OCEAN: A Non-Conventional Parameter Free Clustering Algorithm Using Relative Densities of Categories

Iffat Gheyas[a], Simon Parkinson[b], Saad Khan[b]

[a] *Secure Societies Institute, School of Human and Health Sciences, University of Huddersfield Queensgate, Huddersfield HD1 3DH, UK*
[b] *Department of Computer Science, School of Computing and Engineering, University of Huddersfield Queensgate, Huddersfield HD1 3DH, UK*

Abstract

In this paper, we propose a fully autonomous density-based clustering algorithm named 'Ocean', which is inspired by the oceanic landscape and phenomena that occur in it. Ocean is an improvement over conventional algorithms regarding both distance metric and the clustering mechanism. Ocean defines the distance between two categories as the difference in the relative densities of categories. Unlike existing approaches, Ocean neither assigns the same distance to all pairs of categories, nor assigns arbitrary weights to matches and mismatches between categories that can lead to clustering errors. Ocean uses density ratios of adjacent regions in multidimensional space to detect the edges of the clusters. Ocean is robust against clusters of identical patterns. Unlike conventional approaches, Ocean neither makes any assumption regarding the data distribution within clusters, nor requires tuning of free parameters. Empirical evaluations demonstrate improved performance of Ocean over existing approaches.

Keywords: autonomous clustering, categorical data, density based clustering, unsupervised learning, distance metric

1. Introduction

Clustering is an important research area in the field of data mining and machine learning [1], and as a result, numerous learning algorithms have been proposed. The majority of clustering algorithms are based on the use of distance measures and cluster detection techniques. In terms of distance measurements, they are often not appropriate for different data types (e.g., categorical) and arbitrary weights are often used. In terms of clustering mechanism, the use of distance measures during cluster identification, and the requirements to fine-tune input parameters, result in the algorithms being more suitable for certain types of application and require incremental trial-and-error utilisation.

In this paper, we present an improved density-based clustering algorithm which is accurate and parameter free. Furthermore, the technique is applicable to different types of data (categorical, ordinal, continuous or mixed types). We name this algorithm *Ocean* since its cluster identification process is inspired by the challenge of ocean boundary-making. An ocean is a continuous body of saltwater that is connected to other bodies of water with different water compositions. Oceanic gateways are where two oceans meet; however, the two waters of the two different oceans do not mix because of the differing rates of densities (different mineral content). The proposed Ocean clustering technique assumes that like oceanic boundaries, the cluster boundaries are recognisable by the differing rates of densities of the clusters' data points in a multi-dimensional space. We compare our algorithm with three popular clustering algorithms and their variants on forty one public datasets. Furthermore, we also empirically compare six well-known conventional dissimilarity measures. The specific contributions presented in this study are:

- A clustering algorithm named *Ocean*, with novelties lying in three different aspects. First, the conversion of non-continuous data to continuous data by replacing the categories with their relative densities. Second, the

Email addresses: iffat.gheyas@strath.ac.uk (Iffat Gheyas), s.parkinson@hud.ac.uk (Simon Parkinson), saad.khan@hud.ac.uk (Saad Khan)

technique only labels a region as a cluster if it has more than one unique pattern. Third, if and only if the ratio of densities of two adjacent clusters are lower than 2, they are merged into a cluster. This results in Ocean not having any free parameters that require tuning.

- Comparative evaluation (in terms of classification accuracy and speed) with six other clustering algorithms on 41 publicly available data sets, where Ocean excels in terms of classification accuracy and reduced run-time.

The rest of the paper is organised as follows: a brief review of the related work is provided in Section 2, followed by the details of our proposed algorithm in Section 3. An empirical performance evaluation strategy is included in Section 4, followed by a summary of results in Section 5. A discussion is presented in Section 6 where we theoretically compare the clustering algorithms to explore various possibilities as potential causes of the empirical findings. Finally, a conclusion is presented in Section 7 where we discuss the possible limitations of Ocean.

2. Related work

Clustering is an important research area in the field of data mining and machine learning.In hard clustering, each data point is assigned to one and only one cluster, whereas in fuzzy clustering each data point can have varying degrees of membership to several or all available clusters. A fuzzy-approach is useful for supervised classification when the data is highly heterogeneous. However, fuzzy logic is less useful for pure unsupervised learning, where no label information is available [2]. However, if a previously unseen event/object holds memberships in more than one cluster, the interaction strategy will be difficult to formulate through a splitting or averaging strategy. Hence, fuzzy clustering is not within the scope of our study, which concentrates on hard clustering only. In this section, we assess theoretically the effectiveness of existing algorithms against two key requirements of the clustering task.

2.1. Requirement One: distance measurement

A key function of a clustering algorithm is to measure the distance between two data points. Each data point is usually comprised of the observed values of different types of attributes (e.g., categorical, ordinal, and continuous attributes) of an object or event. Therefore, the ability to correctly measure the distance between data instances with different types of attributes is a key requirement of a clustering algorithm. Most existing algorithms use Euclidean metric as a way of measuring the distance between two data points [3], which is appropriate for continuous attributes. However, it is not suitable for handling categorical or ordinal attributes. In order to handle non-continuous attributes, a common practice is to convert each category into a separate binary variable and then the Euclidean distance measure is applied to the binary variables. This approach is called the 'Overlap' measure [4]. The major limitation of this approach is that it assumes that the distances between the categories are the same. However, this assumption is often not valid as all objects have varying degrees of similarities and differences from each other.

It is crucial to quantify the similarities (or dissimilarities) that exist between the categories of a categorical variable. To address this issue, a number of alternative distance measures are proposed in literature to compute the distance between two categorical data instances including: Goodall, Eskin, Inverse Occurrence Frequency (IOF), Occurrence Frequency (OF), and Lin [4, 5, 6]. These measures are the functions of either the absolute or relative densities (frequencies) of the categories of a categorical attribute. In the Eskin measure, the distance between any two distinct categories available in the categorical attribute is the same, exactly like the overlap measure. Hence, the Eskin measure does not offer much improvement over the overlap measure. In addition, it assigns higher distance weights for the attributes having a small number of categories than the attributes having a large number of categories. Therefore, the Eskin measure favours attributes with fewer categories over attributes having a large number of categories, even though the latter is more informative.

Like the overlap and Eskin measures, the distance between any two distinct categories of a categorical attribute is also the same as with the Goodall measure. However, the most concerning aspect of both Goodall and Lin measures is that they assign distance weights to matches. This implies that the distance between two identical data points will not be zero. The Goodall measure gives varying weights to the matches. It assigns a lower weight to a match if the category is infrequent rather than if the category is frequent. Lin's measure, on the other hand, assigns varying weights to both matches and mismatches. In Lin's measure, the highest possible weight is assigned to the match or mismatch between the two most common categories and the lowest weight is assigned to the match or mismatch between the

two most infrequent categories. Among the existing distance metrics, IOF and OF appears to be the best measures in the sense that they do not assign any distance to matches (i.e., when two data points have the same category value for an attribute) and does not assign the same distance to all pairs of distinct categories of an attribute. However, IOF measure arbitrarily assigns lower weights to mismatches on more frequent values, whereas OF measure assigns higher weights to mismatches on more frequent values. Several comparative studies have been conducted to determine the impact of such arbitrary weighting schemes [7, 8]. However, most of these studies are conducted on a small number of datasets and consequently, their findings are inconsistent. More comparative studies are required to further investigate the best distance measure.

2.2. Requirement Two: cluster detection

Another crucial requirement for high-quality clustering is the implementation of a robust cluster-detection mechanism. Since the target class information is not available, these cluster-detection mechanisms define a set of rules to identify clusters. Most popular clustering algorithms typically follow one of the following two principles: (1) principle of maximisation of inter-cluster distances and minimisation of intra-cluster distances [9], and (2) principle of density-based clustering [10, 11]. The most popular algorithms that work on inter-cluster and intra-cluster distance include K-means clustering (KMC) [12] and Kohonen self-organising map (SOM) [13, 14]. In contrast, based on density-based clustering principle, clusters are dense regions in the data space, separated by regions of lower object density. The cluster centres have the maximum local density, whereas, the neighbouring data points are lower in local density and they are at larger distance from the data points with a higher local density [15].

DBSCAN [16] and its variations, for example, RNN-DBSCAN [17], enhanced DBSCAN [18], GDBSCAN [19], HDBSCAN [20] and several others are widely used density-based clustering algorithms as evident in other research studies. The major drawback of DBSCAN is that it has two free parameters, *Minpts* (the minimum number of neighbours of a point that is not noise) and *Eps* (neighbourhood radius). Both these values need to be specified based on domain expertise before the clustering is performed. One of the latest developments in density-based clustering algorithms is ADD_clustering, which is fully autonomous as it does not require any user-specific parameters [16]. However, the major drawback of ADD_clustering is that they require a position in the data density to identify the cluster borders. Another issue with the density-based clustering technique is the poor performance on data with varying density distribution due to sparse cluster loss and cluster fragmentation. However, this issue has been resolved by Domain-Adaptive Density Clustering (DADC) algorithm [21] by automatically determining the density peaks of different density regions. In fact, the major drawback of both the maximal inter-cluster and minimal intra-cluster principle and the density-based clustering principle is that they implicitly assume that the distribution of data points within the cluster is normal – the normality assumption is essential for the validity of the existing clustering algorithms, but this assumption is not a strong one. There are no hard and fast rules in nature why the points within a cluster cannot be uniformly or otherwise distributed. More points may be located towards the edge than the centre and clusters may adjoin one another. A highly dense cluster may be surrounded by an even more denser clusters without any gap between two adjacent clusters. However, when the normality assumption is not met, existing clustering algorithms provide very few rules as how to proceed.

3. Details of the proposed algorithm (OCEAN)

In this section, we explain how Ocean works and how it satisfies two essential requirements for high quality cluster generation: (1) similarity measurements between data points – described in subsection 3.1, and (2) cluster detection approach – described in subsection 3.2.

3.1. Ocean's approach to the measurement of distances between data points

Every data point has a location in multidimensional space. For distance measurement purposes, we must know the exact positions of the data points in the multidimensional space for comparison purposes. However, the discrete attributes that are in a categorical scale or in an ordinal scale do not provide all the information required to locate the values on the axis [22]. Hence, Ocean converts all discrete attributes into continuous versions and then uses a Euclidean metric to measure distances. In order to compensate for the limited information encoded by discrete attributes, Ocean uses relative densities of categories to measure distances. The novelty of the proposed distance

measure is that it does not artificially assign weights to matches and non-matches, as existing approaches do. In the following subsections, explanation is provided detailing how Ocean handles categorical, ordinal and continuous features, including those with missing values.

3.1.1. Categorical Attributes

Categorical attributes provide little useful information about which category is higher or lower than the other and what is the distance between two categories. Ocean implements the following steps to convert categorical attributes to continuous attributes.

Step 1: Treat all missing values as a separate category: Missing values are common in real-world data. A common practice is to replace missing values with neighbouring non-missing values by an imputation algorithm. However, values may be missing not at random (MNAR) [23]. That means missing values occur because of the missing data item itself or other missing values. In such cases, the substitution of missing values based on observed values may introduce bias. Hence, Ocean treats missing values as a separate category instead of replacing the missing values using an imputation method.

Step 2: List unique categories within the categorical attribute.

Step 3: Estimate the relative density (*RD*) of each category using the following formula 1:

$$RD = \frac{m}{n} \tag{1}$$

Where, m is the total number of data points with the category of interest and n is the total number of data points in the dataset.

It is possible for two or more categories to have exactly the same relative density and thus be indistinguishable from one another in uni-dimensional relative density space. In those cases, Ocean implements step 4 (otherwise Ocean goes straight to step 5) to arbitrarily differentiate the relative densities of the categories in order to avoid losing vital information. The main justification behind step 4 is that clustering is carried out on multidimensional space and these categories may be critical due to complex interactions among multiple attributes. It should be noted that step 4 is designed in such a way that it will slightly change the relative densities making them a bit different from each other, but the difference will be almost negligible. In this context, the quite natural question may arise: what will happen if all categories of all attributes in a dataset have the same relative density? In the first place, we turn to relative densities of categories since discrete categories do not reveal much. If the relative densities are the same for all categories then they are also not revealing any useful information. Under the circumstances, all distance metrics as well as ours that are based on absolute or relative densities of categories are only as good as the overlap measure. It should be highlighted that the overlap measure also arbitrarily assigns the same distance between any two categories.

Step 4: Ocean executes this step only if two or more distinct categories in one attribute have the same relative density. It performs the following calculations using equation 2:

$$SN = RD \times N \tag{2}$$

where N is the number of categories of an attribute has the same relative densities, RD is the relative density of each of the N-categories (with the same relative density shown in equation 3 and equation 4), and SN is the total relative frequency of those categories with the same RD. Next, the following two values are calculated:

$$sn1 = \frac{4}{5} \times SN \tag{3}$$

$$sn2 = SN - sn1 \tag{4}$$

where $\frac{4}{5}$ is a constant value. Ocean randomly assigns a unique index $i \in [1, N]$ for each category. The sum of the indices of the four categories is calculated as $s1 = sum([1 : N])$. Ocean divides $sn2$ into $s1$ equal parts using $ep = \frac{sn2}{s1}$. The revised estimates of the relative densities of N categories (with randomly assigned indices i) is presented in equation 5:

$$RD_i = (\frac{sn1}{N}) + (ep \times i) \tag{5}$$

Where $i = \{1, \cdots, N\}$. Ocean keeps iterating step 4 until a unique relative density estimate is obtained for every category within the attribute.

Step 5: Replace each category in the dataset with its relative density attributes. Relative densities are always between zero and one. Hence, there is no need to standardise the values of this attribute.

3.1.2. Ordinal Attributes

Ordinal attributes give information about the order (greater/lesser, bigger/smaller) of their values, but not the exact distance between two ordinal values (or ranks). Ocean uses the following steps to handle the ordinal variables.

Step 1: It makes a temporary vector, T, with all the non-missing values of the attribute and then make a list (called A) of the unique values in T.

Step 2: Calculate the relative density (RD) of each unique value in A using equation 1, where m is the total number of data points with the value of interest and n is the total number of data points in T.

Step 3: To preserve the rank order relations, Ocean calculates cumulative relative density of each unique value in A. The value is the sum of the relative densities of that value and all the ones that precede it.

Step 4: Ocean replaces the values of the attribute in the original data matrix with their respective cumulative relative densities. Cumulative relative densities are always between zero and one. Hence there is no need for further standardisation.

3.1.3. Continuous Attributes

Continuous attributes naturally express the exact distance between values. Hence, Ocean does not need to convert continuous attributes. Ocean normalises continuous attributes using Min-Max scaling, so that all values are within the range of 0 and 1.

3.1.4. Imputation of Missing Values in Continuous and Ordinal attributes

Once all features have been converted into a numerical scale, Ocean imputes missing values in (original) continuous and ordinal attributes. In ordinal attributes, we cannot replace the missing values with the cumulative relative densities of missing values treated as single category, since rank order relations of the missing values in the attribute are unknown. Similarly, in continuous attributes, if we create a category for the missing values, we have to approximate not only the rank of the category in relation to the other continuous values of the corresponding attributes but also the precise distances of the category from the other values. Hence, Ocean cannot impute the missing values of continuous and ordinal attributes in the same way as it does in categorical attributes. If these attributes contain missing values, Ocean imputes missing values using k-Nearest Neighbour Imputation (kNNI) algorithm [24, 25]. kNNI imputes each missing attribute value of a data point with the average of the corresponding attribute values in the k-nearest non-missing neighbour data points. Ocean uses a MATLAB function for imputing missing values[1].

To address MNAR missing values, Ocean adds a new indicator variable (for each ordinal and continuous variable with the missing values) to the dataset. The indicator variable contains two categories that is 1 if the value of the attribute for the current row is missing and 2 if the value is not missing. Ocean replaces these two categories with their respective relative density values using equation 1. Furthermore, if the relative densities of two categories are the same, Ocean triggers step 4 described in subsection 3.1.1 of this section to incorporate uncertainty regarding the missing data mechanism.

3.2. Clustering mechanism of Ocean

As previously mentioned, the proposed clustering algorithm is named 'Ocean', since it is inspired by the oceanic landscape and phenomena that occur in it. This section is organised as follows. Section 3.2.1 describes two rules for cluster detection that Ocean follows. Section 3.2.2 presents a step-by-step discussion of how Ocean detects clusters.

[1]MATLAB function to inpute missing values using the following command: *F = fillmissing(Dataset, 'nearest', K)*, where `F` is the imputed dataset, and `Dataset` is the dataset with missing values in continuous and ordinal attributes.

3.2.1. Two rules Ocean uses when detecting clusters

Rule No. 1: This rule answers whether a region may be considered as a cluster or not. To explain this rule, we need to discuss the clustering mechanism of Ocean, which uses differences in density to separate different clusters. Ocean uses the following formula 6 to compute the density of a region/cluster:

$$density = \frac{population}{area} \tag{6}$$

Where, *population* is the total number of data points in the region, and *area* of a cluster is determined by the farthest Euclidean distance between the cluster's centroid and a member in the cluster.

It could happen that all data points (e.g., multiple patients with the same phenotype and fitness) are in the same segment. However, this segment cannot be marked as a cluster because in such cases the area of the cluster would be zero. If the denominator is zero, the fraction is not valid. More specifically, the denominator and/or numerator of equation 6 cannot be zero since Ocean works with density ratios. Hence, a region can only be considered as a cluster if its area as well as population are positive real numbers greater than zero. This implies that a cluster must have at least two unique data points – this is the first assumption rule that Ocean uses to detect clusters. This rule lies at the core of the purpose of clustering. The very purpose of the clustering is improving the generalisation capability of the machine so that they can successfully generalise to new variations. If all the members in a cluster are identical, the cluster is reduced to a point which is detrimental to generalisation. The clusters are characterised as having a surface area of greater than zero.

Rule No. 1: This rule determines whether two regions are located within the same cluster. Ocean uses the ratios of densities of adjacent regions to obtain the clusters. It should be noted that the ratio of the two regions cannot be lower than 1, since Ocean always places the higher density in the numerator. According to this second rule, the two regions belong to the same cluster if the following two conditions are met:

1. the density ratio of the two regions is lower than 2; and
2. when the two regions are merged into one, the density ratio of the new region and any of the original regions is lower than 2.

If the first condition holds but the second condition does not, this means that the two (original) regions are topologically separated and hence distinct. If any of the above conditions do not hold, the original regions do not belong to the same cluster. The logic for choosing 2 as the clustering threshold is that a ratio of 2 or above means that the numerator and denominator of the fraction are not the same, whereas a ratio of 1 (or below 2) means the numerator and denominator are roughly the same. The density ratio between regions can be 1, 2, 3, and so on.

3.2.2. Step-by-step description of the clustering mechanism of Ocean

Ocean detects clusters in two phases: (i) an initial cluster formation phase, and (ii) a merging phase.

Initial Cluster Formation Phase – In this phase each data point forms the smallest possible cluster (according to Rule 1 described in section 3.2.1). The points with higher density get priority over those with lower density for cluster formation. This phase comprises the following steps:

Step 1.1: Create a matrix U with the unique patterns/data points.

Step 1.2: Calculate the cumulative distance of each unique data point by summing the square of distances of the unique data point from all other data points based on equation 7.

$$CD_i = \sum_{j=1}^{N} (f_i \times dis_{ij}) \tag{7}$$

Where $i, j = \{1, 2, \cdots, N\}$ and CD_i is the cumulative distance of the i^{th} unique data point, N is the total number of unique data points, dis_{ij} is the Euclidean distance between the i^{th} unique data point and the j^{th} unique data point, and f_j is the frequency of the j^{th} unique pattern in the original dataset (i.e., the total number of times the j^{th} unique pattern appears in the original dataset).

Step 1.3: The cumulative distance of a point is inversely related to the density of the data point. Arrange the data points in matrix U in the ascending order of their cumulative distances. That is, the data points with the lowest

cumulative distances (or the highest densities) will be at the top of the list and the data points with the highest cumulative distances (or the lowest densities) will be at the bottom of the list.

Step 1.4: According to Rule 1, at least two unique data points are required to form a cluster. At this stage, each data point x_i, starting from the top of the matrix U to the bottom and systematically progressing down the list, sequentially forms its own initial cluster with its nearest unique data point x_j. Once a cluster is formed, members of the new cluster (x_i and x_j) will be removed from matrix U, so that every unique data point belongs to only one cluster.

Step 1.5: Select the unique pattern with the smallest cumulative distance from all data points within each cluster as the centre for that cluster. Use equation 8 to calculate the cumulative distance. In this context, N is the total number unique points within the cluster.

Step 1.6: For the given set of centres, reassign each unique pattern (or data point) to its closest centre.

Step 1.7: Find if there are any clusters with only one unique pattern (i.e., only the cluster centre) in them. If yes, then remove those clusters and place those unique patterns to their closest clusters.

Step 1.8: Update the cluster centres so that the sum of squares of distances between the centre and the cluster members is minimised.

Step 1.9: Check if there are any changes between the previous and current sets of centres. If any change is detected, go back to step 1.6. Otherwise proceed to the next phase.

Merging Phase – During this phase, Ocean merges those clusters that meet rule 2 (described in section 3.2.1 above).

Step 2.1: Calculate the densities of the clusters using equation 6.

Step 2.2: Estimate the ratios of the densities for all possible pairs of clusters.

Step 2.3: Create a multi-dimensional array, M, and store cluster pairs with the corresponding ratios in the array.

Step 2.4: Sort the pairs of clusters in ascending order based on their density ratios so that the pair with the smallest ratio is at the beginning of the column and the pair with the largest ratio at the bottom.

Step 2.5: Remove the pairs of clusters from the array whose ratio is 2 or higher. The resulting array M contains the pairs of clusters that can potentially be merged.

Step 2.6: Starting from the first row in M that contains the pair with the smallest ratio, Ocean goes through the array and checks each pair of clusters to find whether the clusters in the pair can be merged. To accomplish this task, Ocean performs the following sub-steps:

1. Group all the data points of two clusters (herein named Cluster A and Cluster B) in the pair together and form a new temporary cluster: Cluster C;
2. Find the centre of cluster C that has the smallest sum of squares of the distances from all member data points in the cluster. If more than one point has the same smallest distance, Ocean selects randomly one of these points as the centre of the cluster;
3. Calculate the density of Cluster C using equation 6;
4. Calculate the density ratios between Cluster C and Cluster A (let the ratio be $r1$) and between Cluster C and Cluster B (let the ratio be $r2$). If any of the two ratios (i.e., $r1$, and $r2$) are equal to 2 or above, the merging between the pair of clusters (Cluster A and Cluster B) is not possible, because this indicates that they are topologically separated. Ocean will reject the new cluster C and keep clusters A and B. Otherwise, if both ratios ($r1$, and $r2$) are smaller than 2, Ocean will accept the merged cluster C, and remove the old clusters (A, and B). Every time a merger is accepted, Ocean replace the old potential merger list M with the new list. The merging phase continues providing M contains the pairs of clusters to be merged; and
5. When no merging happens any more, repeat steps 1.6 to 1.8 of the first phase (i.e., initial cluster formation phase). Keep repeating the step until the previous and the current sets of centres are the same.

4. Comparative Evaluation

We compare Ocean with the following conventional clustering algorithms: (i) SOM(PSO): a self-organising map (SOM) optimised by Particle Swarm Optimisation (PSO) [26, 27], (ii) SOM(HCA): SOM optimised by a classical hill-climbing approach (HCA) [28, 29], (iii) KMC(PSO): K-means clustering (KMC) optimised by PSO, (iv) KMC(HCA): KMC optimised by HCA, (V) DBSCAN, and (VI) ADD_clustering [16]. We compare the algorithms on 41 real-world

datasets where class labels are known. The datasets are downloaded from the UCI repository [2] and individual dataset names are provided in Table 4, located in the Appendix.

In this section, we discuss how we evaluate the performance of Ocean and benchmark measures (discussed below in section 4.1), as well as how we compare six popular distance metrics for non-continuous attributes so that we can use the best one for the benchmark clustering algorithms (discussed in section 4.2).

4.1. Description of the Performance metrics

The following set of performance metrics are used to reflect the performance of clustering algorithms as fairly as possible. We apply the statistical test 'comparison of Groups or Conditions with a Control' [30] to determine the overall performance of the clustering algorithms assessed using the following performance criteria.

Criterion 1 – Classification Accuracy: Data clustering is usually undertaken in situations where class labels are uncertain. As a result, it is very difficult to measure the performance of clustering algorithms without using classification performance metrics. Hence, we compare the clustering algorithms across a collection of 41 known classification problems. However, it should be noted that the class information is not revealed to the algorithms. Although the algorithms do not know the true class labels, we know the class labels of the data instances, which we can use to determine accuracy. Hence, we assess the algorithms based on the classification accuracy on 10-fold cross validation (CV). In 10-fold CV, the unique data points of each dataset is randomly partitioned into 10 disjoint sets such that each set has (roughly) the same distribution (i.e., the ratios of different classes) and all instances of each unique pattern are placed in only one of the 10 sets, so the training and test instances will never exactly be the same and each unique pattern will only be tested once. The classifier is trained 10 times, using a different fold for testing and the other 9 folds combined for training. During the training phase, the clustering algorithms discover clusters.

In the testing phase, the test instances are labelled with a cluster class if they are located within the area of the cluster. The area of a cluster is determined by the farthest distance between the cluster centre and a member in the cluster. DBSCAN is the only algorithm whose clusters do not have any centre. In DBSCAN, the test instances are assigned to a cluster if the points fall within the epilson distance of any member point of the cluster. A test instance is considered to misclassify if any of the following occurs: (i) the actual class of the test record is not the same as the cluster class, (ii) the test instance is not located within the area covered by any cluster, (iii) the test instance is located in a cluster that has no majority class. Classification accuracy is estimated using the following equation 8:

$$ClassificationAccuracy(\%) = \frac{\sum_{k=1}^{10} m_k}{\sum_{k=1}^{10} n_k} \times 100 \tag{8}$$

Where, $\sum_{k=1}^{10} m_k$ is the total number correctly classified data points in the k^{th} validation (i.e., test) sample and n_k is the total number of data points in the k^{th} validation sample.

Criterion 2 – Percentage of observed classes identified by the algorithm: Classification accuracy alone is not sufficient to determine the true performance of an algorithm [31]. This is because sometimes class inequalities are so wide that if an algorithm misses one or two classes altogether (i.e., incorrectly classification of all instances of a minority class), the performance difference will be almost negligible, which is a problem when the minority classes are the classes of interest. Hence, the algorithms are also compared based on percentage of all observed classes that they have successfully discovered. We assume that the class label for each cluster to be the class label of the majority of the data points within the cluster. There should be at least one cluster per observed class where that class is the majority. For measuring the performance of clustering algorithms in terms of the percentage of classes identified, the entire datasets (i.e., no training-validation splits) are presented to the algorithms for clustering all sequences instead of only training instances. This is because some classes have only a few instances. We then label the clusters based on the class of majority members within the cluster. We exclude the clusters with the following characteristics from the performance measure: (i) no majority class is found to be the label of the cluster, and/or (ii) all members in the cluster are identical.

The percentage of actual classes discovered ($class_{discovered}$) is calculated based on the following equation 9:

$$Class_{discovered}(\%) = (\frac{m}{n}) \times 100 \tag{9}$$

[2] Centre for Machine Learning and Intelligent Systems, Machine Learning Repository:`https://archive.ics.uci.edu/ml/index.php`

Where m represents the total number of actual classes for which at least one cluster is detected where the majority of the member data points (in the corresponding cluster) actually belong to the given class; and n represents the total number of output classes in the dataset.

Criterion 3: Total running time: The maximum running time is fixed to five hours (18,000 seconds) for the following algorithms: Ocean, DBSCAN, ADD_clustering, KMC(HCA), and SOM (HCA); and ten hours (i.e., 36,000 seconds) for KMC (PSO) and SOM(PSO) since global search algorithms (such as PSO) generally require a large number of iterations for convergence. These time limits have been established through early testing and it is anticipated that the algorithms will complete in less than the overall time limit.

Memory usage and computational speed of the algorithms are evaluated on a PC with 16 GB RAM, Intel Core i7-7500 CPU@2.90 GHz, 64-bit operating system, and x-64 based processor. It should be mentioned that the speed (i.e., the clustering time of each algorithm) on each dataset is measured over the entire dataset (i.e., no training-validation split), and not on the training set only. Each runtime is recorded from reading data to obtaining the final set of clusters with MATLAB's tic-toc timer. Runtime includes data pre-processing and optimisation of model parameters.

All of the three above criteria together will enable a systematic performance of the clustering algorithms.

4.2. Comparative analysis of conventional dissimilarity metrics

The algorithms used to benchmark use the Euclidean distance metric for continuous features. To find the best existing dissimilarity measure for non-continuous attributes, we empirically test the following most popular distance measures: (1) overlap measure, (2) OF measure, (3) Eskin measure, (4) IOF measure, (5) Lin measure, and (6) Goodall's measure. The details of these dissimilarity measures are provided in Table 5 in the Appendix.

To find the best distance measure, we observe the performance of KMC algorithm with each of the above-mentioned six dissimilarity measures, while keeping the other things constant (i.e., the number of clusters). The dissimilarity metrics are tested on 41 known classification datasets.

In this particular comparative study, the number of classes observed in the dataset is specified as the value of K (i.e., the number of clusters) in KMC. The datasets we use in our study originally contains both continuous and non-continuous attributes. Since this comparative analysis is for non-continuous attributes, we transform the continuous attributes into discrete ones by grouping the normalised values into five categories which are: Category-1 $(0 - 0.2)$, category-2 $(0.2 - 0.4)$, category-3 $(0.4 - 0.6)$, category-4 $(0.6 - 0.8)$ and category-5 $(0.8 - 1)$. It should be emphasised that we convert the continuous attributes into categorical ones only for the comparison of dissimilarity metrics, not for the comparison of clustering algorithms. We compare the distance metrics based on the classification accuracy (Criterion 1 described in Section 4.1) on 10-fold-cross validation.

5. Results

In this section we provide a summary of the comparative results based on summary statistics.

The preferred conventional distance metrics for non-continuous attributes in descending order of classification accuracy are as follows: OF, Eskin, IOF, Lin and Goodall. The overall accuracy of each distance metric can be see in Figure 1). However, these performance differences are marginal, which could result from the fact that in this comparative study, the parameter of KMC (i.e., the number of clusters) is given as input (not optimised) since the goal is to compare the relative performance of the distance metrics for non-continuous attributes, not the performance of the clustering mechanism. In this particular comparative study, we set the number of clusters to the actual number of classes in the datasets (i.e., we use domain knowledge) and also the continuous attributes are categorised. However, the performance of KMC is consistently increased for all datasets when we optimise the parameter K using a search algorithm (see the overall performance of KMC (HCA)) in Figure 2 and its performances for individual datasets provided in Table 4). This fact suggests that classification and clustering are two completely different tasks. Clustering identifies homogeneous groups. Classification identifies the groups that belong to a particular class. The population of a group is often heterogeneous. As a result, personalised treatments are required to address the heterogeneity of these classes, hence the importance of clustering.

The fact that classification and clustering are two different things is made even clearer in the next comparative analysis where we compare clustering algorithms. The number of classes in these datasets, on an average, is $5(\pm 6)$ and the standard deviation is given in parenthesis. The numbers of clusters found by the clustering algorithms do

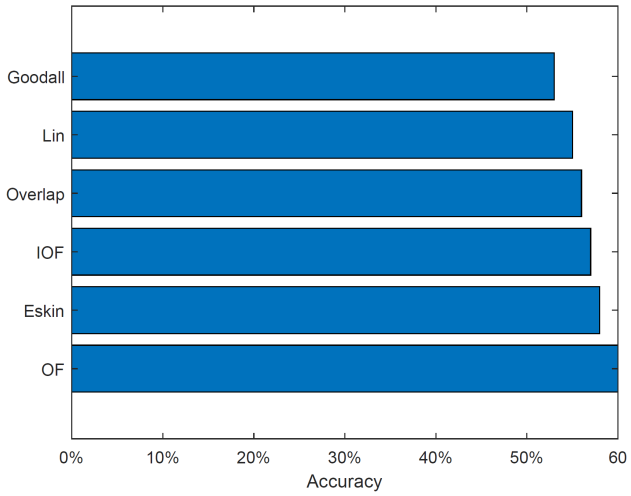

Figure 1: Performance comparison of popular distance metrics for categorical data. The y-axis represents the distance metrics and x-axis represents the classification accuracy (based on performance criterion1) in percentage. The blue bars represent the mean accuracy on the 41 datasets.

not coincide with the actual number of classes. The clustering algorithms in ascending order of the average number of clusters found are: ADD_clustering: 279(±444), Ocean: 308(±472), SOM(PSO): 475(±797), SOM(HCA): 675 (±1337), KMC(HCA): 894(±1775), KMC (PSO): 894±(1775), and DBSCAN: 2498(±13178). Based on the summary statistics, DBSCAN is the producer of the largest number of clusters. However, this result is slightly misleading in that DBSCAN actually produces the smallest number of clusters (by a wide margin) in most of the datasets (see Table 4) that is reflected on the large standard deviation observed for DBSCAN.

The best performing clustering algorithms (in descending order of performance) in terms of classification accuracy (performance criterion 1: see section 4.1) on unseen cases (10-fold cross-validation) for the 41 known classification problems are: Ocean, SOM (PSO), SOM (HCA), KMC (HCA), ADD_clustering, and DBSCAN (overall performance presented in Figure 2 and Table 4 shows the performance of each dataset). Ocean performs better than others because there is an optimal level of variation in its clusters due to its first rule (see section 3.2.1). Closer inspection reveals that the benchmark algorithms create many clusters with identical points. As a result, these algorithms often do not find a cluster for a new pattern. This is particularly true for KMC(PSO), KMC(HCA), SOM(HCA) and SOM(PSO). On the other hand, DBSCAN often creates clusters without any majority class. Hence, new points that lie inside these clusters are automatically considered to be misclassify. We attribute this problem to the way the parameters (especially minpts) of DBSCAN are commonly determined. For example, the Lung Cancer dataset has 32 instances and 56 attributes. DBSCAN returns only one cluster for this dataset. The reasons behind the bad performance of ADD_clustering is discussed in section 6.

The clustering algorithms in descending order with regards to the percentage of classes detected (2nd performance criterion: see section 4.1) are?: Ocean, SOM(PSO), SOM (HCA), KMC(HCA), KMC(PSO), ADD_clustering, and DBSCAN. The results are presented in Figure 2 and Table 4 provides the performance of each individual dataset. Ocean is particularly good at recognising clusters. We attribute this quality to the second rule of Ocean (see section 3.2.1). Ocean uses ratios between adjacent regions to identify clusters. Like existing ones, Ocean does not make any assumptions regarding the shape of the cluster. A majority of the points needs not to be at the middle of the clusters and the boundary of clusters are not required to be marked by the presence of a small number of points. There need not be any space gap between two adjacent clusters.

The clustering algorithms in descending order according to the quickest runtime (performance criterion 3: see section 4.1) are: Ocean, KMC(HCA), DBSCAN, SOM(HCA), ADD_clustering, KMC(PSO), and SOM(PSO) as presented in Figure 3 and Table 4 includes the performance on each individual dataset. It should be noted that we set the time limit to 10hrs for SOM(PSO) and KMC (PSO). The time limit for all other algorithms is 5hrs (see section 4.1). It seems that Ocean is slightly faster than KMC(HCA). Generally, KMC is fairly fast when the number of clusters is specified. However, KMC(HCA) and KMC(PSO) are required to optimise the parameter through repeatedly running

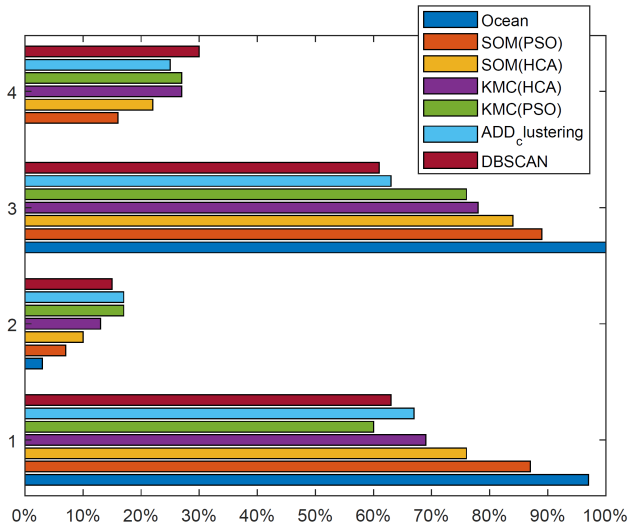

Figure 2: Comparison of Clustering Algorithms in terms of Performance Criteria 1 and 2. The X-axis represents accuracy. The bars marked 1 & 2 on the Y-axis represent the mean classification accuracy (performance criterion 1) of clustering algorithms and the corresponding standard deviations, respectively. The bars marked 3 & 4 represent the means and standard deviations, respectively, of the clustering algorithms on the performance criterion 2 (i.e., the percentage of the total number of classes detected.

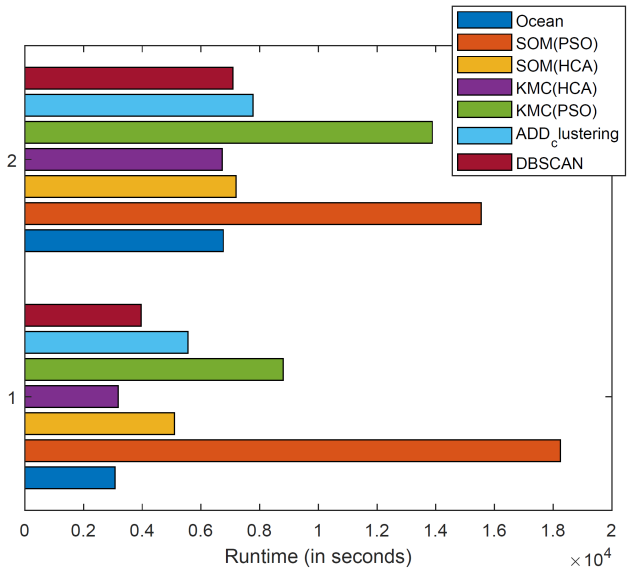

Figure 3: Comparison of clustering algorithms in terms of runtime. The X-axis represents accuracy. The X-axis represents the runtime in seconds. The group of bars marked 1 on the Y-axis compare the average runtime of the algorithms and the group of bars marked 2 compare the corresponding standard deviations.

k-means many times with different *k* values. They also need to calculate the Davies-Bouldin (DB) index [32] value after every *k*-means execution for cluster validity assessment and compare it to the previous DB index value(s).

The DB index is the ratio of the average within-cluster distance to the average between-cluster distance. These make them slow. It is noteworthy to mention that Ocean performs the k-means clustering twice during the clustering task – the first time after the formation of the initial clusters (steps 1.6 – 1.9 in the 'Initial Cluster Formation Phase') and the second time at the end of merging phase. But that is only twice per clustering task – just to make sure that data points belong to their closest clusters.

Like all other algorithms, Ocean's runtime increases exponentially with the number of instances in the dataset.

Table 1: Time Complexity of Ocean

Total number of Instances	Average runtime (seconds)	Standard deviation of runtime (seconds)
0-500	1.7	130.28
501-1000	10.82	4.71
1001-2000	85	56.75
4000-4600	1441	272.94
above 4600	18000	0

Table 2: The number of algorithms outperformed by an algorithm is the rank of the algorithm. The larger the number of outperformed algorithms, the better is the algorithm (of interest).

Algorithm of interest	Number of algorithms outperformed	Algorithms outperformed by the algorithm of interest
Ocean	6	KMC(HCA), SOM(HCA), SOM(PSO), DBSCAN, KMC(PSO), ADD_clustering
KMC (HCA)	5	SOM(HCA), SOM(PSO), DBSCAN, KMC(PSO), ADD_clustering
SOM (HCA)	4	SOM(PSO), DBSCAN, KMC(PSO), ADD_clustering
SOM (PSO)	3	DBSCAN, KMC(PSO), ADD_clustering
DBSCAN	2	KMC(PSO), ADD_clustering
KMC (PSO)	1	ADD_clustering
ADD_clustering	0	N/A

Table 1 shows the average runtime of Ocean on different sizes of datasets.

To compare the overall performance of the proposed and conventional clustering algorithms on all datasets, we apply the statistical test 'Comparison of groups or conditions with a control' [30]. The algorithms are ranked from the best to worst, taking into account the number of algorithms significantly outperformed by it at the 0.05 level. Table 2 shows the overall ranking results based on all three performance criteria: classification accuracy, the percentage of actual classes identified, and the time during which the algorithm is running. The first two criteria together reflect the accuracy of the clustering algorithm, whereas the last performance criterion indicates the algorithm's speed. Table 3 shows the final ranking results based on two performance criteria – excluding the performance criterion of runtime efficiency. The results reveal that the best algorithm and the worst algorithm in the current set of seven clustering algorithms are Ocean and ADD_clustering, respectively. The rankings of the following three algorithms vary according to the performance criteria chosen: SOM(PSO), SOM(HCA) and KMC(HCA). Based on the accuracy alone, the next best algorithm to Ocean is SOM(PSO), whereas, based on both accuracy and speed, KMC(HCA) is the second-best, followed by SOM(HCA) and SOM(PSO). In other words, KMC is preferable to SOM (even though SOM is more accurate) and there is no comparative advantage in optimising SOM by global stochastic algorithms (e.g., PSO) when we take into account the runtime. It should be mentioned, although different time limits are set for SOM(PSO) and SOM (HCA), SOM(PSO) could converge early. Time limits are set unevenly for the two reasons: (1) Global optimisation algorithms require substantially more time than local optimisation algorithms (e.g. HCA), and (2) we have the problem of scarcity in computing resources.

KMC(HCA) are found to perform significantly better than KMC(PSO), based on the evidence provided in Table 2, Table 3 and Table 4. We note that SOM(PSO) with SOM(HCA). SOM(PSO) performs significantly better than SOM(HCA) in terms of accuracy (Table 3). Closer inspection reveals that KMC(PSO) produces more clusters with identical patterns compared to KMC(HCA). We believe this is because of the performance index they use to find clusters. They minimise intra-cluster distances and maximise inter-cluster distances. This type of performance measure (e.g., DB Index) favours clusters with identical members. DB index landscape is also full of local optima. Thus KMC(HCA) converges to the nearest local optimum, where it converges. PSO, on the other hand, is more robust to local optima. Hence, KMC(PSO) ends up with many clusters of identical patterns. We note that SOM(PSO) outperforms SOM(HCA), which we believe is due to the facto that unlike KMC, SOM does not need to select centres from data. SOM randomly initialises cluster centres and then adjusts the cluster centres to best fit the data. Second, SOM has multiple model parameters which play a part in selecting an optimal number of centres. We believe these

Table 3: The number of algorithms outperformed by an algorithm is the rank of the algorithm. The larger the number of outperformed algorithms, the better is the algorithm (of interest).

Algorithm	**Number of algorithms outperformed**	**Outperformed algorithms**
Ocean	6	SOM (PSO), SOM(HCA), KMC(PSO), KMC(HCA), ADD_clustering, DBSCAN
SOM(PSO)	5	SOM(HCA), KMC(PSO), KMC(HCA), ADD_clustering, DBSCAN
SOM(HCA)	4	KMC(PSO), KMC(HCA), ADD_clustering, DBSCAN
KMC(HCA)	3	KMC(PSO), ADD_clustering, DBSCAN
DBSCAN	2	KMC(PSO), ADD_clustering
KMC(PSO)	1	ADD_clustering
ADD$_{c}$*lustering*	0	N/A

features of SOM make it relatively more resistant to clusters with identical patterns.

6. Discussion

In this section, we explore the rationale behind the superior performance of our proposed algorithm Ocean compared to that of the benchmark clustering algorithms. The performance of a clustering algorithm depends primarily on the choice of distance metric and clustering procedure. Hence, we theoretically assess the strength of these seven clustering algorithms in subjects with distance metric (discussed in Section 6.1 and clustering process (discussed in Section 6.2).

6.1. Comparison of the algorithms in terms of distance metric

A similarity metric maps data points to a high-dimensional feature space. If the map is poor, the performance of clustering algorithms cannot be good. In this study, all algorithms (including Ocean) use Euclidean distance metric to determine the closeness of data points based on continuous attributes. All benchmark clustering algorithms use the OF measure for non-continuous data since our empirical results suggests that this metric is better than the other conventional distance metrics. In the OF measure, mismatches on less frequent categories are assigned higher distance weights than mismatches on more frequent categories. In other words, if two data points $x1$ and $x2$ have two different equally rare (infrequent) categories for an attribute value and another two data points $x3$ and $x4$ have two different equally common (frequent) categories in the corresponding attribute space, the OF gives the highest weight to the distance between $x1$ and $x2$, followed by (in no particular order) the distance between $x1$ and $x3$, $x1$ and $x4$, $x2$ and $x3$, and $x2$ and $x4$. It assigns the smallest weight to the distance between $x3$ and $x4$. The conventional distance measure IOF, on the other hand, assigns the highest distance weights to the following pairs of points: ($x3$, $x4$), followed by (in no particular order) ($x1$, $x3$), ($x1$, $x4$), ($x2$, $x3$), and ($x2$, $x4$). IOF assigns the lowest weight to the distance between point $x1$ and point $x2$. In our empirical evaluation, IOF performs worse than OF. In contrast, Ocean replaces each category with its relative density value and then estimates the distance between two data points by computing the difference between the relative densities of corresponding categories for the two data points. Ocean implicitly assigns higher weights to the distances between the following pairs of points: ($x1$, $x3$), ($x1$, $x4$), ($x2$, $x3$), and ($x2$, $x4$) than to the distances within the following pairs of points: ($x1$, $x2$) and ($x3$, $x4$). Our empirical study suggests that Ocean's distance measuring approach works well. In Ocean, the higher the difference between the relative densities (frequencies) of categories of an attribute, the more distant they are from each other.

In the following text, we assess the soundness of these distance measures. It can be stated that a frequent point ($x1$) and an infrequent point ($x3$) have a high distance measure. However, two infrequent points $x1$ and $x2$ may or may not be far away from each other and it is simply not possible to know. Likewise, two frequent points $x3$ and $x4$ may or may not be far away from each other. It is difficult to determine by just looking at the absolute and relative densities of the categories. The empirical results of this study do not back up such arbitrary weighting schemes. For this reason, Ocean's distance metric empirically performs better than the existing ones.

6.2. Comparison of the clustering algorithms in terms of clustering procedure

In this section, we theoretically validate the relative quality of clustering strategies used by different clustering algorithms, starting with the pair of ADD_clustering and Ocean. Among all the clustering algorithms compared, the performance gap between Ocean and ADD_clustering is interesting as they are the closest in terms of technical approach. More specifically, both are fully automatic density-based clustering algorithms. ADD_clustering performs clustering in three stages; (1) initial centre and radius formation, (2) centre and radius updating, and (3) cluster final adjustment), whereas Ocean performs clustering in two stages (initial cluster formation phase and merging phase). The clustering process is guided by eight rules in ADD_clustering and two rules in Ocean. Now it may be asked: How come two such theoretically similar algorithms can show such a different performance on the real data? Our study reveals three major weaknesses in ADD_clustering. First, ADD_clustering calculates the density of a cluster as the inverse of the cumulative distance between points within the cluster. When the algorithm splits the data space into partitions to detect regions with different densities, sometimes one or more regions are formed by the process that contains all identical data points. These regions are problematic for the technique. This is because as the cumulative distance between the data points within those regions becomes zero, the denominator of the density expression is zero. If the denominator is zero, the fraction has no meaning or is considered undefined. As a result, ADD_clustering fails to produce clusters on 12 out of 41 datasets (error messages appear), which has contributed to its lower mean accuracy. Our proposed algorithm Ocean is robust against this problem because of its first rule.

Second, ADD_clustering applies a moving window average difference operation to detect the edge between two adjacent clusters that adversely affects its ability to detect small clusters within large clusters. This weakness has been reflected in its empirical performance.

Third, ADD_clustering (just as other existing algorithms) typically assumes that a majority of the members of a cluster are located at the middle and as the edge approaches the density of members gets thinner and thinner. ADD_clustering uses this assumption for detecting the boundary of each cluster. Their performances are bound to decrease where the distribution within a cluster is non-Gaussian. Ocean, on the other hand, does not make such assumptions regarding the distributions of members in cluster. It assumes that if the ratio of densities of two adjacent regions is 2 or greater, they belong to two different clusters; otherwise they belong to the same cluster. It is necessary to understand that every clustering algorithm relies on assumptions to perform clustering. We believe Ocean's assumption is more logical compared to the assumptions existing algorithms make. A ratio of 2 or above suggests that both numerator and denominator are significantly different from each other, whereas a ratio of 1 (or below 2) suggests they are (roughly) equal. A comparison with experimental results suggests the validity of the Ocean's assumption.

We attribute the inferior performance of the KMC-KMC(HCA) and KMC(PSO) to the fact that it uses the intra-cluster compactness and inter-cluster separateness scheme (DB index) cluster detection. The findings of our study suggest that DB index favours clusters with single unique pattern. As a result, they often fail to assign a new instance to a cluster. In contrast, Ocean does not create clusters of identical patterns due to its first rule. Most of the time Ocean can assign a new instance to a cluster. The question may arise as to Why a new instance needs to be located within the distance between the centre and its furthest member for being assigned to the cluster? Instead, we could assign it to its nearest cluster. If a point can be assigned to a cluster only because it is closer to the point than other existing clusters, then we can skip the clustering task entirely. Every unique pattern can be considered as a cluster. When a new event hits, we can assign it to its nearest cluster. However, the purpose of clustering is more than that. Clusters must have a definite area so that it can be confidently known whether the new event belongs to an existing cluster.

SOM (PSO & HCA) is found to be the second-best performing algorithm. We believe its performance is good considering its disadvantages when compared to other algorithms. SOM has multiple free parameters (dimensions, layer topology function, initial neighbourhood size, etc.) that need to be tuned for a specific noise environment. SOM needs a search algorithm for the discovery of optimal parameters. Users face two choices: local search algorithm (i.e., HCA) or global search algorithm (i.e., PSO). Any local search algorithm gets trapped into the nearest local minimum. Global search algorithms, on the other hand, have various escape mechanisms to avoid local minima. However, these algorithms themselves have many free parameters to tune. Again, there is no hard and fast rule regarding how to optimise these parameters. We left most of the parameters of PSO at default. Search algorithms require an objective function to measure how good a candidate solution is. In clustering problems, the standard practice is to use inter-cluster and intra-cluster distance metrics as objective functions; however, these metrics are found to be highly problematic in practice. They tend to favour clusters of identical patterns. In addition, existing global search algorithms are computationally demanding but we have time limit.

The situation is much more complicated with DBSCAN. DBSCAN is designed in such a way that domain knowledge is required to optimise its free parameters. However, it is not clear what is meant by domain knowledge. Guessing the threshold number of neighbour points to each point a priori is challenging. A common practice is to set the parameters to the total number of attributes plus one. Our empirical results suggest that this practice is not very effective. It often fails to create clusters with a single dominant class. In contrast, Ocean does not have any free parameters that need tuning.

7. Conclusion

In this paper, we propose a novel clustering algorithm named Ocean. The novelties of Ocean lie in three aspects. First, Ocean converts non-continuous data to continuous data by replacing the categories with their relative densities. Second, it only labels a region as cluster if it has more than one unique pattern. Third, if and only if the ratio of densities of two adjacent clusters is lower than 2, it merges the clusters. Ocean does not have any free parameters that need tuning.

In terms of limitations of the proposed technique: the threshold ratio of densities of two adjacent clusters has be set to 2, and it might be better to optimise the threshold ratio for each specific problem. It is however challenging to optimise variables for any problem that lacks input instances with output labels. Hence, all clustering algorithms make their own specific assumptions and rules in order to segment the landscape of interest into clusters. A ratio of 1 (or below 2) means that the numerator roughly equals the denominator (we place the larger density in the numerator). We believe our assumption is reasonable and more realistic, given that existing algorithms make assumptions regarding the distribution of points within a set of clusters.

In categorical attributes, Ocean replaces the categories with the relative densities. Ocean arbitrarily changes the values of relative densities when two or more categories of the same attribute have the same relative density in order to keep them distinguishable from each other. It is important because clustering is made based on the density in multidimensional space and not based on the density in uni-dimensional space. However, we would like to bring attention to the fact that Ocean induces almost negligible changes in the relative densities.

If all categories of the attributes have exactly the same relative densities, the absolute and relative densities (or frequencies) of the categories do not provide any useful information on distances between categories. All existing distance metrics (including that in Ocean) will suffer severe performance degradation. The best strategy, under the circumstances, is to keep categories distinct from one another for the multidimensional space which contains the clustering solutions. This is the strategy that Ocean follows.

The number of clusters that each of the algorithms (including Ocean) captures may appear to be too high if we take into consideration the actual numbers of classes in the datasets. We would like to stress the fact that clustering and classification are two completely different tasks. The goal of clustering is to partition a heterogeneous multidimensional dataset into homogeneous regions (i.e., clusters). The aim of classification is to select the clusters that belong to a certain class.

If dataset contains many redundant or irrelevant features, Ocean does not optimise the feature subset. We attribute the poor performance of existing algorithms to their free parameters that require optimisation. We apply the algorithms on the UCI datasets [33] that are commonly used to evaluate feature subset selection algorithms, which indicates that these datasets contain redundant or irrelevant attributes. Ocean and SOM(PSO) perform very well on these datasets without optimising the feature subsets.

Finally, Ocean does not perform any clustering on the datasets that contain less than four unique patterns. Although this may seem like a clear limitation, we come to the conclusion that such datasets are too simple to be considered seriously for a clustering problem.

8. Availability

Annotated source code for Ocean is available at the following address:`https://selene.hud.ac.uk/scomsp2/docs/OceanSourceCode.pdf`. The code was written for execution in the Matlab environment and Iffat Gheyas should be contacted for all enquires.

References

[1] D. J. Bora, A. K. Gupta, A comparative study between fuzzy clustering algorithm and hard clustering algorithm, CoRR abs/1404.6059 (2014). URL: http://arxiv.org/abs/1404.6059. arXiv:1404.6059.

[2] S. Sharma, P. Sharma, Comparative study on supervised and unsupervised fuzzy approach for image classification, International Journal of Engineering Research 3 (2014).

[3] P.-E. Danielsson, Euclidean distance mapping, Computer Graphics and image processing 14 (1980) 227–248.

[4] P. M. Bhagat, P. S. Halgaonkar, V. M. Wadhai, Review of clustering algorithm for categorical data, International Journal of Engineering and Advanced Technology 3 (2013).

[5] D. J. Weller-Fahy, B. J. Borghetti, A. A. Sodemann, A survey of distance and similarity measures used within network intrusion anomaly detection, IEEE Communications Surveys & Tutorials 17 (2014) 70–91.

[6] S. Boriah, V. Chandola, V. Kumar, Similarity measures for categorical data: A comparative evaluation, in: Proceedings of the 2008 SIAM international conference on data mining, SIAM, 2008, pp. 243–254.

[7] G. Tzortzis, A. Likas, Kernel-based weighted multi-view clustering, in: 2012 IEEE 12th international conference on data mining, IEEE, 2012, pp. 675–684.

[8] A. T. Wilson, P. A. Chew, Term weighting schemes for latent dirichlet allocation, in: human language technologies: The 2010 annual conference of the North American Chapter of the Association for Computational Linguistics, Association for Computational Linguistics, 2010, pp. 465–473.

[9] Y. Shim, J. Chung, I.-C. Choi, A comparison study of cluster validity indices using a nonhierarchical clustering algorithm, in: International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06), volume 1, IEEE, 2005, pp. 199–204.

[10] R. Xu, D. Wunsch, Survey of clustering algorithms, IEEE Transactions on neural networks 16 (2005) 645–678.

[11] D. Sisodia, L. Singh, S. Sisodia, K. Saxena, Clustering techniques: a brief survey of different clustering algorithms, International Journal of Latest Trends in Engineering and Technology (IJLTET) 1 (2012) 82–87.

[12] N. Sharma, N. Gaud, K-modes clustering algorithm for categorical data, International Journal of Computer Applications 127 (2015) 46.

[13] J. Vesanto, E. Alhoniemi, et al., Clustering of the self-organizing map, IEEE Transactions on neural networks 11 (2000) 586–600.

[14] D. Alahakoon, S. K. Halgamuge, B. Srinivasan, Dynamic self-organizing maps with controlled growth for knowledge discovery, IEEE Transactions on neural networks 11 (2000) 601–614.

[15] A. Rodriguez, A. Laio, Clustering by fast search and find of density peaks, Science 344 (2014) 1492–1496.

[16] K. Khan, S. U. Rehman, K. Aziz, S. Fong, S. Sarasvady, Dbscan: Past, present and future, in: The fifth international conference on the applications of digital information and web technologies (ICADIWT 2014), IEEE, 2014, pp. 232–238.

[17] A. Bryant, K. Cios, Rnn-dbscan: A density-based clustering algorithm using reverse nearest neighbor density estimates, IEEE Transactions on Knowledge and Data Engineering 30 (2018) 1109–1121.

[18] H. Tandon, M. Gupta, Enhancement in dbscan using sift technique, International Journal of Advanced Research in Computer Science and Software Engineering 6 (2016) 935–941.

[19] J. Sander, M. Ester, H.-P. Kriegel, X. Xu, Density-based clustering in spatial databases: The algorithm gdbscan and its applications, Data mining and knowledge discovery 2 (1998) 169–194.

[20] R. J. Campello, D. Moulavi, J. Sander, Density-based clustering based on hierarchical density estimates, in: Pacific-Asia conference on knowledge discovery and data mining, Springer, 2013, pp. 160–172.

[21] J. Chen, P. Yu, A domain adaptive density clustering algorithm for data with varying density distribution, IEEE Transactions on Knowledge and Data Engineering 1 (2019) 1–12. doi:10.1109/TKDE.2019.2954133.

[22] S. Guha, R. Rastogi, K. Shim, Rock: A robust clustering algorithm for categorical attributes, Information systems 25 (2000) 345–366.

[23] R. J. Little, D. B. Rubin, Statistical analysis with missing data, volume 793, John Wiley & Sons, 2019.

[24] R. Pan, T. Yang, J. Cao, K. Lu, Z. Zhang, Missing data imputation by k nearest neighbours based on grey relational structure and mutual information, Applied Intelligence 43 (2015) 614–632.

[25] P. Jonsson, C. Wohlin, An evaluation of k-nearest neighbour imputation using likert data, in: 10th International Symposium on Software Metrics, 2004. Proceedings., IEEE, 2004, pp. 108–118.

[26] I. C. Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, Information processing letters 85 (2003) 317–325.

[27] Q. Bai, Analysis of particle swarm optimization algorithm, Computer and information science 3 (2010) 180.

[28] S. B. Kjær, Evaluation of the "hill climbing" and the "incremental conductance" maximum power point trackers for photovoltaic power systems, IEEE Transactions on Energy Conversion 27 (2012) 922–929.

[29] A. Lim, R. Brian, F. Xiao, Integrated genetic algorithm with hill climbing for bandwidth minimization problem, in: Genetic and Evolutionary Computation Conference, Springer, 2003, pp. 1594–1595.

[30] S. Sidney, Nonparametric statistics for the behavioral sciences, The Journal of Nervous and Mental Disease 125 (1957) 497.

[31] S. Ben-David, M. Ackerman, Measures of clustering quality: A working set of axioms for clustering, in: Advances in neural information processing systems, 2009, pp. 121–128.

[32] D. L. Davies, D. W. Bouldin, A cluster separation measure, IEEE transactions on pattern analysis and machine intelligence (1979) 224–227.

[33] I. University of California, Uci machine learning repository, 2019. URL: https://archive.ics.uci.edu/ml/index.php, accessed on 18-Dec-2019.

9. Appendix

9.1. Performance of Clustering Algorithms on Individual Datasets

Table 4: Evaluation of the Performance of Clustering Algorithms (C1: performance criterion1—average classification accuracy on 10-fold cross-validation, C2: performance criterion 2: the percentage of the total number of actual classes discovered—is measured on the entire dataset, C3: Performance Criterion 3: the runtime (in seconds) of the algorithm(measured on the entire dataset) and NC is the number of clusters found (measured on the entire dataset)

Dataset	**Ocean**	**SOM (PSO)**	**SOM (HCA)**	**KMC (HCA)**	**KMC (PSO)**	**DBSCAN**	**ADD_Clustering**
Abalone	C1: 99% C2:100% C3:1668 NC:1234	C1: 94% C2: 50% C3:36000 NC:1589	C1: 94% C2: 50% C3:18000 NC:1826	C1: 94% C2: 50% C3: 2906 NC:2316	C1: 94% C2: 50% C3:36000 NC:2689	C1: 94% C2: 50% C3:18000 NC:464	C1: 94% C2: 50% C3:18000 NC:1017
Absenteeism	C1: 96% C2:100% C3:13.63 NC:217	C1: 85% C2:100% C3:36000 NC:362	C1: 76% C2: 100% C3: 5215 NC:397	C1: 70% C2: 100% C3:140.5 NC:365	C1: 64% C2: 100% C3:3505 NC:430	C1: 64% C2:100% C3:543 NC:33	Error Message
Adult	C1: 97% C2:100% C3:18000 NC:549	Out of Memory	Out of Memory	C1: 59% C2: 100% C3:18000 NC:1187	Out of Memory	C1: 56% C2:100% C3:18000 NC:3489	Error Message
Annealing	C1: 98% C2: 100% C3:11 NC:275	C1: 90% C2: 100% C3:30870 NC:294	C1: 68% C2: 80% C3:2705 NC:317	C1: 65% C2: 60% C3:18.5 NC:403	C1: 48% C2: 60% C3:671 NC:442	C1: 55% C2: 40% C3:438 NC:19	C1: 57% C2: 60% C3:1739 NC:269
Balance Scale	C1: 95% C2:100% C3: 11 NC:290	C1: 80% C2: 100% C3:25620 NC:295	C1: 64% C2: 100% C3:2708 NC:299	C1: 58% C2: 100% C3:142 NC:350	C1: 51% C2: 100% C3:3481 NC:378	C1: 43% C2: 66% C3:438.27 NC:156	C1: 44% C2: 66% C3:1813 NC:197
Breast Cancer Wisconsin (original)	C1: 99% C2: 100% C3:4 NC:135	C1: 91% C2: 100% C3:29190 NC:183	C1: 78% C2: 100% C3:1315 NC:154	C1: 72% C2: 100% C3:158 NC:191	C1: 67% C2: 100% C3:7075 NC:228	C1: 69% C2: 100% C3:307 NC:69	Error Message
Breast Cancer Wisconsin (prognistic)	C1: 94% C2:100% C3:1.77 NC:59	C1: 88% C2: 100% C3:4002 NC:89	C1: 83% C2: 100% C3:443 NC:82	C1: 81% C2: 100% C3:20.65 NC:90	C1: 76% C2: 100% C3:510 NC:105	C1:76% C2: 100% C3:70.02 NC:6	C1:77% C2: 100% C3:392 NC:56
Car Evaluation	C1: 98% C2:100% C3:131 NC:504	C1: 91% C2: 75% C3:36000 NC:607	C1: 80% C2: 50% C3:18000 NC:666	C1: 72% C2: 50% C3:552.5 NC:680	C1: 60% C2: 50% C3:15368 NC: 699	C1: 70% C2: 25% C3:4917 NC: 288	C1: 75% C2: 50% C3:18000 NC:488
Cervical Cancer	C1: 100% C2: 100% C3:11 NC:265	C1: 97% C2: 100% C3:30590 NC:326	C1: 94% C2: 50% C3:2581 NC:409	C1: 94% C2: 50% C3:26.5 NC:434	C1: 94% C2: 50% C3: 853 NC:488	C1: 94% C2: 50% C3:438 NC:23	C1: 94% C2: 50% C3:1808 NC:158
Congressional Voting	C1: 99% C2: 100% C3:1.67 NC:105	C1: 92% C2: 100% C3:15545 NC:124	C1: 80% C2: 100% C3:523 NC:149	C1: 70% C2: 100% C3:55.51 NC:191	C1: 63% C2: 100% C3:1301 NC:204	C1: 64% C2: 100% C3:65.44 NC:25	Error Message
Credit Approval	C1: 98% C2: 100% C3: 8 NC:207	C1: 79% C2: 100% C3:19740 NC:288	C1: 62% C2: 100% C3:1944 NC:297	C1: 60% C2: 100% C3:112 NC:295	C1: 57% C2: 100% C3:2730 NC:354	C1: 55% C2: 50% C3:318 NC: 43	C1: 57% C2: 100% C3:1316 NC:189

Table 4: Evaluation of the Performance of Clustering Algorithms (C1: performance criterion1—average classification accuracy on 10-fold cross-validation, C2: performance criterion 2: the percentage of the total number of actual classes discovered—is measured on the entire dataset, C3: Performance Criterion 3: the runtime (in seconds) of the algorithm(measured on the entire dataset) and NC is the number of clusters found (measured on the entire dataset)

Dataset	**Ocean**	**SOM (PSO)**	**SOM (HCA)**	**KMC (HCA)**	**KMC (PSO)**	**DBSCAN**	**ADD_Clustering**
Cryptotherapy	C1: 95% C2: 100% C3: 5 NC: 33	C1: 90% C2: 100% C3: 7770 NC: 42	C1: 71% C2: 100% C3: 1157 NC: 46	C1: 66% C2: 100% C3: 7.5 NC: 45	C1: 54% C2: 100% C3: 284 NC:57	C1: 55% C2: 100% C3:199.58 NC: 11	C1: 62% C2: 100% C3:712 NC: 29
Cylinder Bands	C1: 97% C2: 100% C3:7.94 NC:171	C1: 78% C2: 100% C3:19232 NC:202	C1: 70% C2: 100% C3:1834 NC:256	C1: 67% C2: 100% C3:57.91 NC:279	C1:61% C2: 100% C3:1498 NC:284	C1: 62% C2: 100% C3:316 NC: 12	Error Message
Dermatology	C1: 98% C2: 100% C3:3.59 NC: 99	C1: 91% C2: 75% C3:5837 NC:187	C1:83% C2: 75% C3:943 NC:240	C1: 55% C2: 75% C3:37.39 NC:167	C1: 47% C2: 75% C3:932 NC:280	C1: 59% C2: 25% C3:142 NC:10	C1: 77% C2: 75% C3:625 NC: 84
Ecoli	C1: 93% C2:100% C3: 2 NC:106	C1: 80% C2: 63% C3:6020 NC:190	C1: 69% C2: 50% C3:530 NC:202	C1: 56% C2: 37% C3: 39 NC:186	C1: 48% C2: 37% C3:933 NC:203	C1: 59% C2: 37% C3:79 NC:37	Error Message
Fertility	C1: 99% C2:100% C3: 0.5 NC: 35	C1: 93% C2: 100% C3:1575 NC: 44	C1: 89% C2: 100% C3:140 NC: 62	C1: 90% C2: 100% C3:16.75 NC: 58	C1: 89% C2: 100% C3:392 NC: 79	C1: 88% C2: 50% C3:19.54 NC: 9	C1: 88% C2: 50% C3: 22 NC: 27
Flag	C1: 92% C2:100% C3:1.29 NC: 61	C1: 70% C2: 71% C3:3436 NC: 86	C1: 51% C2: 71% C3:312 NC:104	C1: 49% C2: 25% C3:7.94 NC:100	C1: 37% C2: 25% C3:210 NC:116	C1: 50% C2: 25% C3:50.84 NC: 6	C1: 52% C2: 25% C3:262 NC: 42
Glass Identification	C1: 96% C2:100% C3:2.29 NC: 68	C1: 84% C2: 100% C3:4486 NC:115	C1: 74% C2: 100% C3:579 NC:132	C1: 70% C2: 67% C3:37.44 NC:124	C1: 46% C2: 67% C3: 904 NC:148	C1: 71% C2: 67% C3:90.78 NC:19	C1: 74% C2: 67% C3:424 NC: 60
Haberman's Survival	C1: 97% C2: 100% C3:1.75 NC: 98	C1: 90% C2: 100% C3:5653 NC:175	C1: 78% C2: 100% C3:484 NC:193	C1: 63% C2: 100% C3:68.93 NC:118	C1: 59% C2: 100% C3:1692 NC:193	C1: 61% C2: 100% C3:72.63 NC: 76	Error Message
Hayes-Roth	C1: 92% C2: 100% C3:0.39 NC: 26	C1: 78% C2: 100% C3:6873 NC: 35	C1: 66% C2: 100% C3:172 NC: 39	C1: 60% C2: 100% C3:14.94 NC: 37	C1: 52% C2: 100% C3:1979 NC: 49	C1: 62% C2: 67% C3:86.59 NC: 27	C1: 64% C2: 67% C3:382 NC:20
Hepatitis	C1: 99% C2: 100% C3: 1 NC:51	C1: 92% C2: 100% C3:2660 NC:69	C1: 88% C2: 100% C3:236 NC:80	C1: 86% C2: 100% C3:10.5 NC:82	C1: 85% C2: 50% C3:262 NC:105	C1: 85% C2: 50% C3:39.35 NC: 7	C1: 85% C2: 50% C3:85 NC:43
Horse Colic	C1: 97% C2:100% C3: 4.17 NC:83	C1: 82% C2: 100% C3:11590 NC:145	C1: 73% C2: 100% C3:993 NC:169	C1: 60% C2: 100% C3:111 NC:130	C1: 53% C2: 100% C3:2694 NC:187	C1: 57% C2: 100% C3:166 NC:13	C1: 59% C2: 100% C3:756 NC:75

Table 4: Evaluation of the Performance of Clustering Algorithms (C1: performance criterion1—average classification accuracy on 10-fold cross-validation, C2: performance criterion 2: the percentage of the total number of actual classes discovered—is measured on the entire dataset, C3: Performance Criterion 3: the runtime (in seconds) of the algorithm(measured on the entire dataset) and NC is the number of clusters found (measured on the entire dataset)

Dataset	**Ocean**	**SOM (PSO)**	**SOM (HCA)**	**KMC (HCA)**	**KMC (PSO)**	**DBSCAN**	**ADD_Clustering**
Iris	C1: 99% C2:100% C3:0.81 NC: 49	C1: 96% C2: 100% C3:3991 NC:95	C1: 89% C2: 100% C3:209 NC: 86	C1: 58% C2: 100% C3:31.77 NC:104	C1: 39% C2: 100% C3:1366 NC:108	C1: 58% C2: 100% C3:59.22 NC: 5	C1: 86% C2: 100% C3:181 NC: 3
Lung Cancer	C1: 93% C2:100% C3:0.48 NC: 9	C1: 90% C2: 100% C3:706 NC: 16	C1: 68% C2: 100% C3:119 NC: 20	C1: 60% C2: 100% C3: 3 NC:14	C1: 51% C2: 100% C3:255 NC: 21	C1: 40% C2: 33% C3:10.72 NC: 1	C1: 54% C2: 100% C3:59 NC: 7
Lymphography	C1: 95% C2: 100% C3:1.13 NC: 53	C1: 77% C2: 50% C3:1941 NC: 72	C1: 65% C2: 50% C3:279 NC: 83	C1: 44% C2: 50% C3:3.69 NC: 75	C1: 36% C2: 50% C3:110 NC: 101	C1: 48% C2: 50% C3:44.58 NC: 7	C1: 43% C2: 50% C3:234 NC: 47
Mushroom	C1: 100% C2: 100% C3:1800 NC: 478	C1: 93% C2: 100% C3:36000 NC: 2098	C1: 89% C2: 100% C3:18000 NC:3210	C1: 89% C2: 100% C3:18000 NC:3579	C1: 88% C2: 100% C3:36000 NC:3881	C1: 87% C2: 50% C3:18000 NC:354	C1: 87% C2: 50% C3:18000 NC:751
Nursery	C1: 98% C2:100% C3:18000 NC: 690	C1: 89% C2: 80% C3:36000 NC:866	C1: 72% C2: 80% C3:18000 NC:1210	C1: 67% C2: 80% C3:18000 NC:2489	C1: 49% C2: 60% C3:36000 NC:2076	C1: 53% C2: 60% C3:18000 NC:1440	C1: 58% C2: 60% C3:18000 NC: 475
Parkinsons	C1: 96% C2:100% C3: 3 NC: 62	C1: 84% C2: 100% C3:7480 NC: 74	C1: 80% C2: 100% C3:758 NC: 66	C1: 79% C2: 100% C3: 47 NC:77	C1: 76% C2: 100% C3:1138 NC: 91	C1: 76% C2: 100% C3:119 NC: 8	Error Message
Poker Hand	C1: 98% C2:100% C3:18000 NC: 1710	C1: 87% C2: 100% C3:36000 NC:2453	C1: 73% C2: 100% C3:18000 NC:4410	C1: 66% C2: 40% C3:18000 NC:6313	C1: 52% C2: 40% C3:36000 NC:7821	C1: 51% C2: 20% C3:18000 NC:85418	C1: 55% C2: 30% C3:18000 NC:986
Post-Operative Patient	C1: 93% C2:100% C3:0.56 NC: 24	C1: 81% C2: 67% C3:1663 NC: 39	C1: 75% C2: 67% C3:131 NC: 44	C1: 73% C2: 67% C3:10.84 NC: 40	C1: 60% C2: 67% C3:259 NC: 56	C1: 71% C2: 33% C3:21.98 NC: 10	Error Message
Primary Tumour	C1: 97% C2:100% C3: 2 NC:96	C1: 72% C2: 63% C3:13230 NC:96	C1: 61% C2: 29% C3:573 NC: 114	C1: 55% C2: 29% C3: 40 NC:151	C1: 47% C2: 29% C3:956 NC:125	C1: 54% C2: 29% C3:78.85 NC: 18	Error Message
Soybean (large)	C1: 98% C2:100% C3:1.97 NC:105	C1: 89% C2: 84% C3:5626 NC:134	C1: 71% C2: 52% C3:462 NC:157	C1: 47% C2: 26% C3:25.96 NC:165	C1: 32% C2: 26% C3:635 NC:185	C1: 39% C2: 21% C3:77.73 NC:8	C1: 45% C2: 21% C3:344 NC:98
Spambase	C1: 99% C2:100% C3: 1233 NC: 1067	C1: 86% C2: 100% C3:36000 NC:2246	C1: 72% C2: 100% C3:18000 NC:3031	C1: 69% C2: 100% C3:2125 NC:2427	C1: 61% C2: 100% C3:36000 NC:3451	C1: 60% C2: 50% C3:9996 NC: 79	Error Message

Table 4: Evaluation of the Performance of Clustering Algorithms (C1: performance criterion1—average classification accuracy on 10-fold cross-validation, C2: performance criterion 2: the percentage of the total number of actual classes discovered—is measured on the entire dataset, C3: Performance Criterion 3: the runtime (in seconds) of the algorithm(measured on the entire dataset) and NC is the number of clusters found (measured on the entire dataset)

Dataset	**Ocean**	**SOM (PSO)**	**SOM (HCA)**	**KMC (HCA)**	**KMC (PSO)**	**DBSCAN**	**ADD_Clustering**
Statlog (Shuttle)	C1: 100% C2:100% C3:18000 NC: 367	C1: 93% C2: 71% C3:36000 NC: 548	C1: 88% C2: 57% C3:18000 NC:739	C1: 83% C2: 29% C3:18000 NC:785	C1: 81% C2: 29% C3:36000 NC:624	C1: 80% C2: 14% C3:18000 NC:5100	C1: 84% C2: 29% C3:18000 NC:178
Teaching Assistant Evaluation	C1: 93% C2:100% C3:0.44 NC: 33	C1: 82% C2: 100% C3:6037 NC: 37	C1: 76% C2: 100% C3:157 NC: 48	C1: 78% C2: 100% C3:16.97 NC:64	C1: 53% C2: 100% C3:548 NC: 80	C1: 60% C2: 100% C3:23.66 NC:25	C1: 67% C2: 100% C3:168 NC:29
Tic-tac-toe Endgram	C1: 96% C2:100% C3: 20 NC:302	C1: 85% C2: 100% C3:36000 NC:401	C1: 71% C2: 100% C3:4666 NC:456	C1: 70% C2: 100% C3: 60 NC:494	C1: 66% C2: 100% C3:1820 NC:519	C1: 65% C2: 50% C3:798 NC:95	C1: 65% C2: 50% C3:3247 NC:271
Website Phishing	C1: 94% C2:100% C3: 12 NC:227	C1: 82% C2: 100% C3:36000 NC:285	C1: 70% C2: 100% C3:3441 NC:342	C1: 65% C2: 100% C3:477 NC:391	C1: 60% C2: 100% C3:11735 NC:420	C1: 59% C2: 100% C3:589 NC:123	Error Message
Wilt	C1: 100% C2:100% C3:18000 NC:165	C1: 98% C2: 50% C3:36000 NC:211	C1: 98% C2: 50% C3:18000 NC:260	C1: 98% C2: 50% C3:18000 NC:279	C1: 98% C2: 50% C3:36000 NC:224	C1: 98% C2: 50% C3:18000 NC:480	C1: 98% C2: 50% C3:18000 NC:180
Wine	C1: 100% C2:100% C3:1.27 NC:54	C1: 96% C2: 100% C3:4807 NC:68	C1: 84% C2: 100% C3:307 NC:82	C1: 79% C2: 100% C3:17.91 NC:97	C1: 49% C2: 100% C3:436 NC:101	C1: 58% C2: 100% C3:50.08 NC:12	C1: 60% C2: 100% C3:158 NC: 39
Yeast	C1: 92% C2:100% C3: 47 NC: 440	C1: 86% C2: 90% C3:36000 NC: 579	C1: 72% C2: 90% C3:11091 NC:625	C1: 69% C2: 60% C3:199 NC:678	C1: 51% C2: 50% C3:5585 NC:740	C1: 52% C2: 20% C3:1877 NC:164	C1: 61% C2: 40% C3:7845 NC:417
Zoo	C1: 100% C2:100% C3:0.37 NC: 18	C1: 91% C2: 71% C3:4114 NC: 25	C1: 74% C2: 57% C3:98.47 NC: 44	C1: 70% C2:57% C3:14.34 NC: 35	C1: 42% C2: 57% C3:726 NC:44	C1: 55% C2: 42% C3:31.56 NC: 5	C1: 60% C2: 57% C3:120 NC: 8

9.2. Performance of Clustering Algorithms on Individual Datasets

Table 5: Comparison of conventional dissimilarity metrics

Dataset	**OF**	**Eskin**	**IOF**	**Overlap**	**Lin**	**Goodall**
Abalone	94%	94%	94%	94%	94%	94%
Absenteeism at work	64%	63%	63%	63%	63%	63%
Adult	52%	51%	51%	51%	51%	51%
Annealing	76%	76%	76%	76%	76%	76%
Balance Scale	48%	47%	46%	44%	43%	33%

Table 5: Comparison of conventional dissimilarity metrics

Dataset	**OF**	**Eskin**	**IOF**	**Overlap**	**Lin**	**Goodall**
Breast Cancer Wisconsin (original)	66%	65%	65%	65%	65%	65%
Breast Cancer Wisconsin (prognostic)	76%	76%	76%	76%	76%	76%
Car Evaluation	70%	70%	70%	70%	70%	70%
Cervical Cancer	94%	94%	94%	94%	94%	94%
Congressional Voting	62%	61%	61%	61%	61%	61%
Credit Approval	56%	56%	56%	55%	55%	55%
Cryptotherapy	56%	55%	55%	53%	53%	53%
Cylinder Bands	60%	60%	60%	60%	60%	60%
Dernatology	49%	47%	46%	43%	42%	41%
Ecoli	50%	46%	45%	44%	43%	41%
Fertility	88%	88%	88%	88%	88%	88%
Flags	18%	15%	15%	14%	12%	12%
Glass Identification	54%	49%	49%	48%	47%	45%
Habermen's survival	58%	58%	58%	58%	58%	58%
Hayes-Roth	46%	42%	38%	37%	32%	28%
Hepatitis	85%	85%	85%	85%	85%	85%
Horse Colic	54%	52%	52%	52%	52%	52%
Iris	51%	48%	47%	44%	42%	41%
Lung Cancer	39%	37%	34%	33%	31%	30%
Lymphography	29%	25%	22%	21%	20%	17%
Mushroom	87%	87%	87%	87%	87%	87%
Nursery	46%	44%	43%	42%	39%	38%
Parkinsons	75%	75%	75%	75%	75%	75%
Poker Hand	51%	32%	32%	31%	26%	23%
Post-operative Patient	71%	71%	71%	71%	71%	71%
Primary Tumour	30%	26%	25%	23%	21%	16%
Soybean(large)	27%	24%	23%	20%	16%	15%
Spambase	61%	60%	60%	60%	60%	60%
Statlog(shuttle)	80%	80%	80%	80%	80%	80%
Teaching Assistant Evaluation	55%	53%	52%	51%	50%	50%
Tic-tac-toe Endgram	65%	65%	65%	65%	65%	65%

Table 5: Comparison of conventional dissimilarity metrics

Dataset	**OF**	**Eskin**	**IOF**	**Overlap**	**Lin**	**Goodall**
Twitter Dataset	56%	53%	52%	50%	48%	33%
Website Phishing	60%	59%	59%	59%	59%	59%
Wilt	98%	98%	98%	98%	98%	98%
Wine	44%	38%	36%	35%	33%	21%
Yeast	48%	43%	41%	40%	37%	26%