Int. J. Appl. Math. Comput. Sci., 2018, Vol. 28, No. 3, 545–557
DOI: 10.2478/amcs-2018-0042

LINGUISTICALLY DEFINED CLUSTERING OF DATA

JACEK M. LESKI [a,*], MARIAN P. KOTAS [b]

[a]Institute of Medical Technology & Equipment ITAM, Roosevelta 118, 41-800 Zabrze, Poland
e-mail: jacek.leski@polsl.pl

[b]Institute of Electronics
Silesian University of Technology, Akademicka 16, 44-100 Gliwice, Poland

This paper introduces a method of data clustering that is based on linguistically specified rules, similar to those applied by a human visually fulfilling a task. The method endeavors to follow these remarkable capabilities of intelligent beings. Even for most complicated data patterns a human is capable of accomplishing the clustering process using relatively simple rules. His/her way of clustering is a sequential search for new structures in the data and new prototypes with the use of the following linguistic rule: *search for prototypes in regions of extremely high data densities and immensely far from the previously found ones*. Then, after this search has been completed, the respective data have to be assigned to any of the clusters whose nuclei (prototypes) have been found. A human again uses a simple linguistic rule: *data from regions with similar densities, which are located exceedingly close to each other, should belong to the same cluster*. The goal of this work is to prove experimentally that such simple linguistic rules can result in a clustering method that is competitive with the most effective methods known from the literature on the subject. A linguistic formulation of a validity index for determination of the number of clusters is also presented. Finally, an extensive experimental analysis of benchmark datasets is performed to demonstrate the validity of the clustering approach introduced. Its competitiveness with the state-of-the-art solutions is also shown.

Keywords: clustering, possibility theory, linguistic rules, data analysis.

1. Introduction

"All the real knowledge which we possess, depends on methods by which we distinguish the similar from the dissimilar" reverberates in numerous publications the famous sentence of the Swedish scholar C. Linnaeus. Indeed, one of the fundamental faculties of living creatures, and particularly of the reasoning ones, is the ability to cluster similar objects, cases or events for their later classification and naming. The idea of connecting similar objects forms a basis for creation of natural languages which help us to remember and exchange ideas concerning various types of objects and phenomena. In a natural language, groups of objects with some specific features are named using different nouns. This kind of clustering enables us (or highly simplifies) to communicate, thereby making progress. In science, clustering is one of the fundamental approaches to data processing. It plays an important role in many engineering fields, such as pattern recognition, computer vision, machine learning, image analysis, communication, knowledge discovery, data mining and so on (Everitt *et al*., 2011; Király *et al*., 2016; Leski and Kotas, 2015; Leski, 2016; Nguyen and Choi, 2015; Pancerz *et al*., 2015; Zok *et al*., 2015). It also enables discoveries and development in other branches of science, such as medicine, economy, psychiatry, biology, marketing, education, archeology, chemistry and many more.

2. Related works

Owing to their widespread and diverse applications, many clustering methods have been developed and practically implemented. Because of their multitude, it would be arduous to review all of them; however, we can divide them generally into the following types (Everitt *et al*., 2011; Zaki and Meira, 2014): representative-based clustering (including k-means, fuzzy c-means and their kernel versions, and expectation-maximization

*Corresponding author

clustering), hierarchical clustering, density-based clustering (including DBSCAN (Ester *et al*., 1996) and kernel density estimation (Fukunaga and Hostetler, 1975; Comaniciu and Meer, 2002) as well as spectral and graph clustering. An overview of these methods can be found in the classical monographs by Duda *et al*. (2001), Ripley (1996), Tou and Gonzalez (1974), or Webb (1999), and in the paper by Jain *et al*. (1999). Obviously, there exist methods that can be assigned to more than one among the above-mentioned types of clustering, e.g., to representative-based and hierarchical one (Pedrycz *et al*., 2015), or to density-based and graph clustering (Rodriguez and Laio, 2014).

The latter proposition is particularly interesting because it tries to solve a very fundamental problem encountered in clustering of multidimensional data whose properties cannot easily be scrutinized. The problem is that even if for a given set of data there is a method, among a multitude of methods, that would be able to cluster them correctly, it is difficult (if at least possible) to choose it. It inspires to the endeavor to develop a method that would be able to cluster the data of different properties (with clusters of similar or different sizes, of the same or different shapes, being well separated as well as overlapping).

This study was also inspired by the work of Leski (2015), which set forth a method of finding initial prototypes for fuzzy clustering that lie on the boundary of the convex hull of the clustered data. Such an initialization of prototypes makes convergence of the calculations faster and lessens possibilities of their getting stuck in a local minimum of the criterion function. A simple algorithm to find such initial prototypes is based on linguistically defined rules, which are as follows (Leski, 2015): First the mean value over the dataset is calculated, and the datum that is most distant from this mean is chosen as the first initial prototype. Next the distances between this prototype and each datum are calculated. Then a datum as far as possible from the previous prototype AND as far as possible from the mean value is chosen as the next prototype. Each new prototype should be as far as possible from the previously chosen prototypes AND from the mean value. The algebraic product as the t-norm modeling the AND connection is used.

In the work of Rodriguez and Laio (2014) the search for the clusters centers was also defined linguistically: "Cluster centers should be of the highest density among their nearest neighbors AND should by of relatively large distances from the points of higher density." Since the theory of fuzzy sets was not applied in that work, the AND connection was realized graphically with the participation of a human.

It seems that for a human a more natural linguistic definition of cluster centers (prototypes) could be as follows: The first prototype should be placed in an area with the highest density. The successive prototypes should be placed in areas with a very, very high density AND should have very, very large distances from the previous prototypes. The search should be ended when the condition has been very weakly satisfied (or, in other words, when its degree of truth has become very low). In such an approach, the prototypes are searched successively and the condition can also be used to determine their number. In this study, however, we propose to apply a linguistically specified validity index.

The goal of this work is threefold. Firstly, it introduces a new linguistically defined clustering method, based on the fuzzy sets and possibility theories; secondly, it investigates the clustering quality for benchmark datasets; and thirdly, it compares the method proposed to the state-of-the-art solutions, well-known from the literature, with respect to their clustering quality.

The remainder of this paper is organized as follows: Section 3 shows simple linguistically specified rules for iterative search for cluster prototypes/medoids. Section 4 shows an algorithm assigning data to the previously established prototypes. It is also based on a simple linguistically defined rule. Section 5 presents a new linguistic validity index which performs an assessment of the cluster consistency. In Section 6, a comparative study of the proposed method with some reference ones, known from the literature and regarded as classical, is made. The investigations are performed and discussed for real-world and synthetic benchmark datasets. Finally, conclusions are drawn in Section 7.

3. Linguistically defined clustering: Finding prototypes

The proposed clustering method divides a set of N observations (input vectors) $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \in \mathbb{R}^n$ into c clusters denoted by $\Omega_1, \Omega_2, \ldots, \Omega_c$. We assume that the number of clusters is not known and must be estimated based on the data. Additionally, we assume that the cluster prototypes must be selected from the data clustered, which means that they are medoids. For clusters Ω_i, $i = 1, 2, \ldots, c$ the prototypes (medoids) are denoted by $\mathbf{x}_{\mathcal{M}(i)}$, where $\mathcal{M} : \{1, 2, \ldots, c\} \to \{1, 2, \ldots, N\}$ is a medoid index. Thus, $\mathcal{M}(k) = j$ will mean that the j-th datum is the medoid of the k-th cluster. Denote by $R_{ij} = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^\top (\mathbf{x}_i - \mathbf{x}_j)}$ the distance between the i-th and the j-th datum. Of course, many different definitions of the distance can be applied here, e.g. the Hamming one, but in this work the most commonly applied Euclidean one is used.

Since the proposed method is based on a search for prototypes (or, more precisely, medoids) in high density areas, we must begin the method description by defining the notion of the data density. Analogously to the visual search for dense regions by a human, define the density

around the i-th datum on the basis of its distances to the other data. These distances are aggregated. Of the greatest impact are the data located nearest to the considered i-th datum. If the aggregated distance from the i-th datum to the remaining ones is small, we can conclude that the data density in this region is high. As the aggregation operator we can apply the generalized ordered mean (Yager and Filev, 1999)

$$\begin{aligned} \uplus\left(R_{i1}, R_{i2}, \cdots, R_{iN}\right) &= \biguplus_{j=1}^{N} R_{ij} \\ &\triangleq \Big(\sum_{j=1}^{N} \beta_j \left(R_{i[j]}\right)^{\alpha}\Big)^{\frac{1}{\alpha}}, \end{aligned} \tag{1}$$

where the rank-ordered arguments satisfy the conditions

$$R_{i[1]} \leq R_{i[2]} \leq \cdots \leq R_{i[N]}. \tag{2}$$

By a suitable choice of weights $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_N$ and $\alpha \in \mathbb{R} \setminus \{0\}$, we can obtain many interesting methods for data density assessment. For each $\mathbf{x}_i$,

$$\Delta\left(\mathbf{x}_i\right) \triangleq 1 / \biguplus_{j=1}^{N} R_{ij}. \tag{3}$$

In this work one of the simplest choices was applied, i.e., $\alpha = 1$ and

$$\beta_j = \begin{cases} 1/\xi & \text{for } j = 1, 2, \ldots, \xi, \\ 0 & \text{for } j = \xi+1, \xi+2, \ldots, N, \end{cases} \tag{4}$$

where $\xi > 0$ is a parameter. In other words, the density is estimated as the reciprocal of the average distance to the nearest $\xi - 1$ neighbors (note that in the first position, after distances sorting, we will always have the distance to the datum of the same index, $R_{ii} = 0$). Additionally, we normalize the so defined densities

$$\bar{\Delta}\left(\mathbf{x}_i\right) = \frac{\Delta\left(\mathbf{x}_i\right)}{\max_{j=1}^{N} \Delta\left(\mathbf{x}_j\right)}, \tag{5}$$

and then the membership of the i-th datum to 'dense regions' can be defined as

$$\mu_{\text{dense}}\left(\mathbf{x}_i\right) = \mathcal{S}\left(\bar{\Delta}\left(\mathbf{x}_i\right); 0, 1\right), \tag{6}$$

where $\mathcal{S}(x; a, b)$ denotes the $\mathcal{S}$-type membership function with parameters a and b

$$\begin{aligned} &\mathcal{S}(x; a, b) \\ &= \begin{cases} 0 & \text{for } x \leq a, \\ 2\left(\frac{x-a}{b-a}\right)^2 & \text{for } a < x \leq \frac{a+b}{2}, \\ 1 - 2\left(\frac{x-b}{b-a}\right)^2 & \text{for } \frac{a+b}{2} < x \leq b, \\ 1 & \text{for } x > b. \end{cases} \end{aligned} \tag{7}$$

In (6) we can apply many other types of the membership functions, e.g., trapezoidal or sigmoidal; however, in this work we will only apply the functions $\mathcal{S}$, $\mathcal{Z}$ and Π to pay homage to the founder of fuzzy set theory (Zadeh, 1965) and possibilistic theory (Zadeh, 1978), which are fundamental for this paper. In (6) the linguistic value 'dense' was determined on the linguistic variable 'normalized density.' For clustering we will need a definition of the term 'very, very dense', which we can easily obtain using the modifier rule 'very', well-known from possibility theory (Zadeh, 1978)

$$\mu_{\text{very\_very\_dense}}\left(\mathbf{x}_i\right) = \left(\mu_{\text{dense}}\left(\mathbf{x}_i\right)\right)^4. \tag{8}$$

Note that selecting the membership function (7) and the clustering rule with the linguistic term 'very, very', the algebraic product as the t-norm (13) and the values of the parameters used in (11), (14) and (17) are reasonable, but arbitrary. The aim of the work is to show that such a reasonable choice leads to excellent clustering effects. A separate problem, not addressed here due to the volume of the work, is the optimization of the clustering procedure with respect to the above-mentioned functions and parameters. The proposal for further work in this direction will be given in the conclusions. Let us now define the linguistic term 'very, very large distance.' In our application, we refer to the distance from a set of the previously found cluster prototypes. If we denote by γ the current number of the found prototypes, a distance from the following set of points $\{\mathbf{x}_i \,|i \in \underline{\mathcal{M}}_{\gamma}\}$ is considered, where $\underline{\mathcal{M}}_{\gamma} = \{\mathcal{M}(1), \mathcal{M}(2), \ldots, \mathcal{M}(\gamma)\}$ denotes the set of indexes to the medoids that have already been found. By checking the term 'very, very large distance', we want to preclude selecting a new prototype near the previously found ones. Thus, in fact, by the distance from the set of the prototypes we mean the distance to the nearest of them. Therefore, we rather regard the linguistic term 'very, very large distance from the nearest prototype' which can be defined as

$$D_i \triangleq \min_{j=1}^{\gamma} \bar{R}_{i\mathcal{M}(j)}, \tag{9}$$

where $\bar{R}_{ij}$ is the normalized distance,

$$\bar{R}_{ij} = \frac{R_{ij}}{\max_{i=1}^{N} \max_{j=1}^{N} R_{ij}}. \tag{10}$$

The large distance from the prototypes is now defined with the use of the S-type membership function

$$\mu_{\text{far}}\left(\mathbf{x}_i\right) = \mathcal{S}\left(D_i; 0, 1\right). \tag{11}$$

Then, just as before, we obtain the term 'very, very large distance'

$$\mu_{\text{very\_very\_far}}\left(\mathbf{x}_i\right) = \left(\mu_{\text{far}}\left(\mathbf{x}_i\right)\right)^4. \tag{12}$$

In the linguistic definition of the proposed clustering method, we use a compound statement 'very, very large density AND very, very large distance from the previously found prototypes.' On the basis of possibility theory, we model conjunction AND with the use of the t-norm; one of the basic t-norms, i.e., the algebraic product, will be applied

$$\mu_{\text{very\_very\_far\_AND\_very\_very\_dense}}(\mathbf{x}_i) = \mu_{\text{very\_very\_far}}(\mathbf{x}_i) \cdot \mu_{\text{very\_very\_dense}}(\mathbf{x}_i). \quad (13)$$

Based on above ideas and considerations, we are ready to present an iterative algorithm for finding the prototypes (medoids) which is defined by simple linguistic rules.

Algorithm 1. Finding prototypes.

Require: R_{ij}, $\mu_{\text{very\_very\_dense}}(\mathbf{x}_i)$, c.
Ensure: Indexes of the prototypes.
1: $\mathcal{M}(1) := \arg\max_{i=1}^{N} \mu_{\text{very\_very\_dense}}(\mathbf{x}_i)$,
2: $\gamma := 1$, $\underline{\mathcal{M}}_\gamma := \{\mathcal{M}(1)\}$,
3: **while** $\gamma < c$ **do**
4: **for** the current prototype indexes $\underline{\mathcal{M}}_\gamma$ **and** $i = 1, 2, \ldots, N$ **do**
5: Calculate $\mu_{\text{very\_very\_far}}(\mathbf{x}_i)$,
6: Calculate $\mu_{\text{very\_very\_far\_AND\_very\_very\_dense}}(\mathbf{x}_i)$,
7: **end for**
8: $\gamma := \gamma + 1$,
9: $\mathcal{M}(\gamma) := \arg\max_{i=1}^{N} \mu_{\text{very\_very\_far\_AND\_very\_very\_dense}}(\mathbf{x}_i)$,
10: $\underline{\mathcal{M}}_\gamma := \underline{\mathcal{M}}_{\gamma-1} \cup \{\mathcal{M}(\gamma)\}$,
11: **end while**
12: **return** $\underline{\mathcal{M}}_c$ {Returns a set of prototype indexes.}

At Point 5 of Algorithm 1 we determine the 'distance from the nearest prototype' using (9). Owing to the associative property of the 'min' operator, calculations of D_i's can be performed iteratively. The final number of clusters c can be known *a priori* or established by examining the values of a validity index. In Section 4, a linguistically defined validity index is proposed.

4. Linguistically defined clustering: Assignment of data to clusters

Applying Algorithm 1 to data clustered, we select prototypes $\mathbf{x}_{\mathcal{M}(i)}$ whose indexes are gathered in set $\underline{\mathcal{M}}_c = \{\mathcal{M}(1), \mathcal{M}(2), \ldots, \mathcal{M}(c)\}$. The next operation, necessary to complete the clustering task, is the assignment of individual data points to one of the respective clusters, represented by medoids. The easiest solution would be to assign the data on the basis of the nearest distance. In such a case, however, as in the fuzzy c-means method, we would favor clusters in the form of hyperballs of the same radius. To avoid such limitations, we applied a different solution, analogous to that used by a human, based on a reasonable linguistic term. We assumed that an intelligent being would act according to the following statement: 'very, very near data with similar densities should be assigned to the same cluster.' Apart from that, this propagation of membership starts from prototypes, i.e., data with highest memberships, towards data with lower one.

Beginning with the results produced by Algorithm 1, which form the nuclei of the clusters, the successive data will be assigned iteratively, one after another, with the order dependent on the degree with which they satisfy the term 'very, very near distance AND similar density'. Thus, in much the same way as before, we have to specify the linguistic variables 'distance' and 'density difference', and for them define subsequently linguistic terms 'near', 'very, very near' and 'similar density'.

The normalized distance between the i-th and j-th data pieces is equal to $\bar{R}_{ij}$ (see (10)). Thus the value of the term 'near' referring to this distance is defined with the use of the S-type (or Z-type) membership function in the following way:

$$\begin{aligned}\mu_{\text{near}}(\mathbf{x}_i, \mathbf{x}_j) &= 1 - \mu_{\text{far}}(\mathbf{x}_i, \mathbf{x}_j) \\ &= 1 - \mathcal{S}\left(\bar{R}_{ij}; 0, 1\right) \\ &= \mathcal{Z}\left(\bar{R}_{ij}; 0, 1\right).\end{aligned} \quad (14)$$

Specifically, $\mu_{\text{near}}(\mathbf{x}_i, \mathbf{x}_j)$ is the membership function of a two-dimensional fuzzy set, or simply the fuzzy relation. Of course, we can obtain the term 'very, very near' as before,

$$\mu_{\text{very\_very\_near}}(\mathbf{x}_i, \mathbf{x}_j) = \left(\mu_{\text{near}}(\mathbf{x}_i, \mathbf{x}_j)\right)^4. \quad (15)$$

Let us now handle the variable 'normalized density difference' for the i-th and j-th data pieces

$$\partial\bar{\Delta}(\mathbf{x}_i, \mathbf{x}_j) = \bar{\Delta}(\mathbf{x}_i) - \bar{\Delta}(\mathbf{x}_j), \quad (16)$$

which takes values from interval $[-1, +1]$. The linguistic value 'small density difference' is obtained using the Π-type membership function

$$\mu_{\text{small\_density\_difference}}(\mathbf{x}_i, \mathbf{x}_j) = \Pi\left(\partial\bar{\Delta}(\mathbf{x}_i, \mathbf{x}_j); -0.5, -0.2, 0.02, 0.25\right), \quad (17)$$

where

$$\Pi(x; a_1, b_1, a_2, b_2) = \begin{cases} \mathcal{S}(x; a_1, b_1) & \text{for} \quad x \le b_1, \\ 1 & \text{for} \quad b_1 < x \le a_2, \\ 1 - \mathcal{S}(x; a_2, b_2) & \text{for} \quad x > a_2, \end{cases} \quad (18)$$

and a_1, b_1, a_2, b_2 are parameters. In the above definition, after scrutinizing the values of parameters a_1, b_1, a_2, b_2,

we can be puzzled by the asymmetry of the linguistic value 'small density difference'! Such a membership function corresponds to a greater extent to the term 'small density difference but usually negative.' And so it is as it should be! The method of the search for prototypes does not assure their finding in the regions of the highest density (they are rather found in the regions of the highest product of the density and the distance from the previously found prototypes). Thus, in the stage of the data assignment to clusters, we have to allow for the possibility to assigning data points with a similar but higher density. This will be exceptional, however, and we will usually search for points with a similar but lower density.

In a linguistic definition used for data assignment to clusters, there is a compound term 'very, very near distance AND small density difference.' Just as before, we model conjunction AND using the algebraic product as the t-norm

$$\begin{aligned}&\mu_{\text{very\_very\_near\_AND\_small\_density\_difference}}\left(\mathbf{x}_i,\mathbf{x}_j\right)\\&=\mu_{\text{very\_very\_near}}\left(\mathbf{x}_i,\mathbf{x}_j\right)\\&\quad\cdot\mu_{\text{small\_density\_difference}}\left(\mathbf{x}_i,\mathbf{x}_j\right).\end{aligned}\tag{19}$$

After the above presentation of the introductory notions, we are ready for the description of the algorithm assigning data to clusters. Define the function indicating the cluster to which the i-th datum belongs as $\mathcal{C}:\{1,2,\ldots,N\}\to\{0,1,2,\ldots,c\}$, where c is the number of clusters; zero means that the datum has not been assigned to any group yet. Thus $\mathcal{C}(i)=k$ will mean that the i-th datum belongs to the k-th cluster, i.e., $\mathbf{x}_i\in\Omega_k$. Obviously, we begin with the assignment of the cluster prototypes (medoids). Each of them is a nucleus of a new cluster: $\mathcal{C}(\mathcal{M}(i))=i$ for $i=1,2,\ldots,c$. Define the function $\mathcal{W}(i)\in[0,1]$ determining the degree of membership of the i-th datum to the cluster indicated by $\mathcal{C}(i)$. Of course, function $\mathcal{W}(i)$ takes on a value of one for each cluster prototype: $\mathcal{W}(\mathcal{M}(i))=1$ for $i=1,2,\ldots,c$. It will also be helpful to define a set of indexes of the data that are not assigned to the clusters yet $\mathcal{R}=\{i\,|\mathcal{C}(i)=0,i=1,2,\ldots,N\}$, and of its complement, i.e., the set of indexes of the data already assigned to clusters $\overline{\mathcal{R}}=\{1,2,\ldots,N\}\setminus\mathcal{R}$. Of course, at the beginning, when we only have the clusters nuclei (prototypes/medoids), $\mathcal{R}=\{1,2,\ldots,N\}\setminus\underline{\mathcal{M}_c}$ and $\overline{\mathcal{R}}=\underline{\mathcal{M}_c}$.

After clusters nuclei have been formed on the basis of the prototypes, we assume the following recurrent rule for modification of $\mathcal{C}(i)$ and $\mathcal{W}(i)$. We search for data indexes for which

$$\mathcal{W}(i)\wedge\mu_{\text{very\_very\_near\_AND\_small\_density\_difference}}\left(\mathbf{x}_i,\mathbf{x}_j\right)$$

attains a maximum, subject to the condition that the first argument has already been assigned to a cluster and the second has not. Thus we set

$$\begin{aligned}(k,\ell)=\arg\max_{i\in\overline{\mathcal{R}},\,j\in\mathcal{R}}\;&[\mathcal{W}(i)\\&\wedge\mu_{\text{very\_very\_near\_AND\_small\_density\_difference}}\left(\mathbf{x}_i,\mathbf{x}_j\right)],\end{aligned}\tag{20}$$

where $\wedge$ denotes the minimum operation. After finding such a pair of indexes (k,ℓ) the cluster containing the k-th datum is extended with the ℓ-th one. Thus

$$\mathcal{C}(\ell)=\mathcal{C}(k).\tag{21}$$

The degree of membership for the new datum is determined as

$$\begin{aligned}&\mathcal{W}(\ell)\\&=\mathcal{W}(k)\wedge\mu_{\text{very\_very\_near\_AND\_small\_density\_difference}}\left(\mathbf{x}_k,\mathbf{x}_\ell\right).\end{aligned}\tag{22}$$

Of course, after such an assignment the sets of indexes are updated as follows:

$$\begin{cases}\mathcal{R}\leftarrow\mathcal{R}\setminus\{\ell\},\\ \overline{\mathcal{R}}\leftarrow\overline{\mathcal{R}}\cup\{\ell\}.\end{cases}\tag{23}$$

We continue assigning the data until $\mathcal{R}$ is an empty set. After completing the assignment process, for each datum we can determine a path (chain) along which the cluster was extended, beginning at the cluster prototype and ending at the datum considered. Note that according to (22) the degree of the data membership to a cluster propagates on the basis of the 'weakest link' of this chain. Altogether, data assignment to clusters can be expressed as Algorithm 2.

Algorithm 2. Data assignment to clusters.

Require: $\mu_{\text{very\_very\_near\_AND\_small\_density\_difference}}\left(\mathbf{x}_i,\mathbf{x}_j\right)$.
Ensure: Assignment/Membership of data to clusters.

1: $\mathcal{C}(i):=0,\mathcal{W}(i):=0$ for $i=1,2,\ldots,N$,
2: $\mathcal{C}(\mathcal{M}(i)):=i$ and $\mathcal{W}(\mathcal{M}(i)):=1$ for $i=1,2,\ldots,c$,
3: $\mathcal{R}:=\{1,2,\ldots,N\}\setminus\{\mathcal{M}(1),\mathcal{M}(2),\ldots,\mathcal{M}(c)\}$, $\overline{\mathcal{R}}:=\{\mathcal{M}(1),\mathcal{M}(2),\ldots,\mathcal{M}(c)\}$,
4: **while** $\mathcal{R}\neq\emptyset$ **do**
5: $(k,\ell):=\arg\max_{i\in\overline{\mathcal{R}},\,j\in\mathcal{R}}\;[\mathcal{W}(i)\wedge\mu_{\text{very\_very\_near\_AND\_small\_density\_difference}}\left(\mathbf{x}_i,\mathbf{x}_j\right)]$,
6: $\mathcal{C}(\ell):=\mathcal{C}(k)$,
7: $\mathcal{W}(\ell):=\mathcal{W}(k)\wedge\mu_{\text{very\_very\_near\_AND\_small\_density\_difference}}\left(\mathbf{x}_k,\mathbf{x}_\ell\right)$,
8: $\mathcal{R}:=\mathcal{R}\setminus\{\ell\},\overline{\mathcal{R}}:=\overline{\mathcal{R}}\cup\{\ell\}$,
9: **end while**
10: **return** $\mathcal{C}$, $\mathcal{W}$ {Returns data assignment $\mathcal{C}$ and memberships $\mathcal{W}$.}

Algorithm 2 transforms the set of prototypes $\{\mathcal{M}(1),\mathcal{M}(2),\ldots,\mathcal{M}(c)\}$ into a function indicating

a class to which each datum belongs, that is to say, $\mathcal{C} : \{1,2,\cdots,N\} \rightarrow \{0,1,2,\ldots,c\}$ and the degree of this membership, $\mathcal{W} : \{1,2,\ldots,N\} \rightarrow [0,1]$. Obviously, the proposed method of the linguistically defined clustering (LDC) first executes Algorithm 1 and then Algorithm 2. The computational complexity of the R matrix calculation is $\mathcal{O}(0.5N^2)$, of the density calculation using quicksort is $\mathcal{O}(N^2 \log N)$, and of the determination of the linguistic terms 'very, very dense', 'very, very far', 'very, very near and small density difference' is $\mathcal{O}(N)$ and $\mathcal{O}(cN)$, $\mathcal{O}(0.5N^2)$, respectively. Finally, the computational complexity of the procedure for data assignment to clusters is $\mathcal{O}(0.5(N-1)N)$, Thus, the total computational complexity of the LDC algorithm is $\mathcal{O}((1.5N + N\log N)N)$.

5. Linguistically defined validity index

Application of the LDC algorithm, described in the previous two sections, leads to clustering data $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \in \mathbb{R}^n$ into c clusters, denoted as $\Omega_1, \Omega_2, \cdots, \Omega_c$ and represented by medoids $\{\mathcal{M}(1), \mathcal{M}(2), \ldots, \mathcal{M}(c)\}$. We also obtain a function indicating a class to which each datum belongs $\mathcal{C} : \{1,2,\ldots,N\} \rightarrow \{0,1,2,\ldots,c\}$ and the degree of this membership $\mathcal{W} : \{1,2,\ldots,N\} \rightarrow [0,1]$. The aim of this section is to present a method for estimation of the appropriate number of clusters. The method uses a classic approach based on a validity index, with the exception that the index will be based on a linguistic description. It will be used to assess the quality of the obtained data partition on the basis of internal consistency of the clusters. This linguistic specification can be formulated in the following way: *if any of the clusters contains an isolated data region AND the data cardinality in this cluster is big AND the density within this region is high, then the cluster internal consistency is low and, consequently, bigger number of clusters should be assumed*.

In order to facilitate a mathematical description, we introduce the following notation for the set of indexes of the data that belong to the j-th cluster: $\mathcal{I}_j = \{i \,|\mathcal{C}(i) = j\}$. Obviously, $\Omega_j = \{\mathbf{x}_i \,|i \in \mathcal{I}_j\}$. Investigation if the j-th data cluster is internally consistent will be based on the analysis of the distribution of memberships $\mathcal{W}(i)$ for $i \in \mathcal{I}_j$ and results from the following observation. According to (22) a membership propagates with the use of the operation of 'minimum'. When a cluster consists of two isolated sub-clusters, for the first one, containing the cluster medoid, the memberships are high; however, while propagating to the second, they abruptly decrease (because of the lower linguistic value of the term $\mu_{\text{very\_very\_near\_AND\_small\_density\_difference}}$ of the between sub-clusters transition points). After this decrease, the membership remains constant for the most closely distributed points of the second sub-cluster (related with greater linguistic values of the term $\mu_{\text{very\_very\_near\_AND\_small\_density\_difference}}$). It is thus advisable to find the most frequent value of $\mathcal{W}(i)$ for $i \in \mathcal{I}_j$. The number of its occurrences is a measure of the size of the isolated region (sub-cluster).

More formally, for $\ell \in \mathcal{I}_j$ define the quantity

$$\mathcal{N}_\ell = \frac{\text{Card}\,\{i \in \mathcal{I}_j \,|\mathcal{W}(i) = \mathcal{W}(\ell)\}}{\min\limits_{1\le k\le c} \text{Card}\,\{\mathcal{I}_k\}}, \tag{24}$$

which is the number of occurrences of the membership value $\mathcal{W}(\ell)$ referred to the cardinality of the least frequent cluster. The linguistic value 'large region' is defined as follows:

$$\mu_{\text{large\_region}}(\ell) = \mathcal{S}(\mathcal{N}_\ell; 0.1, 1). \tag{25}$$

Hence we assume that if the cardinality of the isolated data region (sub-cluster) is less than 10% of the cardinality of the least frequent cluster, we should not create a new cluster. If this value is exceeded, such an operation should be considered but other features of the isolated region should be taken into account, i.e., the degree of isolation and the density of this region.

The degree of isolation is also assessed based on the $\mathcal{W}(i)$ values of the data cluster. Observe that if we want to assess the degree of isolation between the data region represented by the found value $\mathcal{W}(\ell)$ (and the smaller ones) and the region of higher memberships, we can find the minimal difference between those higher memberships ($\mathcal{W}(i) > \mathcal{W}(\ell)$) and $\mathcal{W}(\ell)$. Thus we define

$$\mathcal{T}_\ell = \frac{\min\limits_{\{k\in\mathcal{I}_j|\mathcal{W}(k)>\mathcal{W}(\ell)\}} \mathcal{W}(k) - \mathcal{W}(\ell)}{\max\limits_{1\le i\le N} \mathcal{W}(i) - \min\limits_{1\le i\le N} \mathcal{W}(i)}, \tag{26}$$

where obviously the search is carried out for memberships $\mathcal{W}(i)$ of the j-th cluster. Normalization to the greatest difference between memberships from the whole dataset was also applied. The linguistic value 'well separated region' is defined as

$$\mu_{\text{well\_separated\_region}}(\ell) = \mathcal{S}(\mathcal{T}_\ell; 0, 1). \tag{27}$$

What still remains is to determine the density of the isolated region. To this end, we search for the maximal value of $\mu_{\text{very\_very\_dense}}(\mathbf{x}_k)$ for the data that belong to the isolated region ($\mathcal{W}(k) \le \mathcal{W}(\ell)$) of the j-th cluster

$$\mathcal{D}_\ell = \max_{\{k\in\mathcal{I}_j|\mathcal{W}(k)\le\mathcal{W}(\ell)\}} \mu_{\text{very\_very\_dense}}(\mathbf{x}_k), \tag{28}$$

We are now prepared for a numerical assessment of the quality of the j-th cluster (denoted as $\mathcal{Q}_j$), by combining information on the presence of an isolated data region with a high degree of isolation AND with high

cardinality AND with high density. Again, we use the algebraic product to model conjunction AND,

$$\mathcal{Q}_j = \max_{\ell \in \mathcal{I}_j} \mathcal{N}_\ell \cdot \mathcal{T}_\ell \cdot \mathcal{D}_\ell. \tag{29}$$

The greater $\mathcal{Q}_j$, the worse the quality of the j-th cluster because it contains better isolated data regions of higher density.

If we form a partition into c clusters, the quality of the whole partition $\mathcal{U}(c)$ can be assessed based on the quality of the 'worst' cluster,

$$\mathcal{U}(c) = \max_{1 \le j \le c} \mathcal{Q}_j. \tag{30}$$

Finally, index $\mathcal{U}(c)$ is used as other indexes of the same type, i.e., we form partitions into clusters whose number varies within the assumed range and we regard as the best the partition into a number $c^{\#}$ of clusters, for which $\mathcal{U}(c^{\#})$ is smallest. If there are several values of c for which the same, smallest value of $\mathcal{U}(c)$ was achieved, we usually choose the smallest c (the simplest solution is best).

6. Numerical experiments and discussion

All experiments were performed on NTT Intel® Core™ i5-3570 CPU @ 3.40 GHz with 6 GB RAM, running Windows 8 and Matlab™ 7.5 environment. The Matlab implementation of the 'clustering by fast search and find of density peaks (CFSFDP) method (Rodriguez and Laio, 2014) was obtained (`https://people.sissa.it/~laio/Research/Research.php`). All algorithms introduced in the paper were implemented as Matlab m-files, too. For the experiments, 8 well-known benchmark datasets were used, obtained from the SIPU (School of Computing, University of Eastern Finland) repository (`http://cs.uef.fi/sipu/datasets`), where numerous data sets, applied by various authors to investigate clustering algorithms, have been gathered:

1. *Aggregation* (788 data points, containing 7 clusters with different shapes and sizes, both well separated and overlapping ones) (Gionis *et al*., 2007).

2. *Compound* (399 data points, containing 6 clusters with different shapes, sizes and densities, both well separated and overlapping ones) (Zahn, 1971).

3. *D31* (3100 data points, containing 31 clusters with circular shapes, similar sizes, and different degrees of overlapping) (Veenman *et al*., 2002).

4. *Flame* (240 data points, containing 2 overlapping clusters with different sizes and shapes) (Fu and Medico, 2007).

5. *Jain's data* (373 data points, containing 2 clusters with similar crescent shapes but different sizes and densities) (Jain and Law, 2005).

6. *Pathbased* (300 data points, containing 3 overlapping clusters with different shapes and sizes) (Chang and Yeung, 2008).

7. *R15* (600 data points, containing 15 mostly well separated clusters with circular shapes and similar sizes) (Veenman *et al*., 2002).

8. *Spiral* (312 data points, containing 3 well separated clusters with spiral shapes) (Chang and Yeung, 2008).

These datasets are easily available and their primary feature is a large variety of shapes, cardinalities and sizes of the clusters, and of the degree of overlap and of the number of clusters. Therefore, it is an excellent set to compare the proposed method with the other methods known from the literature with respect to their efficiency of clustering. Only a few among those methods can work well on such a diverse set of data. All these datasets are depicted in Fig.1. For each dataset, information is provided on the true data partition. This allowed us to assess the clustering quality with the use of the measures comparing the obtained data partition with the true one. Such measures are usually categorized into set-matching or pair-counting ones. In this study representatives of both of these types have been applied, i.e., the purity measure and the Jaccard coefficient. The purity measure can be regarded as a fraction of correctly clustered data. The Jaccard coefficient focuses on pairwise agreement between partitions. For each possible pair of data, the Jaccard coefficient evaluates how similarly the two partitions treat them. The closer these measures to unity, the closer the obtained data partition to the true one. The upper bound is equal to the one which corresponds to a perfect match between the partitions.

The purpose of the first experiment is to choose a proper value of parameter ξ which influences data density estimation (1)–(4). To this end, we performed the clustering of the described datasets with ξ varying from 3 to 15. Clustering quality was assessed with the use of the purity and the Jaccard coefficient. For both of these measures, in Fig. 2 we presented the minimal and the mean of their values obtained for the respective datasets (as functions of parameter ξ).

Analyzing these plots, we can notice that for a wide range of ξ (from 7 to 11) the mean values of the purity index are similar. For the mean values of the Jaccard coefficient, the plot is flat in a much narrower range, i.e., from 8 to 9. At the same time, the both indexes achieved the highest of their minimal values for $\xi = 9$. Hence this value was chosen for further experiments.

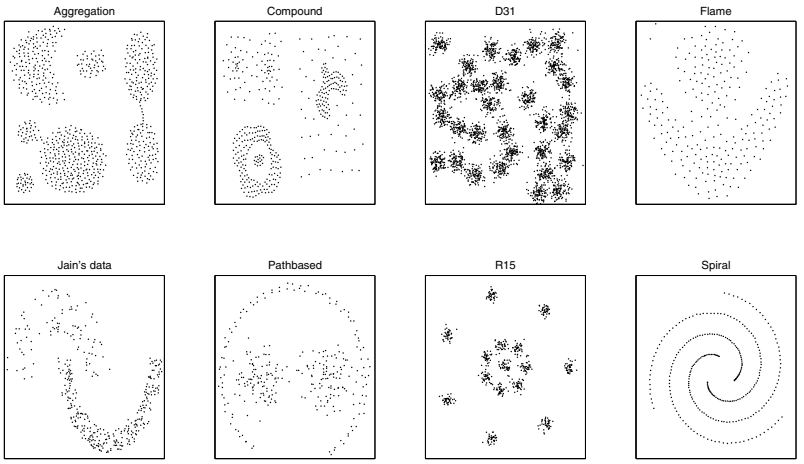


Fig. 1. Distribution of points in the test datasets.

Table 1 shows the values of the purity index, the Jaccard coefficient and the execution time obtained for the method proposed in the paper (LDC) in an experiment on all the datasets described. In these experiments, we assumed that the true number of clusters was known. In the final part of this section, the problem of determining the number of clusters will be tested. Results from Table 1 justify an opinion that the LDC algorithm copes effectively with all data. In the worst case (of the *Compound* dataset) we obtained the purity index equal to 0.87. However, even this slight decrease in the index value can, to some extent, be defended if we pay attention to the particular properties of this dataset (see Fig. 1). One can notice that the dataset contains one cluster with a much lower density (with respect to the densities of the other clusters). Even a human being, while analyzing these data, wonders if this low density cluster should not be merged with one of the other clusters! A similar value of the purity index was obtained for the *Pathbased* dataset. In this case, a cluster resembling a segment of a circle contains a discernible break and, besides, it overlaps with the other two clusters. For three datasets (*Flame*, *Jain's data* and *Spiral*) we have obtained partitions exactly consistent with the true ones. For the other datasets, the results obtained are almost faultless (with the purity above 0.96).

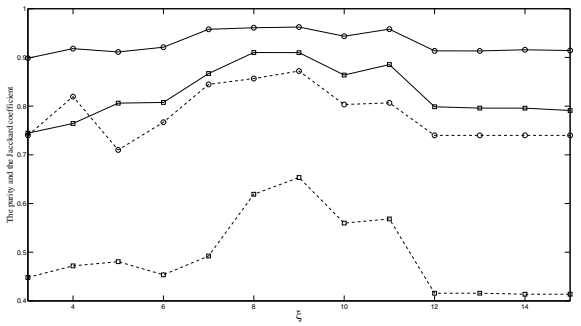


Fig. 2. Mean values (solid lines) and the minimal ones (dashed lines) of the purity (marked by circles) and the Jaccard coefficient (squares) as functions of parameter ξ. The values were obtained in tests on all datasets from Fig. 1.

Of course, to make a reliable appraisal of these results, we have to compare them with the analogous results of other algorithms. To this end, the following methods were selected: 'clustering by fast search and find of density peaks' (CFSFDP) (Rodriguez and Laio, 2014), agglomerative hierarchical clustering with either a single or a complete linkage, fuzzy c-means (FCM) clustering, k-means clustering (Jain *et al*., 1999), DBSCAN (Ester *et al*., 1996), and the mean-shift (MS) method (Comaniciu and Meer, 2002). For the CFSFDP method, the number of groups is selected interactively by the user. In experiments, the number of groups was selected to be equal to the true value. Table 2 shows the values of the purity index, the Jaccard coefficient and the execution time obtained for the CFSFDP method (for all the datasets described). Scrutinizing these results, we can notice that for three datasets (*Aggregation*, *Flame* and *Spiral*) we have obtained a faultless partition into clusters. For two datasets (*D31* and *Pathbased*) the results are not satisfactory (the values of purity are equal to 0.58 and 0.77, respectively). For the remaining datasets, the values of this index exceed 0.83. Comparing the execution times, we can see that they are usually longer for the CFSFDP method, with the exception for the *D31* dataset, for which CFSFDP was slightly faster than LDC.

Table 3 shows the purity index, the Jaccard coefficient and the execution time obtained for all the described datasets by the method of hierarchical clustering with a single linkage. We can see that in this case only for the *Spiral* dataset the partition is exactly the

Table 1. Purity, the Jaccard coefficient and the execution times obtained for the proposed method.

Name	Purity	Jaccard	Time [s]
Aggregation	0.9937	0.9809	0.86
Compound	0.8722	0.8165	0.12
D31	0.9671	0.8797	56.67
Flame	1.0000	1.0000	0.04
Jain's data	1.0000	1.0000	0.10
Pathbased	0.8800	0.6534	0.06
R15	0.9867	0.9487	0.34
Spiral	1.0000	1.0000	0.07

Table 2. Purity, the Jaccard coefficient and the execution times obtained for CFSFDP.

Name	Purity	Jaccard	Time [s]
Aggregation	1.0000	1.0000	1.98
Compound	0.8321	0.4756	3.31
D31	0.5856	0.3319	52.27
Flame	1.0000	1.0000	2.67
Jain's data	0.8928	0.7135	2.99
Pathbased	0.7333	0.4866	2.30
R15	0.9967	0.9866	4.60
Spiral	1.0000	1.0000	2.35

same as the true one. Quite high values, i.e., the purity index exceeding 0.8, we obtained for two datasets only (*Aggregation* and *Jain's data*). For the remaining five datasets, the results are not satisfactory. On the other hand, the method is competitive with the proposed one with respect to the execution time, which was more than 40 times shorter in the case of the *D31* dataset! Of course, for the hierarchical method, clustering was interrupted for the true number of clusters.

Table 4 shows the values of the purity index, the Jaccard coefficient and the execution time obtained for all the described datasets by the method of hierarchical clustering with complete linkage. We can see that for this method, we achieved high consistency of the obtained partitions with the true ones (the purity index exceeding the value of 0.94) for four datasets (*Aggregation*, *D31*, *Jain's data*, *R15*). Unfortunately for two datasets (*Pathbased* and *Spiral*) the results are very poor (for the latter dataset, the purity index was equal to 0.4). The execution times are comparable to those of the method of hierarchical clustering with a single linkage. In these experiments, the grouping was also interrupted for the true number of clusters.

Table 5 shows the values of the purity index, the Jaccard coefficient and the execution time obtained for all the described datasets by the fuzzy c-means method (for the true number of clusters and for the standard parameter values used by Matlab). For this method the results are very stable, i.e., the purity index belongs always to the range from 0.74 to 0.88. There are only two exceptions: for the *R15* dataset we obtained an almost perfect result (0.99) and for the *Spiral* dataset the worst result of 0.34. The execution times are comparable to those of the method proposed; however, for the *D31* dataset they were 8 times shorter.

Table 6 shows the values of the purity index, the Jaccard coefficient and the execution time obtained for all the described datasets by the k-means method (for the true number of clusters and for the standard parameter values used by Matlab). For this method the results are very similar to those obtained for the fuzzy c-means method (the mean purity index remains almost unchanged), but the execution times are about twice shorter.

Table 7 we can see the same set of values as in Table 6 but obtained by the DBSCAN method (here the minimal number of points considered as a cluster equals 10). For this method the results are rather good, i.e., the purity index belongs always to the range from 0.36 to 0.82. There are only two exceptions: for the *Spiral* dataset we obtained a perfect result (1.00) and for the *D31*

Table 3. Purity, the Jaccard coefficient and the execution times obtained for the method of hierarchical clustering with single linkage.

Name	Purity	Jaccard	Time [s]
Aggregation	0.8274	0.7430	0.28
Compound	0.7469	0.6912	0.14
D31	0.2652	0.1241	1.36
Flame	0.6458	0.5357	0.10
Jain's data	0.8097	0.6553	0.15
Pathbased	0.3733	0.3316	0.12
R15	0.7317	0.4126	0.25
Spiral	1.0000	1.0000	0.12

Table 4. Purity, the Jaccard coefficient and the execution times obtained for the method of hierarchical clustering with complete linkage.

Name	Purity	Jaccard	Time [s]
Aggregation	0.9530	0.7031	0.27
Compound	0.8246	0.7386	0.15
D31	0.9619	0.8626	1.28
Flame	0.6375	0.4425	0.10
Jain's data	0.9464	0.8533	0.14
Pathbased	0.6833	0.4192	0.12
R15	0.9900	0.9606	0.20
Spiral	0.4071	0.2331	0.12

Table 5. Purity, the Jaccard coefficient and the execution times obtained for the fuzzy c-means method.

Name	Purity	Jaccard	Time [s]
Aggregation	0.8807	0.5152	0.70
Compound	0.8321	0.4627	0.33
D31	0.8806	0.6934	7.21
Flame	0.8500	0.6032	0.15
Jain's data	0.7748	0.5218	0.16
Pathbased	0.7433	0.4888	0.21
R15	0.9967	0.9866	0.63
Spiral	0.3397	0.1957	0.29

Table 6. Purity, the Jaccard coefficient and the execution times obtained for the k-means method.

Name	Purity	Jaccard	Time [s]
Aggregation	0.9112	0.6201	0.42
Compound	0.8697	0.7009	0.26
D31	0.8487	0.6581	2.88
Flame	0.8375	0.5822	0.14
Jain's data	0.7828	0.5315	0.18
Pathbased	0.7433	0.4911	0.17
R15	0.8583	0.6554	0.37
Spiral	0.3429	0.1957	0.23

dataset the worst result of 0.19. The execution times are comparable to those of the hierarchical methods.

Table 8 shows the results obtained by the mean-shift method (the Gaussian kernel bandwidth is determined as in the work of Rodriguez and Laio (2014)). For this method the results are very good, i.e., the purity index belongs always to the range from 0.80 to 0.99. Only for the *Spiral* dataset we obtained the value beyond this range; it was the worst result of 0.64. The execution times are comparable to those of the hierarchical methods. At first sight, the LDC algorithm bears a similarity to the clustering algorithms family known from the literature, i.e., to the mean-shift (MS) algorithms (Fukunaga and Hostetler, 1975; Comaniciu and Meer, 2002). Of course, both types of algorithms belong to the same group of the density-based ones. However, after a more in-depth analysis, we can notice some fundamental differences. The mean-shift algorithm iteratively shifts a datum to the nearest local peak of the density function (cluster center). Thus, all the data points near a cluster center converge to it by iterative climbing up the density function. In contrast to it, the LDC algorithm uses the density function separately to find cluster prototypes and to form clusters. Moreover, this formation of clusters runs from the highest densities to the smallest, and from the smallest differences in distance and density to the highest ones. Additionally, in the LDC algorithm the clustering rules are formulated in a linguistic way.

The observations regarding the values of the purity index obtained by the compared clustering methods can to a large extent be applied to the obtained values of the Jaccard coefficient. To facilitate a comprehensive appraisal of the respective methods, a juxtaposition of the minimal and the mean values of both indexes and of the cumulative execution times was made in Table 9. The method proposed (LDC) achieved the greatest of the presented mean values of both the indexes. Also the smallest of the values obtained for different datasets are greatest for this method. As far as the execution times are considered, the method can be regarded as rather time consuming. For instance, its computations are about 6 times longer than those of the FCM method. However, it is significantly less time consuming than the CFSFDP method. Concluding, it seems that a good compromise has been achieved between the contradictory requirements of the efficiency and computational speed of the method developed.

In the above described experiments, we assumed that the true number of clusters is known. Of course the problem of clustering is more complicated if this number is not known and should be established. In real applications of clustering, we usually deal with such

Table 7. Purity, the Jaccard coefficient and the execution times obtained for the DBSCAN method.

Name	Purity	Jaccard	Time [s]
Aggregation	0.8274	0.7486	0.29
Compound	0.7393	0.6895	0.15
D31	0.1932	0.1027	1.44
Flame	0.6375	0.5359	0.10
Jain's data	0.7399	0.6141	0.13
Pathbased	0.3667	0.3329	0.12
R15	0.5333	0.2071	0.23
Spiral	1.0000	1.0000	0.12

Table 8. Purity, the Jaccard coefficient and the execution times obtained for the mean-shift method.

Name	Purity	Jaccard	Time [s]
Aggregation	0.9112	0.6472	0.30
Compound	0.8296	0.7283	0.15
D31	0.9435	0.8112	1.25
Flame	0.9208	0.7677	0.12
Jain's data	0.8070	0.4161	0.15
Pathbased	0.7012	0.4714	0.12
R15	0.9950	0.9801	0.23
Spiral	0.6473	0.1478	0.22

Table 9. Summary of the clustering method investigation.

Method	Min Purity	Mean Purity	Min Jaccard	Mean Jaccard	Time [s]
Hierarchical, single	0.2652	0.6750	0.1241	0.5616	2.52
Hierarchical, complete	0.4071	0.8004	0.2331	0.6516	**2.38**
CFSFDP	0.5856	0.8800	0.3319	0.7492	72.49
FCM	0.3397	0.7872	0.1957	0.5584	9.68
k-MEANS	0.3429	0.7743	0.1957	0.5543	4.65
DBSCAN	0.1932	0.6296	0.1027	0.5288	2.58
MS	0.7012	0.8445	0.1478	0.6212	2.54
LDC	**0.8722**	**0.9624**	**0.6534**	**0.9099**	58.26

a problem. That is why the next experiment concerns the linguistically defined validity index, described in Section 4, developed to solve this problem. In the experiment, we performed the clustering of all datasets with the number of clusters varying from 2 to 35. As the estimated number of clusters, we selected the value for which the quality of clustering was best (the validity index calculated according to (30) was lowest). In Table 10, we presented both the true and the estimated numbers of clusters, for all datasets. Analyzing these results, we can see that for six datasets the correct number of clusters was found. For the *Pathbased* dataset, we obtained too small a number (2 clusters instead of 3). We suppose that it was caused by a visible break within the cluster resembling the section of a circle (see Fig. 1). Consequently, for 2 clusters we obtained their best inner consistency.

In the second case of failure, the error is more serious. For the *D31* dataset we obtained 2 clusters instead of 31. However, if we scrutinize this dataset carefully (see Fig. 1), we can find that it contains a number of overlapping (to various extents) Gaussian distributions. Only on the basis of the information on the true number of clusters, we know that there are 31 of them. A careful observer can discern digits '9' and '7' or other well isolated groups of clusters, e.g., the one in the lower right corner (consisting of 8 overlapping sub-clusters). Single well isolated clusters can be noticed in the middle of the figure and in the top left corner. This can be a reason behind a low value of the proposed validity index for small numbers of clusters (from 2 to 5). On the other hand, for the number of clusters equal to 31, we obtained the second locally minimal value of this index (only slightly larger than for 2). Concluding, it seems right to say that the proposed linguistically defined validity index copes well with determination of the number of clusters for different types of data, and only for rather complicated distributions, which may cause problems even for a human observer, it may 'make' wrong decisions.

Table 10. Performance of the linguistically defined validity index: the true numbers of clusters and the estimated values.

Name	True	Estimated
Aggregation	7	7
Compound	6	6
D31	31	2
Flame	2	2
Jain's data	2	2
Pathbased	3	2
R15	15	15
Spiral	3	3

7. Conclusions

The paper introduced a new method of data clustering based on simple linguistically defined rules: (i) search for prototypes should be carried out in regions of very, very high data density very, very far from the previously found ones; (ii) data from the regions with similar densities that are located very, very near to each other should belong to one cluster. The first rule is applied to the sequential search for cluster prototypes, or rather medoids (we assume that prototypes are chosen from the data clustered). The second rule helps in assigning data to the selected medoids and, thereby, to form data clusters. Both the rules try to embrace the natural clustering capabilities of human beings and to transfer them to the clustering algorithms. The rules are expressed formally with the use of notions from the possibility theory such as linguistic variables/values, linguistic modifiers, and the modeling of a conjunction with a t-norm.

The goal of our work was to show that an algorithm based on such simple rules can be competitive with classical ones, known from the literature. The investigations were performed on benchmark datasets, containing both simulated and real data. The results obtained are very favorable and they inspire a further development of linguistically defined algorithms, not only for clustering but also for solving other problems for which a human expert can specify a solution linguistically.

As an example, consider a problem of detecting the

so-called QRS complexes in a noisy electrocardiogram. A human uses a simple rule in this case too. He or she detects QRS complexes if the signal energy in the frequency band around about a dozen hertz, within a time segment of more or less a hundred milliseconds, is big. Yet another example of such a linguistic formulation of algorithms was presented, when we defined (linguistically) a validity index for determination of a number of clusters. In this case, we check the inner consistency of clusters by applying the linguistically specified rule: if a cluster contains an isolated region of data AND their cardinality is high AND their density in this region is high then the cluster inner consistency is low. We have obtained promising results of using such a validity index.

Concluding, we have tried not only to show that linguistically defined algorithms can be competitive with classical ones in data clustering but also to encourage the readers to the construction of other algorithms for which a human can easily formulate linguistic rules, transferring in this way the extraordinary faculties of his brain to the computer algorithms. Finally, let us focus on the problem of selecting the membership function and formulating the linguistic rules for clustering. It seems interesting to apply evolutionary methods to make this choice. This would be a reflection of the fact that each of us uses his or her inherited and derived experience in real world phenomena clustering.

Acknowledgment

This research was supported by the National Science Centre in Poland under the grant no. 2017/27/B/ST6/01989. The authors thank the referees for their useful comments and suggestions, which helped to improve the quality of this manuscript.

References

Chang, H. and Yeung, D. (2008). Robust path-based spectral clustering, *Pattern Recognition* **41**(1): 191–203.

Comaniciu, D. and Meer, P. (2002). Mean shift: A robust approach toward feature space analysis, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(5): 603–619.

Duda, R., Hart, P. and Stork, D. (2001). *Pattern Classification*, Wiley, New York, NY.

Ester, M., Kriegiel, H.-P., Sander, J. and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise, *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, KDDM 1996, Portland, OR, USA*, pp. 226–231.

Everitt, B., Landau, S., Leese, M. and Stahl, D. (2011). *Cluster Analysis, 5th Edn.*, Wiley, London.

Fu, L. and Medico, E. (2007). Flame, a novel fuzzy clustering method for the analysis of DNA microarray data, *BMC Bioinformatics* **8**(3): 1–15.

Fukunaga, K. and Hostetler, L. (1975). The estimation of the gradient of a density function, with application to pattern recognition, *IEEE Transactions on Information Theory* **21**(1): 32–40.

Gionis, A., Mannila, H. and Tsaparas, P. (2007). Clustering aggregation, *ACM Transactions on Knowledge Discovery From Data* **1**(1): 1–30.

Jain, A. and Law, M. (2005). Data clustering: A user's dilemma, *in* S.K. Pal *et al.* (Eds.), *Pattern Recognition and Machine Intelligence*, Lecture Notes in Computer Science, Vol. 3776, Springer, New York, NY, pp. 1–10.

Jain, A., Murty, M. and Flynn, P. (1999). Data clustering: A review, *ACM Computing Surveys* **31**(3): 264–323.

Király, A., Vathy-Fogarassy, A. and Abonyi, J. (2016). Geodesic distance based fuzzy c-medoid clustering searching for central points in graphs and high dimensional data, *Fuzzy Sets and Systems* **286**(1): 157–172.

Leski, J. (2015). Fuzzy $(c + p)$-means clustering and its application to a fuzzy rule-based classifier: Towards good generalization and good interpretability, *IEEE Transactions on Fuzzy Systems* **23**(4): 802–812.

Leski, J. (2016). Fuzzy c-ordered-means clustering, *Fuzzy Sets and Systems* **286**(1): 114–133.

Leski, J. and Kotas, M. (2015). On robust fuzzy c-regression models, *Fuzzy Sets and Systems* **279**(1): 112–129.

Nguyen, S. and Choi, S.-B. (2015). Design of a new adaptive neuro-fuzzy inference system based on a solution for clustering in a data potential field, *Fuzzy Sets and Systems* **279**(1): 64–86.

Pancerz, K., Lewicki, A. and Tadeusiewicz, R. (2015). Ant-based extraction of rules in simple decision systems over ontological graphs, *International Journal of Applied Mathematics and Computer Science* **25**(2): 377–387, DOI: 10.1515/amcs-2015-0029.

Pedrycz, W., Al-Hmouz, R., Balamash, A. and Morfeq, A. (2015). Hierarchical granular clustering: An emergence of information glanules of higher type and higher order, *IEEE Transactions on Fuzzy Systems* **23**(6): 2270–2283.

Ripley, B. (1996). *Pattern Recognition and Neural Networks*, Cambridge University Press, Cambridge.

Rodriguez, A. and Laio, A. (2014). Clustering by fast search and find of density peaks, *Science* **344**(6191): 1492–1496.

Tou, J. and Gonzalez, R. (1974). *Pattern Recognition Principles*, Addison-Wesley, London.

Veenman, C., Reinders, M. and Backer, E. (2002). A maximum variance cluster algorithm, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(9): 1273–1280.

Webb, A. (1999). *Statistical Pattern Recognition*, Arnold, London.

Yager, R. and Filev, D. (1999). Induced ordered weighted averaging operators, *IEEE Transactions on Systems, Man and Cybernetics: Cybernetics* **29**(2): 141–150.

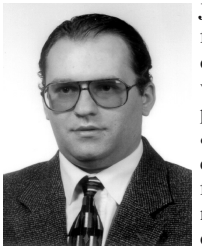
Zadeh, L. (1965). Fuzzy sets, *Information and Control* **8**(1): 338–353.

Zadeh, L. (1978). Fuzzy sets as a basis for a theory of possibility, *Fuzzy Sets and Systems* **1**(1): 2–28.

Zahn, C. (1971). Graph-theoretical methods for detecting and describing gestalt clusters, *IEEE Transactions on Computers* **1**(1): 68–86.

Zaki, M. and Meira, W. (2014). *Data Mining and Analysis: Fundamental Concepts and Algorithms*, Cambridge University Press, New York, NY.

Zok, T., Antczak, M., Riedel, M., Nebel, D., Villmann, T., Lukasiak, P., Blazewicz, J. and Szachniuk, M. (2015). Building the library of RNA 3D nucleotide conformations using the clustering approach, *International Journal of Applied Mathematics and Computer Science* **25**(3): 689–700, DOI: 10.1515/amcs-2015-0050.



Jacek M. Leski is a professor of biomedical information processing at the Silesian University of Technology, and currently the head of the Division of Biomedical Electronics. He is also a professor at the Institute of Medical Technology & Equipment ITAM. His research interests include digital processing of biomedical signals, fuzzy and neuro-fuzzy modeling, pattern recognition and learning theory. He is a senior member of the IEEE and a member of the Polish Society of Theoretical and Applied Electrotechnics.

Marian P. Kotas was born in Bielsko-Biala, Poland, in 1965. He received his MSc, PhD and DSc degrees from the Silesian University of Technology, Gliwice, Poland. Since 2012 he has been an associate professor at the Institute of Electronics, Division of Biomedical Electronics, Silesian University of Technology. His current research interests include linear and nonlinear filtering of biomedical signals, multivariate data processing and pattern recognition.

Received: 17 July 2017
Revised: 11 December 2017
Re-revised: 22 February 2018
Accepted: 16 April 2018