## *Fuzzy-Rough Intrigued Harmonic Discrepancy Clustering*

Yue, Guanli; Qu, Yanpeng; Yang, Longzhi; Shang, Changjing; Deng, Ansheng; Chao, Fei; Shen, Qiang

*Published in:*
IEEE Transactions on Fuzzy Systems

*DOI:*
10.1109/TFUZZ.2023.3247912

*Publication date:*
2023

*Citation for published version (APA):*
Yue, G., Qu, Y., Yang, L., Shang, C., Deng, A., Chao, F., & Shen, Q. (2023). Fuzzy-Rough Intrigued Harmonic Discrepancy Clustering. *IEEE Transactions on Fuzzy Systems*, *31*(10), 3305-3318. https://doi.org/10.1109/TFUZZ.2023.3247912

# Fuzzy-Rough Intrigued Harmonic Discrepancy Clustering

Guanli Yue, Yanpeng Qu, Longzhi Yang, Changjing Shang, Ansheng Deng, Fei Chao and Qiang Shen

***Abstract*—Fuzzy clustering decomposes data into clusters using partial memberships by exploring the cluster structure information, which demonstrates comparable performance for knowledge exploitation under the circumstance of information incompleteness. In general, this scheme considers the memberships of objects to cluster centroids and applies to clusters with the spherical distribution. In addition, the noises and outliers may significantly influence the clustering process; a common mitigation measure is the application of separate noise processing algorithms, but this usually introduces multiple parameters which are challenging to be determined for different data types. This paper proposes a new fuzzy-rough intrigued harmonic discrepancy clustering (HDC) algorithm by noting that fuzzy-rough sets offer a higher degree of uncertainty modelling for both vagueness and imprecision present in real-valued datasets. The HDC is implemented by introducing a novel concept of harmonic discrepancy, which effectively indicates the dissimilarity between a data instance and foreign clusters with their distributions fully considered. The proposed HDC is thus featured by a powerful processing ability on complex data distribution leading to enhanced clustering performance, particularly on noisy datasets, without the use of explicit noise handling parameters. The experimental results confirm the effectiveness of the proposed HDC, which generally outperforms the popular representative clustering algorithms on both synthetic and benchmark datasets, demonstrating the superiority of the proposed algorithm.**

***Index Terms*—Rough set, Fuzzy-rough set, Clustering, Harmonic discrepancy.**

## I. Introduction

CLUSTERING refers to dividing existing unlabelled data instances into a number of clusters according to the similarity between objects without any prior information, leading to high inter-cluster similarity and low intra-cluster similarity between instances. Clustering analysis typically uses a precise similarity measure to gauge the similarity between instances and then determines the division of clusters according to specific clustering strategies [1]. A broad spectrum of clustering algorithms have been developed successfully using fuzzy sets and rough sets [2], [3]. They convey two crucial and mutually orthogonal aspects of imprecision implied by data and knowledge; the former qualifies that instances belong to a set to a certain degree, and the latter delivers approximations of concepts under circumstances of incomplete information [4], [5].

Fuzzy clustering offers a soft scheme against the conventional hard measurement methods, which can be generally grouped into three categories according to the types of fuzzy sets: Type-1, Type-2 and Intuitionistic [6]. Amongst the many types of existing fuzzy clustering solutions, fuzzy $c$-means (FCM) is the most representative Type-1 algorithm [7]. Compared with the popular partitional clustering $k$-means [8], FCM classifies the instances into all clusters simultaneously by calculating their (partial) memberships regarding each cluster, which gives the flexibility to consider full, partial or none belonging of a data point to all clusters. Despite its success, FCM is sensitive to noises, outliers, and cluster sizes. Subsequent research has made various improvements, such as possibilistic $c$-means (PCM) [9] and possibilistic fuzzy $c$-means (PFCM) [10]. The representative algorithms of Type-2 fuzzy clustering include T2FCM and kernelised T2FCM (KT2FCM) [11]. Due to the characteristics of Type-2 fuzzy sets, the specific data elements contributing more to the computation of appropriate cluster centroids result in an improvement of FCM, but the challenge remains for the processing of non-spherical and more complex data. By introducing the tangent function and the Lagrangian method, KT2FCM further improves the performance of T2FCM. The third type, Intuitionistic fuzzy set based clustering, merges the hesitation degree and membership, leading to intuitionistic FCM (IFCM), IFCM-$\sigma$, kernelised IFCM (KIFCM) amongst other [12]–[15]. These methods extend conventional FCM by adding intuitionistic features to memberships and objective functions, improving the computational efficiency and clustering performance of non-spherically separable data.

Rough $k$-means (RKM) and its advancements enhances the traditional $k$-means algorithm using the rough set theory [16], such as three-way $k$-means [17], interval Type-2 fuzzy local enhancement based rough $k$-means [16], and spatial rough $k$-means [18]. These algorithms divide the instances that belong to a specific cluster into the lower approximate set and the instances that do not belong to a specific cluster into the boundary set, which well solves the problem of fuzzy and uncertain data clustering and demonstrates a more efficient performance in overlapping datasets. However, RKM uses an artificial setting of fixed weights and thresholds, which may negatively affect the clustering performance in addition to the challenge of determining these parameters. Fuzzy-

G. Yue and A. Deng are with the Information Technology College, Dalian Maritime University, Dalian, 116026, China.

Y. Qu is with the College of Artificial Intelligence, Dalian Maritime University, Dalian, 116026, China. (e-mail: yanpengqu@dlmu.edu.cn).

L. Yang is with the Department of Computer and Information Sciences, Northumbria University, London E1 7HT, U.K.

F. Chao is with the Department of Computer Science, Xiamen University, Xiamen 361005, China.

C. Shang and Q. Shen are with the Department of Computer Science, Faculty of Business and Physical Sciences, Aberystwyth University, Aberystwyth, Ceredigion, SY23 3DB, U.K.

This work is jointly supported by Dalian High-Level Talent Innovation Program (No. 2018RQ70) and a Sêr Cymru II COFUND Fellowship, UK.

Manuscript received; revised

rough $k$-means [19] and rough-fuzzy $k$-means [16] further integrate rough theory and fuzzy theory into FCM and RKM, respectively, to allow the algorithm to enjoy the advantages of both fuzzy clustering and rough clustering.

The occurrences of noisy data points degrade the clustering algorithm significantly. There are two leading solutions to mitigate or address this. One solution uses a separate algorithm to process the noisy data before clustering, such as Gaussian-based statistical detection methods [20], $k$NN distance-based local outliers searching algorithms and density-based detection methods [21]. The other focuses on the reduction of the negative influence of noisy data points in the clustering process. For example, a possibilistic $c$-means (PCM) is proposed to process datasets containing noises and outliers [9], and this approach has a guaranteed convergence [22]. Also, the FCM and PCM algorithms are combined leading to the possibilistic fuzzy $c$-means (PFCM) algorithm; this algorithm is supposed to well handle noises and outliers by its possibilistic terms but avoid coincident clusters and sensitivity to initialisation by its fuzzy terms [10]. However, the experiments do not show much better performance as expected. In addition, an improved PFCM algorithm is presented for noisy data by modifying the objective function of the PFCM algorithm [23]. Although this algorithm is more accurate than the FCM, PCM, and PFCM based on the experimental results, it suffers from high computational complexity and thus long running time.

In addition to the aforementioned methods, which only consider the centroids of clusters, further improvements are made to partitional clustering in view of the distributions of clusters, including inter-cluster and intra-cluster. In [24], a dissimilarity measure is recommended and incorporated for the benefit of considering the inter-cluster difference of clusters. In [25], a new scheme for scaling the membership degrees of the chosen samples is suggested to boost the effect of the in-cluster samples and to weaken the effect of the out-of-cluster samples in the clustering process. This scheme not only accelerates the convergence of the algorithm but also maintains the high clustering quality. When it comes to the intra-cluster distribution, in [26], an elastic fuzzy c-means (EFCM) is proposed to better recognise intrinsic cluster structure. EFCM provides a sparser description for reliable points and a fuzzier description for marginal points of clusters, thus, the roles of reliable and margin points are more balance. In [27], Gaussian mixture model and collaborative technology are combined with FCM to enhance the ability of recognising the distribution of intra-cluster. This approach is effective in dealing with noise, non-spherical clusters, size-imbalanced clusters. In [28], the local densities of instances in intra-cluster are considered in FCM, and the instances with the local maximum density are used as the initial centroids to improve the stability of FCM.

This paper proposes a new concept of harmonic discrepancy to allow the full consideration of the distributions of clusters when evaluating the dissimilarity between a data instance and foreign clusters. In addition, a new cluster centroid updating scheme is proposed by ignoring the abnormal data elements of a cluster during the cluster centroid updating process. These jointly leads to a novel fuzzy-rough intrigued harmonic discrepancy clustering (HDC) algorithm in an effort to address the aforementioned challenges. The proposed HDC algorithm is applied to a set of synthetic and benchmark datasets and gone through a comparative study by employing existing popular clusters. The experimental results confirm a better stability of the proposed HDC algorithm on real-world datasets in comparison with its competitors. The contribution of the paper is threefold:

- Proposing the novel concept of harmonic discrepancy through an innovative application of fuzzy-rough approximation to enable the comprehension of cluster distributions during cluster centroid updating,
- Developing a non-parameterised noise and outlier processing method which effectively reduces the negative impact of abnormal data instances in clustering and improves the practical applicability;
- Establishing the HDC algorithm with its superiority confirmed through a comparative study and statistical analysis.

The remainder of the paper is structured as follows. Section II briefly reviews the preliminaries of the rough set and fuzzy-rough set. The proposed harmonic discrepancy clustering algorithm is described in Section III. Results of comprehensive experiments are presented in Section IV, leading to conclusions in Section V.

## II. Preliminaries

This section reviews the concepts concerning rough sets and fuzzy-rough sets which underpins the proposed discrepancy metric.

### A. Rough Set

The rough set theory provides a methodology to extract knowledge from a domain in a concise way by minimising information loss whilst reducing the amount of information involved [16]. Central to rough set theory is the concept of indiscernibility. Let $(\mathbb{U}, \mathbb{A})$ be an information system, where $\mathbb{U}$ is a set of objects and $\mathbb{A}$ is a set of attributes such that $a : \mathbb{U} \to V_a$ for every $a \in \mathbb{A}$; $V_a$ is the set of values that attribute $a$ may take. For each feature subset $P \subseteq \mathbb{A}$, an associated $P$-indistinguishable relation can be determined by:

$$IND(P) = \{(x, y) \in \mathbb{U}^2 \mid \forall a \in P, a(x) = a(y)\}. \quad (1)$$

Obviously, $IND(P)$ is an equivalence relation on $\mathbb{U}$. The partition of $\mathbb{U}$ determined by $IND(P)$ is herein denoted by $\mathbb{U}/P$ which can be defined as:

$$\mathbb{U}/P = \otimes\{\mathbb{U}/a | a \in P\}, \quad (2)$$

where $\otimes$ regarding fuzzy sets $A$ and $B$ is defined as follows:

$$A \otimes B = \{X \cap Y | X \in A, Y \in B, X \cap Y \neq \varnothing\}. \quad (3)$$

For any object $x \in \mathbb{U}$, the equivalence class determined by $IND(P)$, is denoted by $[x]_P$. Let $X \subseteq \mathbb{U}$. $X$ can be approximated using only the information contained in $P$ by constructing the $P$-lower and $P$-upper approximations of $X$ [29]:

$$\underline{P}X = \{x \mid [x]_P \subseteq X\}, \quad (4)$$

$$\overline{P}X = \{x \mid [x]_P \cap X \neq \emptyset\}. \tag{5}$$

The pair $\langle \underline{P}X, \overline{P}X \rangle$ is called a rough set. Informally, the former depicts the set of those objects which can be said with certainty to belong to the concept to be approximated, and the latter is the set of objects which either definitely or possibly belong to the concept to be approximated. The difference between the upper and lower approximations is the area known as the boundary region and thus, representing the area of uncertainty. When the boundary region is empty, there is no uncertainty regarding the concept which is being approximated and all objects belong to the subset of objects of interest with full certainty.

### B. Fuzzy-Rough Set

Fuzzy-rough sets encapsulate the related but distinct concepts of vagueness (usually concerned by fuzzy sets) and indiscernibility (usually concerned by rough sets) [30], both of which occur as a result of uncertainty in data or knowledge. Compared to rough sets, fuzzy-rough sets offer a higher degree of flexibility in enabling the vagueness and imprecision present in real-valued data to be simultaneously and effectively modelled. In fuzzy-rough sets, the fuzzy lower and upper approximations to approximate a fuzzy concept $X$ can be defined as:

$$\mu_{\underline{R_P}X}(x) = \inf_{y \in \mathbb{U}} \mathcal{I}(\mu_{R_P}(x,y), \mu_X(y)), \tag{6}$$

$$\mu_{\overline{R_P}X}(x) = \sup_{y \in \mathbb{U}} \mathcal{T}(\mu_{R_P}(x,y), \mu_X(y)), \tag{7}$$

where $\mathcal{I}$ is a fuzzy implicator and $\mathcal{T}$ is a $T$-norm. $R_P$ is a $T$-transitive fuzzy similarity relation induced by the subset of features $P$:

$$\mu_{R_P}(x,y) = \mathcal{T}_{a \in P}\{\mu_{R_a}(x,y)\}, \tag{8}$$

where $\mu_{R_a}(x,y)$ represents the degree to which objects $x$ and $y$ are similar to each other based on feature $a$. This degree may be defined in a number of ways such as:

$$\mu_{R_a}(x,y) = 1 - \frac{|a(x) - a(y)|}{|a_{max} - a_{min}|}, \tag{9}$$

$$\mu_{R_a}(x,y) = \exp(\frac{-(a(x) - a(y))^2}{\delta_a^2}), \tag{10}$$

where $\delta_a^2$ indicates the variation for feature $a$. The fuzzy lower and upper approximations express the same physical meaning with their crisp counterparts. In particular, $\mu_{\underline{R_P}X}(x)$ shows the extent to which the object $x$ must belong to the approximated fuzzy concept $X$, whilst $\mu_{\overline{R_P}X}(x)$ represents the extent to which the object $x$ may belong to the approximated fuzzy concept $X$.

## III. Fuzzy-Rough Intrigued Harmonic Discrepancy Clustering

The existing partitional clustering algorithms, e.g., $k$-means [8] and FCM [7], group an instance into a cluster if it a full or the highest membership as induced by the nearest prototype or expectation of the cluster. These types of clustering methods are usually performed by considering the memberships of the objects to the cluster centroids and ignoring the distributions of the clusters; and a small amount of noisy data points or outliers can have significant, and often negative, impact to the clustering results. A novel harmonic discrepancy clustering (HDC) strategy is presented in this section to ease the restriction of the partitional clustering algorithms by an innovative application of fuzzy-rough sets, for a sound and robust clustering performance.

### A. Discrepancy Inspired by Fuzzy-Rough Set

In this paper, discrepancy refers to the degree of separating an object from a cluster. Given an information system $(\mathbb{U}, \mathbb{A})$, suppose that there are $n$ instances, i.e., $\mathbb{U} = \{x_1, \ldots, x_n\}$, and $k$ clusters will be partitioned, including $C_1, \ldots, C_k$. The degree to which a data instance $x_i$ belongs to a cluster $C_j$ with regard to attributes $\mathbb{A}$ can be gauged by the fuzzy lower approximation $\mu_{\underline{R_\mathbb{A}}C_j}(x_i)$.

Fuzzy implication $\mathcal{I}$ calculates the fulfillment degree of a fuzzy rule:

$$\text{IF } p \text{ is } X \text{ THEN } q \text{ is } Y, \tag{11}$$

where the antecedent ($p$ is $X$) and the consequence ($q$ is $Y$) are fuzzy. For any fuzzy implication , it holds that:

$$\mathcal{I}(p,1) = 1. \tag{12}$$

That is, if the consequence establishes in any case (i.e., $q = 1$), the truth value of the fuzzy rule (11) is 1. Due to Eq. (12), $\mu_{\underline{R_\mathbb{A}}C_j}(x_i)$ can be simplified into:

$$\begin{aligned}
&\mu_{\underline{R_\mathbb{A}}C_j}(x_i) \\
=&\inf_{y \in \mathbb{U}} \mathcal{I}(\mu_{R_\mathbb{A}}(x_i,y), \mu_{C_j}(y)) \\
=&\min\left\{ \min_{y \in C_j} \left\{\mathcal{I}\left(\mu_{R_\mathbb{A}}\left(x_i,y\right),1\right)\right\}, \min_{y \notin C_j} \left\{\mathcal{I}\left(\mu_{R_\mathbb{A}}\left(x_i,y\right),0\right)\right\} \right\} \\
=&\min_{y \notin C_j} \left\{\mathcal{I}\left(\mu_{R_\mathbb{A}}\left(x_i,y\right),0\right)\right\}.
\end{aligned} \tag{13}$$

Moreover, let $\mathcal{N}$ be a strong negation (i.e., a continuous, strictly decreasing, involutive function such that $\mathcal{N}(0) = 1$), $\mathcal{I}$ is contrapositive symmetry with respect to $\mathcal{N}$ if and only if

$$\mathcal{I}(p,q) = \mathcal{I}(\mathcal{N}(p), \mathcal{N}(p)). \tag{14}$$

As proved in [31], if $\mathcal{I}$ belongs to $S$-implications, $QL$-implications or $R$-implications which enjoys the contrapositive symmetry, the equation $\mathcal{I}(x,0) = \mathcal{N}(x)$ holds with $\mathcal{N}$ being a strong negator to induce $\mathcal{I}$. By considering the classical strong negation $\mathcal{N}_C(x) = 1 - x$, Eq. (13) can be further modified to:

$$\mu_{\underline{R_\mathbb{A}}C_j}(x_i) = \min_{y \notin C_j} \left\{1 - \left(\mu_{R_\mathbb{A}}\left(x_i,y\right)\right)\right\}. \tag{15}$$

In particular, typical fuzzy implicators of $S$-implications, $QL$-implications or $R$-implications include but not limited to:

- Łukasiewicz implicator: $\mathcal{I}_L(p,q) = \min(1-p+q, 1)$,
- Kleene-Dienes implicator: $\mathcal{I}_{KD}(p,q) = \max(1-p, q)$,
- Reichenbach implicator: $\mathcal{I}_R(p,q) = 1-p+pq$,
- Zadeh implicator: $\mathcal{I}_Z(p,q) = \max(1-p, \min(p,q))$,
- Willmott implicator: $\mathcal{I}_W(p,q) = \min(\max(1-p, q), \max(p, 1-q), \min(q, 1-p))$,
- Klir-Yuan implicator: $\mathcal{I}_{KY}(p,q) = 1-p+p^2q$.

Respectively, by replacing "$\min$" and "$y \notin C_j$" in Eq. (15) with "$\max$" and "$y \in C_j$", the discrepancy of a data instance $x_i$ in reference to a cluster $C_j$ with regard to attributes $\mathbb{A}$ can be expressed as:

$$\xi_{R_\mathbb{A}C_j}(x_i) = \max_{y \in C_j}\{1 - \mu_{R_\mathbb{A}}(x_i, y)\}. \tag{16}$$

It makes intuitive sense that the discrepancy function indicates the degree to which the most dissimilar data instance $y$ in cluster $C_j$ to the referencing data instance $x_i$. In fact, the discrepancy function as given in Eq. (16) can be deemed as the max-link distance [32] between $x_i$ and $C_j$. However, if a cluster suffers from a decentralised distribution, the discrepancy of a data instance to it may undergo a high probability of inaccuracy. In this case, the discrepancy function as expressed in Eq. (16) can be improved by taking into account the distribution of cluster $C_j$.

Let $M = [m_{ij}]_{n\times k}$ be the partition matrix of $\mathbb{U}$, i.e.,

$$m_{ij} = \begin{cases} 0, & x_i \notin C_j, \\ 1, & x_i \in C_j. \end{cases} \tag{17}$$

Each element of $m_{ij} = 1$ indicate that the $i$-th instance is assigned to the $j$-th cluster. Based on $M$, the centroid $c_j$ of cluster $C_j$ can be calculated by:

$$c_j = \frac{\sum_{i=1}^{n} m_{ij}x_i}{\sum_{i=1}^{n} m_{ij}}, \ j = 1, \ldots, k. \tag{18}$$

Since the centroid of a cluster represents the expectation of the instances belonging to this cluster, in this paper, the distribution of cluster $C_j$ is approximated via the membership of $c_j$ to $C_j$. If cluster $C_j$ enjoys a compact distribution, the extent to which $c_j$ belongs to $C_j$ is supposed to be large accordingly. In the light of the concept of fuzzy-rough sets, this membership can be represented by fuzzy upper approximation $\mu_{\overline{R_\mathbb{A}}C_j}(c_j)$.

$T$-norm $\mathcal{T}$ generalises the logical conjunction to fuzzy logic. For two fuzzy variables $p$ and $q$, $\mathcal{T}(p,q)$ indicates an "and" operator to metric a unified truth degree when $p$ and $q$ are established at the same time. For any $T$-norm operator $\mathcal{T}$, it holds that:

$$\mathcal{T}(p, 1) = p, \tag{19}$$

$$\mathcal{T}(p, 0) = 0. \tag{20}$$

Eq. (19) indicates that if a term $q$ is established in any case (i.e., $q = 1$), the degree of meeting both $p$ and $q$ depend on the value of $p$. Eq. (20) indicates that if a term $q$ acts as null element (i.e., $q = 0$), the chance to meet both $p$ and $q$ is 0. Due to Eqs. (19) and (20), $\mu_{\overline{R_\mathbb{A}}C_j}(c_j)$ can be simplified to:

$$\begin{aligned} &\mu_{\overline{R_\mathbb{A}}C_j}(c_j) \\ =&\sup_{y\in\mathbb{U}} \mathcal{T}\left(\mu_{R_\mathbb{A}}(c_j, y), \mu_{C_j}(y)\right) \\ =&\max\left\{\max_{y\in C_j}\{\mathcal{T}\left(\mu_{R_\mathbb{A}}(c_j, y), 1\right)\}, \max_{y\notin C_j}\{\mathcal{T}\left(\mu_{R_\mathbb{A}}(c_j, y), 0\right)\}\right\} \\ =&\max_{y\in C_j}\{\mathcal{T}\left(\mu_{R_\mathbb{A}}(c_j, y), 1\right)\} \\ =&\max_{y\in C_j}\{\mu_{R_\mathbb{A}}(c_j, y)\}. \end{aligned} \tag{21}$$

Based on Eq. (21), $\mu_{\overline{R_\mathbb{A}}C_j}(c_j)$ can be interpreted as the similarity of $c_j$ to its nearest neighbour in cluster $C_j$. Therefore, it is rational to use this metric as the indicator of cluster distribution.

To synthesise the roles of both Eqs. (16) and (21), a representative object $\tilde{y}_{ji}$ is sought to metric the harmonic discrepancy (HD) of $x_i$ to cluster $C_j$, which is designed as follows:

$$\tilde{y}_{ji} = \underset{y\in C_j}{\operatorname{argmax}}\{\frac{2\times(1-\mu_{R_\mathbb{A}}(x_i,y))\times\mu_{R_\mathbb{A}}(c_j,y)}{(1-\mu_{R_\mathbb{A}}(x_i,y))+\mu_{R_\mathbb{A}}(c_j,y)}\}. \tag{22}$$

The harmonic average of $1-\mu_{R_\mathbb{A}}(x_i,y)$ and $\mu_{R_\mathbb{A}}(c_j,y)$ in Eq. (22), is illustrated in Fig. 1. It can be seen that by maximising this harmonic average, both $1-\mu_{R_\mathbb{A}}(x_i,y)$ or $\mu_{R_\mathbb{A}}(c_j,y)$ can be guaranteed large values, which are consistent with Eqs. (16) and (21), respectively. Therefore, Eq. (22), can locate a sample $y \in C_j$ which is distant to $x_i$ (i.e., a large value of $1-\mu_{R_\mathbb{A}}(x_i,y)$) but close to the centroid $c_j$ of cluster $C_j$ (i.e., a large value of $\mu_{R_\mathbb{A}}(c_j,y)$), from both separability and rationality perspectives.

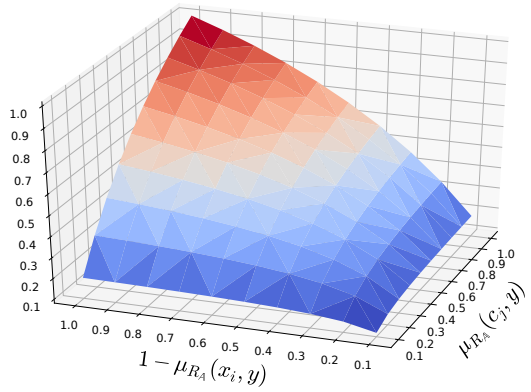


Fig. 1. Harmonic average of $1-\mu_{R_\mathbb{A}}(x_i,y)$ and $\mu_{R_\mathbb{A}}(c_j,y)$.

With the support of Eq. (22), the HD value of $x_i$ to cluster $C_j$ can be expressed as:

$$\delta_{R_\mathbb{A}C_j}(x_i) = 1 - \mu_{R_\mathbb{A}}(x_i, \tilde{y}_{ji}). \tag{23}$$

### *B. Anomaly Reduction*

Despite the comprehensive strategy of HD to distinguish the degrees of data instances belonging to a cluster, its efficacy may still be compromised by the anomalies associated with the decentralised distribution of the cluster. The misclustering

of the peripheral objects always triggers a chain of reactions in subsequent iterations, resulting in unexpected centroid deviation and thus poor clustering results. A novel cluster centroid updating strategy is therefore proposed with an aim to reduce the negative effects of peripheral objects. In particular, the core and peripheral objects are distinguished after each iteration, and the centroid update will depend only on the core objects by ignoring the peripheral instances.

To identify the peripheral objects of cluster $C_j$, $j \in \{1, \ldots, k\}$, the HD acceptance threshold of a data instance to a cluster $C_j$ is defined as:

$$\epsilon_j = ave(\delta_{R_{\mathbb{A}}C_j}(x)) + std(\delta_{R_{\mathbb{A}}C_j}(x)), \tag{24}$$

where $ave(\delta_{R_{\mathbb{A}}C_j}(x))$ and $std(\delta_{R_{\mathbb{A}}C_j}(x))$ represent the average and the standard deviation of $\delta_{R_{\mathbb{A}}C_j}(x)$ for all $x \in C_j$, respectively. With the use of this threshold $\epsilon_j$, if a sample $x \in C_j$ suffers from $\delta_{R_{\mathbb{A}}C_j}(x) > \epsilon_j$, its affiliation with $C_j$ can be considered as unreliable, naturally. Therefore, all such instances are regarded as the peripheral objects of $C_j$; otherwise, they are labelled as a member of the core set. By omitting the peripheral objects, if there is any, during the calculation of the centroid of each cluster, the likelihood of the occurrence of the offset centroid is mitigated.

The peripheral object detection procedure is outlined in Algorithm 1. The main structure of the procedure is a loop over the $k$ clusters to be identified, as shown between Lines 1 and 10. Within this loop, the current clustering is provided in Line 2, and acceptance threshold $\epsilon_j$ of $C_j$ is calculated by using Eq. (24) in Line 3. The inner loop between Lines 4 and 9 compares the HD value of each instance in cluster $C_j$ with the value of the calculated threshold $\epsilon_j$ to determine whether the instance is a peripheral object. Form this, the memberships of all identified peripheral objects $m_{ij}$ are set to 0, to bypass these objects in the calculation of the centroid of $C_j$ in the next iteration. After the main loop, the algorithm returns the updated partition matrix $M$.

**Algorithm 1** Peripheral Object Detection (POD)

**POD** ($\mathbb{U}$, $M$, $k$, $\delta$)
**Input:**
$\mathbb{U}$, input space containing $n$ objects,
$M$, partition matrix,
$k$, number of clusters,
$\delta$, set of HD.
**Output:** $M$, partition matrix.

1: **foreach** $j = 1$ to $k$ **do**
2: $C_j \leftarrow$ set of instances where $m_{ij} = 1, i = 1, ..., n$.
3: $\epsilon_j = ave(\delta_{R_{\mathbb{A}}C_j}(x_i)) + std(\delta_{R_{\mathbb{A}}C_j}(x_i)), (\forall x_i \in C_j)$.
4: **foreach** $x_i$ in $C_j$ **do**
5: **if** $\delta_{R_{\mathbb{A}}C_j}(x_i) > \epsilon_j$ **then** $m_{ij} = 0$
6: **else**
7: **continue**
8: **end**
9: **end**
10: **end**
11: **return** $M$

### C. Harmonic Discrepancy Clustering

To avoid the accident that the centroid are initialised in the peripheral region of data, the random partition (RP) algorithm [33] is used to initialise clusters. As shown in Algorithm 2, rather than straightly initialises centroids. the RP algorithm randomly assigns each instance to a cluster by initialising the partition matrix. In so doing, the RP algorithm avoids selecting outliers to act as centroids from the border areas. And the centroids, resulted from the initialised partition matrix, are concentrated in the central area of the data due to the averaging.

**Algorithm 2** Random Partition (RP)

**RP** ($\mathbb{U}$, $k$)
**Input:**
$\mathbb{U}$, input space containing $n$ objects,
$k$, number of clusters,
**Output:** $M$, partition matrix.

1: **Initialise** $M = \{m_{ij}\}_{n \times k}$, where $m_{ij} = 0$.
2: **foreach** $i = 1$ to $n$ **do**
3: $j \leftarrow$ a random integer in $[1, k]$.
4: $m_{ij} = 1$
5: **end**
6: **return** $M$

Inspired by fuzzy-rough sets, the membership of an instance belonging to a cluster can be obtained from the discrepancy expressing the degree of separating the instance from other clusters. Intuitively, the more significant discrepancy of an instance in reference to other clusters, the greater the membership of this instance to the current computing cluster. According to the definition of HD as expressed in Eq. (23), the membership of $x_i$ to $C_j$ can be defined as:

$$\tau_{R_{\mathbb{A}}C_j}(x_i) = \frac{1}{k-1} \sum_{l \neq j} \delta_{R_{\mathbb{A}}C_l}(x_i). \tag{25}$$

With the support of Eq. (25), the proposed HDC algorithm is summarised as Algorithm 3. Firstly, Line 1 initialises the iteration counter $iter$ as 0. In Line 2, the partition matrix $M$ is initialised by the RP algorithm. Next, a referencing partition matrix $T$ indicating the partition result of the current iteration is prepared for future use (as the algorithm will terminate when there is no change on clusters between two consecutive iterations) in Line 3.

Lines 4-20 show the overall iterative process for clustering. The HD values of all instances in reference to all clusters is obtained by the inner loop expressed in Lines 5-11. After computing the discrepancy values, the memberships of all instances to each cluster are calculated by applying the HD values to Eq. (25) as depicted in Line 12. Then, the partition matrix can be readily updated by the following equation:

$$m_{ij} = \begin{cases} 1 & j = \underset{1 \leq j \leq k}{\operatorname{argmax}} \{\tau_{R_P C_j}(x_i)\}, \\ 0 & \text{else.} \end{cases} \tag{26}$$

By using Eq. (26), both the core objects and the peripheral objects detected via Algorithm 1, are assigned into the clusters with the most significant memberships Eq. (25).

**Algorithm 3** Harmonic Discrepancy Clustering (HDC)

**HDC** ($\mathbb{U}$, $k$, $Max$)
**Input:**
$\mathbb{U}$, input space containing $n$ objects,
$k$, number of clusters,
$Max$, maximum iterations.
**Output:** $M$, final partition matrix.

1: **Initialise** $iter = 0$.
2: $M =$ **RP** ($\mathbb{U}$, $k$) //Algorithm 2
3: $T = M$
4: **repeat**
5: **foreach** $j = 1$ to $k$ **do**
6: $c_j = \sum_{i=1}^{n} m_{ij}x_i / \sum_{i=1}^{n} m_{ij}$
7: **foreach** $i = 1$ to $n$ **do**
8: $\tilde{y}_{ji} \leftarrow$ Eq. (22)
9: $\delta_{R_{\mathbb{A}}C_j}(x_i) = 1 - \mu_{R_{\mathbb{A}}}(x_i, \tilde{y}_{ji})$
10: **end**
11: **end**
12: $M \leftarrow$ Eqs. (25) and (26) //Update $M$
13: **if** $iter == Max$ or $T == M$ **then**
14: **break**
15: **else**
16: $T = M$
17: $M =$ **POD** ($\mathbb{U}$, $M$, $k$, $\delta$) //Algorithm 1
18: **end**
19: $iter = iter + 1$
20: **end**
21: **return** $M$

The algorithm will jump out of the loop either after reaching the maximum number of iterations or the clusters are stabilised, as controlled in Lines 13 and 14; otherwise, the algorithm will move to Lines 16 and 17, which reset the referencing partition matrix $T$ and invokes Algorithm 1 to detect peripheral objects. Correspondingly, the $m_{ij}$ values of the detected peripheral objects in the partition matrix are all set to 0, so as to avoid the influence of such objects in the next iteration of centroid update. In so doing, the peripheral objects are detected progressively over iterations. Instances that have been assigned as peripheral objects in the past iteration may become core objects in the subsequent iterations. Likewise, the previously identified core objects may be transferred to the peripheral set. Regardless of the shifts, the centroid update always relies only on the core set, to maximally guarantee that the centroids resulted from each iteration has a minimal influence from the outliers or other peripheral instances.

Finally, the stabilisation of clusters is examined by comparing the partition matrices between the current iteration and the previous iteration in this work. If the partition matrices between two consecutive iterations are exactly the same, the algorithm will terminate.

The proposed algorithm is able to automatically identify outliers without the use of any pre-defined parameters and prior knowledge about outliers. Thus it can effectively avoid the often negative influence of peripheral objects, but require additional computational resources for this extra functionality. In HDC, each data instance needs to traverse all elements of all clusters when finding the largest memberships. The time complexity of this part is $O(n^2)$. The algorithm needs to be iterated $t$ times, and denoising is performed in each iteration, which computationally contributes to $O(nk)$. Note that $k$ is usually much smaller than $n$, so the final time complexity of the proposed HDC algorithm is $O(n^2t)$.

To illustrate the proposed HDC algorithm, some exemplar instances are given in Table I and displayed in Fig. 2.

TABLE I
THE EXEMPLAR INSTANCES

| Sample | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ | $x_7$ | $x_8$ | $x_9$ | $x_{10}$ | $x_{11}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $x$-axis | 2.3 | 0.8 | 1.5 | 1.3 | 1.4 | 1.1 | 2.3 | 2.5 | 2.0 | 2.0 | 2.0 |
| $y$-axis | 0.5 | 3.0 | 2.0 | 3.2 | 2.4 | 2.6 | 2.4 | 3.0 | 3.2 | 1.8 | 2.2 |

| Sample | $x_{12}$ | $x_{13}$ | $x_{14}$ | $x_{15}$ | $x_{16}$ | $x_{17}$ | $x_{18}$ | $x_{19}$ | $x_{20}$ | $x_{21}$ | $x_{22}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $x$-axis | 4.0 | 5.6 | 3.5 | 4.5 | 5.7 | 4.0 | 5.0 | 9.4 | 11.7 | 12.2 | 12.8 |
| $y$-axis | 5.0 | 1.8 | 4.5 | 4.0 | 2.5 | 3.0 | 2.5 | 7.75 | 7.0 | 3.6 | 3.6 |

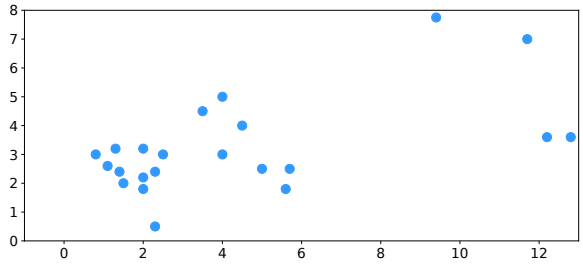


Fig. 2. The unlabelled instances.

Let the number of clusters to be 2. By using the RP algorithm, the partition matrix $M$ and the associated unlabelled instances are initialised as shown in Table II and Fig. 3, respectively. Specifically, in Fig. 3, two pentacles represent the resulting centroids of the respective clusters. Due to the use of the RP algorithm, both of these two centroids locate in the central region of the data, roughly. Thus, the risk of the initialised centroids falling into the border is reduced, effectively.

TABLE II
THE INITIALISATION OF THE PARTITION MATRIX $M$

| $j$ \ $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| 2 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

| $j$ \ $i$ | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 2 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |

Given the initialisation in Fig. 3, by taking Line 17, i.e. the POD algorithm, out of the proposed HDC algorithm, the resulting clusters are illustrated in Fig. 4. Here, the HDC algorithm is implemented by using the Algebraic $T$-norm:

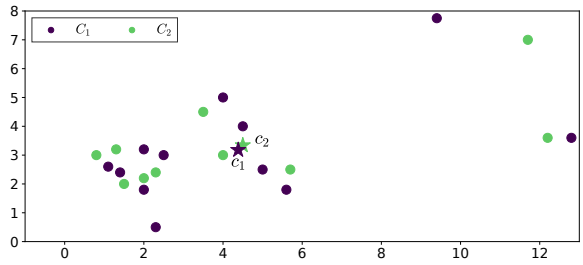


Fig. 3. Initialisation via random partition.

$\mathcal{T}_P(a,b) = ab$ and the fuzzy similarity in Eq. (10). It can be observed that, without the process of anomaly reduction, a small number of peripheral objects form a independent category and their underlying connections to other instances are ignored.

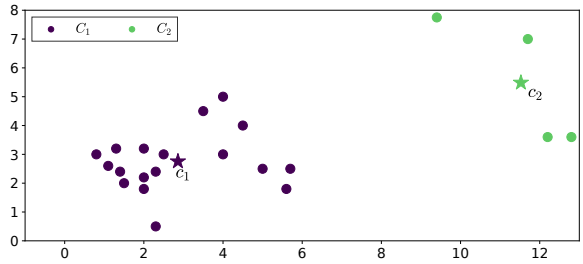


Fig. 4. Clusters without using the POD algorithm.

On the contrary, when using the complete HDC algorithm, $C_2$ in Fig. 4 is merged with the right part of $C_1$ in Fig. 4, as depicted in Fig. 5. And the HD values of all instances at the last iteration are summarised in Table III.

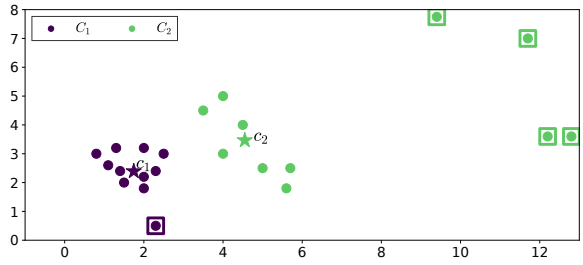


Fig. 5. Clusters by using the complete HDC algorithm.

TABLE III
HD OF THE INSTANCES IN $C_1$ AND $C_2$

| Sample | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ | $x_7$ | $x_8$ | $x_9$ | $x_{10}$ | $x_{11}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $\delta_{R_{\mathbb{A}}C_1}(x_i)$ | 0.77 | 0.37 | 0.27 | 0.35 | 0.20 | 0.25 | 0.30 | 0.32 | 0.30 | 0.37 | 0.27 |
| Sample | $x_{12}$ | $x_{13}$ | $x_{14}$ | $x_{15}$ | $x_{16}$ | $x_{17}$ | $x_{18}$ | $x_{19}$ | $x_{20}$ | $x_{21}$ | $x_{22}$ |
| $\delta_{R_{\mathbb{A}}C_2}(x_i)$ | 0.72 | 0.65 | 0.75 | 0.65 | 0.75 | 0.52 | 0.64 | 0.99 | 0.99 | 0.99 | 0.99 |

By applying Eq. (24), the acceptance thresholds of these instances are

$$ave(\delta_{R_{\mathbb{A}}C_1}(x)) = 0.34,\ std(\delta_{R_{\mathbb{A}}C_2}(x)) = 0.14 \Rightarrow \epsilon_1 = 0.49.$$
$$ave(\delta_{R_{\mathbb{A}}C_2}(x)) = 0.78,\ std(\delta_{R_{\mathbb{A}}C_2}(x)) = 0.17 \Rightarrow \epsilon_2 = 0.96.$$

As shown in Table III, the HD value of $x_1$ is larger than $\epsilon_1$ and those of $x_{19}$, $x_{20}$, $x_{21}$ and $x_{22}$ are larger than $\epsilon_2$. Therefore, these instances are deemed respective peripheral objects for $C_1$ and $C_2$ and marked with squares in Fig. 5. The remaining samples are the members in the core set of $C_1$ and $C_2$. This results demonstrates the ability of HDC to detect peripheral objects and exploit the latent cluster structures.

## IV. Experimental Evaluation

The experimental processes and results are reported in four parts in this section. The specification of the experiment is detailed first. Then, the proposed HDC algorithm is applied to lane segmentation with the clustering results visually analysed. This is followed by a comparative study of HDC in reference to other competitive methods on a set of benchmark datasets. Finally, a statistical analysis is performed to show any statistical significance between different approaches. The codes of HDC can be downloaded from the Github release page[1].

### A. Experimental Setup

All datasets used in this work are derived from the Google image and UCI[2] repository, including three common lane images, and the benchmark datasets *Iris*, *Heart*, *Led7digit*, *Glass*, *Newthyroid*, *Seeds*, *Hepatitis*, *Breast* and *Wine*. Specifically, the lane images include a solid line image (i.e., *Image1*), a mixture of solid and dashed lines image (i.e., *Image2*), and a curve line image (i.e., *Image3*) as shown in Fig. 6, are employed to test the practical applicability of HDC in lane line segmentation. The lane lines in these images are extracted from the original location and represented as two-dimensional datasets. The corresponding results are visualised as subfigures below the original images in Fig. 6. Overall, the details of the used datasets are summarised in Table IV.

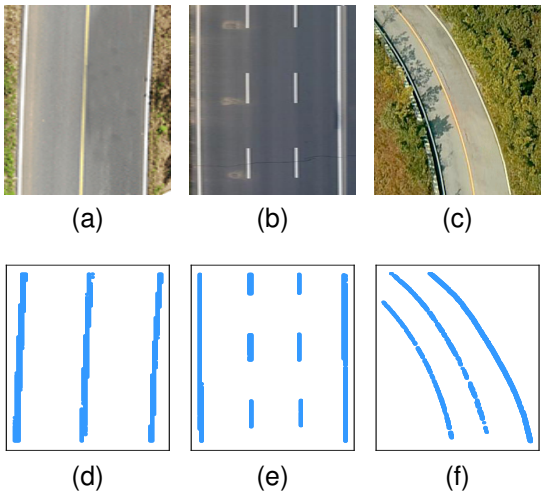


Fig. 6. Original road images and extracted road separation lines. (a) *Image1*. (b) *Image2*. (c) *Image3*. (d) *Road1*. (e) *Road2*. (f) *Road3*.

[1]https://github.com/guanliyue/hdc
[2]https://archive.ics.uci.edu/ml/datasets.php

TABLE IV
DATASETS USED FOR EVALUATION

| Datasets | Attributes | Class | Size |
|---|---|---|---|
| *Road1* | 2 | 3 | 940 |
| *Road2* | 2 | 4 | 1117 |
| *Road3* | 2 | 3 | 1486 |
| *Iris* | 4 | 3 | 150 |
| *Heart* | 13 | 2 | 270 |
| *Led7digit* | 7 | 10 | 500 |
| *Glass* | 9 | 7 | 214 |
| *Newthyroid* | 5 | 3 | 215 |
| *Seeds* | 7 | 3 | 210 |
| *Hepatitis* | 19 | 2 | 80 |
| *Breast* | 9 | 2 | 683 |
| *Wine* | 13 | 3 | 178 |

For a comprehensive evaluation of the effect of the proposed method, four types of algorithms are employed in a comparative study: 1) partitional clustering algorithms, including $k$-means [8], Mean Shift (MS) [34], FCM [7], PCM [9], Enhanced PCM (EPCM) [35], Interval Type-2 Possibilistic FCM (IT2PFCM) [36]; 2) hierarchy-based Agglomerative Clustering (AC) [37]; 3) density-based methods, including Density Peak Clustering (DPC) [38] and Density-based Spatial Clustering of Applications with Noise (DBSCAN) [39]; 4) ensemble clustering methods, specifically Spectral Ensemble Clustering (SEC) [40] and Locally Weighted Evidence Accumulation (LWEA) [41]. In particular, PCM, EPCM and IT2PFCM enjoy the ability to reduce of the negative influence of noisy data points in the clustering process.

Note that redundant features may present in the original datasets. Principal Component Analysis (PCA) [42] is used for all datasets. For the lane image datasets, the first two principal components are extracted to identify the underlying dependencies and reduce the feature correlation of the two road line dimensions. For benchmark datasets, the accumulative contribution rate is set to 90%. To ensure the fairness of the experiment, this settings are used for all comparison algorithms. Also, all instances are randomised and standardised to ensure that clustering results are not affected by the order of data instances.

The parameter settings of the compared algorithms are implemented based on the recommendation in the original publications or optimal settings in the parameters pool. There are no extra parameters for $k$-means except the number of clusters. For MS, the maximum number of iterations is set to 300. The density peaks for DPC are selected automatically using the scheme reported in [38]. It is challenging for DBSCAN to deal with various datasets with fixed parameters; in this work, the value of $minpts$ is set to $\lfloor \ln|n| \rfloor$ as recommended in [43] and the value of $eps$ is set to the optimal set in the pool of $[1, 5]$ with the step being valued as 0.2. For PCM, the fuzzy parameter and $error$ are set to 1.2 and 0.001, respectively. For EPCM, parameters $m$, $\theta$ are set to 2, 3, respectively. In the case of IT2PFCM, parameters $m_1$, $m_2$ are set to 2, 4, respectively. As for the ensemble approaches SEC and LWEA, the required parameters $\mu$ and $\theta$ are set to 1 and 0.4, respectively, in line with the recommendation as reported in [40], [41]. For the proposed HDC, the Algebraic *T*-norm and Eq. (10) are used as the metric for fuzzy similarity calculation and the similarity parameters are set to the optimal value of the standard deviation and variance; the maximal number of iterations $t$ is set to 15. Likewise, FCM, PCM, EPCM, IT2PFCM and traditional $k$-means all use 15 as the maximal number of iterations.

Each clustering algorithm was run 100 times on each dataset. Normalised mutual information (NMI) [41] and homogeneity score (HS) [44] are used as the evaluation criteria of all datasets for a more objective comparison. More specifically, the NMI measure provides a sound indication of the shared information between the real and predicted clusters, which is a normalisation of the mutual information (MI) score scaling the results between 0 (no mutual information) and 1 (perfect correlation). As for the HS, a clustering result satisfies homogeneity if all of its clusters contain only data points which are members of a single class; HS is ranged from 0 to 1, where 1 represents perfectly homogeneous labelling. For clear description, the best results are highlighted in bold in Tables V, VI and VII.

### *B. Performance on Road Line Segmentation*

The best experimental results of each approach for the three types of lanes are illustrated in Figs. 7, 8 and 9. Note that for MS and DBSCAN, there are no parameters about the number of clusters, so the cluster number may not be consistent with the number of lane separation lines when selecting the optimal parameters (such as that shown in Fig. 9b). It can be seen that only DBSCAN and HDC have generated the correct clustering results, but all other methods were not able to segment the lane separation lines correctly. More specifically, for *Road1* and *Road3*, both HDC and DBSCAN have successfully clustered the lane separation lines in the sampled roads. However, the lane segmentation problem is too challenging for other clusters. For *Road2*, the DBSCAN can separate each short line into a cluster when selecting appropriate parameters. Nonetheless, due to the intervals between dotted lines, the two middle lane separation lines cannot be clustered into a complete line no matter how the parameters are set for DBSCAN. Interestingly, the proposed HDC is the only clustering approach leading to correct clustering results, demonstrating superior clustering performance over others.

Moreover, for each lane image dataset, four images are captured from the process of clustering to show the changes of centroids and noises. The respective results are recorded in Figs. 10, 11 and 12. Among these four images, the first one is the moment of initialisation; the last one indicates the resulting clusters; the second and the third ones show the main segments in the process of clustering. And for each cluster, "$\star$" indicates the centroid, and "$\times$" indicates the peripheral object. It can be seen that due to the use of the RP method, the centroid of each cluster is initialised in the center region of

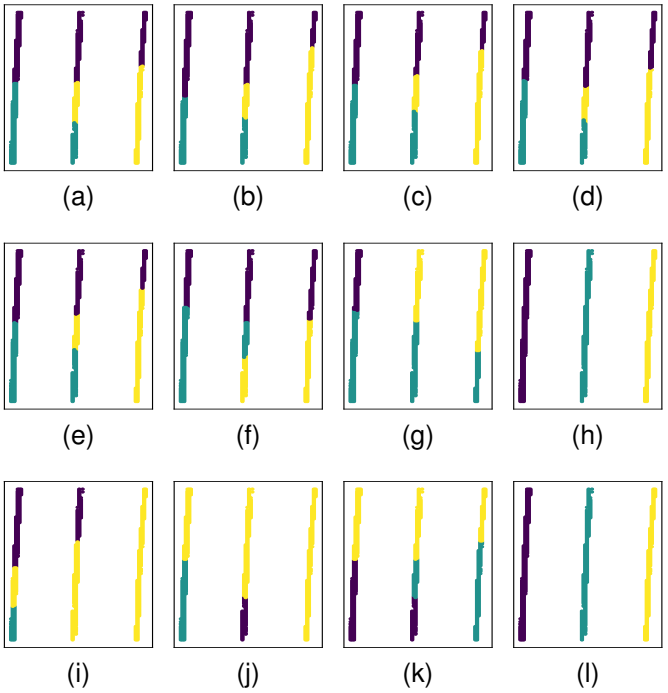


Fig. 7. The visual results of different clustering algorithms for *Road1*. (a) $k$-means. (b) MS. (c) FCM. (d) PCM. (e) EPCM. (f) IT2PFCM. (g) AC. (h) DBSCAN. (i) DPC. (j) SEC. (k) LWEA. (l) HDC.

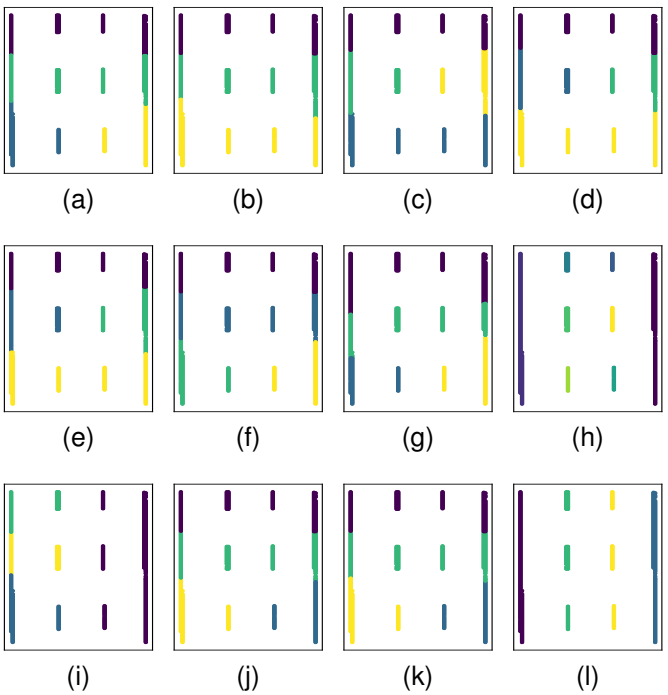


Fig. 8. The visual results of different clustering algorithms for *Road2*. (a) $k$-means. (b) MS. (c) FCM. (d) PCM. (e) EPCM. (f) IT2PFCM. (g) AC. (h) DBSCAN. (i) DPC. (j) SEC. (k) LWEA. (l) HDC.

each image. In the process of clustering, these centroids move towards the intermediate area of the corresponding lane lines, gradually. During the segmentation process of each lane line, it is interesting to note that the proposed HDC algorithm intends

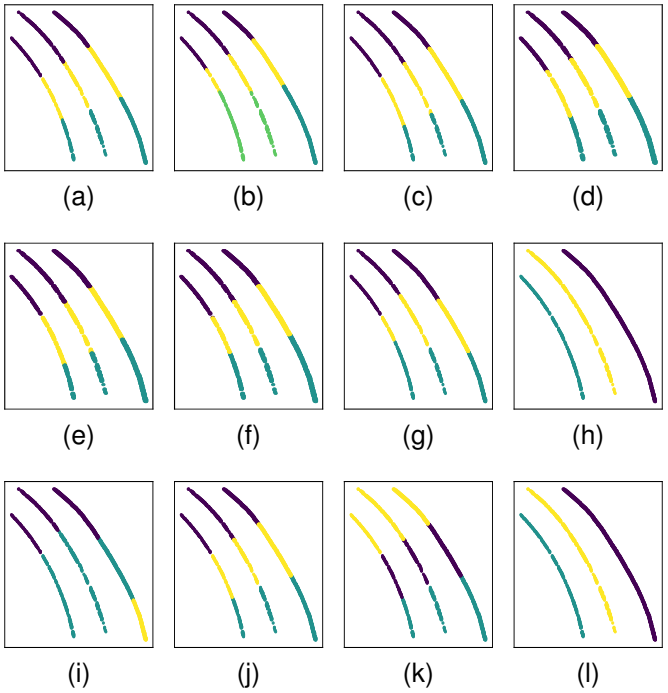


Fig. 9. The visual results of different clustering algorithms for *Road3*. (a) $k$-means. (b) MS. (c) FCM. (d) PCM. (e) EPCM. (f) IT2PFCM. (g) AC. (h) DBSCAN. (i) DPC. (j) SEC. (k) LWEA. (l) HDC.

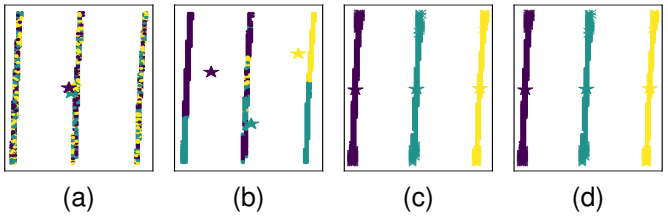


Fig. 10. The clustering process of *Road1*. (a) Initialisation. (b), (c) Two main segments in the process of clustering. (d) The final result.

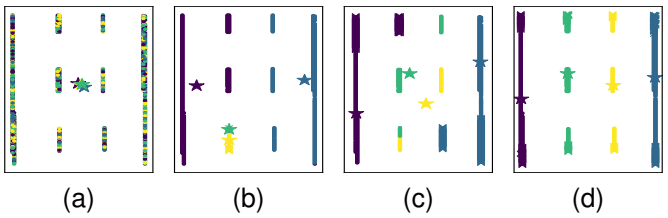


Fig. 11. The clustering process of *Road2*. (a) Initialisation. (b), (c) Two main segments in the process of clustering. (d) The final result.

to treat the upper and lower ends of lines as the peripheral objects of each cluster. In the light of Algorithm 3, the noises won't be used to update the centroids at each iteration. But at last, the noise objects are assigned to the clusters according to the maximum membership principle shown in Eq. (26).

The corresponding average results of NMI and HS over 100 experiments are detailed in Table V. HDC shows the best performance based on both metrics, given that the value of 1 indicates the method has always successfully clustered all lane separation lines in all repeated experiments. The second best

TABLE V
NMI AND HS RESULTS OF DIFFERENT CLUSTERING ALGORITHMS ON EXTRACTED LANE SEPARATION LINES

| Dataset | Index | $k$-means | MS | FCM | PCM | EPCM | IT2PFCM | AC | DBSCAN | DPC | SEC | LWEA | HDC |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *Road1* | NMI | 0.27 | 0.00 | 0.30 | 0.30 | 0.30 | 0.27 | 0.28 | **1.00** | 0.29 | 0.22 | 0.25 | **1.00** |
| | HS | 0.27 | 0.00 | 0.30 | 0.30 | 0.30 | 0.27 | 0.28 | **1.00** | 0.26 | 0.18 | 0.27 | **1.00** |
| *Road2* | NMI | 0.22 | 0.04 | 0.23 | 0.23 | 0.22 | 0.58 | 0.22 | 0.90 | 0.58 | 0.16 | 0.23 | **1.00** |
| | HS | 0.22 | 0.03 | 0.24 | 0.23 | 0.22 | 0.56 | 0.23 | **1.00** | 0.56 | 0.14 | 0.24 | **1.00** |
| *Road3* | NMI | 0.01 | 0.00 | 0.02 | 0.01 | 0.02 | 0.02 | 0.02 | **1.00** | 0.10 | 0.02 | 0.01 | **1.00** |
| | HS | 0.01 | 0.00 | 0.02 | 0.01 | 0.03 | 0.02 | 0.02 | **1.00** | 0.10 | 0.02 | 0.02 | **1.00** |

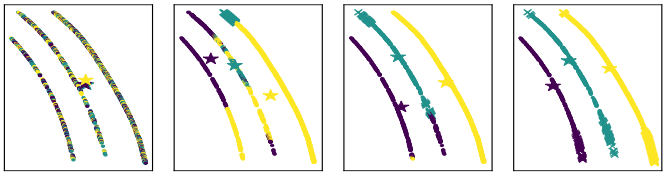


Fig. 12. The clustering process of *Road3*. (a) Initialisation. (b), (c) Two main segments in the process of clustering. (d) The final result.

performer is DBSCAN, which does not show the best result for *Road2*, but it has got a tie with the proposed HDC for the other two roads. None of the rest referenced approaches has demonstrated any close performance. This clearly shows the superiority of the proposed HDC clustering algorithm.

### C. Performance on Benchmark Datasets

To verify the noise-resistance ability of the studied algorithms, 5%, 10%, and 15% random Gaussian noises are added to each benchmark dataset following the experiments reported in [45]. Each newly added noise is assigned to the class where the centroid is closed to this noise. All of these noise samples will be taken into account by the evaluation metrics NMI and HS. The average NMI and HS results (and the best NMI and HS results as shown in brackets) of these algorithms over 100 repeated experiments on different datasets are detailed in Tables VI and VII.

Considering the NMI evaluation index, HDC shows an overall better clustering performance on most datasets. Especially for *Newthyroid*, the average results of HDC surpass the results of all other competing methods under different noise conditions. However, as the noise scale increases, the performance of the noise-sensitive algorithms declines quickly. Take *Iris* as an example, the average result of $k$-means decreases from 0.72 (5% added noise) to 0.46 (10% added noise) and 0.18 (15% added noise). For MS, SEC, and LWEA, the performance also drops with the enlargement of noise; especially when the noise ratio reaches 15%, the evaluation values of those contrasting methods are reduced to the lowest. For datasets *Newthyroid*, *Seeds* and *Wine*, the performance of $k$-means, AC, SEC and LWEA also degrades as the noise level rises. As for the robust clustering algorithms: PCM, EPCM, and IT2PFCM, the obtained results are more steady no matter how much noise is included, and occasionally, able to surpass HDC even with 15% added noise. Nevertheless, without noise parameters, HDC still outperform these robust clustering algorithms in most cases. It is worth noting that for datasets *Led7digit* and *Glass*, with the expansion of noise, the performance of diverse approaches does not change significantly, which is related to the added random noise and the specific distribution of the data.

Interestingly, the density-based algorithms DPC and DBSCAN seem to suffer less from the increased noise, although negatively affected by noise like other compared approaches. Again, take the *Iris* dataset as an example, the effect of noises to these two algorithms is marginal, and other datasets also show similar trends, indicating that the density-based clustering approach is more resistant to data noises.

In terms of the best NMI values, the results of $k$-means are rather different from the corresponding average on many datasets, such as *Iris*, *Hepatitis*, and *Wine*, when the noise level is 15%. This shows its instability to deal with noise. For the remaining comparison strategies and the proposed HDC, although there is a gap between the average and best values, it is not significantly noticeable. Especially for MS and AC, their clustering performance is highly stable, and the same results can always be observed in repeated experiments. Overall, HDC has a stable ability to outperform the compared algorithms.

Regarding the HS metric, the performance of each algorithm is consistent to its NMI results under most circumstances. Again, HDC displays outperformance in comparison with other competitor. Especially for datasets *Newthyroid*, and *Wine*, HDC outperforms nearly all the compared methods in terms of both average and best results, and it exceeds more than half of the compared approaches for the remaining datasets in most cases. In general, many clustering algorithms are noise-sensitive, which means they have difficulties to well handle datasets with noises and/or outliers. For the robust clustering algorithms, the noise preprocessing helps provide more stable results but it may challenge to determine the noise threshold and it often requires bespoke optimisation for different datasets. Experimental results show that HDC can better deal with noise data without the requirement of predefined noise parameters.

### D. Statistical Analysis

Paired $t$-test is applied to all the experiments to explore any statistically significant differences between the proposed HDC algorithm and the referenced clustering approaches. The threshold of significance is set to 0.05 for all experiments, which ensures that the results are not obtained by chance. The

TABLE VI
NMI RESULTS OF DIFFERENT CLUSTERING ALGORITHMS ON BENCHMARK DATASETS WITH DIFFERENT PROPORTIONS OF NOISE

| Dataset | *Iris* | | | *Heart* | | | *Led7digit* | | |
|---|---|---|---|---|---|---|---|---|---|
| Noise(%) | 5 | 10 | 15 | 5 | 10 | 15 | 5 | 10 | 15 |
| $k$-means | 0.72 (0.72) * | 0.46 (0.72) * | 0.18 (0.72) * | 0.27 (0.36) * | 0.10 (0.10) * | 0.10 (0.10) * | 0.46 (0.48) * | 0.45 (0.47) * | 0.45 (0.46) * |
| MS | 0.67 (0.67) * | 0.64 (0.64) * | 0.16 (0.16) * | 0.07 (0.07) * | 0.12 (0.12) * | 0.15 (0.15) * | 0.08 (0.08) * | 0.14 (0.14) * | 0.18 (0.18) * |
| FCM | **0.80** (**0.81**) v | 0.76 (0.80) * | 0.69 (0.72) * | 0.34 (0.35) | 0.32 (0.35) * | 0.32 (0.33) * | 0.48 (0.49) | 0.47 (0.48) * | 0.45 (0.46) * |
| PCM | 0.63 (0.65) * | 0.68 (0.69) * | 0.56 (0.58) * | 0.25 (0.26) * | 0.10 (0.10) * | 0.09 (0.10) * | 0.45 (**0.51**) * | 0.48 (0.48) * | 0.39 (0.41) * |
| EPCM | 0.75 (0.76) * | **0.78** (0.79) | 0.71 (0.72) * | 0.34 (0.36) * | **0.39** (**0.39**) v | 0.26 (0.33) * | 0.39 (0.41) * | 0.47 (0.48) * | 0.45 (0.48) * |
| IT2PFCM | 0.76 (0.76) * | 0.67 (0.72) * | 0.71 (0.71) * | **0.35** (0.35) | 0.28 (0.30) * | 0.19 (0.19) * | 0.44 (0.48) * | 0.47 (0.47) * | 0.47 (0.48) * |
| AC | 0.72 (0.72) * | 0.13 (0.13) * | 0.17 (0.17) * | 0.06 (0.06) * | 0.10 (0.10) * | 0.10 (0.10) * | 0.46 (0.46) * | 0.44 (0.44) * | 0.44 (0.44) * |
| DBSCAN | 0.66 (0.66) * | 0.62 (0.62) * | 0.60 (0.60) * | 0.18 (0.18) * | 0.17 (0.17) * | 0.16 (0.16) * | **0.51** (**0.51**) v | 0.48 (0.49) * | **0.48** (0.48) |
| DPC | 0.73 (0.73) * | 0.72 (0.72) * | 0.68 (0.68) * | 0.23 (0.23) * | 0.12 (0.12) * | 0.10 (0.10) * | 0.00 (0.00) * | 0.00 (0.00) * | 0.00 (0.00) * |
| SEC | 0.53 (0.74) * | 0.43 (0.69) * | 0.12 (0.68) * | 0.15 (**0.37**) * | 0.02 (0.10) * | 0.02 (0.10) * | 0.44 (0.50) * | 0.46 (0.50) * | 0.42 (0.48) * |
| LWEA | 0.72 (0.72) * | 0.64 (0.64) * | 0.17 (0.17) * | 0.06 (0.06) * | 0.10 (0.10) * | 0.10 (0.10) * | 0.47 (0.47) * | 0.46 (0.46) * | 0.45 (0.45) * |
| HDC | 0.79 (0.80) | **0.78** (**0.81**) | **0.79** (**0.80**) | **0.35** (0.36) | 0.35 (**0.39**) | **0.36** (**0.36**) | 0.49 (**0.51**) | **0.49** (**0.51**) | **0.48** (**0.49**) |
| **Summary** | (10/0/1) | (10/1/0) | (11/0/0) | (9/2/0) | (10/0/1) | (11/0/0) | (9/1/1) | (11/0/0) | (10/1/0) |

| Dataset | *Glass* | | | *Newthyroid* | | | *Seeds* | | |
|---|---|---|---|---|---|---|---|---|---|
| Noise(%) | 5 | 10 | 15 | 5 | 10 | 15 | 5 | 10 | 15 |
| $k$-means | 0.31 (0.34) * | 0.35 (0.36) | 0.36 (0.36) | 0.15 (0.15) * | 0.25 (0.25) * | 0.32 (0.32) * | 0.17 (0.42) * | 0.13 (0.13) * | 0.17 (0.17) * |
| MS | 0.37 (0.37) v | 0.22 (0.22) * | 0.26 (0.26) * | 0.58 (0.58) * | 0.24 (0.24) * | 0.30 (0.30) * | 0.07 (0.07) * | 0.13 (0.13) * | 0.17 (0.17) * |
| FCM | 0.33 (0.38) * | 0.30 (0.37) * | 0.32 (0.38) * | 0.59 (0.62) | 0.31 (0.52) * | 0.35 (0.56) * | **0.72** (**0.73**) v | 0.59 (0.63) * | 0.48 (0.50) * |
| PCM | 0.30 (0.31) * | 0.33 (0.33) * | 0.35 (0.36) * | 0.16 (0.16) * | 0.29 (0.30) * | 0.31 (0.33) * | 0.51 (0.51) * | 0.13 (0.13) * | 0.17 (0.17) * |
| EPCM | 0.29 (0.34) * | 0.26 (0.30) * | 0.31 (0.32) * | 0.49 (0.49) * | 0.23 (0.25) * | 0.30 (0.32) * | **0.72** (0.72) v | 0.48 (0.48) * | 0.49 (0.50) * |
| IT2PFPCM | 0.36 (0.38) v | 0.32 (0.33) * | 0.33 (0.35) * | 0.57 (0.58) * | 0.40 (0.40) * | 0.32 (0.32) * | 0.66 (0.66) v | 0.60 (0.61) * | 0.47 (0.48) * |
| AC | 0.30 (0.30) * | 0.33 (0.33) * | 0.34 (0.34) * | 0.15 (0.15) * | 0.25 (0.25) * | 0.32 (0.32) * | 0.08 (0.08) * | 0.13 (0.13) * | 0.17 (0.17) * |
| DBSCAN | **0.39** (**0.39**) v | **0.39** (**0.39**) v | 0.36 (0.36) | 0.46 (0.48) * | 0.47 (0.49) * | 0.48 (0.49) * | 0.08 (0.08) * | 0.13 (0.13) * | 0.17 (0.17) * |
| DPC | 0.33 (0.33) * | 0.33 (0.35) * | 0.34 (0.36) * | 0.53 (0.53) * | 0.56 (0.56) * | 0.56 (0.56) * | 0.70 (0.72) v | 0.60 (0.60) * | 0.44 (0.49) * |
| SEC | 0.23 (0.34) * | 0.23 (0.31) * | 0.22 (0.30) * | 0.01 (0.11) * | 0.05 (0.25) * | 0.07 (0.32) * | 0.23 (0.42) * | 0.02 (0.09) * | 0.04 (0.12) * |
| LWEA | 0.33 (0.33) * | 0.36 (0.36) | 0.36 (0.36) | 0.15 (0.15) * | 0.25 (0.25) * | 0.32 (0.32) * | 0.08 (0.08) * | 0.13 (0.13) * | 0.17 (0.17) * |
| HDC | 0.35 (0.36) | 0.35 (0.36) | **0.37** (**0.39**) | **0.60** (**0.64**) | **0.64** (**0.68**) | **0.64** (**0.69**) | 0.63 (0.63) | **0.61** (**0.64**) | **0.52** (**0.56**) |
| **Summary** | (8/0/3) | (8/2/1) | (8/3/0) | (10/1/0) | (11/0/0) | (11/0/0) | (7/0/4) | (11/0/0) | (11/0/0) |

| Dataset | *Hepatitis* | | | *Breast* | | | *Wine* | | |
|---|---|---|---|---|---|---|---|---|---|
| Noise(%) | 5 | 10 | 15 | 5 | 10 | 15 | 5 | 10 | 15 |
| $k$-means | 0.23 (0.23) * | 0.30 (0.30) * | 0.08 (0.29) * | 0.11 (0.56) * | 0.13 (0.13) * | 0.08 (0.14) * | 0.12 (0.59) * | 0.13 (0.13) * | 0.13 (0.59) * |
| MS | 0.22 (0.22) * | 0.23 (0.23) * | 0.25 (0.25) * | 0.67 (0.67) v | 0.13 (0.13) * | 0.16 (0.16) * | 0.43 (0.43) * | 0.13 (0.13) * | 0.16 (0.16) * |
| FCM | 0.23 (0.23) * | 0.30 (0.30) * | **0.29** (0.29) | **0.71** (0.71) v | 0.52 (0.53) * | 0.46 (0.46) * | 0.76 (0.76) * | 0.63 (0.72) * | 0.74 (0.74) * |
| PCM | 0.23 (0.23) * | 0.29 (0.30) * | **0.29** (0.29) | **0.71** (**0.72**) v | 0.13 (0.13) * | 0.14 (0.14) * | 0.42 (0.44) * | 0.13 (0.13) * | 0.49 (0.49) * |
| EPCM | 0.18 (0.19) * | **0.32** (**0.32**) v | 0.28 (0.29) | 0.70 (0.71) v | 0.55 (0.56) v | 0.53 (0.54) | 0.65 (0.66) * | 0.62 (0.65) * | **0.78** (**0.78**) v |
| IT2PFCM | 0.23 (0.23) * | 0.16 (0.16) * | 0.21 (0.21) * | 0.62 (0.65) v | 0.54 (0.55) | **0.55** (**0.55**) v | 0.48 (0.51) * | 0.52 (0.56) * | 0.56 (0.61) * |
| AC | 0.23 (0.23) * | 0.30 (0.30) * | 0.04 (0.04) * | 0.07 (0.07) * | 0.13 (0.13) * | 0.07 (0.07) * | 0.04 (0.04) * | 0.13 (0.13) * | 0.11 (0.11) * |
| DBSCAN | **0.29** (**0.29**) v | 0.26 (0.26) * | 0.25 (0.25) * | 0.67 (0.67) v | 0.64 (0.64) v | **0.55** (**0.55**) v | 0.59 (0.59) * | 0.55 (0.55) * | 0.53 (0.53) * |
| DPC | 0.22 (0.22) * | 0.25 (0.25) * | 0.25 (0.25) * | 0.64 (0.64) v | **0.65** (**0.66**) v | 0.26 (0.27) * | 0.65 (0.69) * | 0.64 (0.64) * | 0.62 (0.66) * |
| SEC | 0.01 (0.18) * | 0.04 (0.30) * | 0.01 (0.04) * | 0.23 (0.56) * | 0.02 (0.13) * | 0.03 (0.14) * | 0.35 (0.63) * | 0.02 (0.13) * | 0.33 (0.63) * |
| LWEA | 0.18 (0.18) * | 0.30 (0.30) * | 0.04 (0.04) * | 0.07 (0.07) * | 0.13 (0.13) * | 0.07 (0.07) * | 0.04 (0.04) * | 0.13 (0.13) * | 0.11 (0.11) * |
| HDC | 0.26 (0.28) | 0.31 (**0.32**) | **0.29** (**0.30**) | 0.52 (0.52) | 0.54 (0.54) | 0.53 (**0.55**) | **0.83** (**0.83**) | **0.80** (**0.80**) | 0.75 (0.77) |
| **Summary** | (10/0/1) | (10/0/1) | (8/3/0) | (4/0/7) | (7/1/3) | (8/1/2) | (11/0/0) | (11/0/0) | (10/0/1) |

$t$-test results are summarised at the end of each subtable of Tables VI and VII, by counting the number of statistically better, equivalent, or worse cases for HDC in comparison to other compared algorithms; in particular, better and worse cases are indicated by "*" and "v" in the tables whilst equivalent cases are represented by blank spaces. For example, (10/1/0) in the column *Iris* with 10% noise in Table VI expresses that the average clustering result led by the proposed HDC algorithm is better than 10 compared methods, equally well with 1 compared method, and worse than 0 compared method. It can be clearly seen from these tables that the statistical results of HDC are better than other methods in most cases based on both metrics NMI and HS. Especially based on NMI, the proposed HDC algorithm surpasses all other compared approaches on nearly half of the datasets. For other datasets and the HS metric, HDC outperforms most of the compared algorithms as well. Statistical analysis based on 100 repeated experiments proves the better stability of the proposed HDC algorithm in reference to the employed competitors in this work.

TABLE VII
HOMOGENEITY SCORES OF DIFFERENT CLUSTERING ALGORITHMS ON BENCHMARK DATASETS WITH DIFFERENT PROPORTIONS OF NOISE

| Dataset | *Iris* | | | *Heart* | | | *Led7digit* | | |
|---|---|---|---|---|---|---|---|---|---|
| Noise(%) | 5 | 10 | 15 | 5 | 10 | 15 | 5 | 10 | 15 |
| $k$-means | 0.60 (0.60) * | 0.39 (0.62) * | 0.13 (0.62) * | 0.25 (0.33) * | 0.07 (0.07) * | 0.07 (0.07) * | 0.43 (0.44) * | 0.42 (0.45) * | 0.42 (0.44) * |
| MS | 0.60 (0.60) * | 0.62 (0.62) * | 0.12 (0.12) * | 0.05 (0.05) * | 0.10 (0.10) * | 0.13 (0.13) * | 0.05 (0.05) * | 0.09 (0.09) * | 0.13 (0.13) * |
| FCM | **0.79** (**0.80**) v | 0.74 (0.80) * | 0.68 (0.72) * | 0.31 (0.33) * | 0.30 (0.32) * | 0.32 (0.33) * | 0.47 (0.47) | 0.46 (0.46) * | 0.44 (0.44) * |
| PCM | 0.53 (0.54) * | 0.59 (0.60) * | 0.50 (0.51) * | 0.25 (0.26) * | 0.06 (0.06) * | 0.06 (0.07) * | 0.47 (**0.50**) * | 0.44 (0.45) * | 0.34 (0.36) * |
| EPCM | 0.74 (0.75) * | **0.78** (0.78) | 0.63 (0.63) * | 0.32 (**0.36**) * | **0.38** (**0.38**) v | 0.31 (0.33) * | 0.37 (0.43) * | 0.44 (0.47) * | 0.46 (0.48) * |
| IT2PFCM | 0.74 (0.74) * | 0.59 (0.61) * | 0.63 (0.63) * | **0.33** (0.34) | 0.26 (0.27) * | 0.18 (0.18) * | 0.46 (0.48) * | 0.44 (0.44) * | 0.46 (0.47) * |
| AC | 0.60 (0.60) * | 0.09 (0.09) * | 0.12 (0.12) * | 0.03 (0.03) * | 0.07 (0.07) * | 0.07 (0.07) * | 0.42 (0.42) * | 0.40 (0.40) * | 0.41 (0.41) * |
| DBSCAN | 0.57 (0.57) * | 0.56 (0.56) * | 0.58 (0.58) * | 0.24 (0.24) * | 0.23 (0.23) * | 0.22 (0.22) * | **0.50** (**0.50**) v | 0.46 (0.48) * | 0.46 (**0.49**) |
| DPC | 0.71 (0.71) * | 0.62 (0.62) * | 0.60 (0.60) * | 0.22 (0.22) * | 0.14 (0.14) * | 0.07 (0.07) * | 0.00 (0.00) * | 0.00 (0.00) * | 0.00 (0.00) * |
| SEC | 0.42 (0.60) * | 0.36 (0.56) * | 0.09 (0.60) * | 0.15 (0.25) * | 0.01 (0.07) * | 0.01 (0.07) * | 0.40 (0.48) * | 0.42 (0.49) * | 0.38 (0.47) * |
| LWEA | 0.60 (0.60) * | 0.56 (0.56) * | 0.12 (0.12) * | 0.03 (0.03) * | 0.07 (0.07) * | 0.07 (0.07) * | 0.44 (0.44) * | 0.44 (0.44) * | 0.42 (0.42) * |
| HDC | 0.77 (0.79) | 0.77 (**0.81**) | **0.78** (**0.79**) | **0.33** (0.34) | 0.34 (0.36) | **0.36** (**0.36**) | 0.48 (**0.50**) | **0.47** (**0.50**) | **0.47** (**0.49**) |
| **Summary** | (10/0/1) | (10/1/0) | (11/0/0) | (10/1/0) | (10/0/1) | (11/0/0) | (9/1/1) | (11/0/0) | (10/1/0) |

| Dataset | *Glass* | | | *Newthyroid* | | | *Seeds* | | |
|---|---|---|---|---|---|---|---|---|---|
| Noise(%) | 5 | 10 | 15 | 5 | 10 | 15 | 5 | 10 | 15 |
| $k$-means | 0.26 (0.27) * | 0.26 (0.27) * | 0.29 (0.29) * | 0.09 (0.09) * | 0.17 (0.17) * | 0.24 (0.24) * | 0.12 (0.33) * | 0.09 (0.09) * | 0.12 (0.12) * |
| MS | 0.28 (0.28) * | 0.15 (0.15) * | 0.20 (0.20) * | **0.57** (0.57) v | 0.17 (0.17) * | 0.24 (0.24) * | 0.05 (0.05) * | 0.09 (0.09) * | 0.12 (0.12) * |
| FCM | 0.33 (**0.39**) v | 0.30 (**0.38**) | 0.31 (0.35) * | 0.52 (0.55) * | 0.23 (0.43) * | 0.28 (0.48) * | **0.72** (**0.73**) v | 0.56 (0.61) * | 0.43 (0.44) * |
| PCM | 0.21 (0.22) * | 0.26 (0.26) * | 0.28 (0.29) * | 0.10 (0.10) * | 0.17 (0.21) * | 0.23 (0.25) * | 0.44 (0.44) * | 0.09 (0.09) * | 0.12 (0.12) * |
| EPCM | 0.30 (0.34) * | 0.27 (0.30) * | 0.30 (0.30) * | 0.35 (0.38) * | 0.16 (0.17) * | 0.22 (0.24) * | **0.72** (0.72) v | 0.41 (0.41) * | 0.42 (0.44) * |
| IT2PFCM | **0.34** (0.36) v | 0.30 (0.31) | 0.30 (0.32) * | 0.48 (0.50) * | 0.30 (0.31) * | 0.24 (0.25) * | 0.65 (0.66) v | 0.42 (0.44) * | 0.44 (0.45) * |
| AC | 0.21 (0.21) * | 0.25 (0.25) * | 0.27 (0.27) * | 0.09 (0.09) * | 0.17 (0.17) * | 0.24 (0.24) * | 0.05 (0.05) * | 0.09 (0.09) * | 0.12 (0.12) * |
| DBSCAN | 0.30 (0.30) * | 0.30 (0.30) | 0.28 (0.28) * | 0.42 (0.43) * | 0.43 (0.45) * | 0.45 (0.46) * | 0.05 (0.05) * | 0.09 (0.09) * | 0.12 (0.12) * |
| DPC | **0.34** (0.34) v | **0.34** (0.36) v | 0.33 (**0.36**) | 0.50 (0.52) * | 0.48 (0.50) * | 0.48 (0.48) * | 0.70 (0.72) v | 0.50 (0.55) * | 0.44 (0.44) * |
| SEC | 0.20 (0.30) * | 0.20 (0.28) * | 0.19 (0.26) * | 0.01 (0.06) * | 0.03 (0.17) * | 0.05 (0.24) * | 0.17 (0.33) * | 0.01 (0.05) * | 0.03 (0.08) * |
| LWEA | 0.23 (0.23) * | 0.27 (0.27) * | 0.29 (0.29) * | 0.09 (0.09) * | 0.17 (0.17) * | 0.24 (0.24) * | 0.05 (0.05) * | 0.09 (0.09) * | 0.12 (0.12) * |
| HDC | 0.31 (0.31) | 0.30 (0.30) | **0.34** (**0.36**) | 0.54 (**0.59**) | **0.55** (**0.58**) | **0.57** (**0.61**) | 0.62 (0.62) | **0.57** (**0.63**) | **0.51** (**0.56**) |
| **Summary** | (8/0/3) | (7/3/1) | (10/1/0) | (10/0/1) | (11/0/0) | (11/0/0) | (7/0/4) | (11/0/0) | (11/0/0) |

| Dataset | *Hepatitis* | | | *Breast* | | | *Wine* | | |
|---|---|---|---|---|---|---|---|---|---|
| Noise(%) | 5 | 10 | 15 | 5 | 10 | 15 | 5 | 10 | 15 |
| $k$-means | 0.16 (0.16) * | 0.22 (0.22) * | 0.06 (0.22) * | 0.08 (0.53) * | 0.09 (0.09) * | 0.06 (0.10) * | 0.09 (0.49) * | 0.08 (0.08) * | 0.10 (0.51) * |
| MS | 0.16 (0.16) * | 0.23 (0.23) * | 0.24 (0.24) * | 0.65 (0.65) v | 0.11 (0.11) * | 0.14 (0.14) * | 0.40 (0.40) * | 0.09 (0.09) * | 0.13 (0.13) * |
| FCM | 0.16 (0.16) * | 0.22 (0.22) * | 0.29 (0.30) * | **0.70** (**0.70**) v | 0.48 (0.49) * | 0.45 (0.45) * | 0.76 (0.76) * | 0.60 (0.71) * | 0.73 (0.74) * |
| PCM | 0.16 (0.16) * | 0.21 (0.22) * | 0.22 (0.22) * | 0.69 (0.69) v | 0.09 (0.09) * | 0.24 (0.24) * | 0.38 (0.39) * | 0.08 (0.08) * | 0.43 (0.43) * |
| EPCM | 0.11 (0.12) * | **0.37** (**0.37**) v | 0.28 (0.30) * | 0.69 (**0.70**) v | 0.54 (0.55) v | 0.53 (0.53) v | 0.65 (0.66) * | 0.63 (0.65) * | **0.77** (**0.78**) v |
| IT2PFCM | 0.16 (0.16) * | 0.13 (0.13) * | 0.25 (0.25) * | 0.65 (0.68) v | 0.52 (0.55) v | **0.55** (**0.56**) v | 0.41 (0.43) * | 0.46 (0.47) * | 0.50 (0.53) * |
| AC | 0.16 (0.16) * | 0.22 (0.22) * | 0.03 (0.03) * | 0.04 (0.04) * | 0.09 (0.09) * | 0.05 (0.05) * | 0.02 (0.02) * | 0.08 (0.08) * | 0.08 (0.08) * |
| DBSCAN | **0.29** (0.29) | 0.25 (0.25) * | 0.24 (0.24) * | 0.68 (0.68) v | **0.65** (**0.65**) v | 0.54 (0.54) v | 0.69 (0.69) * | 0.65 (0.65) * | 0.62 (0.62) * |
| DPC | 0.25 (0.25) * | 0.26 (0.26) * | 0.29 (0.29) * | 0.65 (0.65) v | 0.64 (0.64) v | 0.30 (0.31) * | 0.73 (0.73) * | 0.71 (0.71) * | 0.70 (0.70) * |
| SEC | 0.00 (0.12) * | 0.03 (0.22) * | 0.00 (0.03) * | 0.22 (0.52) * | 0.01 (0.09) * | 0.02 (0.10) * | 0.27 (0.49) * | 0.01 (0.08) * | 0.26 (0.50) * |
| LWEA | 0.12 (0.12) * | 0.22 (0.22) * | 0.03 (0.03) * | 0.04 (0.04) * | 0.09 (0.09) * | 0.05 (0.05) * | 0.02 (0.02) * | 0.08 (0.08) * | 0.08 (0.08) * |
| HDC | **0.29** (**0.31**) | 0.28 (0.28) | **0.30** (**0.31**) | 0.48 (0.48) | 0.50 (0.50) | 0.50 (0.53) | **0.83** (**0.83**) | **0.80** (**0.80**) | 0.74 (0.76) |
| **Summary** | (10/1/0) | (10/0/1) | (11/0/0) | (4/0/7) | (7/0/4) | (8/0/3) | (11/0/0) | (11/0/0) | (10/0/1) |

## V. CONCLUSION

Inspired by the fuzzy-rough set theory, this paper proposes the concept of harmonic discrepancy and associated harmonic discrepancy clustering (HDC) algorithm, which clusters data from the perspectives of both separability and rationality of clusters. Also, this HDC algorithm benefits from a non-parametric noise detection strategy for better applicability on noisy datasets. Experimental results demonstrate that HDC enjoys sound effectiveness and stability on the lane segmentation and benchmark datasets.

Whilst promising, the work also opens up an avenue for further development. For instance, it would be interesting to investigate an extension of HDC for multi-density clusters. In addition, an investigation into potential time efficiency improvement remains active research. Moreover, HDC can deal with noise without the assistance of pre-defined noise parameters, which is promising for complex and large-scale real-world datasets. Therefore, further applications, such as medical image analysis [46], [47], would construct the foundation for a broader spectrum of future research.

## ACKNOWLEDGMENTS

The authors would like to thank the editor and anonymous referees for their constructive comments which have been very helpful in revising this work.

## REFERENCES

[1] Z. Bian, F.-L. Chung, and S. Wang, "Fuzzy density peaks clustering," *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 7, pp. 1725–1738, 2021.

[2] T. Lei, X. Jia, Y. Zhang, S. Liu, H. Meng, and A. K. Nandi, "Superpixel-based fast fuzzy c-means clustering for color image segmentation," *IEEE Transactions on Fuzzy Systems*, vol. 27, no. 9, pp. 1753–1766, 2018.

[3] Y. Song, J. Lu, H. Lu, and G. Zhang, "Fuzzy clustering-based adaptive regression for drifting data streams," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 3, pp. 544–557, 2019.

[4] Y. Pan, L. Zhang, Z. Li, and L. Ding, "Improved fuzzy bayesian network-based risk analysis with interval-valued fuzzy sets and d–s evidence theory," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 9, pp. 2063–2077, 2019.

[5] L. Kong, W. Qu, J. Yu, H. Zuo, G. Chen, F. Xiong, S. Pan, S. Lin, and M. Qiu, "Distributed feature selection for big data using fuzzy rough sets," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 5, pp. 846–857, 2019.

[6] A. Gosain and S. Dahiya, "Performance analysis of various fuzzy clustering algorithms: a review," *Procedia Computer Science*, vol. 79, pp. 100–111, 2016.

[7] F. Yang, Z. Liu, X. Bai, and Y. Zhang, "An improved intuitionistic fuzzy c-means for ship segmentation in infrared images," *IEEE Transactions on Fuzzy Systems*, vol. 30, no. 2, pp. 332–344, 2022.

[8] MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, no. 14, 1967, pp. 281–297.

[9] S. D. Xenaki, K. D. Koutroumbas, and A. A. Rontogiannis, "Sparsity-aware possibilistic clustering algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 6, pp. 1611–1626, 2016.

[10] N. R. Pal, K. Pal, J. M. Keller, and J. C. Bezdek, "A possibilistic fuzzy c-means clustering algorithm," *IEEE Transactions on Fuzzy Systems*, vol. 13, no. 4, pp. 517–530, 2005.

[11] P. Kaur, I. Lamba, and A. Gosain, "Kernelized type-2 fuzzy c-means clustering algorithm in segmentation of noisy medical images," in *2011 IEEE Recent Advances in Intelligent Computational Systems*. IEEE, 2011, pp. 493–498.

[12] W.-h. Hou, Y.-t. Wang, J.-q. Wang, P.-F. Cheng, and L. Li, "Intuitionistic fuzzy c-means clustering algorithm based on a novel weighted proximity measure and genetic algorithm," *International Journal of Machine Learning and Cybernetics*, vol. 12, no. 3, pp. 859–875, 2021.

[13] S. Askari, "Fuzzy c-means clustering algorithm for data with unequal cluster sizes and contaminated with noise and outliers: Review and development," *Expert Systems with Applications*, vol. 165, p. 113856, 2021.

[14] X. Zhou, R. Zhang, X. Wang, T. Huang, and C. Yang, "Kernel intuitionistic fuzzy c-means and state transition algorithm for clustering problem," *Soft Computing*, vol. 24, no. 20, pp. 15 507–15 518, 2020.

[15] J. Arora and M. Tushir, "Robust spatial intuitionistic fuzzy c-means with city-block distance clustering for image segmentation," *Journal of Intelligent & Fuzzy Systems*, vol. 35, no. 5, pp. 5255–5264, 2018.

[16] T. Zhang, F. Ma, D. Yue, C. Peng, and G. M. O'Hare, "Interval type-2 fuzzy local enhancement based rough k-means clustering considering imbalanced clusters," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 9, pp. 1925–1939, 2019.

[17] P. Wang, H. Shi, X. Yang, and J. Mi, "Three-way k-means: integrating k-means and three-way decision," *International Journal of Machine Learning and Cybernetics*, vol. 10, no. 10, pp. 2767–2777, 2019.

[18] A. Raj and S. Minz, "Spatial rough k-means algorithm for unsupervised multi-spectral classification," in *International Conference on Information and Communication Technology for Intelligent Systems*. Springer, 2020, pp. 215–226.

[19] G. Schaefer, Q. Hu, H. Zhou, J. F. Peters, and A. E. Hassanien, "Rough c-means and fuzzy rough c-means for colour quantisation," *Fundamenta Informaticae*, vol. 119, no. 1, pp. 113–120, 2012.

[20] A. Smiti, "A critical overview of outlier detection methods," *Computer Science Review*, vol. 38, p. 100306, 2020.

[21] B. Tang and H. He, "A local density-based approach for outlier detection," *Neurocomputing*, vol. 241, pp. 171–180, 2017.

[22] K. D. Koutroumbas, S. D. Xenaki, and A. A. Rontogiannis, "On the convergence of the sparse possibilistic c-means algorithm," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 1, pp. 324–337, 2017.

[23] S. Askari, N. Montazerin, and M. F. Zarandi, "Generalized possibilistic fuzzy c-means with novel cluster validity indices for clustering noisy data," *Applied Soft Computing*, vol. 53, pp. 262–283, 2017.

[24] U. Qamar, "A dissimilarity measure based fuzzy c-means (fcm) clustering algorithm," *Journal of Intelligent & Fuzzy Systems*, vol. 26, no. 1, pp. 229–238, 2014.

[25] S. Zhou, D. Li, Z. Zhang, and R. Ping, "A new membership scaling fuzzy c-means clustering algorithm," *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 9, pp. 2810–2818, 2020.

[26] Y. Gao, Z. Wang, J. Xie, and J. Pan, "A new robust fuzzy c-means clustering method based on adaptive elastic distance," *Knowledge-Based Systems*, vol. 237, p. 107769, 2022.

[27] Y. Gao, Z. Wang, H. Li, and J. Pan, "Gaussian collaborative fuzzy c-means clustering," *International Journal of Fuzzy Systems*, vol. 23, no. 7, pp. 2218–2234, 2021.

[28] J.-j. Liu and J.-C. Fan, "A novel fuzzy c-means clustering algorithm based on local density," in *Intelligent Information Processing X: 11th IFIP TC 12 International Conference, IIP 2020, Hangzhou, China, July 3–6, 2020, Proceedings 11*. Springer, 2020, pp. 46–58.

[29] Y. Yao, "Relational interpretations of neighborhood operators and rough set approximation operators," *Information Sciences*, vol. 111, no. 1-4, pp. 239–259, 1998.

[30] R. Jensen and Q. Shen, "New approaches to fuzzy-rough feature selection," *IEEE Transactions on Fuzzy Systems*, vol. 17, no. 4, pp. 824–838, 2008.

[31] J. C. Fodor, "Contrapositive symmetry of fuzzy implications," *Fuzzy Sets and Systems*, vol. 69, no. 2, pp. 141–156, 1995.

[32] P. Yildirim and D. Birant, "K-linkage: A new agglomerative approach for hierarchical clustering," *Advances in Electrical and Computer Engineering*, vol. 17, no. 4, pp. 77–88, 2017.

[33] P. Fränti and S. Sieranoja, "How much can k-means be improved by using better initialization and repeats?" *Pattern Recognition*, vol. 93, pp. 95–112, 2019.

[34] D. Demirović, "An implementation of the mean shift algorithm," *Image Processing On Line*, vol. 9, pp. 251–268, 2019.

[35] S. Sotudian and M. H. F. Zarandi, "Interval type-2 enhanced possibilistic fuzzy c-means clustering for gene expression data analysis," *arXiv preprint arXiv:2101.00304*, 2021.

[36] A. M. Abdul-Sadah, A. N. Najaf, and N. K. Bachache, "Image enhancing using both interval type-2 fuzzy sets and it2pfcm," in *AIP Conference Proceedings*, vol. 2386, no. 1, 2022, p. 050025.

[37] N. Ding and F. Farokhi, "Developing non-stochastic privacy-preserving policies using agglomerative clustering," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3911–3923, 2020.

[38] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.

[39] K. M. Kumar and A. R. M. Reddy, "A fast dbscan clustering algorithm by accelerating neighbor searching using groups method," *Pattern Recognition*, vol. 58, pp. 39–48, 2016.

[40] H. Liu, J. Wu, T. Liu, D. Tao, and Y. Fu, "Spectral ensemble clustering via weighted k-means: Theoretical and practical evidence," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 5, pp. 1129–1143, 2017.

[41] D. Huang, C.-D. Wang, and J.-H. Lai, "Locally weighted ensemble clustering," *IEEE Transactions on Cybernetics*, vol. 48, no. 5, pp. 1460–1473, 2017.

[42] B. M. S. Hasan and A. M. Abdulazeez, "A review of principal component analysis algorithm for dimensionality reduction," *Journal of Soft Computing and Data Mining*, vol. 2, no. 1, pp. 20–30, 2021.

[43] R. Scitovski and K. Sabo, "Dbscan-like clustering method for various data densities," *Pattern Analysis and Applications*, vol. 23, no. 2, pp. 541–554, 2020.

[44] D. Mustafi, A. Mustafi, and G. Sahoo, "A novel approach to text clustering using genetic algorithm based on the nearest neighbour heuristic," *International Journal of Computers and Applications*, vol. 44, no. 3, pp. 291–303, 2022.

[45] L. M. Ward, A. Neiman, and F. Moss, "Stochastic resonance in psychophysics and in animal behavior," *Biological Cybernetics*, vol. 87, no. 2, pp. 91–101, 2002.

[46] Y. Qu, Q. Fu, C. Shang, A. Deng, R. Zwiggelaar, M. George, and Q. Shen, "Fuzzy-rough assisted refinement of image processing pro-

cedure for mammographic risk assessment," *Applied Soft Computing*, vol. 91, p. 106230, 2020.

[47] Y. Qu, G. Yue, C. Shang, L. Yang, R. Zwiggelaar, and Q. Shen, "Multi-criterion mammographic risk analysis supported with multi-label fuzzy-rough feature selection," *Artificial Intelligence in Medicine*, vol. 100, p. 101722, 2019.