Fuzzy Clustering with Uninorm-Based Distance Measure

Evgeny Kagan *, Alexander Novoselsky, Alexander Rybalov

Posted Date: 17 April 2025

doi: 10.20944/preprints202504.1454.v1

Keywords: fuzzy c-means clustering; uninorm; absorbing norm

 doi:10.20944/preprints202504.1454.v1

Article

Fuzzy Clustering with Uninorm-Based Distance Measure

Evgeny Kagan [1,*], Alexander Novoselsky [2] and Alexander Rybalov [3]

[1] Ariel University, Kiryat ha-Mada, Ariel 4070000, Israel
[2] Independent researcher; Tel Aviv 6158101, Israel
[3] LAMBDA Lab, Tel Aviv University, Ramat Aviv, Tel Aviv 6997801, Israel
* Correspondence: evganyk@ariel.ac.il

Abstract: In the paper we suggest an algorithm of fuzzy clustering with uninorm-based distance measure. The algorithm follows a general scheme of fuzzy *c*-means (FCM) clustering, but in contrast to the existing algorithm it implements logical distance between data instances. The centers of the clusters calculated by the algorithm are less deviated and are concentrated in the areas of the actual centers of the clusters that results in more accurate recognition of the number of clusters and of data structure.

Keywords: fuzzy *c*-means clustering; uninorm; absorbing norm

MSC: 62A86; 62H30; 91C20

1. Introduction

In general, decision making starts with formulation of possible alternatives, collecting required data, data analysis, formulation of the decision criteria and choice of the alternatives based on the formulated criteria [1].

In simple cases of decision making with certain information and univariate data, decision making process is reduced to the choice of alternatives which minimize the payoff or maximize the reward. In cases of uncertain information, decision making is more complicated and deals with the expected payoffs and rewards using probabilistic or other methods for handling uncertainty. Finally, in cases of multivariate data and multicriteria choices, decision making is hardly solvable by exact methods and can be considered as a kind of art.

For example, Triantaphyllou in his book [2] describes several methods of multicriteria decision making without suggesting any preferred technique and indicates that each specific decision-making problem requires specific method for its solution. Such an approach is supported by the authors of the book [3] who stress "a growing need for methodologies that, on the basis of the ever-increasing wealth of data available, are able to take into account all the relevant points of view, technically called criteria, and all the actors involved" [3] (p. ix) and then present classification of the methods for solving different decision-making problems.

To apply the methods of decision making the raw data are preprocessed with the aim of recognition of possible patterns and consequently – to decrease the number of instances and, if it is possible, the number of dimensions [4]. The basic preprocessing procedure which decreases the number of instances used in the next steps of decision making is data classification [5,6] or, if any additional information is unavailable – data clustering [7,8].

The popular clustering method is the *k*-means algorithm suggested independently by Lloyd [9] and by Forgy [10] and is known as Lloyd-Forgy algorithm. This algorithm obtains n instances and creates k clusters such that each instance is included into a single cluster for which Euclidean distance between the instance and the cluster's center is minimal. The clusters are formed from the instances for which the within-cluster variances are also minimal. The FORTRAN implementation of

 doi:10.20944/preprints202504.1454.v1

the algorithm was published by Hartigan and Wong [11]. Currently, this algorithm is implemented in most statistical and mathematical software tools; for example, in MATLAB® it is implemented in the Statistics and Machine Learning Toolbox™ in the function `kmeans` [12].

Main disadvantage of the *k*-means algorithm and its direct successors is an inclusion of each instance only in a single cluster that often restricts recognition of the data patterns and interpretation of the obtained clusters.

To overcome this disadvantage Bezdek [13,14] suggested the fuzzy clustering algorithm widely known as the fuzzy *c*-means (FCM) algorithm. Following this algorithm, the clusters are considered as fuzzy sets, and each instance is included in several clusters with certain degrees of membership. In its original version, the algorithm uses Euclidean distances between the instances and the degrees of membership are calculated based on these distances. The other versions of the algorithm follow its original structure but use the other distance measures. For example, the Fuzzy Logic Toolbox™ of MATLAB® includes the function `fcm` [15], which implements the *c*-means algorithm with three distance measures: Euclidean distance, distance measure based on Mahalanobis distance from the instances and the cluster centers [16] and exponential distance measure normalized by the probability of choosing the cluster [17]. For the overview of different versions of the *c*-means algorithm and the other fuzzy clustering methods see the paper [18] and the book [19].

Along with the advantages of the fuzzy *c*-means clustering techniques, it inherits the following disadvantage of the *k*-means algorithm: both algorithms search for the predefined number of clusters and separate the cluster centers even in cases where such separation does not follow from the data structures. For example, if the data is a set of normally distributed instances, then it is expected that the clustering algorithm will recognize a single cluster, and the centers of all clusters will be placed close to the center of distribution. However, the *c*-means algorithm with the known distance measures does not provide such a result.

The suggested *c*-means algorithm with the uninorm-based distance solves this problem. Similar to the other fuzzy *c*-means algorithms, the suggested algorithm has the same structure as original Bezdek algorithm, but, in contrast to the known methods, it considers the normalized instances as truth values of certain propositions and implements fuzzy logical distance between them. The suggested distance is based on the probability-based uninorm and absorbing norm [20] which already demonstrated their usefulness in a wide range of decision-making and control tasks [21].

The suggested algorithm can be used for recognition of the patterns in raw data and for data preprocessing in different decision-making problems.

The rest of the paper is organized as follows. In section 2.1 we briefly outline the Bezdek fuzzy *c*-means algorithm and in section 2.2 describe the uninorm and absorbing norm which will be used for construction the logical distance. Section 3 presents the main results: the distance measure based on the uninorm and absorbing norm (section 3.1) and the resulting algorithm for fuzzy clustering (section 3.2). In section 3.3 we illustrate the activity of the algorithm by numerical simulations with different data distributions and compare the obtained results with the results provided by the known fuzzy *c*-means algorithms. Section 4 includes some discussable issues.

2. Materials and Methods

The suggested algorithm follows the structure of the Bezdek fuzzy *c*-means algorithm [13,14] but, in contrast to the existing algorithms, it uses the logical distance based on the uninorm and absorbing norm [20]. Below we briefly describe this algorithm and the used norms.

2.1. Fuzzy c-Means Algorithm

Assume that the raw data is given by d-dimensional vectors of real numbers

$$X = (x_1, x_2, \dots, x_n) \subset \mathbb{R}^d,$$

where $x_i = (x_{i1}, x_{i2}, \dots, x_{id}),\ i = 1, 2, \dots, n,\ n \geq 1,\ d \geq 1$, stands for the observation or instance of the data.

 doi:10.20944/preprints202504.1454.v1

Given a number $m \le n$, denote by

$$Y = (y_1, y_2, \dots, y_m) \subset \mathbb{R}^d$$

a vector of the cluster centers, where $y_j = (y_{j1}, y_{j2}, \dots, y_{jd})$, $j = 1, 2, \dots, m$, $m \ge 1$, $d \ge 1$, denotes the center of the jth cluster.

The problem is to define the centers y_j of the clusters and to define the membership degree $\mu_{ij} \in [0, 1]$ of each instance x_i to each cluster j.

The fuzzy *c*-means algorithm which solves this problem is outlined as follows [13] (see also [14,15,17]).

Algorithm 1. Fuzzy c-means algorithm.

Input: d-dimensional data vectors $X = (x_1, x_2, \dots, x_n)$, $n \ge 1$,
number of clusters m,
termination criterion $\varepsilon \in (0, 1)$,
weighting exponent $q > 1$,
function dist^d.

Output: vector $Y = (y_1, y_2, \dots, y_m)$, $m \ge 1$, of cluster centers,
matrix $\text{M} = (\mu_{ij})$, $i = 1, 2, \dots, n$, $j = 1, 2, \dots, m$, of membership degrees.

1. Initialize the cluster centers y_j, $j = 1, 2, \dots, m$, by random.
2. Initialize the membership degrees μ_{ij}, $i = 1, 2, \dots, n$, $j = 1, 2, \dots, m$,

$$\mu_{ij} = \left(\frac{1}{\text{dist}^d(x_i, y_j)}\right)^{1/(q-1)} \Big/ \sum_{k=1}^{m} \left(\frac{1}{\text{dist}^d(x_i, y_k)}\right)^{1/(q-1)}.$$

3. Do
4. Save membership degrees $\mu'_{ij} = \mu_{ij}$, $i = 1, 2, \dots, n$, $j = 1, 2, \dots, m$,
5. Calculate cluster centers

$$y_j = \sum_{i=1}^{n} (\mu_{ij})^q x_i / \sum_{i=1}^{n} (\mu_{ij})^q,$$

6. Calculate membership degrees

$$\mu_{ij} = \left(\frac{1}{\text{dist}^d(x_i, y_j)}\right)^{1/(q-1)} \Big/ \sum_{k=1}^{m} \left(\frac{1}{\text{dist}^d(x_i, y_k)}\right)^{1/(q-1)},$$

where $\text{dist}^d(x_i, y_j)$ is the distance between the instance x_i and the cluster center y_j,

7. While $\max_{i,j}(\mu'_{ij} - \mu_{ij}) > \varepsilon$.
8. Return $Y = (y_1, y_2, \dots, y_m)$ and $\text{M} = (\mu_{ij})$.

In the original version [13], the fuzzy *c*-means algorithm uses the Euclidean distance

$$\text{dist}^d(x_i, y_j) = \sqrt{\sum_{l=1}^{d} (x_{il} - y_{jl})^2},$$

and in the succeeding versions [16,17] the other distance measures were applied.

In this paper, we suggest to use the logical distance measure based on the uninorm and absorbing norm which results in more accurate recognition of the cluster centers.

2.2. Uninorm and Absorbing Norm

Uninorm [22] and absorbing norm [23] are the operators of fuzzy logic which extend Boolean operators and are used for aggregating truth values.

Uninorm aggregator $\oplus_\theta$ with neutral element $\theta \in [0, 1]$ is a function $\oplus_\theta : [0, 1] \times [0, 1] \to [0, 1]$ satisfying the following properties [22]; $x, y, z \in [0, 1]$:

a. Commutativity: $x \oplus_\theta y = y \oplus_\theta x$,
b. Associativity: $(x \oplus_\theta y) \oplus_\theta z = x \oplus_\theta (y \oplus_\theta z)$,
c. Monotonicity: $x \le y$ implies $x \oplus_\theta z \le y \oplus_\theta z$,

 doi:10.20944/preprints202504.1454.v1

d. Identity: $\theta \oplus_\theta x = x$ for some $\theta \in [0,1]$,

and such that if $\theta = 1$, then $x \oplus_1 y = x \wedge y$ is a conjunction operator and if $\theta = 0$, then $x \oplus_0 y = x \vee y$ is a disjunction operator.

An absorbing norm aggregator $\otimes_\vartheta$ with absorbing element $\vartheta \in [0,1]$ is a function $\otimes_\vartheta : [0,1] \times [0,1] \to [0,1]$ satisfying the following properties [23]; $x, y, z \in [0,1]$:

a. Commutativity: $x \otimes_\vartheta y = y \otimes_\vartheta x$,
b. Associativity: $(x \otimes_\vartheta y) \otimes_\vartheta z = x \otimes_\vartheta (y \otimes_\vartheta z)$,
c. Absorbing element: $\vartheta \otimes_\vartheta x = \vartheta$ for any $x \in [0,1]$.

Operator $\otimes_\vartheta$ is a fuzzy analog of the negated xor operator.

Let $u_\theta : (0,1) \to (-\infty, \infty)$ and $v_\vartheta : (0,1) \to (-\infty, \infty)$ be invertible, continuous, strictly monotonously increasing functions with the parameters $\theta, \vartheta \in [0,1]$ such that

a. $\lim_{x \to 0} u_\theta(x) = -\infty$, $\lim_{x \to 0} v_\vartheta(x) = -\infty$,
b. $\lim_{x \to 1} u_\theta(x) = +\infty$, $\lim_{x \to 1} v_\vartheta(x) = +\infty$,
c. $u_\theta(\theta) = \theta$, $v_\vartheta(\vartheta) = \vartheta$.

Then, for any $x, y \in (0,1)$ hold [24]

$$x \oplus_\theta y = u_\theta^{-1}\big(u_\theta(x) + u_\theta(y)\big),$$
$$x \otimes_\vartheta y = v_\vartheta^{-1}\big(v_\vartheta(x) \times v_\vartheta(y)\big),$$

and for the bounds 0 and 1 the values of the operators $\oplus_\theta$ and $\otimes_\vartheta$ are defined in correspondence to the results of Boolean operators.

If $\theta = \vartheta$ and $u \equiv v$, then the interval $[0,1]$ with the operators $\oplus_\theta$ and $\otimes_\vartheta$ form an algebra [20,25]

$$\mathcal{A} = \langle [0,1],\ \oplus_\theta,\ \otimes_\vartheta \rangle$$

in which uninorm $\oplus_\theta$ acts as a summation operator and absorbing norm $\otimes_\vartheta$ acts as a multiplication operator. In this algebra, neutral element θ is zero value for summation and absorbing element ϑ is a unit value for multiplication. Note that in contrast to the algebra of real numbers, in the algebra $\mathcal{A}$ holds $\theta = \vartheta$ despite different meanings of these values.

The inverse operators in this algebra are

$$x \ominus_\theta y = u_\theta^{-1}\big(u_\theta(x) - u_\theta(y)\big), \qquad x, y \in (0,1)$$
$$x \oslash_\vartheta y = v_\vartheta^{-1}\big(v_\vartheta(x) / v_\vartheta(y)\big), \qquad v(y) \neq 0.$$

In addition, negation operator is defined as

$$\ominus_\theta x = u_\theta^{-1}\big(-u_\theta(x)\big), \qquad x \in (0,1).$$

In the paper [20] it was demonstrated that the functions u_θ and v_ϑ satisfy the requirements of quantile functions of probability distributions. Along with that any function satisfying the indicated above requirements and its inverse are also applicable.

Here we will assume that the functions $u_\theta \equiv v_\vartheta$ are equivalent. In the simulations, we will use the functions

$$u_\theta(x) = \ln \frac{x^\alpha}{1 - x^\alpha}, \qquad x \in (0,1),$$

$$u_\theta^{-1}(\xi) = \left(\frac{e^\xi}{1 + e^\xi}\right)^{1/\alpha}, \qquad \xi \in (-\infty, \infty),$$

where $\alpha = 1 / \log_2 \frac{1}{\theta}$, $\theta \in (0,1)$.

For the other examples of the functions u_θ and v_ϑ see the paper [20] and the book [21].

 doi:10.20944/preprints202504.1454.v1

3. Results

We start with the definition of logical distance based on the uninorm and absorbing norm and then present the suggested algorithm. In the last subsection we consider numerical simulations and compare the suggested algorithm with the known method.

3.1. Logical Distance Based on the Uninorm and Absorbing Norm

Let $x, y \in [0,1]$ be two truth values. Given neutral element θ and absorbing element ϑ, fuzzy logical dissimilarity of the values x and y is defined as follows:

$$\mathrm{disim}_{\theta,\vartheta}(x,y) = \ominus_\theta \left(\left(x \oplus_y \theta \right) \otimes_\vartheta \left(\theta \oplus_x y \right) \right).$$

where the value

$$\mathrm{sim}_{\theta,\vartheta}(x,y) = \left(x \oplus_y \theta \right) \otimes_\vartheta \left(\theta \oplus_x y \right)$$

is a fuzzy logical similarity of the values x and y.

If $\theta = \vartheta$, we will write disim_θ instead of $\mathrm{disim}_{\theta,\vartheta}$ and sim_θ instead of $\mathrm{sim}_{\theta,\vartheta}$.

Lemma 1. *If $\theta = \vartheta$, then the function $disim_\theta$ is a semi-metric in the algebra $\mathcal{A}$.*

Proof. To prove the lemma, we need to check the following three properties:

a. $\mathrm{disim}_{\theta,\vartheta}(x,x) = \theta = \vartheta$.

By commutativity and identity properties of the uninorm holds:

$$x \oplus_x \theta = \theta \oplus_x x = \theta.$$

Then, by the property of absorbing element holds

$$\theta \otimes_\theta \theta = \theta$$

and

$$\ominus_\theta \theta = \theta.$$

b. $\mathrm{disim}_{\theta,\vartheta}(x,y) > \theta = \vartheta$ if $x \neq y$.

If $x \neq y$, then either

$$x \oplus_y \theta > \theta \text{ and } \theta \oplus_x y < \theta$$

or

$$x \oplus_y \theta < \theta \text{ and } \theta \oplus_x y > \theta.$$

Thus,

$$\left(x \oplus_y \theta \right) \otimes_\theta \left(\theta \oplus_x y \right) < \theta$$

and

$$\ominus_\theta \left(\left(x \oplus_y \theta \right) \otimes_\theta \left(\theta \oplus_x y \right) \right) > \theta.$$

c. $\mathrm{disim}_{\theta,\vartheta}(x,y) = \mathrm{disim}_{\theta,\vartheta}(y,x)$.

It follows directly from the commutativity of the absorbing norm.

Lemma is proven. □

Using the dissimilarity function, we define the fuzzy logical distance $\mathrm{dist}_{\theta,\vartheta}$ between the values $x, y \in [0,1]$ as follows ($\theta = \vartheta$):

$$\mathrm{dist}_{\theta,\vartheta}(x,y) = \frac{1}{\theta} \mathrm{disim}_{\theta,\vartheta}(x,y) - 1.$$

Lemma 2. *If $\theta = \vartheta$, then the function $dist_\theta$ is a semi-metric in the algebra of real numbers on the interval* $[0,1]$.

 doi:10.20944/preprints202504.1454.v1

Proof. Since for any $x, y \in [0,1]$ and any $\theta = \vartheta \in [0,1]$ both $x \oplus_\theta y \in [0,1]$, $x \otimes_\vartheta y \in [0,1]$ and $\ominus_\theta x \in [0,1]$, and by the properties a and b of semi-metric, holds

$$\mathrm{disim}_{\theta,\vartheta}(x,y) \in [\theta, 1].$$

a. If $x = y$, then $\mathrm{disim}_{\theta,\vartheta}(x,y) = \theta = \vartheta$ and

$$\mathrm{dist}_{\theta,\vartheta}(x,y) = 0.$$

b. If $x \neq y$, then $\mathrm{disim}_{\theta,\vartheta}(x,y) > \theta = \vartheta$. So $\frac{1}{\theta}\mathrm{disim}_{\theta,\vartheta}(x,y) > 1$ and

$$\mathrm{dist}_{\theta,\vartheta}(x,y) > 0.$$

c. Symmetry

$$\mathrm{dist}_{\theta,\vartheta}(x,y) = \mathrm{dist}_{\theta,\vartheta}(y,x)$$

follows directly from the symmetry of the dissimilarity $\mathrm{disim}_{\theta,\vartheta}$. □

An example of the fuzzy logic distance $\mathrm{dist}_{\theta,\vartheta}(x,y)$ between the values $x, y \in [0,1]$ with $\theta = \vartheta = 0.5$ is shown in Figure 1.a. For comparison, Figure 1.b shows the Euclidean distance between the values $x, y \in [0,1]$.

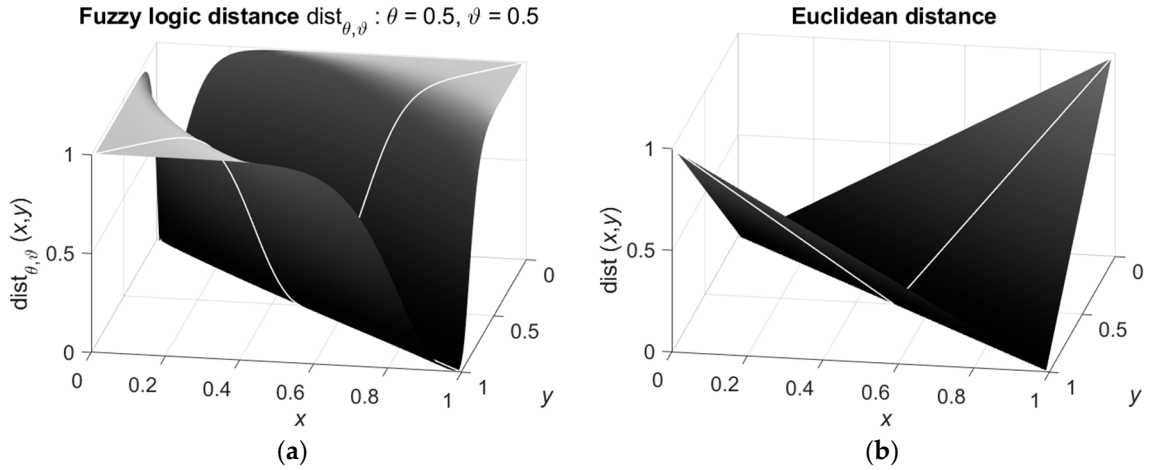


Figure 1. (**a**) Fuzzy logic distance $\mathrm{dist}_{\theta,\vartheta}(x,y)$ between the values $x, y \in [0,1]$ with $\theta = \vartheta = 0.5$; (**b**) Euclidean distance between the values $x, y \in [0,1]$.

It is seen that the fuzzy logical distance better separates the close values and less sensitive to the far values than the Euclidean distance.

Now let us extend the introduced fuzzy logical distance to multidimensional variables.

Let

$$(x_1, x_2, \dots, x_n) \subset [0,1]^d, \ n \geq 1, \ d \geq 1,$$

be d-dimensional vectors such that each vector $x_i = (x_{i1}, x_{i2}, \dots, x_{id})$, $i = 1, 2, \dots, n$, is a point in a d-dimensional space.

The fuzzy logical dissimilarity of the points x_i and x_j, $i, j = 1, 2, \dots, n$, is defined as follows:

$$\mathrm{disim}^d_{\theta,\vartheta}(x_i, x_j) = \ominus_\theta \left\{ \oplus_\theta {}^d_{l=1} \left[\left(x_{il} \oplus_{x_{jl}} \theta \right) \otimes_\vartheta \left(\theta \oplus_{x_{il}} x_{jl} \right) \right] \right\},$$

where, as above, the value

$$\mathrm{sim}^d_{\theta,\vartheta}(x_i, x_j) = \oplus_\theta {}^d_{l=1} \left[\left(x_{il} \oplus_{x_{jl}} \theta \right) \otimes_\vartheta \left(\theta \oplus_{x_{il}} x_{jl} \right) \right]$$

is a fuzzy logical similarity of the points x_i and x_j.

Let $\theta = \vartheta$. Then, as above, the fuzzy logical distance between the points x_i and x_j, $i, j = 1, 2, \dots, n$, is

 doi:10.20944/preprints202504.1454.v1

$$\mathrm{dist}_{\theta,\vartheta}^{d}(x_i, x_j) = \frac{1}{\theta}\mathrm{disim}_{\theta,\vartheta}^{d}(x_i, x_j) - 1.$$

Lemma 3. *If* $\theta = \vartheta$, *then the function* $disim_{\theta}^{d}$ *is a semi-metric in the algebra* $\mathcal{A} = \langle [0,1]^d,\ \oplus_\theta,\ \otimes_\vartheta \rangle,\ d \geq 1$.

Proof. This statement is a direct consequence of lemma 1 and the properties of the uninorm.

a. $\mathrm{disim}_{\theta,\vartheta}^{d}(x_i, x_j) = \theta = \vartheta$.

If $x_i = x_j$ then for each $l = 1,2,\dots,d$

$$\left(x_{il} \oplus_{x_{jl}} \theta\right) \otimes_\theta \left(\theta \oplus_{x_{il}} x_{jl}\right) = \theta.$$

Then,

$$\oplus_{\theta\, l=1}^{d} \left[\left(x_{il} \oplus_{x_{jl}} \theta\right) \otimes_\vartheta \left(\theta \oplus_{x_{il}} x_{jl}\right)\right] = \oplus_{\theta\, l=1}^{d}\, \theta = \theta$$

and finally

$$\ominus_\theta\, \theta = \theta.$$

b. $\mathrm{disim}_{\theta,\vartheta}^{d}(x_i, x_j) > \theta$ if $x \neq y$.

If $x_i \neq x_j$, then for each $l = 1,2,\dots,d$

$$\left(x_{il} \oplus_{x_{jl}} \theta\right) \otimes_\theta \left(\theta \oplus_{x_{il}} x_{jl}\right) < \theta$$

Then,

$$\oplus_{\theta\, l=1}^{d} \left[\left(x_{il} \oplus_{x_{jl}} \theta\right) \otimes_\vartheta \left(\theta \oplus_{x_{il}} x_{jl}\right)\right] < \theta$$

and

$$\ominus_\theta \left\{\oplus_{\theta\, l=1}^{d} \left[\left(x_{il} \oplus_{x_{jl}} \theta\right) \otimes_\vartheta \left(\theta \oplus_{x_{il}} x_{jl}\right)\right]\right\} > \theta.$$

c. $\mathrm{disim}_{\theta,\vartheta}^{d}(x_i, x_j) = \mathrm{disim}_{\theta,\vartheta}^{d}(x_j, x_i)$.

It follows directly from the symmetry of the dissimilarity for each $l = 1,2,\dots,d$ and commutativity of the uninorm. □

Lemma 4. *If* $\theta = \vartheta$, *then the function* $dist_{\theta,\vartheta}^{d}$ *is a semi-metric on the hypercube* $[0,1]^d,\ d \geq 1$.

Proof. The proof is literally the same as the proof of lemma2. □

The suggested algorithm uses the introduced function $\mathrm{dist}_{\theta,\vartheta}^{d}$ as a distance measure between the instances of the data.

3.2. The c-Means Algorithm with Fuzzy Logical Distance

The suggested algorithm considers the instances of the data as truth values and uses Algorithm 1 with the fuzzy logical distance $\mathrm{dist}_{\theta,\vartheta}^{d}$ on these values.

As above, assume that the raw data is represented by the d-dimensional vectors

$$X = (x_1, x_2, \dots, x_n) \subset \mathbb{R}^d,$$

where $x_i = (x_{i1}, x_{i2}, \dots, x_{id}),\ i = 1,2,\dots,n,\ n \geq 1,\ d \geq 1$, is a data instance.

Since function $\mathrm{dist}_{\theta,\vartheta}^{d}$ requires the values from the hypercube $[0,1]^d$, d ≥ 1, vector X must be normalized

$$\tilde{X} = \mathrm{norm}(X), \quad X \subset \mathbb{R}^d, \quad \tilde{X} \subset [0,1]^d,$$

and the algorithm should be applied to the normalized data vector $\tilde{X}$. After definition of the cluster centers $\tilde{Y}$, the inverse normalization must be applied to the vector $\tilde{Y}$

 doi:10.20944/preprints202504.1454.v1

$$Y = \text{norm}^{-1}(\tilde{Y}), \quad \tilde{Y} \subset [0,1]^d, \quad Y \subset \mathbb{R}^d.$$

Normalization can be conducted by several methods; in Appendix A we present a simple Algorithm 3 of normalization by linear transformation. The inverse transformation is provided by the Algorithm 4 also presented in Appendix A.

In general, the suggested Algorithm 2 follows the Bezdek fuzzy *c*-means algorithm 1, however differs in the distance function and in the initialization of the cluster centers $\tilde{y}_j$, $j = 1,2,\dots,m$, and, consequently, – in the definition of the number of clusters.

Algorithm 2. Fuzzy *c*-means algorithm with fuzzy logical distance measure

Input: d-dimensional data vectors $X = (x_1, x_2, \dots, x_n) \subset \mathbb{R}^d$, $n \geq 1$, $d \geq 1$,
termination criterion $\varepsilon \in (0,1)$,
weighting exponent $q > 1$,
precision $\xi \in (0,1)$,
distance function $\text{dist}_{\theta,\vartheta}^d$ with the parameters $\theta, \vartheta \in [0,1]$.

Output: vector $Y = (y_1, y_2, \dots, y_m)$, $m \geq 1$, of cluster centers,
matrix $\text{M} = (\mu_{ij})$, $i = 1,2,\dots,n$, $j = 1,2,\dots,m$, of membership degrees.

1. Calculate normalized data vectors $\tilde{X} = (\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_n) \subset [0,1]^d$ and the values x_{min}, $\tilde{x}_{max}$ using Algorithm 3.
2. Initialize vector $\tilde{Y} = (\tilde{y}_1, \tilde{y}_2, \dots, \tilde{y}_m) \subset [0,1]^d$ of the cluster centers as a d-dimensional grid with the step ξ in the hypercube $[0,1]^d$. The number m of clusters is defined as a number of nodes in this grid.
3. Initialize the membership degrees μ_{ij}, $i = 1,2,\dots,n$, $j = 1,2,\dots,m$,
$$\mu_{ij} = \left(\frac{1}{\text{dist}_{\theta,\vartheta}^d(\tilde{x}_i, \tilde{y}_j)}\right)^{1/(q-1)} \Big/ \sum_{k=1}^m \left(\frac{1}{\text{dist}_{\theta,\vartheta}^d(\tilde{x}_i, \tilde{y}_k)}\right)^{1/(q-1)}.$$
4. Do
5. Save membership degrees $\mu_{ij}' = \mu_{ij}$, $i = 1,2,\dots,n$, $j = 1,2,\dots,m$,
6. Calculate cluster centers
$$\tilde{y}_j = \sum_{i=1}^n (\mu_{ij})^q \tilde{x}_i / \sum_{i=1}^n (\mu_{ij})^q,$$
7. Calculate membership degrees
$$\mu_{ij} = \left(\frac{1}{\text{dist}_{\theta,\vartheta}^d(\tilde{x}_i, \tilde{y}_j)}\right)^{1/(q-1)} \Big/ \sum_{k=1}^m \left(\frac{1}{\text{dist}_{\theta,\vartheta}^d(\tilde{x}_i, \tilde{y}_k)}\right)^{1/(q-1)},$$
where $\text{dist}_{\theta,\vartheta}^d(\tilde{x}_i, \tilde{y}_j)$ is the fuzzy logical distance between the instance $\tilde{x}_i$ and the cluster center $\tilde{y}_j$,
8. While $\max_{i,j}(\tilde{\mu}_{ij} - \mu_{ij}) > \varepsilon$.
9. Calculate renormalized vector $Y = (y_1, y_2, \dots, y_m) \subset \mathbb{R}^d$ from the vector $\tilde{Y} = (\tilde{y}_1, \tilde{y}_2, \dots, \tilde{y}_m) \subset [0,1]^d$ using Algorithm 4.
10. Return $Y = (y_1, y_2, \dots, y_m)$ and $\text{M} = (\mu_{ij})$.

The main difference between the suggested Algorithm 2 and the original Bezdek fuzzy *c*-means Algorithm 1 [13,14] and the known Gustafson-Kessel [16] and Gath-Geva [17] is the use of the fuzzy logical distance $\text{dist}_{\theta,\vartheta}^d$. The use of this distance requires normalization and renormalization of the data.

The other difference is in the need of the initialization of the cluster centers as a grid in the algorithm's domain. Such initialization is required because of quick convergence of the algorithm; thus, the even distribution of the initial cluster centers avoids missing the clusters. A simple algorithm 5 for creating a grid in the square $[0,1]^2$ is outlined in Appendix A.

Let us consider two main properties of the suggested algorithm.

 doi:10.20944/preprints202504.1454.v1

Theorem 1. *Algorithm 2 converges.*

Proof. Convergence of the algorithm 2 follows directly from the fact that $\mathrm{dist}_{\theta,\vartheta}^{d}$ is a semi-metric (see lemma 4).

In fact, in the lines 3 and 7 of the algorithm holds

$$0 < \mu_{ij} \leq 1,$$

$$\mathrm{dist}_{\theta,\vartheta}^{d}(x_i, x_j) < \mathrm{dist}_{\theta,\vartheta}^{d}(x_p, x_q) \Rightarrow \mu_{ij} < \mu_{pq},$$

and the algorithm converges. □

Theorem 2. *Time complexity of the Algorithm 2 is* $\mathcal{O}(dmnk)$*, where* d *is the dimensionality of the data,* m *is the number of clusters,* n *is the number of instances in the data and* k *is the number of iterations.*

Proof. At first, consider calculation of the fuzzy logical distances $\mathrm{disim}_{\theta,\vartheta}^{d}(x_i, y_j)$, $i = 1, 2, \dots, n$, $j = 1, 2, \dots, m$. Complexity of this operation is $\mathcal{O}(1)$ for each dimension $l = 1, 2, \dots, d$; thus, calculation of the distances has a complexity $\mathcal{O}(d)$.

Now let us consider the lines of the algorithm. Normalization of the data vector (line 1) requires $\mathcal{O}(dn)$ steps and initialization of the cluster centers (line 2) requires $\mathcal{O}(mn)$ steps (see Algorithm 3 in Appendix A).

Initialization of the membership degrees (line 3) requires $\mathcal{O}(mn)$ steps for each dimension that gives $\mathcal{O}(dmn)$.

In the do-while loop, saving the membership degrees (line 5) requires $\mathcal{O}(mn)$, calculation of the cluster centers given the membership degrees (line 6) requires $\mathcal{O}(mn)$ steps and calculation of the membership degrees (line 7) requires, as above, $\mathcal{O}(mn)$ steps for each dimension that gives $\mathcal{O}(dmn)$.

Finally, renormalization of the vector of the cluster centers (line 9) requires $\mathcal{O}(dn)$ steps.

Thus, initial (lines 1-3) and final (9) operations of the algorithm require

$$\mathcal{O}(dn) + \mathcal{O}(mn) + \mathcal{O}(dmn) + \mathcal{O}(dn) = \mathcal{O}(dmn).$$

steps and each iteration requires

$$\mathcal{O}(mn) + \mathcal{O}(mn) + \mathcal{O}(mn) + \mathcal{O}(dmn) = \mathcal{O}(dmn).$$

Then, for k iterations it is required

$$\mathcal{O}(dmn) + k\mathcal{O}(dmn) = \mathcal{O}(dmnk)$$

steps. □

Note that in the considerations above we assumed that $\theta = \vartheta$ which supported the semi-metric properties of the function $\mathrm{dist}_{\theta,\vartheta}^{d}$. Along with that, in practice these parameters can differ and, despite the absence of formal proofs, the use of the function $\mathrm{dist}_{\theta,\vartheta}^{d}$ with $\theta \neq \vartheta$ can provide essentially better clustering.

3.3. Numerical Simulations

In the first series of simulations, we will demonstrate that the suggested Algorithm 2 with fuzzy logical distance results in more precise centers of the clusters than the original Algorithm 1 with Euclidean distance.

For this purpose, in the simulations we generate a single cluster as normally distributed instances with the known center and apply Algorithms 1 and 2 to these data. As a measure of the quality of the algorithms we use the mean squared error (MSE) in finding the center of the cluster center and the means of standard deviations of the calculated clusters centers.

 doi:10.20944/preprints202504.1454.v1

To avoid the influence of the instance values, in the simulations we considered the normalized data. An example of the data (white circles) and the results of the algorithms (gray diamonds for Algorithm 1 and black pentagrams for Algorithm 2) are shown in Figure 2.

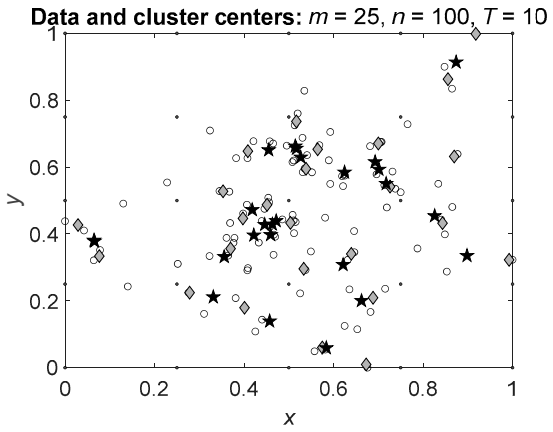


Figure 2. Example of the data and the resulting cluster centers for a single cluster of normally distributed instances. Number of instances is $n = 100$, number of clusters is $m = 25$, and number of iterations $T = 10$. Initially, the cluster centers are in the nodes of the grid and are depicted by black points. The instances are distributed normally around the mean $\mu = 0.5$ and are shown by white circles. The cluster centers calculated by the Algorithm 1 with Euclidean distance are depicted by gray diamonds and the cluster centers calculated by the Algorithm 2 with fuzzy logical distance are depicted by black pentagrams.

The simulations were conducted with different numbers of clusters by the series of 100 trials. The results were tested and compared by the one-sample and two-sample Student's t-tests.

In the simulations we assumed that the algorithm which for a single cluster provides less deviated cluster centers is more precise in the calculation of the cluster centers. In other words, the algorithm which results in the clusters centers concentrated near the actual cluster center is better than the algorithm which results in more dispersed cluster centers.

Results of simulations are summarized in Table 1. In the table, we present the results of clustering of two-dimensional data distributed around the mean $\mu = 0.5$.

Table 1. Results of simulations for each dimension: means μ_x, μ_y of the instances which are the centers of the cluster, mean squared errors e_x, e_y of the centers calculated by the Algorithms 1 and 2, means of the standard deviations σ_x, σ_y of the cluster centers calculated by the Algorithms 1 and 2 and significance of the difference between standard deviations of the cluster centers calculated by the Algorithms 1 and 2.

#instances, #clusters	Source	Mean μ_x, μ_y	MSE e_x, e_y	Mean STD σ_x, σ_y	Significance of the STD's difference
$n = 100$, $m = 25$	Data	0.499 0.507	–	–	–
	Algorithm 1	0.498 0.506	9.3×10^{-4} 9.4×10^{-4}	0.254 0.247	Significant, $\alpha = 0.95$
	Algorithm 2	0.494 0.503	3.7×10^{-4} 2.6×10^{-4}	0.211 0.206	
$n = 100$, $m = 16$	Data	0.491 0.490	–	–	–
	Algorithm 1	0.493 0.499	1.14×10^{-3} 9.16×10^{-4}	0.247 0.251	Significant, $\alpha = 0.95$
	Algorithm 2	0.488 0.489	4.03×10^{-4} 4.22×10^{-4}	0.210 0.211	
$n = 100$,	Data	0.501 0.487	–	–	

 doi:10.20944/preprints202504.1454.v1

$m = 9$	Algorithm 1	*0.505* *0.489*	9.17×10^{-4} 6.13×10^{-4}	*0.223* *0.213*	Significant, $\alpha = 0.95$
	Algorithm 2	*0.502* *0.488*	3.33×10^{-4} 2.06×10^{-4}	*0.190* *0.187*	

It is seen that both algorithms result in cluster centers close to the actual cluster center and the errors in calculating the centers are extremely small. Additional statistical testing by Student's t-test demonstrated that the differences between the obtained clusters centers are not significant with $\alpha = 0.95$.

Along with that, the suggested Algorithm 2 results in smaller standard deviation than the Algorithm 1 and this difference is significant with $\alpha = 0.95$. Hence, the suggested Algorithm 2 results in more precise cluster centers than Algorithm 1.

To illustrate these results, let us consider simulations of the algorithms on the data with several predefined clusters.

Consider application of Algorithms 1 and 2 to the data with two predefined clusters with the centers in the points $(0.3, 0.3)$ and $(0.7, 0.7)$. The resulting cluster centers with $m = 25$ and $m = 9$ clusters are shown in Figure 3. Notation in the figure is the same as in Figure 2.

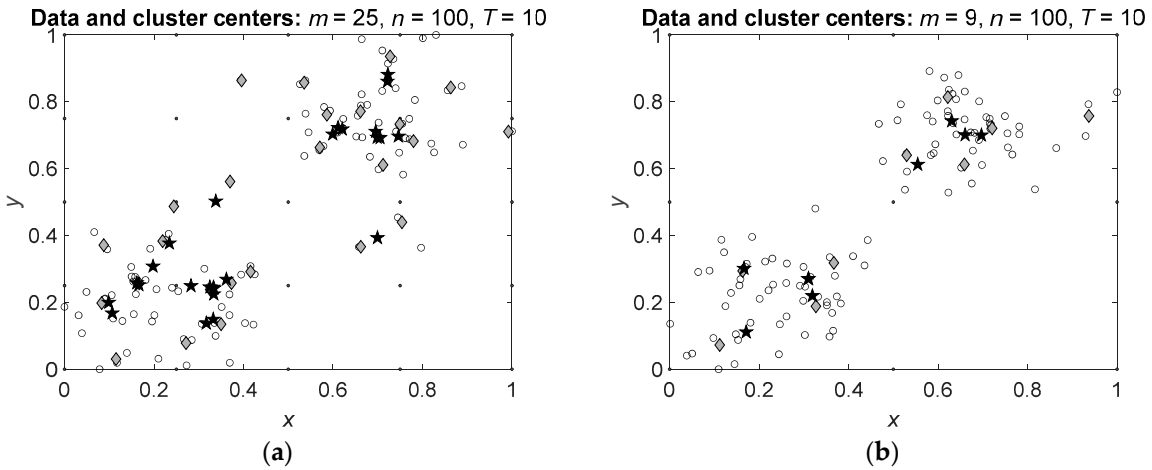


Figure 3. Data with two predefined clusters and cluster centers calculated by the Algorithms 1 and 2: (**a**) number of clusters $m = 25$; (**b**) number of clusters $m = 9$.

It is seen that clusters centers obtained by Algorithm 2 (black pentagrams) are concentrated closer to the actual cluster centers than the cluster centers obtained by Algorithm 1 (gray diamonds). Moreover, some of the cluster centers obtained by Algorithm 2 are located in the same points while all cluster centers obtained by Algorithm 1 are located in different points.

More clearly the observed effect is seen on the data with several clusters. Figure 4 shows the results of Algorithms 1 and 2 applied to the data with 10 predefined clusters.

 doi:10.20944/preprints202504.1454.v1

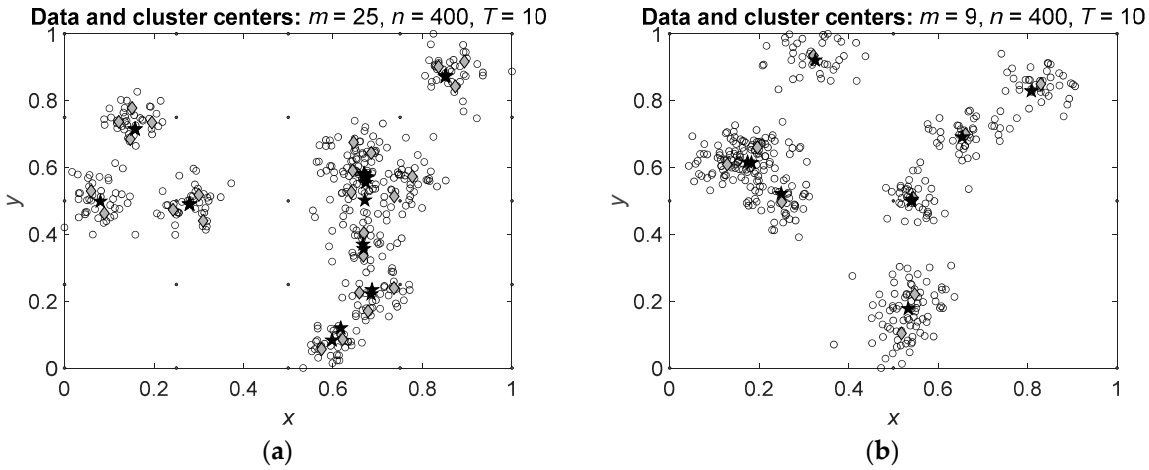
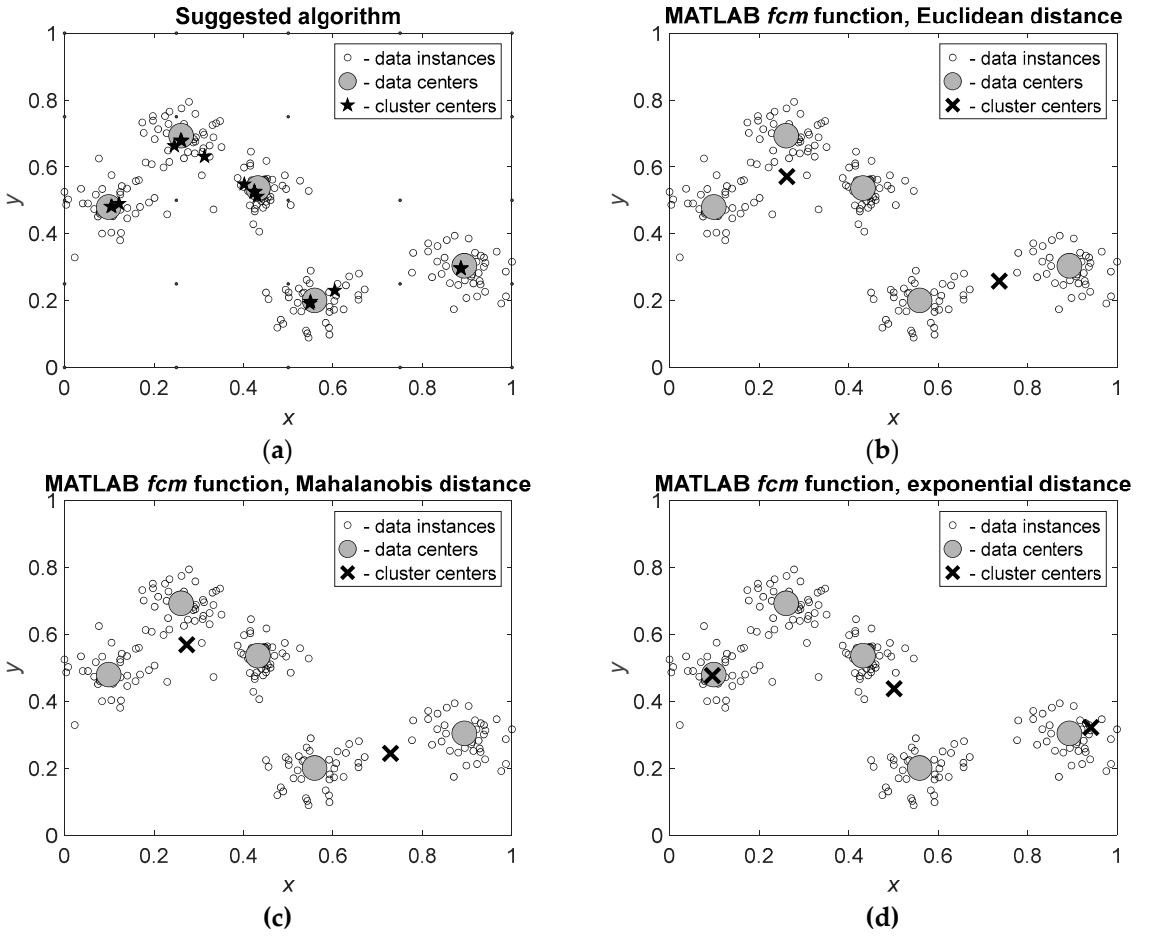


Figure 4. Data with ten predefined clusters and cluster centers calculated by the Algorithms 1 and 2: (**a**) number of clusters $m = 25$; (**b**) number of clusters $m = 9$.

It is seen that the cluster centers calculated by Algorithm 2 are concentrated at the real centers of the clusters and, as above, several centers are located at the same points. Hence, the suggested Algorithm 2 allows more correct definition of the cluster centers and consequently, more correct clustering.

To illustrate the usefulness of the suggested algorithm in recognition of the number of clusters and analysis of the data structure, let us compare the results of the algorithm with the results obtained by the MATLAB® `fcm` function [15] with three possible distance measures: Euclidean distance, Mahalanobis distance [16] and exponential distance [17]. These algorithms were applied to $n = 200$ data instances distributed around 5 centers; the obtained results are shown in Figure 5.



 doi:10.20944/preprints202504.1454.v1

Figure 5. Cluster centers calculated by the suggested Algorithm 2 (**a**) and by the MATLAB® `fcm` function with Euclidean (**b**), Mahalanobis (**c**) and exponential (**d**) distance measures. In all cases, the real number of clusters is 5 and the number of data instances is $n = 200$. Algorithm 2 starts with $m = 25$ clusters and is terminated after 10 iterations, and function `fcm` defines an optimal number of clusters by trials with the number of clusters from 2 to 11.

It is seen that the suggested algorithm (Figure 5.a) correctly recognizes the real centers of the clusters and locates the cluster centers close to these real centers. In contrast, the known algorithms implemented in MATLAB® do not recognize real centers of the clusters and, consequently, do not define correct number of the clusters and locations of their centers. The function `fcm` with Euclidean and Mahalanobis distance measures (Figure 5.b,c) results in two cluster centers and the function `fcm` with exponential distance (Figure 5.d) results in three cluster centers. Note that the use of Euclidean and Mahalanobis distance measures leads to very similar results.

Thus, recognition of the real cluster centers which are the centers of distributions of the data instances can be conducted in two stages. At first, the cluster centers are defined by application of the suggested algorithm to the raw data, and at second, the cluster centers by application of *k*-means to the cluster centers found at the first stage. The resulting cluster centers indicate the real cluster centers.

4. Discussion

The suggested algorithm of fuzzy clustering follows the line of *c*-means fuzzy clustering algorithms and differs from the known methods in the used distance measure.

The suggested fuzzy logical distance is a semi-metric based on the introduced semi-metric in the algebra of truth values with uninorm and absorbing norm. The meaning of these semi-metrics is the following.

Assume that some statements A and B are considered by a group of d observers and each of the observers expresses an opinion about the truthiness of these statements. Assuming that the observers are independent, the statements A and B can be considered as point in the d-dimensional space and then the suggested semi-metric is a distance between these statements.

In other words, the fuzzy logical distance measures allow comparing the statements based on their subjective truthiness.

In the paper, we considered the fuzzy logical distance based on the uninorm and absorbing norm which, in their turn, use the sigmoid generating functions. As a result, the fuzzy logical distance effectively separates the data instances which leads to more precise calculation of the cluster centers and more quick convergence of the algorithm.

An additional advantage of the algorithm is the possibility of tuning its activity by two parameters: neutral element θ and absorbing element ϑ. As indicated above, better results of the clustering can be obtained using non-equal values θ and ϑ. An inequality of these parameters slightly disturbs the semi-metric properties of the distance measures, however, leads to better separation of close but different points.

The weakness of the algorithm is the need of normalization of the data and then renormalization of the obtained clusters centers. However, since both operations are conducted in polynomial time, this disadvantage is not a serious drawback.

The suggested algorithm of fuzzy clustering can be used for solving the clustering problems instead of or together with the known algorithms, for recognition of the centers of distributions of the data instances and can form a basis for development of the methods of comparison of multivariate samples.

 doi:10.20944/preprints202504.1454.v1

Author Contributions: Conceptualization, E.K. and A.N.; methodology, A.R.; software, E.K. and A.N.; validation, E.K., A.N. and A.R.; formal analysis, E.K. and A.R.; investigation, E.K. and A.N.; resources, E.K. and A.N.; data curation, A.N.; writing—original draft preparation, E.K.; writing—review and editing, A.N. and A.R.; supervision, A.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

Appendix includes three supplementary algorithms which are used in the suggested Algorithm 2.

Algorithm 3. Normalization by linear transformation.

Input: data vector $X = (x_1, x_2, \dots, x_n) \subset \mathbb{R}^d,\ n \geq 1,\ d \geq 1.$

Output: normalized data vector $\tilde{X} = (\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_n) \subset [0,1]^d,\ n \geq 1,\ d \geq 1,$ $x_{min},\ \tilde{x}_{max}.$

1. Calculate minimal data value

$$x_{min} = \min_l \left\{ \min_i \{x_{il}\} \right\},\ i = 1, 2, \dots, n,\ l = 1, 2, \dots, d.$$

2. For each $l = 1, 2, \dots, d$ do
3. For each $i = 1, 2, \dots, n$ do

$$\tilde{x}_{il} = x_{il} - x_{min}$$

4. End for
5. End for.
6. Calculate intermediate maximal value

$$\tilde{x}_{max} = \max_l \left\{ \max_i \{\tilde{x}_{il}\} \right\},\ i = 1, 2, \dots, n,\ l = 1, 2, \dots, d.$$

1. For each $l = 1, 2, \dots, d$ do
2. For each $i = 1, 2, \dots, n$ do

$$\tilde{x}_{il} = \tilde{x}_{il} / \tilde{x}_{max}$$

3. End for
4. End for.
5. Return $\tilde{X} = (\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_n),\ x_{min}$ and $\tilde{x}_{max}.$

Algorithm 4. Inverse transform of the normalized vector.

Input: normalized vector $\tilde{X} = (\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_n) \subset [0,1]^d,\ n \geq 1,\ d \geq 1,$ $x_{min},\ \tilde{x}_{max}$

Output: vector $X = (x_1, x_2, \dots, x_n) \subset \mathbb{R}^d,\ n \geq 1,\ d \geq 1.$

1. For each $l = 1, 2, \dots, d$ do
2. For each $i = 1, 2, \dots, n$ do

$$x_{il} = \tilde{x}_{il} \times \tilde{x}_{max} + x_{min}$$

3. End for
4. End for.
5. Return $X = (x_1, x_2, \dots, x_n).$

 doi:10.20944/preprints202504.1454.v1

Algorithm 5. Creating a grid in the square $[0,1]^2$

Input:	precision $\xi \in [0,1]$.
Output:	vector $Y = (y_1, y_2, \dots, y_m) \subset [0,1]^2$, number m of clusters.

1. Set $\tilde{m} = \lceil 1/\xi \rceil$ and $m = \tilde{m}^2$.
2. If $m = 1$ then
3. Set $y_{11} = 1/2$ and $y_{21} = 1/2$,
4. Else
5. For $i = 1$ to m do
6. Set $j = \lceil 1/\tilde{m} \rceil$ and $k = (i-1) \bmod \tilde{m} + 1$,
7. Set $y_{1i} = j$ and $y_{2i} = k$,
8. End for.
9. End if.
10. Return vector $Y = (y_1, y_2, \dots, y_m)$ and number m of clusters.

References

1. Raiffa, H. *Decision Analysis. Introductory Lectures on Choices under Uncertainty*. Addison-Wesley: Reading, MA, USA, 1968.
2. Triantaphyllou, E. *Multi-Criteria Decision Making Methods: A Comparative Study*. Springer Science + Business Media: Dordrecht, Netherlands, 2000.
3. López, L. M.; Ishizaka, A.; Qin, J.; Carrillo, P. A. A. *Multi-Criteria Decision-Making Sorting. Methods Applications to Real-World Problems*. Academic Press / Elsevier: London, UK, 2023.
4. Han, J.; Kamber, M.; Pei, J. *Data Mining: Concepts and Techniques*. Morgan Kaufmann / Elsevier: Waltham, MA, USA, 2012.
5. Ghanaiem, A; Kagan, E.; Kumar, P.; Raviv, T.; Glynn, P.; Ben-Gal, I. Unsupervised classification under uncertainty: the distance-based algorithm. *Mathematics* 2023, 11 (23), 4784.
6. Ratner, N.; Kagan, E.; Kumar, P.; Ben-Gal, I. Unsupervised classification for uncertain varying responses: the Wisdom-In-the-Crowd (WICRO) algorithm. *Knowledge-Based Systems* 2023, 272, 110551.
7. Everitt, B. S.; Landau, S.; Leese, M.; Stahl, D. *Cluster analysis*. Jonh Wiley & Sons: Chichester, UK, 2011.
8. Aggarwal, C. C. An introduction to cluster analysis. In *Data Clustering: Algorithms and Applications*; Aggarwal, C. C., Reddy C. K., Eds.; Chapman & Hall / CRC / Taylor & Francis, 2014.
9. Lloyd, S. P. Least squares quantization in PCM. *IEEE Trans. Information Theory*, 1982, 28(2), 129-137.
10. Forgy, E. W. Cluster analysis of multivariate data: efficiency vs. interpretability of classifications; Abstract. *Biometrics*, 1965, 21(3), 768-769.
11. Hartigan, J. A.; Wong, M. A. A k-means clustering algorithm. *J. Royal Statistical Society, Series C*, 1979, 28(1), 100-108.
12. MATLAB® Help Center. *k-Means Clustering*. Available online: https://www.mathworks.com/help/stats/k-means-clustering.html (accessed on 12.04.2025).
13. Bezdek, J. C. *Fuzzy Mathematics in Pattern Classification*. Ph.D. Thesis, Cornell University, Ithaca, NY, USA, 1973.
14. Bezdek, J. C. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Springer: New York, NY, USA, 1981.
15. MATLAB® Help Center. *Fuzzy Clustering*. Available online: https://www.mathworks.com/help/fuzzy/fuzzy-clustering.html (accessed on 12.04.2025).
16. Gustafson, D.; Kessel, W. Fuzzy clustering with a fuzzy covariance matrix. In *Proceedings of IEEE Conference on Decision and Control*, San Diego, CA, USA, 10-12 Jan 1979.
17. Gath, I.; Geva, A. B. Unsupervised optimal fuzzy clustering. *IEEE Trans. Pattern Analysis and Machine Intelligence* 1989, 11 (7), 773-780.

 doi:10.20944/preprints202504.1454.v1

18. Li, J.; Lewis, H. W. Fuzzy clustering algorithms – review of the applications. In *Proceedings of IEEE International Conference on Smart Cloud*, New York, NY, USA, 18-20 Nov 2016.
19. Höppner, F.; Klawonn, F.; Kruse, R.; Runkler, T. *Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition*. Jonh Wiley & Sons: Chichester, UK, 2019.
20. Kagan, E.; Rybalov, A.; Siegelmann, H.; Yager, R. Probability-generated aggregators. *Int. J. Intelligent Systems* 2013, 28 (7), 709-727.
21. Kagan, E.; Rybalov, A.; Yager, R. *Multi-valued Logic for Decision-Making under Uncertainty*; Springer-Nature / Birkhäuser: Cham, Switzerland, 2025.
22. Yager, R. R.; Rybalov, A. Uninorm aggregation operators. *Fuzzy Sets and Systems* 1996, 80, 111-120.
23. Rudas, I. J. New approach to information aggregation. *Zbornik Radova* 2000, 2, 163–176.
24. Fodor, J.; Yager, R.; Rybalov, A. Structure of uninorms. *Int. J. Uncertainty, Fuzziness and Knowledge-Based Systems* 1997, 411-427.
25. Fodor, J.; Rudas, I. J.; Bede, B. Uninorms and absorbing norms with applications to image processing. In *Proceedings of the 4th Serbian-Hungarian Joint Symposium on Intelligent Systems*, Subotica, Serbia, 29-30 Sept 2006.