Clusterval: Python package for determining the number of clusters in Longitudinal Datasets

Nuno Miguel Canhoto da Silva
nuno.da.silva@tecnico.ulisboa.pt
Instituto Superior Técnico, Lisbon, Portugal
October 2021

ABSTRACT
Clustering is concerned with finding patterns in a given data. Ideally, clustering algorithms output the correct clusters of elements, although not always that is achieved. The clustering structure chosen by the algorithm should then be evaluated, and a good criterion to follow when doing cluster validation is that objects in the same partition should be close to each other. On the other hand, different clusters are noticeably distant in respect to each other. The knowledge that can be obtained from the data patterns has the potential to be useful in virtually any area. In this work we focus on a tool created in 2019, called AliClu, that combines alignment of medical sequences with clustering to analyse longitudinal data. Our goal is to enhance the clustering validity aspect of AliClu through the use of evaluation metrics. Furthermore, we will work on automating the generation of the patients sequences. Finally, we make a contribution to the clustering community and broaden interest as well, by making the clustering process easily available, in the form of a Python library which we call *clusterval*. The work we present follows two main criteria for clustering validation (external and internal), which will be presented with respective background. Following, the *clusterval* package is introduced and, its functionality explained. In the same chapter we explain the automation behind the clusters visual representation. Additionally, an extensive experiment is made with the described indices, making use of synthetic and real-world datasets.

Author Keywords
clusterval, clustering, indices, cluster validation

INTRODUCTION

The project started in [29], that presents a method that combines Temporal Needleman-Wunsch [37] and agglomerative clustering [20] to find groups in longitudinal datasets, a tool called AliClu. Although quality work was made it should be possible to build a stronger and more reliable algorithm, with the goal of achieving the best results possible and a way that can be done is by developing the clustering validation aspect of the work.

The idea behind clustering algorithms is to try find the best partition of the data so that in each of these data groups we get very similar data points, in terms of its attributes and features. There is a variety of clustering algorithms currently available, each with input parameters that we can vary in order to obtain the best structure. As a consequence of experimenting with these variations, we are likely going to have to consider more than one possible partition. In case we want to choose only one of the possible clustering results, we have to know which one represents better our problem context. At this step of the clustering process is where clustering validation measures can be used, and this is the subject of our work.

In the literature, numerous clustering validity indices exist that we can use to evaluate the results of a clustering algorithm. Fundamentally, all metrics have the same goal (compact and well separated clusters). However, some indices might be better suited for specific clustering algorithms, others can be very computationally expensive and slow down the algorithm and some work well with a particular type of data. The more knowledge we have on the many ways that exist to analyze the resulting clusters the more efficient will be the work done and might lead us to achieving more accurate insights.

However, most of the clustering indices are not easily accessible for usage, which would be of immense value to the potential users, since many times they have to write these indices from scratch. Most libraries for clustering contain some indices to evaluate clustering, but in some cases having more insights could make a difference. Therefore, with this work, we expect to also address the lack of readily available clustering validity measures in the scientific community.

With the goal of providing users with numerous metrics to evaluate generated clusterings and achieve the best possible results, we propose constructing a Python library with clustering validity indices that should be readily available, easy to use and reliable. The code can be seen in https://github.com/Nuno09/clusterval. Additionally, we want to contribute to the AliClu tool by adding one more step for evaluation of the results, plus automation of the clusters visual representation by using graphs.

We will go through the previous work done on clustering and validation methods as well as a presentation of the AliClu tool in Section 2, and Section 3 presents in detail the contents of the library and how to use it, along with a description on the representation of the final clusters resulting from AliClu. Section 4 shows the results obtained when using the library's methods on synthetic, real-world datasets and the Reuma.pt dataset. Finally, in Section 5 we will share some conclusions on the work done.

OVERVIEW OF CLUSTERING

The objective is to develop an automatic algorithm that will discover the natural grouping in unlabeled data.

Therefore, clustering can be defined as the problem of determining the structure of clustered data, without prior knowledge of the number of clusters or any other information about their composition [14].

The development of data clustering methods has been a multidisciplinary feat [20]. Computer scientists, biologists, taxonomists, medical researchers, and others who gather data and analyse it have in some way contributed to clustering methodology.

Many books have been published about data clustering [21, 36, 4, 20] and from this pool of knowledge we can find two approaches to clustering: **Hierarchical** and **Partitional**.

Hierarchical clustering

Hierarchical clustering algorithms produce a nested series of partitions. The clustering structure obtained can be represented as a dendrogram. The algorithms that produce the clustering structure follow one of two approaches: **Agglomerative** clustering, where the algorithm begins with partitions of single objects that are successively merged until a termination condition is met or only one cluster remains. Alternatively, we can use **Divisive** clustering, which will start with all objects belonging to one unique clustering that will be split at every iteration of the algorithm until a stopping criterion is satisfied or what is left is single object clusters.

Regardless of the approach taken, merging and splitting of elements follow the similarity between each of the clusters or objects within them. The calculation of the similarities is a distance problem and to calculate those we can, for instance, use metrics from the Minkowski family, like the Euclidean distance , which is simple to calculate. Furthermore, when joining clusters we need to build on top of these more basic metrics. These criteria for linking clusters is what differentiate algorithms and we can use the **single link** method, which takes the distance between two clusters as the *minimum* distance between all pairs of patterns drawn from the two clusters, **complete link** takes the distance between two clusters as the *maximum* of all pairwise distances between patterns in the two clusters, **average link** considers the distance from one cluster to another as the *mean* distance between elements of each cluster, **centroid link** measures the distance between the centroid of one cluster and the centroid of another cluster, finally, **ward's method** uses the *Ward variance minimization* algorithm, which measures how much the sum of squares will increase if we merge two clusters.

Partitional clustering

A partitional clustering algorithm obtains a single partition of the data instead of a clustering structure. Hierarchical techniques are popular in biological, social, and behavioral sciences because of the need to construct taxonomies. Partitional are more frequently used in engineering applications where single partitions are important. Moreover, partitional algorithms have advantages when dealing with large datasets for which the building of a dendrogram by an hierarchical algorithm would be computationally expensive and even hard to read by the human eye. One restriction that accompanies partitional algorithms is the choice of the number of desired output clusters.

The most intuitive and frequently used criterion function is the *Error Square*, based on the use of the distance between objects. The general idea of this types of algorithms is to obtain the partition which, for a fixed number of clusters, minimizes the square error. Suppose we want to organize a set of objects $\mathbf{x}_j, j = 1,...,N$, into k subsets $C = C_1,...,C_k$. The error squared criterion J then is defined as:

$$J(C) = \sum_{i=1}^{k} \sum_{j=1}^{N} \|x_j - c_i\|^2. \quad (1)$$

The most popular algorithm that uses *Error Squared* is **K-means** [26]. K-means initially creates k random centroids and assigns each point to the closest centroid. The next step will be to recalculate those centroids and iterate this step until no changes happen or the error function converges.

K-means is popular much thanks to being very easy to implement, although, it has the drawback of requiring the input of the number of clusters K beforehand, which is not always possible to know accurately in real-world applications. It is also affected by the randomly chosen centroids and initial partitions.

Other algorithms have been proposed, tackling the weak points of *K-means*. Examples of those are ISODATA [5], PAM [22] and K-medoids [12].

OVERVIEW OF CLUSTERING EVALUATION METRICS

Logically, there should be some kind of validation of the clustering resulting from applying a clustering algorithm. We can find in the literature two main methods: **internal** and **external** validation [3].

One of the first papers published on the topic was [34], where some methods for comparison of dendrograms start to arise, for instance, the cophenetic index [35] measures the connectness of data points in the clusters. [6] provides statistical criteria, such as entropy, average distance from nearest cluster center and coefficient of belongingness. In [1], it is presented a mathematical criteria and related with statistical theory for finding the "best" partition, for instance, some of the criteria is the minimum pairwise distance considering all partitions, and a measure for scatterness using a total scatter matrix of the N dataset points. [31] proposes the first objective method for clustering validation, with the Rand index, that like many new indicies after it, compares two different clustering results based on the similarity between pairs of points. Dunn index [11] follows, expanding the idea of pairwise comparison to evaluate two clusterings. Davies and Bouldin [7] introduced a new idea and from it a new index (DB index) which considers the dispersion inside the clusters and also the distance between the clusters. From the developments we talked about so far, many other indices were derived, each trying to improve the

quality of the clustering produced and reducing the drawbacks from previous work.

Going back to the two families of clustering validation indices (CVIs), **external validation** measures are based on the comparison of partitions, one generated by the clustering algorithm, another from clustering of a random sample of the data, a good partition of the data will be stable, independent of the subset chosen. The second approach, **internal validation**, is based on calculating properties of the resulting clusters, such as compactness, separation and roundness. In the sequel we will describe the fundamentals for each of the two approaches and list the several indices that will be part of our work.

External validation

In order to evaluate the resulting structure, we use statistical tests, more specifically the average. The metrics that we use for this approach will evaluate the resulting clustering C, by comparing it to the clustering of a random subset of the data, P.

Let us introduce the contingency matrix that is represented in Table 1, which contains information on the clusterings overlap. Each entry n_{ij} indicates the number of elements that are common to cluster C_i and P_j

		Partition C				
		C_1	C_2	...	$C_{k'}$	$\sum$
Partition P	P_1	n_{11}	n_{12}	...	$n_{1k'}$	$n_{1\cdot}$
	P_2	n_{21}	n_{22}	...	$n_{2k'}$	$n_{2\cdot}$
	.	.	.	...	.	.
	P_k	n_{k1}	n_{k2}	...	$n_{kk'}$	$n_{k\cdot}$
	$\sum$	$n_{\cdot 1}$	$n_{\cdot 2}$	...	$n_{\cdot k'}$	n

Table 1: The contingency matrix

The information in the contingency matrix can be transformed into a pairwise mismatch agreement between the clusters, shown in Table 2.

		Partition C		
	Number of pairs	In the same cluster	In different clusters	Sums
Partition P	In the same cluster	a	b	$a+b$
	In different clusters	c	d	$c+d$
	Sums	$a+c$	$b+d$	M

Table 2: The mismatch matrix

The entries a, b, c and d represent counts of pairs among all distinct pairs of the clusters overlap. We can interpret a and d as agreements between the two configurations, and b and c as disagreements between the two clusterings C and P. Following, we present the formulas for twelve external validation indices, which make use of the two matrices presented.

- **Adjusted Rand index** [19]

$$AR = \frac{a+d-n_c}{M-n_c}, \tag{2}$$

where, using the contigency matrix:

$$n_c = \frac{n(n^2+1)-(n+1)\sum n_{i\cdot}^2-(n+1)\sum n_{\cdot j}^2+\sum\sum \frac{n_{ij}^2}{n}}{2(n-1)}. \tag{3}$$

- **Jaccard index** [16]

$$J = \frac{a}{a+b+c}. \tag{4}$$

- **Fowlkes and Mallows index** [13]

$$FM = \sqrt{\frac{a}{a+b}.\frac{a}{a+c}}. \tag{5}$$

- **F-Measure** [2]

$$F = \frac{2a}{2a+b+c}. \tag{6}$$

- **Adjusted Wallace index** [33]

Firstly, lets define the Wallace coefficients [39]:

$$AW_{C\to P} = \frac{a}{a+b}, \tag{7}$$

$$AW_{P\to C} = \frac{a}{a+c}. \tag{8}$$

The index is defined as follows:

$$AW_{C\to P} = \frac{W_{C\to P}-W_{i(C\to P)}}{1-W_{i(C\to P)}}, \tag{9}$$

where $W_{i(C\to P)}$ is the expected Wallace coefficient under independence and is computed as:

$$W_{i(C\to P)} = 1 - SID_P, \tag{10}$$

where SID_P is the Simpson's index of diversity of the clustering P given by:

$$SID_P = 1 - \frac{1}{n(n-1)}\sum_{j=1}^{k} n_{\cdot j}(n_{\cdot j}-1). \tag{11}$$

- **Kulczynski index** [24]

$$K = \frac{1}{2}\left(\frac{a}{a+c}+\frac{a}{a+b}\right). \tag{12}$$

- **Phi index** [8]

$$Phi = \frac{ad-bc}{(a+b)(a+c)(b+d)(c+d)}. \tag{13}$$

- **Rogers-Tanimoto index** [8]

$$RT = \frac{a+d}{a+d+2(b+c)}. \tag{14}$$

- **Sokal-Sneath index** [8]

$$SS = \frac{a}{a+2(b+c)}. \tag{15}$$

- **Hubert index** [19]

  The Hubert's index used is the normalized version of the original metric, represented by Γ can be defined as:

$$\Gamma = \frac{Ma-(a+b)(a+c)}{\sqrt{(a+b)(a+c)(d+b)(d+c)}}. \tag{16}$$

- **Variation of Information index** [27]

$$VI = -\sum_i p_i \log p_i - \sum_j p_j \log p_j - 2\sum_i \sum_j p_{ij} \log \frac{p_{ij}}{p_i p_j}, \tag{17}$$

  where $p_{ij} = n_{ij}/n$, $p_i = n_{i.}/n$, $p_j = n_{.j}$.

- **Van Dongen index** [9, 38]

$$VD = \left(2n - \sum_i \max_j n_{ij} - \sum_j \max_i n_{ij}\right) \Big/ 2n. \tag{18}$$

Internal validation

Internal validation measures are often based on the following two properties:

- **Compactness.** Measures how closely related the objects in a cluster are. Some metrics measure this based on variance. Lower variance indicates better compactness. In addition, numerous measures estimate the cluster compactness based on distance, such as maximum or average pairwise distance, and maximum or average center-based distance.
- **Separation.** Measures how distinct or well-separated a cluster is from other clusters. For example, the pairwise distance between cluster centers and the pairwise minimum distances between objects in different clusters are widely used as a measure of separation. Also, measures based on density are used in some indices.

Following, we will define the eight internal validation indices we will use for our work.

- **Dunn index** [17, 11]

$$D(n_c) = \min_{i=1,\ldots,n_c} \left\{ \min_{j=i+1,..,n_c} \left(\frac{diss(c_i,c_j)}{\max_{k=1,\ldots,n_c} diam(c_k)} \right) \right\}, \tag{19}$$

  where $diss(c_i,c_j)$ is the dissimilarity function between two clusters c_i and c_j defined as:

$$diss(c_i,c_j) = \min_{x \in c_i, y \in c_j} d(x,y), \tag{20}$$

  where d is the distance between x and y, which elements in cluster c_i and c_j, respectively.

  In Eq. (19), $diam(c)$ is the diameter of a cluster, which may be considered as a measure of clusters' dispersion. The diameter of a cluster C can be defined as follows:

$$diam(C) = \max_{x,y \in C} d(x,y). \tag{21}$$

- **Davies-Bouldin index** [7]

  The similarity measure is

$$R_{ij} = (s_i + s_j)/d_{ij}, \tag{22}$$

  where s_i, s_j are the measures of dispersion of clusters C_i and C_j, respectively, and d_{ij} the distance between the two centroids.

  Then the index is defined as:

$$DB(n_c) = \frac{1}{n_c} \sum_{i=1}^{n_c} \max_{i=1,\ldots,n_c, i \neq j} R_{ij}. \tag{23}$$

- **SD validity index** [30]

$$Scat(n_c) = \frac{1}{n_c} \sum_{i=1}^{n_c} \frac{\|\sigma(c_i)\|}{\|\sigma(X)\|}, \tag{24}$$

  where c_i is the center of cluster i, X is a data set and $\sigma(c_i), \sigma(X)$ are the variance of cluster i and variance of dataset X, respectively.

$$Dis(n_c) = \frac{D_{max}}{D_{min}} \sum_{k=1}^{n_c} \left(\sum_{z=1}^{n_c} \|c_k - c_z\| \right)^{-1}, \tag{25}$$

  where $D_{max} = max(\|c_i - c_j\| \forall_{i,j} \in 1,2,3,...,n_c$ and $D_{min} = min(\|c_i - c_j\| \forall_{i,j} \in 1,2,3,...,n_c$, the maximum and minimum distance between cluster centers, respectively.

  Now we can define the index based on Eq. (24) and Eq. (25):

$$SD(n_c) = \alpha Scat(n_c) + Dis(n_c), \tag{26}$$

  where α is a weighting factor to normalize the two terms of the equation calculated using:

$$\alpha = Dis(max\, n_c). \tag{27}$$

- **SDbw validity index** [18]

$$SDbw(n_c) = Scat(n_c) + Dens\_bw(n_c), \tag{28}$$

  The first term, which measures compactness, is equal to Eq. (24)

Inter-cluster separation is defined by density between clusters:

$$Dens\_bw(n_c) = \frac{1}{n_c(n_c-1)} \sum_{i=1}^{n_c} \left(\sum_{j=1, i \neq j}^{n_c} \frac{density(u_{ij})}{max\{density(c_i), density(c_j)\}} \right), \tag{29}$$

where c_i, c_j are, respectively, the center of clusters i and j, u_{ij} the middle point of the line segment defined by the clusters' centers c_i and c_j. The term density is defined as:

$$density(u) = \sum_{l=1}^{n_{ij}} f(x_l, u), \tag{30}$$

where n_{ij} = number of tuples that belong to clusters c_i and c_j i.e., $x_1 \in c_i \cup c_j$ represents the number of points in the neighbourhood of u. We define this neighbourhood as a hyper-sphere with center u and radius as the average standard deviation of the clusters, $stdev$. The function $f(x,u)$ is defined as:

$$f(x,u) = \begin{cases} 0, & \text{if } d(x,u) > stdev \\ 1, & \text{otherwise} \end{cases}. \tag{31}$$

- **CVNN index** [25]

  Cluster separation measurement is given by the following equation:

$$Sep(n_c,k) = \max_{i=1,2,\ldots,n_c} \left(\frac{1}{n_i} \sum_{j=1}^{n_i} \frac{q_j}{k}\right), \tag{32}$$

  where k is the number of nearest neighbours, n_i is the number of objects in ith cluster C_i, and q_j is the number of nearest neighbours of the jth object in C_i that are not in C_i.

  To measure the compactness within the clusters:

$$Comp(n_c) = \sum_i \left[\frac{2}{n_i \cdot (n_i - 1)} \sum_{x,y \in C_i} d(x,y) \right], \tag{33}$$

  where n_i is the number of objects in the ith cluster C_i, and x and y are two different objects in C_i.

  Based on the above equations we can define the CVNN index:

$$CVNN(n_c,k) = Sep_{norm}(n_c,k) + Comp_{norm}(n_c,k), \tag{34}$$

  where

$$Sep_{norm}(n_c,k) = \frac{Sep(n_c,k)}{max_{n_{c_{min}} \leq n_c \leq n_{c_{max}}} Sep(n_c,k)}, \tag{35}$$

  and

$$Comp_{norm}(n_c) = \frac{Comp(n_c)}{(max_{n_{c_{min}} \leq n_c \leq n_{c_{max}}} Comp(n_c))}, \tag{36}$$

  The terms are normalized to the same range before adding up.

- **PBM index** [28]

  For a measure of the inter-cluster separation, let us denote by D_C the largest distance between two cluster centers:

$$D_C = max_{i,j \in n_c} d(c_i, c_j), \tag{37}$$

  where c_i and c_j are the centers of clusters i and j, respectively.

  On the other end, considering the within clusters compactness, let us denote by E_C the sum of the distances of the points of each cluster to their center, using:

$$E_C = \sum_{k=1}^{n_c} \sum_{i \in C_k} d(x_i, v_k), \tag{38}$$

  where x_i is a point in cluster C_k and v_k is the center of cluster k.

  Lets denote E_T as the sum of the distances of all the points to the center of the entire dataset that can be defined as:

$$E_T = \sum_{i=1}^{N} d(x_i, G_C). \tag{39}$$

  where where x_i is a point in the dataset and G_C is the center of the dataset, calculated using the average of all cluster centers.

  The PBM index is defined as:

$$PBM = \left(\frac{1}{n_c} \times \frac{E_T}{E_C} \times D_C \right)^2. \tag{40}$$

- **Silhouette index** [32]

$$S = \frac{1}{n_c} \sum_{i=1}^{n_c} \left(\frac{1}{n_i} \sum_{x \in C_i} \frac{b(x) - a(x)}{max[b(x), a(x)]} \right), \tag{41}$$

  where b and a functions are defined in the following way:

$$b(x) = min_{j, j \neq i} \left[\frac{1}{n_j} \sum_{y \in C_j} d(x,y), \right], \tag{42}$$

$$a(x) = \frac{1}{n_i - 1} \sum_{y \in C_i, y \neq x} d(x,y). \tag{43}$$

Function $b(x)$ calculates the average distance of object x to all other objects of a cluster $C_j, j \neq i$ and selects the minimum of those values.

Function $a(x)$ computes the average dissimilarity of object x to all other objects of its cluster C_i.

- **Xie-Beni index** [40][23]

$$XB = \frac{max_{k=1,..,n_c} \frac{\sum_{j=1}^{N} (\|x_j - c_k\|)^2}{n_k}}{min_{i,l \neq i} (\|c_i - c_l\|)^2}. \quad (44)$$

AliClu

Given the increasing availability of electronic medical records (EMRs) with longitudinal data, this data can be used for making clinical decisions so that physicians can choose personalized treatments for the patients. The work done sets out with the goal of finding an efficient way to cluster patients based on their temporal information from medical appointments. It is proposed the application of the Temporal Needlman and Wunsch algorithm [37] to align discrete sequences. The obtained TNW pairwise scores are then used to perform hierarchical clustering. To test the performance of the method, synthetic datasets were generated by continuous-time Markov chains. For datasets with 2 clusters and that were very well separated, the method successfully found the correct clusters with percentage of correct decisions above or equal to 80%. The tests with the Reuma.pt dataset were done in a non-automatic manner, i.e., the results of clustering were analysed individually. It was discovered 18 clusters in the biological sequences where successful separation based on the events and temporal information between them was achieved.

The validation of the resulting clusters, being a very important step in identifying the correct patterns, is not getting enough focus. Only external validation measures are considered and are not very extensive. Therefore, we propose to add on top of AliClu, more CVIs. Hopefully, the new information that will be gathered can make the cluster decision process more automatic and trustworthy.

One last contribution we wish to give, concerns the visualization of the clusters. In previous work, the generation of the patients sequence was done manually, but being a repetitive process and also a fairly important one given the fact that this representation is what gives values to end users, we built a function that produces said clusters.

CLUSTERVAL

The *clusterval* package for python provides an implementation of all the indices described in the preceding sections and can be found in a github repository: https://github.com/Nuno09/clusterval.

The package evaluation of a clustering of data goes through the following steps:

1. Set the range of **k** to be tested;
2. Perform clustering **C** on the data for **k** clusters;
3. Calculate the values of each internal CVI for structure **C**;
4. Set the range of bootstrap samples to produce.
5. Generate a random sample from the original dataset;
6. Perform clustering **P** of the sample, using the same **k**;
7. Calculate the values of each external CVI, considering structures **C** and **P**;
8. Repeat step 5 until the limit of bootstrap samples has been reached;
9. Save the average values of the external CVIs for current **k**;
10. Repeat from step 2 with new **k**, until the max **k** is reached;
11. Each index takes a vote on the number of clusters that gives the best index value. The majority answer is defined as the number of clusters for the dataset.

In Appendices all CVIs are listed with the name in *clusterval*, value ranges and rule for best value.

The *clusterval* package produces *Clusterval* objects that can be used to perform clustering evaluation of any list-type dataset or distance matrix.

The package can be installed by getting the stable version from the Python software repository, PyPi, with the following instruction:

pip install clusterval

Alternatively, the development version can be pulled from GitHub:

pip install git+https://github.com/Nuno09/clusterval.git

Once the package is installed, it can be loaded in a Python environment like so:

from clusterval import Clusterval

The initialization of the instance with empty parameters means that some default attributes will be assigned, but they can also be specified upon creation. The possible attributes are listed in Table 3:

min_k	Integer that sets the minimum number of clusters to test. Default is 2.
max_k	Integer that sets the maximum number of clusters to test. Default is 8.
algorithm	String that sets the clustering algorithm to use. Default is the hierarchical clustering with ward linkage.
bootstrap_samples	Integer that sets the number of bootstrap samples simulated. Default is 250.
index	String (or list of strings) containing the CVIs calculate. By default considers all CVIs (Table 7 and 8).

Table 3: Available *clusterval* parameters.

For information on the parameters and methods present, use the command:

help(Clusterval)

To initialize the object and do clustering evaluation:

c = Clusterval() $\rightarrow$ creates object

c.evaluate(data) → evaluates the dataset, calculating the CVIs

Note that when passing a distance matrix, K-means clustering is not possible and some indices that depend on feature knowledge will not be calculated, these indices are *XB, SD, S_Dbw, PBM and DB*.

To see more information about the evaluation (e.g. CVI scores) there is the command:

print(c.long_info)

To visualize the **dendrogram** produced by the hierarchical clustering [1], use the command:

c.plot()

AliClu - clusters visual representation

An example of a cluster formed by AliClu, with 2 elements, can be seen in Table 4.

id_patient	coded_sequence
3948652	0.H,187.F,785.Z
69331060	0.F,1109.Z

Table 4: Patient's sequences belonging to one cluster after AliClu analysed Reuma.pt

The sequences are interpreted in chain, with each symbol representing a treatment ("Z" meaning the end of treatments) and the numbers the time in days between one treatment and the next. Therefore, the first patient start with treatment "H", which stays for 187 days, after which starts treatment "F", that lasted 785 days and concluded the sequence for this patient. Patient with id=69331060, had only one treatment, "F", in a time interval of 1109 days.

This description follows a chain, that can be translated into a graph scheme, where the nodes are the treatments and the edges direct from one treatment to the next. The edges carry weight that is equal to the time in days from one node to the next.

Our solution is to create a graph for each patient inside each cluster and merge them, resulting in a "average" graph for the specific cluster. We chose to consider this notion of average graph because in theory, the objects in the same cluster share many similarities, therefore, the treatment will be similar. A function "print_nodes" that handles the graph generation was added in AliClu's code and can be seen in https://github.com/Nuno09/AliClu/blob/master/Code/print_results.py.

For the cluster represented in Table 4, the respective graph is shown in Figure 1.

Note that, both sequences have a transition from state "F" to the final state "Z", and the average time being 947. Since the first patient start with a transition from "H" to "F", that information is added to the graph.

Figure 1: Graph representing the cluster in Table 4.

[1] In linux systems, the installation of the library "python3-tk" might be needed.

RESULTS

In order to test the performance of the library, we use synthetically generated datasets, 10 real datasets drawn from UCI repository [10] and the Reuma.pt dataset [15]. The dataset details and results are presented in the following subsections.

Synthetic datasets

The synthetic datastes were created with the goal of covering all possible combinations of 5 factors: number of clusters (**K**), dimensionality of the data (**dim**), **noise**, cluster density (**dens**) and cluster overlapping (**overlap**). Table 5 shows the possible values for each.

Parameter	Value
Nº samples	200-500
K	2, 4, 8
dim	2, 4, 8
noise	0, 0.1
dens	1, 4
overlap	1.5, 5.0

Table 5: Parameter values used for the synthetic datasets generation.

Moreover, we will use *clusterval* with 4 different clustering algorithms, 3 hierarchical ones (single, complete and ward) and 1 partitional (kmeans). Considering all factors we get 72 possible combinations. From each we will create 5 datasets, which gives us 360 synthetic datasets. Multiplying by 4 possible clustering algorithms we end up with **1440 configurations** to consider.

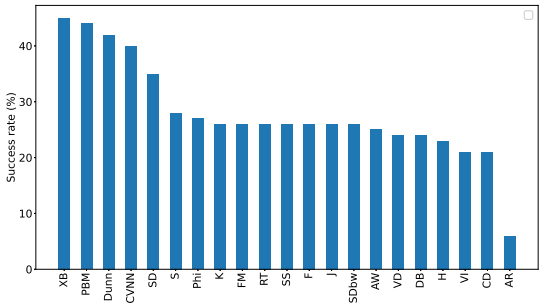

Figure 2: Overall results for the test with synthetic datasets.

The **overall** results for the synthetic datasets are shown in Figure 2. The figure shows the percentage of correct results for the number of clusters from each validation metric, sorted from best to worst performing. From the figure, we can see *Xie-Beni* achieves the best result, 45% success rate. *PBM*, *Dunn* and *CVNN* also show a similar result, 44%, 42% and 41%, respectively. These three metrics are the only that achieve a higher result than 40%. It is noticeable to see that internal indicies perform better than most external indices, with the six best metrics being six out of eight internal validation indices. It is also interesting to see that all external CVIs have very similar results, with the exception of the *Adjusted Rand* which holds the lowest success rate from all CVIs (6%).

For the experiments with other factors we observed that all CVIs, except *AR*, do better when the **number of clusters** is two, which is expected. The average result for k=2 is 71,5%, which drops to 11,1% for k=4 and to 5.4% for k=8. Clearly, the CVIs have difficulties guessing the right number of clusters when this parameter increases and we can not choose one of the metrics as the best in this regard, although internal indices perform relatively better.

With respect to **dimensionality** the results get better with an increase in the number of features in the dataset, which is unexpected due to increase in complexity. All indices, except of *SDbw* and *AR*, perform better with dim=8. Furthermore, we would like to highlight *PBM*, which seems to not be very sensitive to the number of features present, with 38% success rate for dim=2, 47% for dim=4 and 48% when the number of features is 8. On average, the success rate for dim=2 is 22.9%, for dim=4 it is 27.4% and for dim=8 it is 37.7%.

Focusing on the results from the **noise** level experiment, we can say that the addition of noise to the dataset does not affect the CVIs performance severely where we see a drop on the average success rate of 2.7% when introducing noise (from 30.7% to 28%). In fact, some indices (*XB, CVNN, AR*) look to be more successful with a 10% noise addition.

With respect to the **density** of the clusters, like with noise, a small drop happens when introducing cluster density, in this case of 4.5% (31.6% to 27.1%). The overall trend is also mostly kept.

When we compare well separated clusters with **overlapped** clusters the average results decrease by 13.3% (from 36% to 22.7%). Notable exception are the Van Dongen index (VD), which is barely affected by the change in the cluster structure, Variation of Information (VI) and Adjusted Rand (AR) that performed slightly better on datasets with overlap.

Finally, when testing the effect it has the **clustering algorithm** used, no clear pattern can be found, it seems that it does not have a big effect on the overall results. The best average results are obtained when using the K-means algorithm (32.5%), followed by ward algorithm (31.5%), complete (26.8%) and single (26.5%). Moreover we can conclude that single algorithm gives especially bad results for Silhouette (S) and Davies-Bouldin (DB), when compared with the other algorithms.

Real-world datasets

The 10 real-world datasets compared are listed in Table 6. In this case, since we cannot change characteristics of the datasets, comparison is based on the algorithms used for clustering, hence we consider **40 configurations**.

Dataset	N° instances	N° attributes	Classes
Breast tissue	106	9	6
Ecoli	336	7	8
Glass	214	9	7
Haberman	306	3	2
Iris	150	4	3
Parkinsons	195	22	2
Vehicle	846	18	4
Vertebral column	310	6	3
Wine	178	13	3
Yeast	1484	8	10

Table 6: Real datasets from UCI repository.

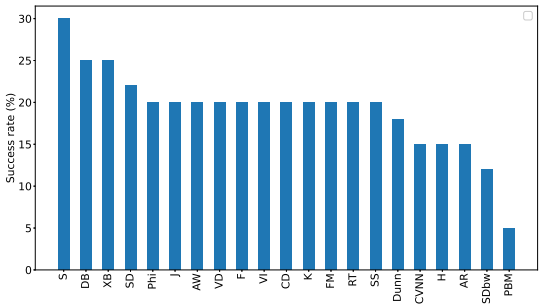

Figure 3: Overall results for the real-world datasets experiment.

Following a similar style to the one from synthetic datasets, in Figure 3 we show the overall results for the real datasets. A quick comparison with the synthetic datasets (Figure 2) shows that there has been some changes in the rankings of the indices, the most glaring one being *PBM* going from second to last in order of success rate. Notably, the external indices in general maintain the same positions and results, while the internal validation metrics continue achieving the best results, except the already mentioned *PBM*, and also *CVNN* and *Dunn*. Worthy of note is that Silhouette performance remained consistent, with 28% for synthetics and 30% for real datasets. *S* jumped from rank 6th for synthetics to 1st for real world datasets. *DB* index had a similar behaviour and is 2nd for real datasets. While in

synthetic results we see a 29.3% average success rate, for real datasets it drops to 20.1%.

Regarding the clustering algorithm used, the results obtained follow the overall trend. Noticeably, the algorithm chosen does not carry any influence when using external CVIs, with exception of Hubert statistic. *PBM* index could only guess correctly the number of clusters when using single linkage hierarchical clustering. Furthermore, ward is the algorithm that provides the best results on average with 22% success rate, followed by k-means (20.5%), complete (19.5%) and single (18.5%).

Since the results are very poor, we analysed the CVIs scores. In Table 9 of the Appendice are displayed the CVI values for each dataset, when using the k-means algorithm. Each dataset has 3 rows, one for the number of clusters selected (NC), other for the index value for that NC and a final one with the value for the correct NC.

We can observe that the majority of external indices chooses NC=2, and internal CVIs come closer to true values. For datasets Haberman and Parkinsons, the overall results are quite good. AR, CVNN, SD, PBM and Dunn guess the correct number for Iris. Regarding the Vertebral column dataset, XB, DB, S and PBM output the correct number of clusters, for Wine dataset only AR, and, finally, for Yeast dataset, CVNN is able to guess the correct number of clusters. In conclusion, we can see that the best value (2nd row for each dataset) is usually very similar to the correct value (3rd row for each dataset), which suggests that the validation cannot perform in a totally automatic way.

Reuma.pt dataset

In this section, we repeat the experiment I that is described in the paper about AliClu [29], which consists on performing alignment with sequences created with the patient treatment history found on Reuma.pt dataset. The goal of this experiment is to try to find similarities between patients based on different variables and, ultimately, be able to stratify them according to their similarity regarding the disease and treatment evolution. The difference in our replication of the experiment is that we have more clustering validation indices (CVIs) that could help us decide on the best clustering structure and an automatic function that generates the chain of treatments for each cluster. Like the previous work, the most relevant results will be presented.

The dataset used was the same as in [29] for experiment I (more information on the dataset in [15]), and we tested for values of gap = [0.0, 0.1], Tp = {0.25, 1.00, 2.00} and 5 hierachical clustering algorithms.

The tests are divided in two: 1) a automatic generation of the results, and 2) a semi-automatic generation of the results, where the user makes the final decision.

The algorithms in AliClu were used to create the sequences and the parameters used for the clustering processed were bootstrap_samples=250, min_k=2, max_k=20, all CVIs will be considered and we will vary the algorithm used. If any parameteres are changed it will be indicated.

From previous work we know that negative gap values do not produce good results, therefore we focus only on gap values that are positive.

Regarding the automatic results, for the Average and Single linkages, clusterval chose two as the number of clusters for tests tp=0.25 and tp=1.00. While for tp=2.00, single link chooses again k=2, but average link ouputs k=7. For this two clustering algorithms, when the results is a two cluster structure, it is generated one very small partition (1-8 elements) and a very big one with all other elements. The resulting dendrogram structure does not have any clear structure and the generated clusters are hard to interpret. Because of this reasons, the results for these algorithms was not accepted.

The most interesting results were given by complete-linkage and ward-linkage hierarchical clustering. Complete linkage, selected k=20 as the cluster number, gap=0.3 and Tp = 0.25. The dendrogram showed a good structure and the generated clusters were also easily interpreted.

On the other hand, for Ward-linkage hierarchical clustering the algorithm chose k=20 as the best number of clusters, gap=0.4 and Tp=0.25. Moreover, the dendrogram showed a good structure and the generated clusters were also easily interpreted.

The second part of the experiment consisted in running the same previous tests but instead of letting the algorithm choose the number of clusters we will do it ourselves.

Most clustering algorithms tend to form few and large clusters that don't seem to carry much information. Much like the automatic results, complete-linkage and ward-linkage hierarchical clustering generated the most interesting cluster structures. For complete linkage, the clustering for gap=0.7 and Tp=0.25 was chosen, and looking at the average index values in Table 4 we can see that almost all indices have better values for high k number (range of 16 to 20), with exceptions of internal CVIs, CVNN and S (range 9 to 12). In the end, k=20 was chosen because it was creating more simple sequence chains that seem to hold more information in the problem context.

Average values of 20 clustering indices
gap: 0.70, Tp: 0.25, complete link

k	AR	FM	J	AW	VD	H	F	VI	K	Phi	RT	SS	CVNN	S	Dunn
2,0	0,99988	0,62051	0,44128	0,10394	0,29625	0,09108	0,60958	1,04801	0,63189	1e-05	0,37757	0,28532	1,02503	0,12552	0,18996
3,0	1,00008	0,56543	0,39538	0,17465	0,3283	0,17199	0,56279	1,30925	0,56809	3e-05	0,41703	0,2493	1,05897	0,15379	0,19912
4,0	0,99997	0,54103	0,37347	0,24203	0,3391	0,25615	0,53881	1,47419	0,54328	4e-05	0,48192	0,23311	1,11672	0,12576	0,20966
5,0	1,00001	0,58547	0,41796	0,40951	0,28042	0,40285	0,58325	1,37167	0,58772	8e-05	0,59679	0,26959	1,04407	0,15871	0,19423
6,0	1,00001	0,60257	0,43438	0,41194	0,25429	0,44791	0,59971	1,29271	0,60544	9e-05	0,63552	0,28262	1,05277	0,19688	0,21077
7,0	1,0	0,64717	0,48187	0,51405	0,22697	0,53569	0,64523	1,18078	0,64913	0,00012	0,71026	0,32248	1,00526	0,20452	0,21323
8,0	1,0	0,66381	0,49726	0,50802	0,21212	0,56684	0,66019	1,12245	0,66747	0,00013	0,73483	0,33508	1,00455	0,22142	0,21886
9,0	1,0	0,66975	0,50005	0,49533	0,2023	0,58099	0,66344	1,07786	0,67615	0,00014	0,74641	0,33681	1,00268	0,24736	0,22662
10,0	1,0	0,67186	0,50099	0,49812	0,20179	0,58996	0,66483	1,05661	0,67899	0,00014	0,75897	0,33703	0,99655	0,23049	0,23025
11,0	1,0	0,75739	0,6125	0,67403	0,1586	0,70651	0,756	0,85654	0,75878	0,00019	0,84409	0,44729	0,91699	0,23037	0,24833
12,0	1,0	0,77736	0,63758	0,74504	0,15092	0,73656	0,77658	0,83105	0,77814	0,00022	0,87058	0,47166	0,89586	0,21569	0,25217
13,0	1,0	0,78691	0,6499	0,73218	0,14565	0,74936	0,78597	0,79956	0,78786	0,00023	0,87933	0,48477	1,71006	0,21581	0,25532
14,0	1,0	0,80151	0,67015	0,76831	0,14023	0,76937	0,80081	0,75943	0,80222	0,00025	0,89473	0,50733	1,69177	0,21147	0,25956
15,0	1,0	0,80384	0,67332	0,77232	0,14165	0,77402	0,80323	0,76193	0,80445	0,00027	0,90121	0,51071	1,68226	0,21535	0,26356
16,0	1,0	0,81145	0,68341	0,76273	0,13609	0,78371	0,81078	0,738	0,81212	0,00028	0,9073	0,52152	1,6804	0,22614	0,26891
17,0	1,0	0,87462	0,77757	0,89383	0,12592	0,85809	0,87374	0,631	0,87551	0,00033	0,94253	0,63955	1,64621	0,21728	0,27381
18,0	1,0	0,88101	0,78806	0,88979	0,12436	0,86554	0,88044	0,617	0,88158	0,00034	0,94628	0,65351	1,64565	0,21179	0,27455
19,0	1,0	0,8903	0,8032	0,89225	0,11993	0,87632	0,88997	0,59119	0,89064	0,00034	0,95141	0,67416	1,64414	0,20014	0,28025
20,0	1,0	0,89812	0,816	0,89254	0,11389	0,88536	0,8979	0,56346	0,89835	0,00035	0,95553	0,69196	1,6427	0,20081	0,28109

Figure 4: Average clustering indices values for complete-linkage with gap=0.7 and Tp=0.25

Finally, for ward-linkage, we chose the clustering for gap=0.2 and Tp=0.25. Analysing the index values from Table 5 we see that k in the range 16 to 20 has the best index values for every CVI (ignoring k=2, which is not desirable for being to general)

but there is not a value in that range that stands out, making the decision of the final number of clusters hard. Nevertheless, following the indices results, k=16 has the best results for the vast majority of the considered CVIs, therefore it was chosen as final k.

Average values of 20 clustering indices
gap: 0.20, Tp: 0.25, ward link

k	AR	FM	J	AW	VD	H	F	VI	K	Phi	RT	SS	CVNN	S	Dunn
2,0	0,99999	0,91158	0,85637	0,7796	0,06155	0,78982	0,91126	0,30504	0,91189	0,00013	0,83702	0,781	1,04531	0,17531	0,14127
3,0	1,0	0,85391	0,74744	0,77175	0,07765	0,77829	0,85384	0,51219	0,85397	0,00014	0,82006	0,60078	1,03919	0,16132	0,15248
4,0	1,0	0,85115	0,74302	0,75931	0,08276	0,79216	0,85062	0,54868	0,85168	0,00015	0,84427	0,59592	1,08904	0,18848	0,15248
5,0	1,0	0,85148	0,74257	0,75149	0,08776	0,79892	0,85021	0,5519	0,85275	0,00016	0,85453	0,59552	1,16851	0,21598	0,15248
6,0	1,0	0,86964	0,77254	0,8221	0,09117	0,8319	0,86869	0,54611	0,87059	0,00019	0,8889	0,63788	1,13245	0,19539	0,15248
7,0	1,0	0,85304	0,75323	0,82924	0,10365	0,81978	0,85262	0,56592	0,85346	0,00022	0,89846	0,62062	1,10951	0,16677	0,15248
8,0	1,0	0,85678	0,75119	0,86303	0,1093	0,83151	0,85592	0,5929	0,85764	0,00026	0,91697	0,60687	1,09169	0,15286	0,15248
9,0	1,0	0,86407	0,76191	0,85332	0,1144	0,84157	0,86377	0,60253	0,86437	0,00027	0,92551	0,61854	1,08833	0,16595	0,15248
10,0	1,0	0,87194	0,77413	0,86276	0,11498	0,85212	0,87171	0,6035	0,87217	0,00029	0,93354	0,63443	1,08924	0,15867	0,15248
11,0	1,0	0,88395	0,7931	0,85948	0,10906	0,86672	0,88378	0,56555	0,88412	0,0003	0,94169	0,65989	1,08307	0,17424	0,15248
12,0	1,0	0,89682	0,81373	0,87761	0,10921	0,88236	0,8967	0,53808	0,89694	0,00032	0,95048	0,6881	1,07423	0,15797	0,15248
13,0	1,0	0,90147	0,82091	0,87196	0,10703	0,88812	0,90127	0,5121	0,90167	0,00033	0,95391	0,69758	1,07146	0,16611	0,15248
14,0	1,0	0,92323	0,85785	0,91156	0,09699	0,91332	0,92319	0,45989	0,92328	0,00035	0,96549	0,75237	1,06662	0,16133	0,15248
15,0	1,0	0,92467	0,86047	0,91065	0,10311	0,91521	0,92454	0,467	0,9248	0,00036	0,96685	0,75685	1,06492	0,15637	0,16146
16,0	1,0	0,92857	0,8688	0,91879	0,09321	0,92012	0,92801	0,43499	0,92913	0,00038	0,9699	0,77414	1,06761	0,15443	0,16146
17,0	1,0	0,90568	0,83166	0,8701	0,09962	0,8953	0,90367	0,45281	0,90773	0,00038	0,96196	0,72587	1,74871	0,15855	0,19034
18,0	1,0	0,86349	0,76365	0,7926	0,11187	0,84974	0,85883	0,48366	0,86822	0,00038	0,94803	0,63828	1,74626	0,15442	0,19034
19,0	1,0	0,86332	0,7604	0,89496	0,10639	0,85152	0,86056	0,48658	0,86612	0,00046	0,95565	0,62241	1,73417	0,14351	0,19034
20,0	1,0	0,88904	0,8012	0,90204	0,09431	0,87997	0,88785	0,45059	0,89025	0,0005	0,96627	0,67368	1,7315	0,14782	0,19034

Figure 5: Average clustering indices values for ward-linkage with gap=0.2 and Tp=0.25

The generated clusters for the results mentioned above are accessible in https://github.com/Nuno09/MscThesis/tree/master/Results_Reuma.

CONCLUSIONS

We proposed a tool, named *clusterval*, that serves the purpose of clustering validation. With 20 metrics included, 12 with an external validation approach and 8 with an internal one. *Clusterval* has the potential to be useful when doing clustering of data, where the partitions produced should be in some way evaluated in order the unravel the clustering structure that best represents the hidden patterns.

In order to test the performance of *clusterval*, synthetic datasets were generated. Data points were randomly assigned to clusters with some variation in parameters for each partition, namely the number of clusters, dimensionality, density, overlap and noise. In total, 1440 different datasets were generated. The results were poor and not expected. For the overall of the datasets generated, XB, PBM, Dunn and CVNN acheived the best results, with 45%, 44%, 42% and 41%, respectively, and the average success rate of all CVIs was 29.3%. When analysing the influence of the factors from Table 5, the conclusion was that when introducing complexity to the datasets, the performance of the CVIs drops, which is expected, although, we also observed that internal CVIs perform much better external CVIs. The latter ones choosing many times very big clusters and small number of those.

The methods were further tested with 10 real-world datasets and in more detail Reuma.pt dataset. The real-world datasets were chosen with the goal of having a good variety of configurations, some small and others large, some datasets having many features and other few, same for the number of clusters. Even though, the datasets are not specific for the clustering problem, but rather for classification, we tested the tool for each, with different clustering alfgorithms. The results reported, in comparison with the synthetic datasets, have poorer results on overall, with some indices performing better for the real-world datasets, but also others performing worse. The average success rate considering all datasets and all clustering algorithms used was 20.1%, which is very poor. Further analysing the individual scores for each CVI we can identify that the values chosen are very close to the true values for the number of clusters, hence it shows that a completely automatic approach to cluster validation can lead to misleading results.

In the experiment with Reuma.pt, *clusterval* was implemented with the AliClu tool [29] and two tests were conducted: 1) automatic generation of the results, and 2) semi-automatic generation of the results, where the user takes the final decision.

For the automatic tests, complete and ward linkage algorithms lead to the most intersting results, the number of clusters selected by both was k=20, and the resulting clusters were easily interpreted and seem to carry useful information for the problem context.

For the semi-automatic tests, similarly to automatic tests, complete linkage and ward linkage had the most intersting results. The final analysis of the CVIs and dendrogram lead us to choose k=20 for complete linkage (like automatic) and for ward linkage, k=16.

In conclusion, *clusterval* is a tool with great usefulness for finding patterns from data, but a fully automatic approach is not able to produce results that shows its effectiveness, hence a more iterative process is recommended. Despite this, there is room for improvement to make the analysis more automatic. For instance, making the algorithm not choose directly the best scores for the CVIs, but rather take into account the problem context or have the voting system for the CVIs assign different weights to those metrics. We also aim at adding more internal validation metrics, which have proven to give better results.

REFERENCES

[1] 1967. On Some Invariant Criteria for Grouping Data. *J. Amer. Statist. Assoc.* 62, 320 (1967), 1159–1178. http://www.jstor.org/stable/2283767

[2] E. Achtert, S. Goldhofer, H. Kriegel, E. Schubert, and A. Zimek. 2012. Evaluation of Clusterings – Metrics and Visual Support. In *2012 IEEE 28th International Conference on Data Engineering*. 1285–1288.

[3] Charu Aggarwal and Chandan Reddy. 2013. *DATA CLUSTERING Algorithms and Applications*. 652 pages.

[4] M.R. Anderberg. 1973. Cluster Analysis for Applications. *Academic Press* (1973). DOI:http://dx.doi.org/https://dx.doi.org/10.1016/c2013-0-06161-0

[5] G.H. Ball and D.J. Hall. 1965. *Isodata, a Novel Method of Data Analysis and Pattern Classification*. Stanford Research Institute. https://books.google.pt/books?id=Ti3BGwAACAAJ

[6] Geoffrey H. Ball. 1965. Data Analysis in the Social Sciences: What about the Details?. In *Proceedings of the November 30–December 1, 1965, Fall Joint Computer Conference, Part I (AFIPS '65 (Fall, part I))*. Association for Computing Machinery, New York, NY, USA, 533–559. DOI:http://dx.doi.org/10.1145/1463891.1463950

[7] David Davies and Don Bouldin. 1979. A Cluster Separation Measure. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* PAMI-1 (05 1979), 224 – 227. DOI: http://dx.doi.org/10.1109/TPAMI.1979.4766909

[8] Bernard Desgraupes. 2017. Clustering Indices.

[9] Stijn Van Dongen. 2000. *Performance Criteria for Graph Clustering and Markov Cluster Experiments*. Technical Report. NATIONAL

RESEARCH INSTITUTE FOR MATHEMATICS AND COMPUTER SCIENCE IN THE.

[10] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. (2017). http://archive.ics.uci.edu/ml

[11] J. C. Dunn†. 1974. Well-Separated Clusters and Optimal Fuzzy Partitions. *Journal of Cybernetics* 4, 1 (1974), 95–104. DOI: http://dx.doi.org/10.1080/01969727408546059

[12] Vladimir Estivill-Castro and Jianhua Yang. 2000. Fast and Robust General Purpose Clustering Algorithms. In *PRICAI 2000 Topics in Artificial Intelligence*, Riichiro Mizoguchi and John Slaney (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 208–218.

[13] E. B. Fowlkes and C. L. Mallows. 1983. A Method for Comparing Two Hierarchical Clusterings. *J. Amer. Statist. Assoc.* 78, 383 (1983), 553–569. http://www.jstor.org/stable/2288117

[14] C. Fraley and A. E. Raftery. 1998. How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis. *Comput. J.* 41, 8 (01 1998), 578–588. DOI: http://dx.doi.org/10.1093/comjnl/41.8.578

[15] A. Faustino H. Canhão and F. M. et al. Jan. 2011. Reuma.pt - The Rheumatic Diseases Portuguese Register. *Acta Reumatologica Portuguesa* 36, 1 (Jan. 2011), 45–56.

[16] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. 2002a. Cluster Validity Methods: Part I. *SIGMOD Rec.* 31, 2 (June 2002), 40–45. DOI:http://dx.doi.org/10.1145/565117.565124

[17] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. 2002b. Clustering Validity Checking Methods: Part II. *SIGMOD Rec.* 31, 3 (Sept. 2002), 19–27. DOI: http://dx.doi.org/10.1145/601858.601862

[18] M. Halkidi and M. Vazirgiannis. 2001. Clustering validity assessment: finding the optimal partitioning of a data set. In *Proceedings 2001 IEEE International Conference on Data Mining*. 187–194. DOI: http://dx.doi.org/10.1109/ICDM.2001.989517

[19] Lawrence Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of Classification* 2 (12 1985), 25. DOI: http://dx.doi.org/10.1007/BF01908075

[20] Anil Jain and Richard Dubes. 1988. *Algorithms for Clustering Data*. Vol. 32. DOI:http://dx.doi.org/10.2307/1268876

[21] Anil K. Jain. 2010. Data clustering: 50 years beyond K-means. *Pattern Recognition Letters* 31, 8 (2010), 651 – 666. DOI: http://dx.doi.org/https://doi.org/10.1016/j.patrec.2009.09.011 Award winning papers from the 19th International Conference on Pattern Recognition (ICPR).

[22] Leonard Kaufman and Peter Rousseeuw. 2009. *Finding Groups in Data: An Introduction to Cluster Analysis*.

[23] Minho Kim and R.S. Ramakrishna. 2005. New indices for cluster validity assessment. *Pattern Recognition Letters* 26, 15 (2005), 2353 – 2363. DOI:http://dx.doi.org/https://doi.org/10.1016/j.patrec.2005.04.007

[24] S. Kulczyński. 1928. *Die Pflanzenassoziationen der Pieninen*. Imprimerie de l'Université. https://books.google.pl/books?id=cmSiHAAACAAJ

[25] Y. Liu, Z. Li, H. Xiong, X. Gao, J. Wu, and S. Wu. 2013. Understanding and Enhancement of Internal Clustering Validation Measures. *IEEE Transactions on Cybernetics* 43, 3 (2013), 982–994. DOI: http://dx.doi.org/10.1109/TSMCB.2012.2220543

[26] J. MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*. University of California Press, Berkeley, Calif., 281–297. https://projecteuclid.org/euclid.bsmsp/1200512992

[27] Marina Meila. 2007. Comparing Clusterings – An Information Based Distance. *Journal of Multivariate Analysis* 98 (05 2007), 873–895. DOI: http://dx.doi.org/10.1016/j.jmva.2006.11.013

[28] Malay K. Pakhira, Sanghamitra Bandyopadhyay, and Ujjwal Maulik. 2004. Validity index for crisp and fuzzy clusters. *Pattern Recognition* 37, 3 (2004), 487–501. DOI:http://dx.doi.org/https://doi.org/10.1016/j.patcog.2003.06.005

[29] Kishan Rama, Helena Canhão, Alexandra Carvalho, and Susana Vinga. 2019. AliClu - Temporal sequence alignment for clustering longitudinal clinical data. *BMC Medical Informatics and Decision Making* 19 (12 2019). DOI:http://dx.doi.org/10.1186/s12911-019-1013-7

[30] M. Ramze Rezaee, B.P.F. Lelieveldt, and J.H.C. Reiber. 1998. A new cluster validity index for the fuzzy c-mean. *Pattern Recognition Letters* 19, 3 (1998), 237 – 246. DOI:http://dx.doi.org/https://doi.org/10.1016/S0167-8655(97)00168-2

[31] William M. Rand. 1971. Objective Criteria for the Evaluation of Clustering Methods. *J. Amer. Statist. Assoc.* 66, 336 (1971), 846–850. DOI:http://dx.doi.org/10.1080/01621459.1971.10482356

[32] Peter J Rousseeuw. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.* 20 (1987), 53–65. DOI:http://dx.doi.org/https://doi.org/10.1016/0377-0427(87)90125-7

[33] Ana Severiano, Francisco R. Pinto, Mário Ramirez, and João A. Carriço. 2011. Adjusted Wallace Coefficient as a Measure of Congruence between Typing Methods. *Journal of Clinical Microbiology* 49, 11 (2011), 3997–4000. DOI: http://dx.doi.org/10.1128/JCM.00624-11

[34] R.R. Sokal and C.D. Michener. 1958. A Statistical Method of Evaluating Systematic Relationships. *The University of Kansas Science Bulletin* 38 (01 1958), 1409–1438.

[35] Robert Sokal and F. Rohlf. 1962. Sokal RR, Rohlf FJ. The comparison of dendrograms by objective methods. Taxon 11: 33-40. *Taxon* 11 (02 1962), 33–40. DOI:http://dx.doi.org/10.2307/1217208

[36] Robert R. Sokal. 1963. THE PRINCIPLES AND PRACTICE OF NUMERICAL TAXONOMY. *TAXON* 12, 5 (1963), 190–199. DOI: http://dx.doi.org/https://doi.org/10.2307/1217562

[37] Haider Syed and Amar Das. 2015. Temporal Needleman-Wunsch. DOI: http://dx.doi.org/10.1109/DSAA.2015.7344785

[38] Silke Wagner and Dorothea Wagner. 2007. Comparing Clusterings - An Overview. *Technical Report 2006-04* (01 2007).

[39] David L. Wallace. 1983. A Method for Comparing Two Hierarchical Clusterings: Comment. *J. Amer. Statist. Assoc.* 78, 383 (1983), 569–576. http://www.jstor.org/stable/2288118

[40] X. L. Xie and G. Beni. 1991. A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13, 8 (1991), 841–847. DOI:http://dx.doi.org/10.1109/34.85677

APPENDICES

Index	Name in clusterval	Range	Rule
Adjusted Rand	AR	[-1,1]	*max*
Jaccard	J	[0,1]	*max*
Fowlkes and Mallows	FM	[0,1]	*max*
F-Measure	F	(0,1]	*max*
Adjusted Wallace	AW	[-1,1]	*max*
Kulczynski	K	(0,1]	*max*
Phi	Phi	(-1,1]	*max*
Rogers-Tanimoto	RT	(0,1]	*max*
Sokal-Sneath	SS	(0,1]	*max*
Hubert	H	(-1,1]	*max*
Variation of Information	VI	[0,2log max(K, K')] [2]	*min*
Van Dongen	VD	[0,1)	*min*

Table 7: External clustering validation Indices.

Index	Name in clusterval	Range	Rule
Xie-Beni index	XB	(0,+∞)	*min*
Dunn index	Dunn	(0,+∞)	*max*
Davies-Bouldin index	DB	(0,+∞)	*min*
SD index	SD	(0, +∞)	*min*
S_Dbw index	S_Dbw	(0, +∞)	*min*
CVNN index	CVNN	(0, +∞)	*min*
PBM index	PBM	(0,+∞)	*max*
Silhouette index	S	(0,+∞)	*max*

Table 8: Internal clustering validation indices.

Table 9: Results for K-means clustering.

	AR	FM	J	AW	VD	H	F	VI	K	Phi	RT	SS	CVNN	XB	SDbw	DB	S	SD	PBM	Dunn	
Breast tissue NC=6	7.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	7.0	2.0	2.0	2.0	2.0	3.0	7.0	2.0	NC
	1.0005	1.0	1.0	1.0	0.0124	1.0	1.0	0.0462	1.0	$0.0358e^{-4}$	1.0	1.0	1.1233	0.0358	0.8558	0.0179	0.9753	0.0068	40953254951.0304	3.3504	Value
	0.9992	0.5749	0.3997	0.2262	0.3231	0.2739	0.5704	0.6707	0.5795	$0.0012e^{-4}$	0.4717	0.2503	1.1793	0.4037	3.8481	0.2808	0.6768	0.0083	36464035150.7726	0.1727	True value
Ecoli NC=8	3.0	2.0	2.0	2.0	2.0	2.0	2.0	13.0	2.0	2.0	2.0	2.0	3.0	2.0	13.0	2.0	2.0	2.0	2.0	3.0	NC
	1.0001	0.5537	0.382	0.051	0.3542	0.0451	0.5528	0.5572	0.5546	$0.00182e^{-7}$	0.3539	0.2361	0.9191	0.545	1.3621	0.4881	0.4274	28.6348	0.1515	0.0874	Value
	1.0	0.0867	0.0339	-0.1301	0.7323	-0.4919	0.0657	1.2508	0.1146	$-0.0306e^{-7}$	0.1049	0.0173	1.1129	1.209	1.5175	0.909	0.2687	49.3509	0.0481	0.0631	True value
Glass NC=7	5.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	5.0	4.0	9.0	4.0	4.0	3.0	4.0	4.0	NC
	1.0011	0.8963	0.8114	0.7856	0.1529	0.7292	0.8959	0.3407	0.8967	$0.0192e^{-6}$	0.7717	0.6826	0.8174	0.6294	0.9303	0.4522	0.5882	7.8025	12.6043	0.1633	Value
	1.0004	0.4942	0.3237	0.1051	0.3532	0.1349	0.4885	0.7273	0.5001	$0.00349e^{-6}$	0.3985	0.1934	1.085	1.2382	1.2943	0.6952	0.4034	9.7968	9.1859	0.06	True value
Haberman NC=2	3.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	5.0	2.0	6.0	2.0	2.0	2.0	7.0	12.0	NC
	1.0001	0.5658	0.3945	0.1285	0.3828	0.1275	0.5658	0.5676	0.5658	$0.00749e^{-7}$	0.3925	0.2457	0.7846	0.5116	0.9972	0.484	0.3967	1.0628	212.8548	0.0476	Value
	0.9999	0.5658	0.3945	0.1285	0.3828	0.1275	0.5658	0.5676	0.5658	$0.00749e^{-7}$	0.3925	0.2457	1.0667	0.5116	1.1687	0.484	0.3967	1.0628	144.9489	0.0262	True value
Iris NC=3	3.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	3.0	2.0	5.0	2.0	2.0	3.0	3.0	3.0	NC
	1.0002	0.5176	0.3487	-0.0065	0.404	-0.006	0.5171	0.5752	0.5181	$-0.00622e^{-7}$	0.3306	0.2112	0.9122	0.2555	0.6892	0.2024	0.7009	8.4861	25.0692	0.0988	Value
	1.0002	0.3236	0.1875	-0.1225	0.5377	-0.1812	0.3157	0.8504	0.3316	$-0.1983e^{-7}$	0.2537	0.1034	0.9122	0.4107	0.7912	0.3919	0.5553	8.4861	25.0692	0.0988	True value
Parkinsons NC=2	4.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	5.0	2.0	5.0	2.0	2.0	2.0	8.0	2.0	NC
	1.0001	0.9351	0.8767	0.8612	0.0748	0.6537	0.9342	0.2101	0.9361	$0.043e^{-6}$	0.8111	0.7809	0.8284	0.2573	2.441	0.2369	0.6623	0.1751	49577.66	0.3069	Value
	1.0	0.9351	0.8767	0.8612	0.0748	0.6537	0.9342	0.2101	0.9361	$0.043e^{-6}$	0.8111	0.7809	1.0	0.2573	5.9713	0.2369	0.6623	0.1751	39996.9435	0.3069	True value
Vehicle NC=4	3.0	2.0	2.0	2.0	2.0	2.0	2.0	20.0	2.0	2.0	2.0	2.0	5.0	2.0	19.0	2.0	2.0	3.0	5.0	19.0	NC
	1.0	0.5905	0.4183	0.1382	0.3251	0.1248	0.5899	-0.3032	0.5912	$0.000125e^{-7}$	0.3916	0.2645	0.7662	0.2482	5.5044	0.2239	0.6597	0.1959	230520.7614	0.0918	Value
	1.0	0.2343	0.1284	-0.2531	0.536	-0.3867	0.2272	1.0603	0.2418	$-0.000409e^{-7}$	0.1748	0.0687	0.8642	0.6249	5.6484	0.4562	0.4524	0.2092	145428.5273	0.0387	True value
Vertebral NC=3	4.0	2.0	2.0	2.0	2.0	2.0	2.0	12.0	2.0	2.0	2.0	2.0	2.0	3.0	2.0	3.0	3.0	4.0	3.0	5.0	NC
	1.0	0.5228	0.3533	-0.0017	0.4059	-0.0016	0.5221	0.5452	0.5235	$-0.0000891e^{-7}$	0.3326	0.2146	1.015	0.5342	0.9588	0.3185	0.6144	1.5009	45095.1951	0.0973	Value
	0.9998	0.4899	0.3242	-0.0119	0.4326	-0.012	0.4888	0.6295	0.491	$-0.000675e^{-7}$	0.328	0.1938	1.9607	0.5342	2.5848	0.3185	0.6144	1.5113	45095.1951	0.0688	True value
Wine NC=3	3.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	4.0	2.0	7.0	2.0	2.0	6.0	7.0	6.0	NC
	1.0001	0.5453	0.3739	0.0315	0.3614	0.0277	0.5442	0.5593	0.5463	$0.0145e^{-7}$	0.3459	0.23	0.5243	0.2687	0.475	0.2394	0.6406	0.0696	1140154.9197	0.0603	Value
	1.0001	0.2661	0.1485	-0.2009	0.5795	-0.3085	0.2585	0.9133	0.2738	$-0.1699e^{-7}$	0.2035	0.0802	0.828	0.5254	2.405	0.3474	0.568	0.0799	473716.2925	0.0163	True value
Yeast NC=10	3.0	2.0	2.0	2.0	2.0	2.0	2.0	27.0	2.0	2.0	2.0	2.0	10.0	3.0	27.0	2.0	6.0	3.0	6.0	27.0	NC
	1.0	0.5944	0.4219	0.1374	0.3184	0.1214	0.5934	-0.9362	0.5954	$0.0001281e^{-7}$	0.3902	0.2674	1.0642	0.6985	1.5093	0.8008	0.2699	60.8302	0.0394	0.0419	Value
	1.0	0.0538	0.0196	-0.1302	0.7760	-0.6036	0.0384	1.6648	0.0756	$-0.00011e^{-7}$	0.0688	0.0099	1.0642	1.3503	1.6670	1.1290	0.2011	100.5881	0.0184	0.0239	True value