td>
<td>min</td>
<td>mnp</td>
<td>prd</td>
<td>min</td>
</tr>
<tr>
<td>art. 3</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>art. 4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>art. 6</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>wine</td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>art. 5</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>wine</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Jaccard</th>
<th>min</th>
<th>minnp</th>
<th>prod</th>
<th>Luka</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>prd</td>
<td>min</td>
<td>mnp</td>
<td>prd</td>
<td>min</td>
</tr>
<tr>
<td>art. 3</td>
<td>4</td>
<td>0</td>
<td>5</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>art. 4</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>art. 6</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>wine</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>art. 5</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>wine</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Fowlkes</th>
<th>min</th>
<th>minnp</th>
<th>prod</th>
<th>Luka</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>prd</td>
<td>min</td>
<td>mnp</td>
<td>prd</td>
<td>min</td>
</tr>
<tr>
<td>art. 3</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>art. 4</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>art. 6</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>wine</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>art. 5</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>wine</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Hubert</th>
<th>min</th>
<th>minnp</th>
<th>prod</th>
<th>Luka</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>prd</td>
<td>min</td>
<td>mnp</td>
<td>prd</td>
<td>min</td>
</tr>
<tr>
<td>art. 3</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>art. 4</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>art. 6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>wine</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>art. 5</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>wine</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4. Results of comparing coincidence matrices with different measures and $t$-norms on the different data sets. In each table the upper header row shows the $t$-norm for combining coincidence matrix entries, the lower header row the $t$-norm for combining membership degrees. Fuzzy $c$-means was used for the first four rows, Gustafson–Kessel clustering for the last two rows of each table.

Among the partition matrix based measures the simple mean squared difference comparison performs fairly reliably, followed by the accuracy computed with the minimum as the $t$-norm. However, none of these measures quite reaches the performance of the properly parameterized Hubert index. Therefore the Hubert index seems to be the best choice if the higher computational complexity of a coincidence matrix based measure is acceptable (see above).
To give an impression of individual results, Tables 5 to 8 show detailed tables for two artificial data sets and the wine data set. The results in Tables 6 and 8 are based on Gustafson–Kessel clustering, the rest on fuzzy c-means clustering. The used t-norms are indicated in the table captions. For each column the global and, if it exists, a relevant local optimum are highlighted.

The results on the wine data set (Table 7) indicate that maybe five clusters are an alternative to the number of classes (three) for fuzzy c-means clustering (spherical clusters). However, this may also be explained by ellipsoidal cluster shapes, where one (ellipsoidal) clusters may cover the same data points for which two spherical clusters are needed. The results shown in Table 8 (where Gustafson–Kessel clustering was used, which makes it possible to find ellipsoidal clusters) actually make this likely, as here no local optima can be observed for five clusters. Even stronger evidence is provided by Figure 7, which shows the results of clustering the wine data set with three clusters using Gustafsson–Kessel clustering and with five clusters using standard fuzzy c-means clustering. This figure indicates fairly clearly how two spherical clusters (right) may be needed to cover the data points that are covered by one ellipsoidal cluster (left), even though the third dimension of the data set is not shown (other projections show a similar structure).

This observation also illustrates that resampling in itself does not make any assumptions about the cluster shapes, but simply uses the assumptions imposed by the used clustering algorithm. Whatever cluster number and cluster structure is compatible with the (implicit) assumptions of the clustering algorithm used and yields the most stable result gets selected.
5. Conclusions

In this paper I transferred resampling ideas that have been used in classical crisp clustering to fuzzy clustering and introduced the mean squared difference as a simple, but effective measure for comparing fuzzy and probabilistic partition matrices. In addition, I explored the influence of different t-norms, which can be used to combine membership degrees and coincidence matrix entries. As the experiments show, the resampling approach is applicable to fuzzy clustering, but one has to be careful which relative cluster evaluation measure to choose and how to parameterize it: not all measures that work with crisp clustering also work with fuzzy clustering. The best results are obtained with the Hubert index, parameterized with either the nil-potent minimum or the Lukasiewicz t-norm to combine the membership degrees,
and with the nil-potent minimum or the product to combine the coincidence matrix entries. A close competitor, which has the advantage of being simple, straightforward and computable in less time, is a direct comparison of the partition matrices based on the mean squared difference of the membership degrees.

6. Software

The software used for the experiments, which includes a program to compare two clustering results with all of the discussed measures on a given data set, is written in C and available free of charge under the GNU Lesser (Library) General License at

http://www.borgelt.net/cluster.html

References

   http://www.ics.uci.edu/~mlearn/MLRepository.html


