An Equilibrium Approach to Clustering: Surpassing Fuzzy C-Means on Imbalanced Data

Yudong He

***Abstract*—Most fuzzy clustering algorithms are based on the well-known Bezdek's fuzzy C-means (FCM). However, FCM fails when the data is imbalanced (class sizes are highly unequal). This issue arises because in FCM all data points have only attraction to each cluster prototype (i.e., centroid), causing centroids to be biased towards the large class with the most data points. This paper proposes a novel equilibrium K-means (EKM) for imbalanced data, where data points exert both attraction and repulsion on centroids. The equilibrium between opposing forces reduces the learning bias towards large clusters, leading to more meaningful clusters on imbalanced data. Unlike FCM, EKM explicitly models the relationship between membership and centroid via equality constraints, avoiding the pitfalls of uniform effect. We derive closed-form centroid update equations proven to converge exponentially fast. Experiments are conducted on four artificial and 16 real-world datasets. The results demonstrate that EKM significantly outperforms 13 state-of-the-art methods on imbalanced datasets while maintaining competitive performance on balanced data. EKM achieves an average improvement of 0.22 in normalized mutual information, 0.31 in adjusted rand index, and 0.21 in clustering accuracy over FCM on 10 real-world imbalanced datasets, with comparable computational efficiency in theory and practice.**

***Index Terms*—K-means, fuzzy clustering, equilibrium clustering, imbalance learning, uniform effect**

I. Introduction

A. Imbalanced Data and Fuzzy Clustering

Imbalanced data refers to the true underlying groups of data having highly unequal sizes, which is common in datasets of medical diagnosis, fraud detection, and anomaly detection. Imbalanced data poses a challenge for learning algorithms because these algorithms tend to be biased towards the majority group [1]. While there is a considerable amount of research on supervised learning from imbalanced data [2]–[4], unsupervised learning has not been as thoroughly explored, because the task is more difficult. Methods like resampling and boosting frequently used in supervised learning cannot be applied in unsupervised learning due to the lack of labels.

Let us consider an important unsupervised learning task that aims to partition data points into a specified number of clusters based on their similarity. This task is primarily accomplished through fuzzy clustering, a more modern and general concept than traditional hard clustering. In hard clustering, each data point can only belong to exactly one cluster, while in fuzzy clustering, each data point can potentially belong to multiple clusters. Membership values are assigned to data points, indicating their degree of belonging to each cluster. Fuzzy C-means (FCM), proposed by Bezdek [5], is one of the most widely used fuzzy clustering algorithms and serves as a generalization of the K-means algorithm proposed by Lloyd [6]. Many fuzzy clustering algorithms build upon FCM, focusing on the management of regularization or penalty terms to obtain expected membership values. For example, Krishnapuram and Keller relaxed the FCM's probabilistic constraint that the sum of membership values of each data point across clusters equals 1, proposing the possibilistic C-means (PCM) [7], where the membership is interpreted as a typicality. Subsequently, the possibilistic fuzzy C-means (PFCM) [8] was proposed as a combination of FCM and PCM to address the noise sensitivity defect of FCM and overcome the coincident clusters problem of PCM. The cutset-type PCM (C-PCM) [9] was proposed to address the coincident clusters problem of PCM by shrinking membership/typicality values of non-core data points to zero. Additionally, maximum-entropy fuzzy clustering (MEFC) [10], [11] was developed to provide a physical interpretation of membership by adding an entropy regularization term. FCM-σ [12] was proposed to improve FCM's performance on data points with uneven variations or non-spherical shapes in individual clusters. The authors in [13], [14] proposed to define a density-based membership value, addressing the limitation of FCM that cannot identify clusters with arbitrary shapes and densities. Fuzzy local information K-means [15] was designed to promote FCM's performance in image segmentation. An improved FCM clustering by varying the fuzziness parameter, called vFCM [16], was proposed to overcome the issue of carefully tuning the fuzziness parameter of FCM. Lohit and Kumar [17] proposed to use a modified picture fuzzy total Bregman divergence as a similarity measure to enhance noise robustness of FCM in magnetic resonance imaging (MRI) segmentation. Bose et al. [18] introduced a fuzzy membership function with Shannon's entropy-based variation to deal with the vagueness problem offered by mixed pixels in MRI. Recently, feature-weighted PCM (FWPCM) [19], [20] was proposed to give non-uniform importance to features. Research of combining fuzzy clustering and kernel mechanisms [21]–[23] has garnered interest, which non-linearly maps data from a low-dimensional space to a high-dimensional space through kernel functions to enhance data separability. Conversely, deep clustering, a modern technique that integrates deep neural networks (DNNs) and fuzzy clustering (or other clustering methods), has been proposed to cluster high-dimensional data by mapping them to a low-dimensional space where the curse of dimensionality is not presented [24]–[27]. Alternatively, a

Y. He is with the Department of Industry Engineering & Decision Analytics, The Hong Kong University of Science and Technology, Hong Kong (e-mail: yhebh@connect.ust.hk).

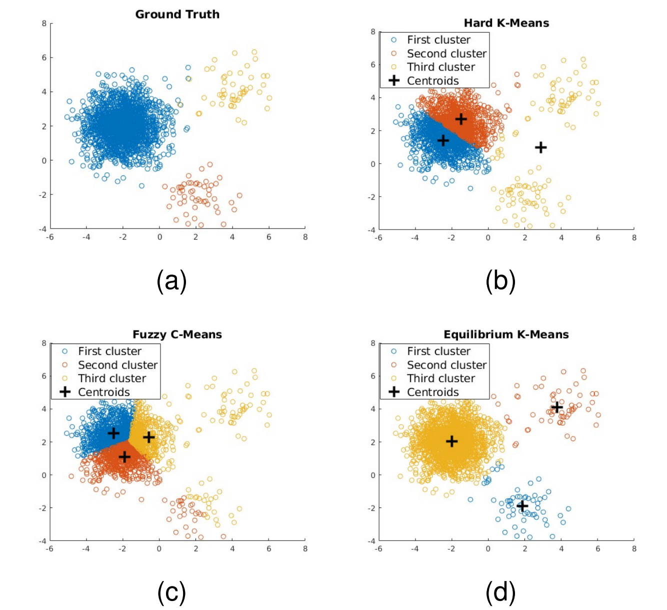

Fig. 1. Clustering results of a highly imbalanced dataset. (a) Ground truth. The colors represent the reference labels of the data points. (b) Clustering by hard K-means [6]. (c) Clustering by fuzzy C-means [5]. (d) Clustering by the proposed equilibrium K-means. Black crosses mark the positions of centroids calculated by algorithms.

hierarchical method based on fuzzy logic was proposed to deal with high-dimensional data [28].

B. Uniform Effect: The Problem of Fuzzy C-Means

The core idea of fuzzy clustering is to find prototypes, or centroids, of clusters. All the aforementioned fuzzy clustering algorithms calculate centroids in the same way - centroids are weighted means of data points, where weights are membership values or their exponents. Membership values, interpreted as levels of belonging, are naturally non-negative and typically positive in practice. A positive weight means that a data point has an attraction to the centroid. However, when dealing with imbalanced datasets where classes have highly unequal sizes, this non-negativity imposes a problem for FCM. Larger clusters with more data points exert a stronger attraction on the centroids than smaller clusters, causing the centroids closer to them. Ultimately, centroids fail to represent the true clusters, leading to poor clustering results. This issue is known as the *uniform effect* [29] and has been observed in FCM [30]. An illustration of the uniform effect of Lloyd's K-means (hereafter referred to as hard K-means, HKM) and FCM is given in Fig. 1, where we can observe that the centroids of HKM and FCM crowd together in the large cluster.

C. Efforts to Overcome Uniform Effect and Their Limitations

There are currently two methods that attempt to overcome the uniform effect. The first method is to scale the membership values according to cluster size. The bias towards larger clusters is reduced by assigning greater weight to data points in smaller clusters. Noordam et al. [31] and Lin et al. [32] adopted this method, proposing a cluster-size insensitive FCM (csiFCM) and a size-insensitive integrity-based FCM (siibFCM), respectively. Liu et al. [33] proposed an improved FCM (iFCM) that defines cluster size by the summation of membership values. Pu et al. [34] enhanced iFCM with edge modification (EM-iFCM) to address the issue that iFCM incorrectly deals with samples on the cluster edge. Priya et al. [35] introduced FkPCM_S_UB that combines a plane-based clustering method and cluster-size insensitivity to overcome the problem of imbalanced and non-spherical data clustering.

Although the method of scaling membership values based on cluster size is intuitive, it is too idealistic. First, scaling requires the true cluster size to which the data points belong. In unsupervised learning, accurately estimating cluster size is difficult. Second, even if the true cluster size is known, correctly scaling is difficult because we do not know which cluster a data point belongs to. Third, cluster sizes are estimated based on the clustering result of the current iteration, but clustering varies in each iteration, making the final performance unreliable. In the worst case, part of the noise data is incorrectly treated as a separate cluster. Studies have reported that both csiFCM and siibFCM are sensitive to noise and outliers [36].

The second method is called multiprototype clustering. This method first groups data into multiple subclusters of similar sizes, and the final clusters are obtained by merging adjacent subclusters. Liu et al. [37] proposed a novel fuzzy clustering validity index, called IMI2, that merges some clusters based on their separation value. Li et al. [38] proposed an adaptive tuning strategy to automatically select the number of multiprototypes and merge them using a convex optimization technique. Liang et al. [39] proposed a multiprototype clustering algorithm that employs an FCM algorithm robust to centroid initialization to generate subclusters. Later, Lu et al. [40] proposed a self-adaptive multiprototype clustering algorithm that automatically adjusts the number of subclusters. However, multiprototype clustering algorithms have a complex process and high time complexity, up to $O(N^2)$ for [39] and [40], where N is the number of data points in the dataset. Thus, they are computationally expensive for large datasets. We should additionally mention that Zeng et al. [41] recently proposed a soft multiprototype clustering algorithm with time complexity linear to N. However, their clustering process remains complex and aims to cluster high-dimensional and complex-structured data rather than imbalanced data.

In summary, existing methods to overcome the uniform effect are limited and do not address the root cause. These methods still compute centroids as non-negatively weighted means of data points. Hence, the problem of uniform effect will still arise for them. To the best of our knowledge, no research fundamentally addresses the uniform effect in FCM, nor is there a simple and effective fuzzy clustering algorithm for imbalanced data. The main challenges lie in (1) the implicit yet prevalent assumption of balanced data distribution, thus conventional paradigms lose effectiveness for imbalanced data distribution; (2) the underlying class distributions are unknown; and (3) the self-reinforcing coupling between membership and centroid drift amplification under imbalance, where traditional regularization (e.g., entropy) fails to decouple these inter-dependencies, causing the minority cluster to be overlooked. These necessitate novel mechanisms to disrupt the reinforcement loop and preserve natural cluster separation while suppressing majority dominance.

D. Our Contributions

In this paper, we propose a novel fuzzy clustering method, robust to imbalanced data as illustrated in Fig. 1d. The main contribution of this work lies in proposing the equilibrium K-means (EKM) algorithm, which fundamentally resolves the long-standing challenge of fuzzy clustering on imbalanced data by introducing a physics-inspired equilibrium mechanism. Specifically, our contribution includes three key aspects.

Firstly, we formulate the proposed EKM algorithm. Unlike the derivation of FCM, we do not treat membership and centroid as two independent optimization variables, but explicitly express membership as a function of the centroid and use it as an equality constraint to optimize the centroid. We appropriately choose the functional form of membership and construct a physically meaningful objective function. The centroid update equations obtained from a quasi-Newton method are as concise as those of FCM, but the weight can be negative. This means that clustering is generated in an equilibrium state, where each cluster exerts attractive and repulsive forces on centroids, and opposing forces are balanced in the final state. This equilibrium state directly counters majority dominance and preserves minority clusters without ad hoc regularization. The proposed EKM alternates between two simple steps and maintains the same time and space complexity as FCM in each iteration.

Secondly, a convergence condition of EKM is given based on a fixed-point theorem. We prove that EKM can converge exponentially fast with proper parameter selection. In addition, we analyze the root cause of why EKM surpasses FCM on imbalanced data.

Finally, a comprehensive study is conducted using four synthetic and 16 real datasets (hence 20 datasets in total) to demonstrate the effectiveness of EKM on balanced and imbalanced data. The results show that EKM is competitive on balanced datasets and significantly outperforms FCM and its many representative variants on imbalanced datasets (paired t-test and Wilcoxson signed-rank test with $p < 0.05$). We also demonstrate the practical computational efficiency, robustness to hyperparameters, and convergence of EKM.

E. Organization

We introduce FCM, and its representative variations, PCM and MEFC, in Section II. The goal is to introduce the traditional centroid update formula in FCM and explain its limitations. In Section III, we describe the proposed EKM algorithm and analyze its convergence and in which aspect it surpasses FCM. We evaluate the performance of EKM on 20 datasets in Section IV. Finally, we conclude in Section V.

II. What's Wrong with Fuzzy C-Means

A. Fuzzy C-Means

We consider the commonly used Euclidean distance as a similarity measure. FCM partitions N data objects into K clusters by minimizing the sum of weighted distances between data objects and centroids, with the following objective function and constraints:

$$\begin{aligned}\min_{\mathbf{c}_1,\cdots,\mathbf{c}_K,\{u_{kn}\}} \quad & \sum_{n=1}^{N}\sum_{k=1}^{K}(u_{kn})^m\|\mathbf{x}_n-\mathbf{c}_k\|_2^2 \\ \text{subject to} \quad & u_{kn}\in[0,\,1]\,\forall k,n, \\ & \sum_{k=1}^{K}u_{kn}=1\,\forall n, \\ & 0<\sum_{n=1}^{N}u_{kn}<N\,\forall k,\end{aligned} \tag{1}$$

where $\mathbf{x}_n$ represents the feature vector of n-th data object (we call $\mathbf{x}_n$ the n-th data point for simplicity), $\mathbf{c}_k$ denotes the k-th centroid, u_{kn} is a coefficient called membership that indicates the degree of $\mathbf{x}_n$ belonging to the k-th cluster, and $m \in (1, +\infty)$ is a hyperparameter controlling the degree of fuzziness level. Approximate optimization of FCM's objective function based on the iteration through the necessary conditions for its local extrema is formulated as [5]:

1) Calculate the membership value of the n-th data point belonging to the k-th cluster:

$$u_{kn}^{(\tau)}=\frac{1}{\sum_{i=1}^{K}\left(\frac{\|\mathbf{x}_n-\mathbf{c}_k^{(\tau)}\|_2}{\|\mathbf{x}_n-\mathbf{c}_i^{(\tau)}\|_2}\right)^{\frac{2}{m-1}}}. \tag{2}$$

2) Recalculate the weighted centroid of the k-th cluster by:

$$\mathbf{c}_k^{(\tau+1)}=\frac{\sum_n(u_{kn}^{(\tau)})^m\mathbf{x}_n}{\sum_n(u_{kn}^{(\tau)})^m}. \tag{3}$$

The time complexity of one iteration of the above two steps is $O(NK^2)$. It is clear that the membership u_{kn} in (2) lies in $[0, 1]$. In the limit of $m \to 1$, the membership u_{kn} converges to 0 or 1, and the objective of FCM coincides with HKM, producing the same data partition. In the following, we introduce two representative variations of FCM, which modify FCM by adding some regularization or penalty terms.

B. Possibilistic C-Means

Krishnapuram and Keller [7] proposed relaxing the probabilistic constraint in FCM, i.e., $\sum_{k=1}^{K} u_{kn} = 1$, and derived PCM. Compared to FCM, PCM assigns smaller membership values to noise data, making it more suitable for noisy datasets. PCM has the following objective function

$$\begin{aligned}\min_{\mathbf{c}_1,\cdots,\mathbf{c}_K,\{u_{kn}\}} \quad & \sum_{n=1}^{N}\sum_{k=1}^{K}(u_{kn})^m\|\mathbf{x}_n-\mathbf{c}_k\|_2^2 \\ & +\sum_{k=1}^{K}\eta_k\sum_{n=1}^{N}(1-u_{kn})^m,\end{aligned} \tag{4}$$

where $\{\eta_k\}_k$ are suitable positive numbers. The second term penalizes small u_{kn} to avoid all membership to 0. Approximate optimization of PCM's objective function based on the iteration through the necessary condition for its local extrema is

1) Calculate the membership value of the n-th data point belonging to the k-th cluster:

$$u_{kn}^{(\tau)} = \frac{1}{1 + (\|\mathbf{x}_n - \mathbf{c}_k^{(\tau)}\|_2^2/\eta_k)^{\frac{1}{m-1}}}. \tag{5}$$

2) Recalculate the weighted centroid of the k-th cluster by:

$$\mathbf{c}_k^{(\tau+1)} = \frac{\sum_n (u_{kn}^{(\tau)})^m \mathbf{x}_n}{\sum_n (u_{kn}^{(\tau)})^m}. \tag{6}$$

It is obvious that u_{kn} in (5) is within the range of $[0, 1]$.

C. Maximum-Entropy Fuzzy Clustering

Karayiannis [10] proposed MEFC in which an entropy term was added to regularize membership, making the membership physically meaningful. MEFC has the following objective function

$$\begin{aligned} \min_{\mathbf{c}_1,\cdots,\mathbf{c}_K,\{u_{kn}\}} \quad & (1-\eta)\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} u_{kn}\|\mathbf{x}_n - \mathbf{c}_k\|_2^2 \\ & + \eta \sum_{n=1}^{N}\sum_{k=1}^{K} u_{kn} \ln u_{kn} \\ \text{subject to} \quad & u_{kn} \in [0,\, 1]\, \forall k, n, \\ & \sum_{k=1}^{K} u_{kn} = 1\, \forall n, \end{aligned} \tag{7}$$

where $\eta \in (0, 1)$ is a hyperparameter, controlling the transition from maximization of the entropy to the minimization of centroid-data distances. Approximate optimization of MEFC's objective function based on iteration through the necessary conditions for its local extrema is given by:

1) Calculate the membership value of the n-th data point belonging to the k-th cluster:

$$u_{kn}^{(\tau)} = \frac{\exp\left(-\lambda\|\mathbf{x}_n - \mathbf{c}_k^{(\tau)}\|_2^2\right)}{\sum_{i=1}^{K}\exp\left(-\lambda\|\mathbf{x}_n - \mathbf{c}_i^{(\tau)}\|_2^2\right)}. \tag{8}$$

2) Recalculate the weighted centroid of the k-th cluster by:

$$\mathbf{c}_k^{(\tau+1)} = \frac{\sum_n (u_{kn}^{(\tau)}) \mathbf{x}_n}{\sum_n (u_{kn}^{(\tau)})}. \tag{9}$$

where $\lambda = \frac{1}{N}\frac{1-\eta}{\eta}$ and u_{kn} in (8) falls within $[0, 1]$.

D. Discussion

FCM, PCM, and MEFC calculate centroids in the same way: centroids are weighted means of all data points, with weights being membership values or their exponents. Such a formula is also employed by most fuzzy clustering algorithms based on FCM. Membership is non-negative by its definition and positive in practice. Consequently, the centroids in FCM are attracted by all data points and tend to be closer to larger clusters with the most data points. This is the core reason causing the uniform effect. The uniform effect is not specific to FCM; it also occurs in PCM and MEFC. This problem is not caused by a poor choice of regularization or penalty terms for the membership; rather, it will arise for any *multiplicative* objective function, which can be expressed as a multiplication of membership and distance between data points and centroids, with membership and centroid being independent optimization variables. The necessary conditions for optimizing such an objective function are the same as those in the theorems for the FCM, resulting in the same centroid update formula. Since the multiplicative objective function is the most intuitive objective function, the uniform effect is prevalent in fuzzy clustering methods.

III. Equilibrium K-Means

A. Objective Function

Considering the aforementioned limitations of FCM, herein, we propose a novel fuzzy clustering framework that does not follow the fashion of FCM. Specifically, we do not treat membership and centroid as two independent optimization variables. Instead, we explicitly express membership as a function of centroids and use it as an equality constraint to optimize centroids. Our proposed objective function is

$$\begin{aligned} \min_{\mathbf{c}_1,\cdots,\mathbf{c}_K} \quad & \sum_{n=1}^{N}\sum_{k=1}^{K} u_{kn} d_{kn}^2 \\ \text{subject to} \quad & u_{kn} = u_{kn}(\mathbf{c}_1, \cdots, \mathbf{c}_K)\, \forall k, n, \end{aligned} \tag{10}$$

where $d_{kn} = \|\mathbf{x}_n - \mathbf{c}_k\|_2$ is the Euclidean distance between n-th data point and k-th centroid, and $u_{kn}(\cdot)$ is an arbitrary real-valued function that turns a vector of K real values into a real value lying in $[0, 1]$. We propose to define $u_{kn}(\cdot)$ as

$$u_{kn}(\mathbf{c}_1, \cdots, \mathbf{c}_K) = \frac{\exp(-\alpha d_{kn}^2)}{\sum_{k=1}^{K}\exp(-\alpha d_{kn}^2)}, \tag{11}$$

where $\alpha \in (0, +\infty)$ is a hyperparameter to control the scaling of distance. The choice of α will be discussed later. Such a definition has several benefits. First, the exponential function is easy to differentiate, facilitating optimization. Second, u_{kn} defined in (11) has non-negative value and $\sum_{k=1}^{K} u_{kn} = 1$, which qualify $\mathbf{U} = \{u_{kn}\}_{k,n}$ as a probabilistic partition matrix. Third, the objective function in (10) has a physical meaning that we will elaborate on in the following sections.

B. Optimization

Substitute the explicit form (11) of u_{kn}, the equality-constrained optimization problem (10) can be reformulated as an unconstrained problem

$$\min_{\mathbf{c}_1,\cdots,\mathbf{c}_K} \quad J_{\text{EKM}}(\mathbf{c}_1, \cdots, \mathbf{c}_K) = \sum_{n=1}^{N}\sum_{k=1}^{K} u_{kn} d_{kn}^2, \tag{12}$$

where readers should keep it in mind that u_{kn} is a function of $\mathbf{c}_1, \cdots, \mathbf{c}_K$. We apply the gradient descent method to optimize the above problem. The objective J_{EKM} possesses the first-order partial derivative of

$$\partial_{\mathbf{c}_k} J_{\text{EKM}} = \frac{\partial J_{\text{EKM}}}{\partial \mathbf{c}_k} = -2\sum_{n=1}^{N} w_{kn}(\mathbf{x}_n - \mathbf{c}_k), \tag{13}$$

where

$$
\begin{aligned}
w_{kn} &= u_{kn}\big[1-\alpha(d_{kn}^2-\sum_{i=1}^{K}u_{in}d_{in}^2)\big]\\
&= u_{kn}\big[1-\alpha(d_{kn}^2-u_{kn}d_{kn}^2-\sum_{i\neq k}u_{in}d_{in}^2)\big]\\
&= u_{kn}\big[1-\alpha\sum_{i\neq k}(d_{kn}^2-d_{in}^2)u_{in}\big]\ \forall k,n.
\end{aligned}
\tag{14}
$$

The minimizer of J_{EKM} can be found using gradient descent iteration

$$
\mathbf{c}_k^{(\tau+1)} = \mathbf{c}_k^{(\tau)} - \gamma_k^{(\tau)}\partial_{\mathbf{c}_k}J_{\text{EKM}}(\mathbf{c}_1^{(\tau)},\cdots,\mathbf{c}_K^{(\tau)}), \tag{15}
$$

where $\gamma_k^{(\tau)}$ is the learning rate at the τ-th iteration. We can choose a fixed learning rate in each iteration, but it will not provide insight into EKM's effectiveness on imbalanced data. We propose to set an adaptive learning rate via a quasi-Newton method. The objective function J_{EKM} possesses the second-order partial derivative given by

$$
\frac{\partial^2 J_{\text{EKM}}}{\partial {\mathbf{c}_k}^2} = 2\sum_{n=1}^{N}(w_{kn}\mathbf{I} - 2\alpha u_{kn}\sum_{i\neq k}w_{in}\mathbf{D}_{kn}), \tag{16}
$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{D}_{kn} = (\mathbf{x}_n-\mathbf{c}_k)(\mathbf{x}_n-\mathbf{c}_k)^{\mathsf{T}}$ is a rank-1 matrix. Observing $w_{in} \propto u_{in}$, hence, $u_{kn}\sum_{i\neq k}w_{in} \propto u_{kn}\sum_{i\neq k}u_{in} = u_{kn}(1-u_{kn})$, which is smaller than u_{kn} especially when clusters are well-separated. Therefore, we can ignore the term of $\mathbf{D}_{kn}$ and approximate the second-order partial derivative of J_{EKM} by

$$
\frac{\partial^2 J_{\text{EKM}}}{\partial {\mathbf{c}_k}^2} \approx 2\sum_{n=1}^{N}w_{kn}\mathbf{I}. \tag{17}
$$

According to the iterative rule of Newton's method, where the Hessian matrix (16) is approximated by (17), the centroids of EKM can be updated by

$$
\begin{aligned}
\mathbf{c}_k^{(\tau+1)} &= \mathbf{c}_k^{(\tau)} - \Big(\frac{\partial^2 J_{\text{EKM}}}{\partial {\mathbf{c}_k}^2}\Big)^{-1}\cdot\partial_{\mathbf{c}_k}J_{\text{EKM}}\\
&\approx \mathbf{c}_k^{(\tau)} + \sum_{n=1}^{N}\frac{w_{kn}^{(\tau)}}{\sum_{n=1}^{N}w_{kn}^{(\tau)}}(\mathbf{x}_n-\mathbf{c}_k^{(\tau)})\\
&= \frac{\sum_{n=1}^{N}w_{kn}^{(\tau)}\mathbf{x}_n}{\sum_{n=1}^{N}w_{kn}^{(\tau)}}.
\end{aligned}
\tag{18}
$$

Similar to FCM, EKM is a two-step iteration algorithm alternating between weight calculation and centroid calculation. For convenience, we summarise the complete procedure of EKM in Algorithm 1. EKM converges when centroids cease to change or the maximum number of iterations is reached. The time complexity of one iteration of EKM is the same as that of FCM, which is $O(NK^2)$. It is important to note that although $\sum_{k=1}^{K}w_{kn}=1$, $\{w_{kn}\}_{k,n}$ cannot be interpreted as membership because some values are negative.

Algorithm 1: Equilibrium K-Means Algorithm

Input: A dataset $X=\{\mathbf{x}_n\}_{n=1}^{N}$, cluster number K, initial centroids $\{\mathbf{c}_k^{(0)}\}_{k=1}^{K}$, parameter α
Output: Centroids $\{\mathbf{c}_1,\cdots,\mathbf{c}_K\}$
$\tau = 0$;
repeat
  Compute weight $w_{kn}^{(\tau)}$ by (14) for all k, n;
  Update centroid $\mathbf{c}_k^{(\tau+1)}$ by (18) for all k;
  $\tau=\tau+1$;
until *convergence*;
return $\{\mathbf{c}_k^{(\tau)}\}_{k=1}^{K}$

C. Physical Interpretation

The second law of thermodynamics asserts that in a closed system with constant external parameters (e.g., volume) and fixed entropy, the internal energy will reach its minimum value at the state of thermal equilibrium. The objective function (12) of EKM follows this minimum energy principle. This connection can be established by envisioning data points as particles with discrete/quantized energy levels, where the number of energy levels is equivalent to the number of centroids, and the energy value corresponds to the squared Euclidean distance between a data point and a centroid. Boltzmann's law states that at the state of thermodynamic equilibrium, the probability of a particle occupying a specific energy level decreases exponentially with the increase of the energy value of that level. Hence, J_{EKM} equals the expectation of the entire system's energy, and EKM seeks centroids to minimize this energy expectation.

D. Why EKM Surpassing FCM

The centroid update equations (18) of EKM are similar to those (3) of FCM, as both are weighted means of data points. The key difference is the weight calculation formula. In FCM, the weight is the exponent of membership (i.e., u_{kn}^m), whereas in EKM, the weight is the membership multiplied by a factor (i.e., $u_{kn}\delta_{kn}$ where $\delta_{kn} = 1-\alpha\beta_{kn}$ and $\beta_{kn} = d_{kn}^2-\sum_{i=1}^{K}u_{in}d_{in}^2$). The use of the exponent of membership in FCM causes the uniform effect, but the situation changes in EKM due to the factor δ_{kn}. Note that β_{kn} can be reformulated as $\beta_{kn}=\sum_{i\neq k}u_{in}(d_{kn}^2-d_{in}^2)$. If the n-th data point is closest to the k-th centroid, β_{kn} is negative, and δ_{kn} will be greater than 1, indicating that the n-th data point attracts the k-the centroid. Conversely, if the n-th data point is furthest from the k-th centroid, β_{kn} is positive. When β_{kn} is positive and α is sufficiently large, δ_{kn} will be negative, indicating that the n-th data point repulses the k-th centroid. In summary, the clustering feature of EKM is that data points close to a centroid will attract that centroid and repel other farther centroids. Because of the offsetting between attraction and repulsion, the learning bias towards large clusters is reduced in EKM. This is why EKM surpasses FCM on imbalanced data.

To better understand the distinction between EKM and FCM, we provide a simple example on an x-axis to illustrate

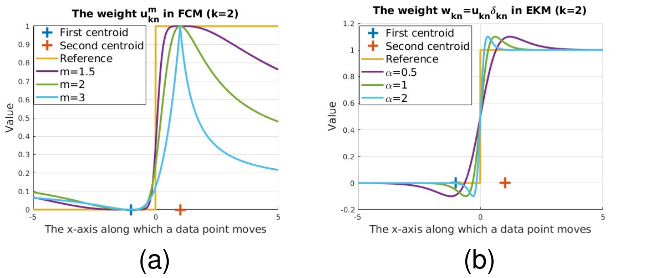

Fig. 2. Weight of the n-th data point to the second centroid as a function of the data point moving along the x-axis. The blue and red crosses mark the positions of the first and second centroids, respectively. The yellow curve is the weight of HKM [6], for reference. The purple, green, and blue curves are weights of FCM and EKM with different parameter values. (a) The weight u_{kn}^m of FCM [5]. (b) The weight $w_{kn} = u_{kn}\delta_{kn}$ of EKM.

how the weight of a data point in EKM and FCM changes with its position. We fix the first centroid at -1 and the second centroid at $+1$. Fig. 2a shows u_{kn}^m of FCM, and Fig. 2b displays $w_{kn} = u_{kn}\delta_{kn}$ of EKM. We focus on the weight of a data point on the second centroid. In addition, we only show the weight of FCM. Other fuzzy clustering algorithms (such as PCM and MEFC) produce similar results and do not alter the conclusions. Therefore, they are omitted to save space. We can see from Fig. 2a that in FCM, if a data point is positioned on the side of the first centroid, its weight to the second centroid is non-negative and generally positive. Therefore, if there are enough data points on the side of the first centroid, the second centroid will be attracted to that side[1]. In contrast, the weight in EKM can be negative. Therefore, regardless of the number of data points on the side of the first centroid, the second centroid in EKM will be less deviated because of the offsetting force.

E. Convergence Analysis

We give a convergence condition of EKM based on the fixed-point theorem. It turns out that EKM can always converge exponentially fast with arbitrary initial centroids as α is sufficiently small or large.

Theorem 1 (Convergence Condition)**:** *The centroid sequence obtained by* (18) *converges exponentially to a stationary point* $\{\mathbf{c}_1^*, \cdots, \mathbf{c}_K^*\}$ *of the objective function* J_{EKM} (12) *if a constant* $0 < v < 1$ *exists with*

$$M_k(\alpha; \{\mathbf{c}_1, \cdots, \mathbf{c}_K\}) \leq v, \quad \forall k, \{\mathbf{c}_1, \cdots, \mathbf{c}_K\} \in \mathbb{S},$$

where

$$\mathbb{S} := \{\{\mathbf{c}_1, \cdots, \mathbf{c}_K\} : \|\mathbf{c}_k - \mathbf{c}_k^*\|_2 \leq \|\mathbf{c}_k^{(0)} - \mathbf{c}_k^*\|_2 \, \forall k\},$$

and

$$M_k := |\sum_{n=1}^N w_{kn}|^{-1} \sum_{n=1}^N 2\alpha u_{kn} |1 - w_{kn}| d_{kn} \cdot \|\mathbf{x}_n - \frac{\sum_n w_{kn}\mathbf{x}_n}{\sum_n w_{kn}}\|_2.$$

[1]Unless the membership is carefully scaled according to the true cluster size, but we have discussed that such scaling is neither feasible nor reliable in unsupervised learning.

Proof: Define $f(\mathbf{c}_k) = (\sum_{n=1}^N w_{kn}\mathbf{x}_n)/(\sum_{n=1}^N w_{kn})$. Observe the gradient of J_{EKM} given in (13), it is clear that any stationary point of J_{EKM} is a fixed point of f and vice versa. Hence, we have

$$\begin{aligned}\|\mathbf{c}_k^{(\tau)} - \mathbf{c}_k^*\|_2 &= \|f(\mathbf{c}_k^{(\tau-1)}) - \mathbf{c}_k^*\|_2 \\ &\leq \|f'(\boldsymbol{\xi}_k)\|_{\text{F}} \|\mathbf{c}_k^{(\tau-1)} - \mathbf{c}_k^*\|_2,\end{aligned}$$

where f' denotes the first derivative of f, $\|\cdot\|_{\text{F}}$ is the Frobenius norm, and $\boldsymbol{\xi}_k$ is a point on the line with $\mathbf{c}_k^{(\tau-1)}$ and $\mathbf{c}_k^*$ as endpoints. The inequality holds according to the median value theorem. The function f has the first derivative of

$$f'(\mathbf{c}_k) = (\sum_{n=1}^N w_{kn})^{-1} \sum_{n=1}^N 2\alpha u_{kn}(1 - w_{kn})(\mathbf{x}_n - \mathbf{c}_k) \cdot (\mathbf{x}_n^\intercal - \frac{\sum_{n=1}^N w_{kn}\mathbf{x}_n^\intercal}{\sum_{n=1}^N w_{kn}})$$

Note that $\|f'(\mathbf{c}_k)\|_{\text{F}} \leq M_k \, \forall \mathbf{c}_k$ by triangular inequality and $\{\boldsymbol{\xi}_1, \cdots, \boldsymbol{\xi}_K\} \in \mathbb{S}$, we have

$$\begin{aligned}\|\mathbf{c}_k^{(\tau)} - \mathbf{c}_k^*\|_2 &\leq v\|\mathbf{c}_k^{(\tau-1)} - \mathbf{c}_k^*\|_2 \leq \dots \\ &\leq v^\tau \|\mathbf{c}_k^{(0)} - \mathbf{c}_k^*\|_2 \, \forall k = 1, \cdots, K.\end{aligned}$$

Since $v < 1$,

$$\lim_{\tau\to\infty} \|\mathbf{c}_k^{(\tau)} - \mathbf{c}_k^*\|_2 \leq \lim_{\tau\to\infty} v^\tau \|\mathbf{c}_k^{(0)} - \mathbf{c}_k^*\|_2 = 0 \, \forall k$$

■

We discuss the value of M_k in two limiting cases ($\alpha \to 0$ and $\alpha \to +\infty$) to provide insight into the convergence property of EKM. When $\alpha \to 0$, $u_{kn} \to 1/K$ and $w_{kn} \to 1/K$ for all k and n, leading M_k to approach 0. This means that convergence requires only one step. Hence, EKM can always converge fast with arbitrary initial centroids as α is sufficiently small. Of course, to avoid a trivial solution, α cannot be set directly to zero. We will discuss the parameter tuning later.

In the case of well-separated data, where each cluster has no overlap and the initial centroids are close to their corresponding clusters, as $\alpha \to +\infty$, we have $\alpha u_{kn}|1 - w_{kn}| = |\sum_{i\neq k} \alpha u_{kn} u_{in}(1 - \alpha\beta_{in})| \to 0$. Consequently, $M_k \to 0$. This occurs because either u_{kn} or u_{in}, one of them approaches 0 and decays exponentially with α. In fact, when $\alpha \to +\infty$, EKM can always converge because the centroid update equations of EKM converge to those of HKM, which are known to converge to a (local) minimum or saddle point [42].

F. The Choice of EKM's parameter

In EKM, only one parameter α needs to be tuned. The optimal choice of α remains unknown, a common issue in unsupervised learning due to limited information. Similarly, FCM struggles with selecting the optimal fuzzifier value, m. Despite numerous studies discussing the selection of m, a widely accepted solution has yet to be found [43].

Nonetheless, we can have a suitable range for α based on the analysis above. On the one hand, the convergence of EKM is always guaranteed when α is sufficiently small. On the other hand, we aim for $\delta_{kn} = 1 - \alpha\beta_{kn}$ to be negative when β_{kn} is positive to introduce repulsion between data points and centroids. Hence, α cannot be too small. Noting that both β_{kn} and M_k involve a distance term, we can estimate α from distance statistics such as variance of datasets. Such estimation ensures that β_{kn} and M_k maintain a certain measure invariance, thus promoting both convergence and the presence of repulsion.

The value of α can be fixed throughout all iterations or vary in each iteration. If the iterations fail to converge, reducing α may help; if the centroids are too close, increasing α might be beneficial. In our experiments, we consistently use a fixed α value for convenience. Both performance and convergence are satisfactory and consistent with our analysis.

To ensure practicality across diverse scenarios, we propose two strategies for selecting α, depending on the user's prior knowledge of the dataset. For datasets with known structural characteristics, a predetermined α can be empirically chosen. For instance, in our experiments on artificial datasets (Section IV-B), we set $\alpha = 0.5$. This strategy is optimal when users have domain expertise or validated default values for similar data structures. For real-world datasets (Section IV-D and Section IV-E) with unknown or complex distributions, we dynamically compute α using the data variance according to our analysis. Unlike tuning-based approaches, this eliminates manual intervention and generalizes across unseen datasets.

IV. Numerical Experiments

A. Experimental Setup

Numerical experiments are conducted to compare the performance of our proposed EKM algorithm with 13 state-of-the-art hard and fuzzy clustering algorithms including (1) HKM [6], (2) FCM [5], (3) PCM [7], (4) MEFC [10], (5) PFCM [8], (6) csiFCM [31], (7) siibFCM [32], (8) C-PCM [9], (9) FWPCM [19], (10) iFCM [33], (11) EM-iFCM [34], (12) vFCM [16], and (13) IMI2 [37]. Multiprototype clustering algorithms (e.g., [39], [40]) are not appropriate as baseline algorithms because they are too complex to be benchmarks for gauging the efficiency of EKM. We also avoid using hybrid methods, such as algorithms combining K-means with kernels proposed in [21]–[23], as benchmarks to ensure fairness. Implementation is conducted in MATLAB R2022a, and the operating system is Ubuntu 18.04.1 LTS with Intel Core i9-9900K CPU @ 3.60GHz X 16 and 62.7 GiB memory. All datasets used and codes are available at https://github.com/ydcnanhe/Imbalanced-Data-Clustering-using-Equilibrium-K-Means.git.

The experimental datasets contain four artificial datasets generated by us (including Data-A, Data-B, Data-C, and Data-D), 13 UCI [44] datasets (consisting of Image Segmentation (IS), Seeds, Wine, Rice, Wisconsin Diagnostic Breast Cancer (WDBC), Ecoli, Htru2, Zoo, Glass, Shill Bidding, Anuran Calls, Occupancy Detection, Machine Failure), and three Kaggle datasets (incorporating Heart Disease, Pulsar Cleaned, and Bert-Embedded Spam). So there are 20 datasets, and Table I provides their information, including name, instance number, feature number, reference class number, and coefficient of variation (CV). CV is used in previous literature [45] to measure the level of data dispersion. It is calculated as the ratio of the standard deviation of class sizes to the mean. Given the number of instances in each class as $N_1, \cdots, N_K$, we have

$$\text{CV} = s/\bar{N}, \tag{19}$$

where

$$\bar{N} = \frac{\sum_{k=1}^{K} N_k}{K}, \; s = \sqrt{\frac{\sum_{k=1}^{K}(N_k - \bar{N})^2}{K-1}}.$$

In [45], a CV value exceeding 1 indicates highly varying class sizes, while a value below 0.3 signifies uniform class sizes. However, there isn't a widely accepted critical CV value implying that the data is imbalanced. For rigorous statements, we establish that if a dataset's CV is less than 0.4, it is considered balanced. Conversely, if the CV exceeds 0.7, the dataset is deemed imbalanced. Consequently, we have six balanced datasets and 14 imbalanced datasets. It should be noted that none of the datasets used has a CV value between 0.4 and 0.7, hence this range remains undefined.

TABLE I
Detailed description of 20 datasets. The larger the CV value, the more imbalanced the dataset

ID	Name	Instances	Feature	Classes	CV
D1	Data-A	2250	2	3	1.4468
D2	Data-B	2250	2	3	1.4468
D3	Data-C	5200	2	2	1.3054
D4	Data-D	5400	2	9	2.7500
D5	IS	2310	19	7	0
D6	Seeds	210	7	3	0
D7	Heart Disease	1125	13	2	0.0373
D8	Wine	178	13	3	0.1939
D9	Rice	3810	7	2	0.2042
D10	WDBC	569	30	3	0.3604
D11	Zoo	101	16	7	0.8937
D12	Glass	214	9	6	1.0767
D13	Ecoli	336	7	8	1.1604
D14	Htru2	17898	8	2	1.1552
D15	Shill Bidding	6321	9	2	1.1122
D16	Anuran Calls	7195	22	10	1.6016
D17	Occupancy Detection	20560	5	2	0.7608
D18	Machine Failure	9815	7	2	1.3318
D19	Pulsar Cleaned	14987	7	2	1.3561
D20	Bert-Embedded Spam	5572	768	2	1.0350

All datasets undergo normalization, ensuring that each feature has zero mean and unit variance. All features are used for clustering purposes. The number of clusters is set to the reference class number[2]. Convergence is achieved when the moving distance of centroids between successive iterations is sufficiently small relative to the magnitude of centroids, i.e.,

$$\frac{\left(\sum_{k=1}^{K}\|\mathbf{c}_k^{(\tau)} - \mathbf{c}_k^{(\tau-1)}\|_2^2\right)^{1/2}}{\left(\sum_{k=1}^{K}\|\mathbf{c}_k^{(\tau)}\|_2^2\right)^{1/2}} \leq 1\text{e}{-3}. \tag{20}$$

We set the maximum number of iterations to 500 to prevent algorithms from infinitely iterating due to failure to converge. Except for sporadic cases, all algorithms can reach the specified convergence within 500 iterations.

[2]When reference class numbers are unavailable, cluster validity indexes, widely used to set the number of clusters when using FCM (e.g., BXI [46], IMI [47], WLI [48]), are directly applicable to EKM due to analogous outputs (centroids via (18) and membership via (11)).

Centroids are initialized using the K-means++ algorithm [49]. Given that each run of K-means++ yields different outputs and the objectives of the tested clustering algorithms are known to be non-convex (i.e., multiple local optima exist), convergence may happen at one of the local optimal points. A common way of finding the global optimum is to carry out a number of replications followed by a selection of the best (lowest) objective value. Hence, each trial includes 100 repetitions, and we select the repetition with the lowest objective value as the final result for that trial. We average the performance of 50 trials to ensure the rationality of the experiment. This means we conduct $50 \times 100 = 5000$ runs for each algorithm. 100 repetitions in each trial is still within the practical executable range, and each trial can give almost consistent results as indicated by a small standard deviation. We utilize widely accepted measures of clustering performance for our evaluation indexes, including the normalized mutual information (NMI) [50], the adjusted rand index (ARI) [51], and the clustering accuracy index (ACC) [52]. Both NMI and ACC range from 0 to 1, with 0 indicating the worst and 1 representing the best. ARI has a range from -1 to 1, in which -1 means two data clusterings are completely dissimilar, 0 indicates that data clusterings are essentially random, and 1 means two data clusterings are perfectly aligned. It should be noted that for imbalanced datasets, a large ACC value is only necessary for good clustering performance, but not sufficient because trivially classifying all data points into one cluster will also lead to a large ACC value. Therefore, ACC must be considered together with NMI and ARI for imbalanced datasets.

If not otherwise stated, all parameters used in the comparative algorithms follow the guidance of their original papers. Specifically, FCM, PCM, PFCM, csiFCM, siibFCM, C-PCM, FWPCM, iFCM, EM-iFCM, and IMI2 employ a typical fuzzifier value of $m = 2$ [53]. The optimal regularization parameter (i.e., λ) of MEFC varies for different datasets, and we set $\lambda = 1$ for all datasets without losing fairness. The $\lambda = 1$ falling in its empirical optimal range [54]. The vFCM starts with $m = 2$ as suggested in [16], and m is automatically adjusted during iterations by its adaptive design. IMI2 will traverse the number of clusters within a given range and merge redundant clusters. We set the minimum number of clusters to K, the maximum number to $2K$, and the number of clusters after merging to K, where K is the reference class number. This setting ensures both effectiveness and efficiency. For EKM, we set $\alpha = 0.5$ for the four artificial datasets. A predetermined value of α can be given because the four datasets have the same dimension and similar data structures. For the 16 real-world datasets, as their feature numbers differ, a suitable α value varies accordingly. To avoid parameter tuning favoring EKM and thus losing fairness to the comparative algorithms, we propose to set α proportional to the data variance, as follows

$$\alpha = 2N / \sum_{n=1}^{N} \|\mathbf{x}_n\|_2^2. \tag{21}$$

Such a selection theoretically benefits convergence and effectiveness of EKM, as explained in Section III-F. For fairness, all statistical tests (Section IV-F) compare EKM with adaptive α against comparative methods using real-world datasets only. Results on artificial datasets are provided solely to validate theoretical properties.

TABLE II
EXPERIMENTAL RESULTS ON FOUR ARTIFICIAL IMBALANCED DATASETS WITH $CV > 1$ (THE BEST PERFORMANCE IS IN BOLD)

Dataset	Measurement	HKM	FCM	PCM	MEFC	PFCM	csiFCM	siibFCM	C-PCM	FWPCM	iFCM	EM-iFCM	vFCM	IMI2	EKM
Data-A	NMI	0.5150 ±0.0001	0.4982 ±0.0001	0.0000 ±0.0000	0.2270 ±0.0002	0.4984 ±0.0001	0.7579 ±0.0471	0.4924 ±0.0503	0.4964 ±0.0001	0.1427 ±0.0488	0.3184 ±0.0000	0.3180 ±0.0919	0.5085 ±0.0011	**0.9547 ±0.0029**	0.9126 ±0.0000
	ARI	0.2906 ±0.0002	0.2485 ±0.0002	0.0000 ±0.0000	0.0756 ±0.0003	0.2489 ±0.0003	0.8545 ±0.0329	0.2974 ±0.0490	0.2441 ±0.0003	0.0116 ±0.0357	0.1693 ±0.0002	0.2243 ±0.1448	0.2739 ±0.0029	**0.9813 ±0.0017**	0.9616 ±0.0000
	ACC	0.6893 ±0.0003	0.6075 ±0.0006	0.8889 ±0.000	0.4818 ±0.0003	0.6089 ±0.0009	0.9429 ±0.0164	0.7443 ±0.0192	0.5933 ±0.0010	0.4562 ±0.0393	0.6964 ±0.0004	0.7799 ±0.0627	0.6633 ±0.0055	**0.9968 ±0.0003**	0.9933 ±0.0000
Data-B	NMI	0.5193 ±0.0000	0.5160 ±0.0000	0.0013 ±0.0000	0.2121 ±0.0001	0.5181 ±0.0001	0.7989 ±0.0677	0.5090 ±0.0419	0.5247 ±0.0686	0.1591 ±0.0577	0.2553 ±0.0011	0.2170 ±0.0864	0.5170 ±0.0000	**1.000 ±0.000**	0.9981 ±0.0057
	ARI	0.2529 ±0.0000	0.2452 ±0.0000	-0.0008 ±0.0000	0.0547 ±0.0000	0.2498 ±0.0003	0.8794 ±0.0406	0.2461 ±0.0486	0.2582 ±0.1070	0.0332 ±0.0497	0.1034 ±0.0009	0.0873 ±0.0817	0.2475 ±0.0000	**1.000 ±0.000**	0.9992 ±0.0023
	ACC	0.6116 ±0.0000	0.5852 ±0.0002	0.8884 ±0.0000	0.4607 ±0.0016	0.6022 ±0.0009	0.9424 ±0.0194	0.6304 ±0.0278	0.5839 ±0.0601	0.4767 ±0.0671	0.5450 ±0.0008	0.5869 ±0.0776	0.5942 ±0.0000	**1.000 ±0.000**	0.9999 ±0.0004
Data-C	NMI	0.0909 ±0.0000	0.0797 ±0.0000	0.0002 ±0.0002	0.0892 ±0.0000	0.0782 ±0.0001	0.1069 ±0.0928	0.0769 ±0.1281	0.0757 ±0.0002	0.0783 ±0.0151	0.0242 ±0.0000	0.0196 ±0.0257	0.0893 ±0.0000	0.3700 ±0.0000	**0.9514 ±0.0000**
	ARI	0.0181 ±0.0000	0.0034 ±0.0001	-0.0002 ±0.0002	0.0159 ±0.0001	0.0015 ±0.0001	0.0728 ±0.1242	0.0199 ±0.1713	-0.0017 ±0.0002	0.0053 ±0.0090	-0.0147 ±0.0000	-0.0435 ±0.0322	0.0160 ±0.0000	0.1221 ±0.0000	**0.9807 ±0.0000**
	ACC	0.5754 ±0.0000	0.5302 ±0.0002	0.9615 ±0.0001	0.5689 ±0.0002	0.5237 ±0.0002	0.6981 ±0.2108	0.8207 ±0.0436	0.5126 ±0.0009	0.5371 ±0.0272	0.5695 ±0.0002	0.8664 ±0.0514	0.5694 ±0.0000	0.5221 ±0.0000	**0.9987 ±0.0000**
Data-D	NMI	0.4799 ±0.0316	0.3185 ±0.0010	0.0007 ±0.0001	0.3963 ±0.0670	0.2353 ±0.0554	0.3984 ±0.0440	0.3022 ±0.0729	0.3310 ±0.0545	0.1733 ±0.0268	0.1790 ±0.0093	0.1452 ±0.0173	0.3456 ±0.0542	0.7135 ±0.0028	**0.9463 ±0.0104**
	ARI	0.1096 ±0.0195	0.0467 ±0.0007	-0.0007 ±0.0002	0.1506 ±0.0360	0.0389 ±0.0183	0.0813 ±0.0266	0.0339 ±0.0260	0.0480 ±0.0190	0.0083 ±0.0068	0.0104 ±0.0056	-0.0082 ±0.0113	0.0559 ±0.0207	0.8094 ±0.0059	**0.9954 ±0.0049**
	ACC	0.3593 ±0.0555	0.2559 ±0.0080	0.9256 ±0.0001	0.4893 ±0.0133	0.3699 ±0.0380	0.3357 ±0.0564	0.2330 ±0.0398	0.2501 ±0.0379	0.2206 ±0.0325	0.2962 ±0.0136	0.2649 ±0.0353	0.2721 ±0.0384	0.9630 ±0.0000	**0.9763 ±0.0045**

The explicit output of EKM is the set of cluster centroids $\{\mathbf{c}_k\}_{k=1}^K$, and the membership matrix $\{u_{kn}\}$ is fully determined by the equilibrium condition (11). For hard cluster assignment, EKM assigns each data point $\mathbf{x}_n$ to the cluster k corresponding to the maximum membership value u_{kn}. This assignment is equivalent to selecting the nearest centroid $\mathbf{c}_k$, as the membership u_{kn} inversely correlates with the distance d_{kn}.

B. Artificial Datasets

Fig. 3 displays scatter plots of the four artificial datasets along with selected clustering results. The average and the standard deviation of evaluation indexes over 50 trials are provided in Table II, where the best results are highlighted in bold. On Data-A, the data points are sampled from Gaussian distributions. While on Data-B the data points are sampled from uniform distributions. Data-C and Data-D are a mixture of Gaussian and uniform distributions. The difference is that the majority of data points on Data-C are sampled from a Gaussian distribution while the majority of data points on Data-D are sampled from a uniform distribution. Additionally, Data-D has more classes than Data-C. All four artificial datasets are highly imbalanced with CV values greater than one.

We can see from Table II that the performance of benchmark algorithms is poor on these artificial datasets, precluding IMI2, which has good performance on Data-A and Data-B, but still performs poorly on Data-C and Data-D. Only EKM is effective on all four datasets. In detail, the best benchmark algorithm (i.e., IMI2) only has 0.3700 and 0.7135 NMI scores on Data-C and Data-D, respectively, while EKM has 0.9514 and 0.9463 NMI scores. As seen in Fig. 3, the benchmark algorithms erroneously divide the majority class into multiple clusters to balance the data size (i.e., the uniform effect) while EKM does not. This experiment also proves that EKM is versatile in handling different data distributions.

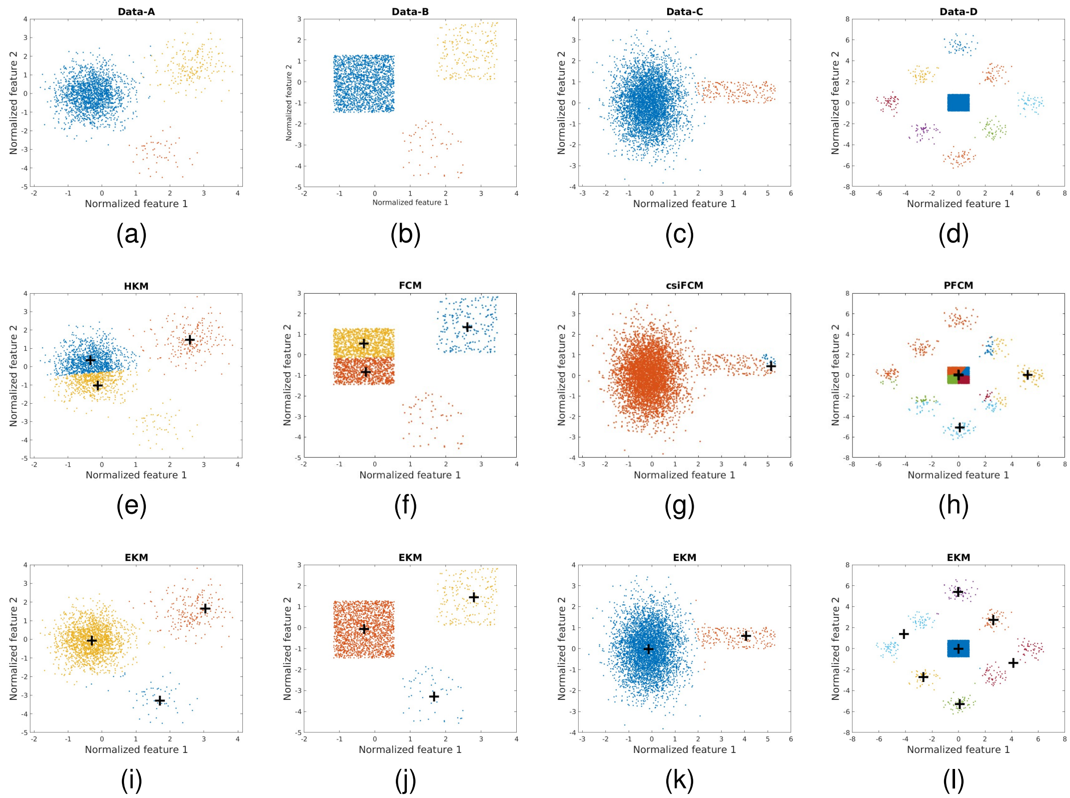

Fig. 3. Scatter diagrams of artificial datasets. (a)-(d) Primitive scatter diagrams with reference class labels (indicated by colors). (e)-(f) Clustering results of some benchmark algorithms (from left to right: HKM [6], FCM [5], csiFCM [31], and PFCM [8]). (i)-(l) Clustering results of the proposed EKM. The black crosses represent centroids obtained by each algorithm.

C. Study of Parameter Impact

We investigate the influence of the parameter α on EKM in this experiment. We perform EKM with α values of $0.05, 0.1, 0.25, 0.4, 0.5, 1, 2.5, 4, 5$ on the four artificial datasets. Fig. 4 presents the corresponding NMI values for these α values. The NMI value of HKM is considered a reference. It is evident that when $\alpha > 2.5$, the NMI of EKM closely aligns with that of HKM. This similarity arises because a sufficiently large α makes the objective of EKM nearly identical to the objective of HKM. As the value of α decreases, the NMI of EKM increases abruptly, which implies the emergence of a repulsive force overcoming the uniform effect. When $\alpha < 0.1$, EKM becomes inferior to HKM. This is because the centroids of EKM tend to overlap when α is particularly small. Fortunately, the range of α that causes the centroids to overlap is narrow.

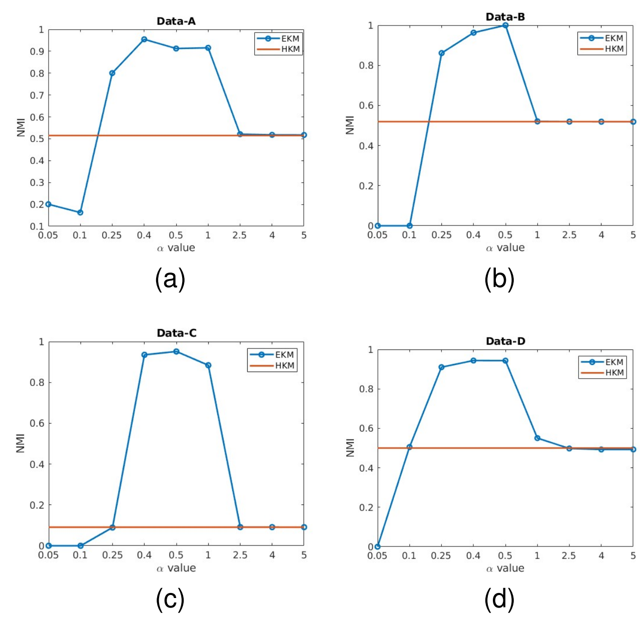

Fig. 4. NMI of EKM with different α values on the four artificial datasets. The blue curve is the NMI of EKM, and the red line is the NMI of HKM as a reference.

D. Real Datasets with Balanced Data

To evaluate EKM's effectiveness on balanced data, we perform it on six real datasets: IS, Seeds, Heart Disease, Wine, Rice, and WDBC. These datasets are routinely used in the field of machine learning and are characterized by uniform class sizes, CV values less than 0.4, and are categorized as balanced datasets as per our protocol. Table III shows the clustering results. EKM delivers superior performance on the Wine dataset and achieves comparable results to the baseline on the remaining five datasets. Although EKM is designed for imbalanced data, this experiment proves that EKM is also competitive on balanced data.

TABLE III
EXPERIMENTAL RESULTS ON THE SELECTED SIX REAL-WORLD BALANCED DATASETS WITH $CV < 0.4$ (THE BEST PERFORMANCE IS IN BOLD)

Dataset	Measurement	HKM	FCM	PCM	MEFC	PFCM	csiFCM	siibFCM	C-PCM	FWPCM	iFCM	EM-iFCM	vFCM	IMI2	EKM
IS	NMI	0.5873 ±0.0008	0.5950 ±0.0057	0.0000 ±0.0000	0.6206 ±0.0009	0.4962 ±0.0241	0.5195 ±0.0317	0.5925 ±0.0371	0.5743 ±0.0190	0.5452 ±0.0664	0.4971 ±0.0170	0.4727 ±0.0169	0.5958 ±0.0078	0.5792 ±0.0762	**0.6618 ±0.0138**
	ARI	0.4608 ±0.0005	0.4960 ±0.0059	0.0000 ±0.0000	0.4979 ±0.0005	0.3650 ±0.0149	0.3472 ±0.0587	0.3776 ±0.0794	0.4844 ±0.0199	0.4022 ±0.0884	0.1666 ±0.0147	0.1422 ±0.0126	**0.5092 ±0.0067**	0.3502 ±0.1368	0.4810 ±0.0089
	ACC	0.5456 ±0.0003	0.6578 ±0.0146	0.1429 ±0.000	0.5943 ±0.0002	0.5276 ±0.0139	0.4977 ±0.0670	0.5840 ±0.0511	0.6208 ±0.0246	0.5576 ±0.0775	0.4230 ±0.0163	0.3948 ±0.0157	**0.6611 ±0.0049**	0.5196 ±0.1212	0.5609 ±0.0118
Seeds	NMI	0.7279 ±0.0000	0.7318 ±0.0056	0.0558 ±0.0162	**0.7496 ±0.0016**	0.7170 ±0.0000	0.7200 ±0.0077	0.6893 ±0.0015	0.7279 ±0.0000	0.5409 ±0.0336	0.7137 ±0.0000	0.6969 ±0.0262	0.7384 ±0.0000	0.7389 ±0.0000	0.7315 ±0.0000
	ARI	0.7733 ±0.0000	0.7768 ±0.0058	0.0008 ±0.0009	**0.7968 ±0.0017**	0.7607 ±0.0000	0.7640 ±0.0082	0.7143 ±0.0034	0.7733 ±0.0000	0.4724 ±0.0513	0.7452 ±0.0000	0.7190 ±0.0461	0.7850 ±0.0000	0.7842 ±0.0000	0.7715 ±0.0000
	ACC	0.9190 ±0.0000	0.9209 ±0.0023	0.3463 ±0.0052	**0.9285 ±0.0007**	0.9143 ±0.0000	0.9156 ±0.0033	0.8952 ±0.0015	0.9190 ±0.0000	0.7069 ±0.0649	0.9095 ±0.0000	0.8969 ±0.0412	0.9238 ±0.0000	0.9238 ±0.0000	0.9190 ±0.0000
Heart Disease	NMI	**0.3162 ±0.0000**	0.0142 ±0.0132	0.2431 ±0.0563	0.2500 ±0.0000	0.2326 ±0.0975	0.1548 ±0.0398	0.0364 ±0.0476	0.1608 ±0.0029	0.0868 ±0.0820	0.0000 ±0.0000	0.0000 ±0.0000	0.2756 ±0.0000	0.2078 ±0.0271	0.2571 ± 0.0000
	ARI	**0.3641 ±0.0000**	0.0033 ±0.0042	0.3085 ±0.0700	0.3140 ±0.0000	0.2995 ±0.1230	0.0509 ±0.0172	0.0163 ±0.0523	0.1985 ±0.0036	0.1030 ±0.1050	0.0000 ±0.0000	0.0000 ±0.0000	0.3363 ±0.0000	0.2592 ±0.0384	0.3162 ± 0.0000
	ACC	**0.8020 ±0.0000**	0.5249 ±0.0202	0.7761 ±0.0325	0.7805 ±0.0000	0.7653 ±0.0689	0.6101 ±0.0284	0.5354 ±0.0549	0.7232 ±0.0019	0.6344 ±0.0898	0.5132 ±0.0000	0.5132 ±0.0000	0.7902 ±0.0000	0.7109 ±0.0499	0.7815 ±0.0000
Wine	NMI	0.8759 ±0.0000	0.8759 ±0.0000	0.0000 ±0.0000	0.8759 ±0.0000	0.8097 ±0.0171	0.6966 ±0.1663	0.5250 ±0.0803	0.7847 ±0.0000	0.3976 ±0.0794	0.6907 ±0.0077	0.4846 ±0.1928	0.8759 ±0.0000	0.8759 ±0.0000	**0.8920 ±0.0000**
	ARI	0.8975 ±0.0000	0.8975 ±0.0000	0.0000 ±0.0000	0.8975 ±0.0000	0.8249 ±0.0142	0.6283 ±0.2549	0.3954 ±0.1252	0.7856 ±0.0000	0.3518 ±0.0987	0.6861 ±0.0065	0.3589 ±0.1551	0.8975 ±0.0000	0.8975 ±0.0000	**0.9134 ±0.0000**
	ACC	0.9663 ±0.0000	0.9663 ±0.0000	0.3989 ±0.0000	0.9663 ±0.0000	0.9396 ±0.0051	0.7797 ±0.1834	0.7129 ±0.0900	0.9270 ±0.0000	0.6476 ±0.0742	0.8897 ±0.0027	0.6456 ±0.1051	0.9663 ±0.0000	0.9663 ±0.0000	**0.9719 ±0.0000**
Rice	NMI	0.5685 ±0.0000	0.5688 ±0.0000	0.0188 ±0.0029	0.5682 ±0.0004	0.5742 ±0.0008	0.5544 ±0.0001	0.5437 ±0.0032	**0.5783 ±0.0009**	0.0536 ±0.0268	0.5108 ±0.0000	0.4332 ±0.0025	0.5656 ±0.0000	0.5688 ±0.0000	0.5659 ±0.0005
	ARI	0.6815 ±0.0000	0.6824 ±0.0000	-0.0024 ±0.0006	0.6817 ±0.0004	0.6876 ±0.0008	0.6634 ±0.0001	0.6471 ±0.0109	**0.6914 ±0.0008**	0.0338 ±0.0134	0.5702 ±0.0000	0.4482 ±0.0034	0.6789 ±0.0000	0.6824 ±0.0000	0.6777 ±0.0004
	ACC	0.9129 ±0.0000	0.9131 ±0.0000	0.5672 ±0.0013	0.9129 ±0.0001	0.9147 ±0.0002	0.9074 ±0.0000	0.9023 ±0.0033	**0.9158 ±0.0002**	0.5941 ±0.0155	0.8780 ±0.0000	0.8356 ±0.0012	0.9121 ±0.0000	0.9131 ±0.0000	0.9117 ±0.0001
WBDC	NMI	0.5547 ±0.0000	0.5612 ±0.0000	0.0000 ±0.0000	0.5547 ±0.0000	**0.5700 ±0.0000**	0.3516 ±0.2432	0.4202 ±0.1246	0.2422 ±0.0102	0.1138 ±0.0626	0.0188 ±0.0000	0.0183 ±0.0040	0.5621 ±0.0000	0.5612 ±0.0000	0.5513 ±0.0025
	ARI	0.6707 ±0.0000	0.6829 ±0.0000	0.0000 ±0.0000	0.6707 ±0.0000	**0.6895 ±0.0000**	0.4093 ±0.3070	0.5250 ±0.1734	0.1371 ±0.0074	0.0281 ±0.0409	0.0024 ±0.0000	0.0022 ±0.0013	0.6765 ±0.0000	0.6829 ±0.0000	0.6444 ±0.0028
	ACC	0.9104 ±0.0000	0.9139 ±0.0000	0.6274 ±0.0000	0.9104 ±0.0000	**0.9156 ±0.0000**	0.7989 ±0.1380	0.8557 ±0.0810	0.6898 ±0.0048	0.5910 ±0.0441	0.6292 ±0.0000	0.6290 ±0.0009	0.9121 ±0.0000	0.9139 ±0.0000	0.9027 ±0.0009

E. *Real Datasets with Imbalanced Data*

The considered datasets here are Ecoli, Htru2, Zoo, Glass, Shill Bidding, Anuran Calls, Occupancy Detection, Machine Failure, Pulsar Cleaned, and Bert-Embedded Spam. Among them, Htru2 and Occupancy Detection have a large amount of data, Glass and Ecoli have 6 and 8 classes respectively, and Zoo has high dimensionality. They have varying class sizes with CV values greater than 0.7. Except for Occupancy Detection and Zoo, the other eight datasets have CV values greater than 1. Bert-Embedded Spam has over 700 features. Since the Euclidean distance is ineffective in high-dimensional space for clustering, we execute principal component analysis on the Bert-Embedded Spam data. The first five principal components are used for clustering, as adding more does not improve and even compromises performance. Table IV displays the results, where the best results are in bold, highlighting the dramatic progress made by the proposed EKM over the state-of-the-art fuzzy clustering methods.

As can be seen from Table IV, in general, EKM obtains the best performances on eight datasets, and achieves notable clustering performances on the remaining two datasets (Zoo and Anuran Calls), as confirmed by all three measured clustering indexes. Due to the structural complexity of real datasets, it is reasonable that some algorithms perform well on certain datasets. However, none of the 13 benchmark algorithms can perform consistently well on all datasets. Specifically, although csiFCM outperforms EKM on the Zoo dataset, it performs substantially worse on Htru2, Occupancy Detection, Shill Bidding, etc. HKM performs well on Anuran Calls but underperforms on Shill Bidding, Machine Failure, and Bert-Embedded Spam. In comparison, only EKM preserves consistently good performance, especially on the Htru2, Shill Bidding, Machine Failure, and Bert-Embedded Spam datasets. Specifically, the best benchmark algorithm only has 0.4577, 0.0491, 0.3214, and 0.1785 NMI scores on the four datasets, respectively, while EKM has 0.5872, 0.6216, 0.5825, and 0.6704 NMI scores.

Fig. 5 illustrates the scatter diagrams of Htru2, Shill Bidding, Machine Failure, and Bert-Embedded Spam datasets, along with some clustering results. We have applied the t-distributed stochastic neighbor embedding (t-SNE) [55] technique to reduce the original feature dimension to 2D for easy visualization. Comparing Fig. 5(e) - 5(h) with Fig. 5(i) - 5(l), again, we can see the uniform effect exists in the benchmark algorithms. Conversely, EKM can accommodate small clusters, implying the existence of a repulsive force overcoming the uniform effect. Overall, the experiment results provide strong evidence to support the superiority and consistent performance of EKM on imbalanced data. Note that we simply set α proportional to the data variance. EKM's performance can be further enhanced by fine-tuning the parameter α.

F. *Statistical Analysis*

To statistically validate the superiority of EKM, we performed the paired t-test and Wilcoxon signed-rank test on the NMI, ARI, and ACC results. On the ten real-world imbalanced datasets, EKM significantly outperforms all comparative algorithms with a significant level of $p < 0.05$. Notably, EKM improves on average 0.22, 0.31, and 0.21 over FCM in NMI, ARI, and ACC, respectively. On the 16 real-world mixed balanced/imbalanced datasets (6 balanced + 10 imbalanced datasets), the statistical advantage remains significant, showing EKM's general applicability for both balanced and imbalanced data.

G. *Empirical Convergence and Algorithm Efficiency*

Table V shows the average number of iterations and the average time (unit: second) per run (each algorithm is run 5000 times). The numbers in brackets in the table are the ranking numbers of the corresponding algorithms. AVK and MDK give the average and median number of iterations and computational time of each algorithm on all test datasets, respectively. IMI2 involves a scheme of merging clusters and is not a pure iterative algorithm. Hence, we leave the number of iterations of IMI2 empty.

As evident from Table V, AVK and MDK are nearly consistent. In terms of AVK, HKM is the fastest algorithm as it does not require membership calculation. MEFC comes second, followed by EKM and FCM. EKM is 30% faster than FCM in AVK and comparable to FCM in MDK. This is because FCM is slow on the Bert-Embedded Spam dataset. In terms of absolute speed, MEFC, FCM, and EKM operate within the same order of magnitude. This aligns with their identical time complexity of $O(NK^2P)$ per iteration, where N is the instance number, K is the cluster number, and P is the data dimension or feature number. The table also demonstrates the empirical convergence of EKM, averaging 44 iterations or a median of 20 iterations to reach the specified convergence (20). In summary, EKM has the same computational cost as FCM.

TABLE IV
EXPERIMENTAL RESULTS ON THE SELECTED TEN REAL-WORLD IMBALANCED DATASETS WITH $CV > 0.7$ (THE BEST PERFORMANCE IS IN BOLD)

Dataset	Measurement	HKM	FCM	PCM	MEFC	PFCM	csiFCM	siibFCM	C-PCM	FWPCM	iFCM	EM-iFCM	vFCM	IMI2	EKM
Zoo	NMI	0.8381 ±0.0268	0.7764 ±0.0019	0.1331 ±0.0000	0.8179 ±0.0000	0.6611 ±0.0806	**0.8657 ±0.0054**	0.7625 ±0.0415	0.8197 ±0.0216	0.6652 ±0.0804	0.5751 ±0.0165	0.5548 ±0.0392	0.7643 ±0.0185	0.7876 ±0.0344	0.7912 ±0.0188
	ARI	0.7546 ±0.0487	0.6235 ±0.0010	-0.0578 ±0.0000	0.6444 ±0.0000	0.4868 ±0.0829	**0.8715 ±0.0034**	0.6423 ±0.1030	0.7811 ±0.0798	0.4743 ±0.0948	0.4061 ±0.0244	0.3644 ±0.0565	0.5623 ±0.0337	0.7730 ±0.0614	0.8181 ±0.0783
	ACC	0.8248 ±0.0310	0.6832 ±0.0000	0.3069 ±0.0000	0.7228 ±0.0000	0.6267 ±0.0536	**0.8703 ±0.0078**	0.7305 ±0.0831	0.7952 ±0.0621	0.5958 ±0.0604	0.7093 ±0.0147	0.6942 ±0.0346	0.6079 ±0.0390	0.8139 ±0.0422	0.8507 ±0.0566
Glass	NMI	0.3140 ±0.0046	0.3073 ±0.0003	0.0000 ±0.0000	0.3120 ±0.0012	0.0721 ±0.0069	0.3390 ±0.0549	0.3424 ±0.0413	0.3217 ±0.0392	0.1654 ±0.0823	0.1966 ±0.0293	0.1672 ±0.0294	0.3500 ±0.0457	0.3709 ±0.0457	**0.3764 ±0.0370**
	ARI	0.1702 ±0.0026	0.1529 ±0.0005	0.0000 ±0.0000	0.1651 ±0.0009	0.0070 ±0.0059	0.1930 ±0.0435	0.1648 ±0.0366	0.1755 ±0.0258	0.0734 ±0.0659	0.0477 ±0.0173	0.0335 ±0.0163	0.1706 ±0.0457	0.1931 ±0.0392	**0.1978 ±0.0193**
	ACC	0.4586 ±0.0076	0.4021 ±0.0011	0.3551 ±0.0000	0.4487 ±0.0007	0.3368 ±0.0054	0.4684 ±0.0476	0.4202 ±0.0270	0.4452 ±0.0354	0.4096 ±0.0584	0.4060 ±0.0078	0.3988 ±0.0090	0.4112 ±0.0457	0.4777 ±0.0358	**0.4796 ±0.0086**
Ecoli	NMI	0.6379 ±0.0040	0.5695 ±0.0115	0.0656 ±0.0805	0.6096 ±0.0076	0.5012 ±0.0649	0.5985 ±0.0241	0.5541 ±0.0464	0.5861 ±0.0243	0.4746 ±0.0424	0.4539 ±0.0180	0.4073 ±0.0304	0.6289 ±0.0055	0.3985 ±0.1704	**0.6426 ±0.0024**
	ARI	0.5032 ±0.0070	0.4142 ±0.0204	0.0119 ±0.0503	0.4815 ±0.0270	0.3357 ±0.0752	0.4807 ±0.0898	0.3697 ±0.0862	0.4002 ±0.0495	0.2832 ±0.0500	0.3679 ±0.0271	0.3187 ±0.0369	0.4949 ±0.0176	0.2779 ±0.2343	**0.5157 ±0.0013**
	ACC	0.6461 ±0.0103	0.5769 ±0.0160	0.4289 ±0.0351	0.6238 ±0.0218	0.5302 ±0.0435	0.6276 ±0.0711	0.5624 ±0.0638	0.5564 ±0.0458	0.4598 ±0.0410	0.6426 ±0.0178	0.6105 ±0.0186	0.6370 ±0.0141	0.5668 ±0.1214	**0.6482 ±0.0043**
Htru2	NMI	0.4068 ±0.0000	0.4100 ±0.0004	0.0000 ±0.0000	0.4075 ±0.0002	0.1289 ±0.0002	0.0204 ±0.0506	0.3671 ±0.0290	0.0415 ±0.0025	0.0660 ±0.0268	0.0229 ±0.0032	0.0179 ±0.0416	0.4577 ±0.0000	0.3220 ±0.0003	**0.5872 ±0.0003**
	ARI	0.6071 ±0.0000	0.5796 ±0.0005	0.0000 ±0.0000	0.6075 ±0.0002	0.0462 ±0.0002	0.0077 ±0.0649	0.5704 ±0.0379	0.0178 ±0.0020	-0.0117 ±0.0074	0.0010 ±0.0134	0.0093 ±0.0392	0.6616 ±0.0000	0.1731 ±0.0001	**0.7333 ±0.0002**
	ACC	0.9366 ±0.0000	0.9252 ±0.0001	0.9084 ±0.0000	0.9366 ±0.0000	0.6091 ±0.0002	0.8888 ±0.0637	0.9337 ±0.0126	0.5679 ±0.0040	0.5170 ±0.0151	0.9057 ±0.0102	0.9069 ±0.0164	0.9489 ±0.0000	0.5223 ±0.0003	**0.9661 ±0.0000**
Shill Bidding	NMI	0.0021 ±0.0001	0.0023 ±0.0000	0.0000 ±0.0000	0.0042 ±0.0002	0.0012 ±0.0000	0.0295 ±0.0359	0.0351 ±0.0047	0.0002 ±0.0000	0.0105 ±0.0330	0.0491 ±0.0002	0.0220 ±0.0093	0.0032 ±0.0000	0.0023 ±0.0000	**0.6216 ±0.0000**
	ARI	-0.0006 ±0.0000	-0.0002 ±0.0000	0.0000 ±0.0000	-0.0002 ±0.0000	-0.0002 ±0.0000	-0.0485 ±0.0482	-0.0854 ±0.0118	-0.0011 ±0.0000	0.0079 ±0.0252	-0.0704 ±0.0002	-0.0529 ±0.0252	0.0108 ±0.0000	-0.0002 ±0.0000	**0.7914 ±0.0000**
	ACC	0.5020 ±0.0002	0.5059 ±0.0000	0.8932 ±0.0000	0.5090 ±0.0003	0.5035 ±0.0002	0.7457 ±0.0961	0.7869 ±0.0182	0.5240 ±0.0001	0.5937 ±0.0750	0.6296 ±0.0004	0.8454 ±0.0261	0.5871 ±0.0000	0.5058 ±0.0001	**0.9674 ±0.0000**
Anuran Calls	NMI	**0.6775 ±0.0195**	0.5699 ±0.0002	0.0101 ±0.0018	0.6155 ±0.0022	0.4270 ±0.0153	0.5575 ±0.0516	0.5178 ±0.0339	0.5254 ±0.0112	0.5159 ±0.0247	0.2773 ±0.0066	0.2758 ±0.0177	0.5149 ±0.0360	0.6610 ±0.0081	0.6229 ±0.0134
	ARI	0.5826 ±0.0174	0.3414 ±0.0009	-0.0029 ±0.0009	0.4017 ±0.0027	0.2254 ±0.0694	0.4329 ±0.1363	0.2368 ±0.0694	0.3696 ±0.0527	0.4268 ±0.0500	0.0842 ±0.0056	0.0909 ±0.0228	0.2997 ±0.0623	**0.7490 ±0.0001**	0.5145 ±0.0329
	ACC	0.6441 ±0.0134	0.4363 ±0.0009	0.4808 ±0.0008	0.5103 ±0.0073	0.3705 ±0.0230	0.5682 ±0.0799	0.4101 ±0.0348	0.4268 ±0.0436	0.5123 ±0.0271	0.5236 ±0.0041	0.5300 ±0.0164	0.4374 ±0.0540	**0.7662 ±0.0216**	0.5821 ±0.0420
Occupancy Detection	NMI	0.4890 ±0.0001	0.4720 ±0.0003	0.0076 ±0.0002	0.4710 ±0.0002	0.0539 ±0.0007	0.0934 ±0.1560	0.2982 ±0.1379	0.2469 ±0.2590	0.0591 ±0.0464	0.2553 ±0.0003	0.1114 ±0.0160	0.4926 ±0.0000	0.3585 ±0.0001	**0.5389 ±0.0002**
	ARI	0.5959 ±0.0001	0.5699 ±0.0004	-0.0043 ±0.0002	0.5670 ±0.0002	0.0209 ±0.0002	0.0521 ±0.2248	0.2175 ±0.2226	0.2780 ±0.3113	0.0363 ±0.0682	0.2922 ±0.0005	0.0681 ±0.0476	0.6008 ±0.0000	0.2659 ±0.0001	**0.6750 ±0.0002**
	ACC	0.8912 ±0.0000	0.8825 ±0.0001	0.7659 ±0.0001	0.8815 ±0.0001	0.5772 ±0.0004	0.7597 ±0.0596	0.7026 ±0.1361	0.7071 ±0.1697	0.6117 ±0.0595	0.8330 ±0.0001	0.7528 ±0.0901	0.8928 ±0.0000	0.5132 ±0.0001	**0.9163 ±0.0001**
Machine Failure	NMI	0.0248 ±0.0000	0.0281 ±0.0000	0.0000 ±0.0000	0.0310 ±0.0000	0.0284 ±0.0003	0.2285 ±0.0569	0.0540 ±0.0292	0.0705 ±0.1150	0.0137 ±0.0161	0.3214 ±0.0034	0.0128 ±0.0421	0.0253 ±0.0000	0.2005 ±0.0636	**0.5825 ±0.1869**
	ARI	-0.0116 ±0.0000	-0.0083 ±0.0000	0.0000 ±0.0000	-0.0052 ±0.0000	0.0007 ±0.0001	0.2217 ±0.0510	-0.0037 ±0.0382	0.0418 ±0.1386	0.0031 ±0.0083	0.3367 ±0.0049	-0.0050 ±0.0464	-0.0111 ±0.0000	0.1396 ±0.1158	**0.6506 ±0.1987**
	ACC	0.5673 ±0.0000	0.5418 ±0.0002	0.9709 ±0.0000	0.5202 ±0.0001	0.5146 ±0.0007	0.9653 ±0.0543	0.6125 ±0.0412	0.6145 ±0.1481	0.5481 ±0.0358	0.9771 ±0.0001	0.9505 ±0.0442	0.5635 ±0.0000	0.7642 ±0.2133	**0.9863 ±0.0068**
Pulsar Cleaned	NMI	0.0311 ±0.0002	0.0167 ±0.0000	0.0000 ±0.0000	0.0201 ±0.0000	0.0236 ±0.0011	0.0523 ±0.0813	0.0044 ±0.0081	0.0139 ±0.0002	0.0090 ±0.0067	0.0019 ±0.0015	0.0050 ±0.0202	0.0301 ±0.0000	0.0159 ±0.0000	**0.0544 ±0.0001**
	ARI	0.0370 ±0.0003	0.0069 ±0.0000	0.0000 ±0.0000	0.0131 ±0.0001	-0.0011 ±0.0002	0.0731 ±0.1193	-0.0067 ±0.0160	-0.0010 ±0.0002	-0.0039 ±0.0080	-0.0078 ±0.0110	0.0029 ±0.0184	0.0316 ±0.0000	0.0446 ±0.0000	**0.1082 ±0.0002**
	ACC	0.7329 ±0.0008	0.5714 ±0.0001	**0.9794 ±0.0000**	0.6171 ±0.0004	0.5019 ±0.0012	0.9488 ±0.0973	0.9043 ±0.1231	0.5045 ±0.0019	0.6036 ±0.0659	0.9232 ±0.0557	0.9777 ±0.0164	0.7091 ±0.0000	0.8367 ±0.0000	0.8834 ±0.0002
Bert-Embedded Spam	NMI	0.1517 ±0.0001	0.1249 ±0.0000	0.0000 ±0.0000	0.0472 ±0.0374	0.1122 ±0.0023	0.0330 ±0.0791	0.0431 ±0.0946	0.1785 ±0.0049	0.0238 ±0.0004	0.0481 ±0.0147	0.0082 ±0.0152	0.0000 ±0.0000	0.1174 ±0.0000	**0.6704 ±0.0000**
	ARI	0.0830 ±0.0003	0.0550 ±0.0000	0.0000 ±0.0000	0.0289 ±0.0265	-0.0026 ±0.0007	0.0080 ±0.1084	0.0296 ±0.1287	0.1019 ±0.0080	-0.0669 ±0.0004	0.0192 ±0.0060	0.0014 ±0.0119	0.0000 ±0.0000	0.0262 ±0.0000	**0.8216 ±0.0000**
	ACC	0.6453 ±0.0002	0.6174 ±0.0000	0.8659 ±0.0000	0.5837 ±0.0390	0.5482 ±0.0012	0.8210 ±0.0816	0.8281 ±0.0832	0.6616 ±0.0068	0.8038 ±0.0002	0.8677 ±0.0006	0.8657 ±0.0029	0.8659 ±0.0000	0.5845 ±0.0000	**0.9664 ±0.0000**

TABLE V
AVERAGE NUMBER OF ITERATIONS AND AVERAGE CALCULATION TIME (UNIT: SECOND) OF EACH ALGORITHM

Dataset	Measurement	HKM	FCM	PCM	MEFC	PFCM	csiFCM	siibCKM	C-PCM	FWPCM	iFCM	EM-iFCM	vFCM	IMI2	EKM
Data-A	Iter	17.3	13.7	17.9	29.0	33.5	30.7	28.0	16.8	44.6	34.2	37.0	56.0	-	12.1
	Time	0.005(1)	0.008(4)	0.017(9)	0.013(7)	0.033(11)	0.021(10)	0.277(14)	0.006(2)	0.016(8)	0.009(5)	0.012(6)	0.138(13)	0.056(12)	0.007(3)
Data-B	Iter	9.2	13.2	29.9	30.1	28.8	15.1	17.4	17.1	24.1	28.0	26.3	34.0	-	17.5
	Time	0.003(1)	0.008(4)	0.019(10)	0.013(9)	0.031(11)	0.011(8)	0.168(14)	0.006(2)	0.010(5)	0.007(3)	0.008(6)	0.084(13)	0.049(12)	0.009(7)
Data-C	Iter	16.3	17.6	17.5	36.0	23.9	35.0	42.1	15.7	73.7	47.1	74.9	94.0	-	23.5
	Time	0.008(2)	0.014(3)	0.028(7)	0.023(6)	0.048(11)	0.040(8)	0.624(14)	0.008(1)	0.035(10)	0.015(4)	0.034(9)	0.354(13)	0.105(12)	0.017(5)
Data-D	Iter	14.8	23.6	37.8	25.4	39.8	24.5	25.2	17.6	7.3	29.1	29.1	83.4	-	15.4
	Time	0.033(2)	0.091(6)	0.229(10)	0.082(5)	0.364(11)	0.142(9)	1.867(14)	0.039(3)	0.030(1)	0.056(4)	0.102(7)	1.515(13)	0.873(12)	0.107(8)
IS	Iter	15.9	40.1	20.4	20.9	53.0	49.8	32.1	24.0	29.6	87.6	70.3	90.0	-	24.1
	Time	0.021(1)	0.110(6)	0.161(8)	0.058(2)	0.218(10)	0.104(5)	0.824(12)	0.068(3)	0.152(7)	0.223(11)	0.195(9)	3.296(14)	1.555(13)	0.074(4)
Seeds	Iter	8.5	10.4	43.3	11.1	13.3	12.6	26.0	11.9	32.5	17.3	18.2	44.0	-	12.0
	Time	0.001(1)	0.001(6)	0.004(10)	0.001(4)	0.003(9)	0.002(7)	0.028(14)	0.001(2)	0.005(11)	0.001(3)	0.001(5)	0.027(13)	0.011(2)	0.002(8)
Heart Disease	Iter	9.9	269.3	9.0	12.5	22.7	500.0	498.8	13.4	3.3	262.3	500	134.0	-	18.7
	Time	0.002(1)	0.074(7)	0.078(8)	0.003(3)	0.081(9)	2.095(11)	1.646(14)	0.004(4)	0.002(2)	0.055(6)	0.120(10)	0.459(13)	0.258(12)	0.005(5)
Wine	Iter	7.0	13.5	21.9	10.6	25.3	15.7	40.4	25.9	19.4	34.1	97.5	60.0	-	11.7
	Time	0.000(1)	0.002(4)	0.003(8)	0.001(2)	0.004(9)	0.002(5)	0.037(13)	0.003(7)	0.005(10)	0.003(6)	0.008(11)	0.054(14)	0.020(12)	0.001(3)
Rice	Iter	8.4	8.7	23.2	8.9	12.7	14.0	17.8	13.2	72.8	20.8	26.7	52.0	-	9.1
	Time	0.004(1)	0.009(5)	0.022(10)	0.006(2)	0.021(9)	0.011(6)	0.214(13)	0.009(4)	0.072(12)	0.011(7)	0.016(8)	0.361(14)	0.068(11)	0.009(3)
WDBC	Iter	7.4	10.8	9.4	10.6	32.4	500.0	500.0	16.1	16.2	33.1	95.0	62.0	-	9.9
	Time	0.001(1)	0.003(4)	0.005(6)	0.002(2)	0.009(8)	2.307(12)	0.933(14)	0.005(5)	0.013(9)	0.008(7)	0.028(10)	0.250(13)	0.034(11)	0.002(3)
Zoo	Iter	4.4	31.0	19.7	23.8	35.0	34.4	37.8	28.7	2.3	33.4	30.0	66.0	-	498.9
	Time	0.001(1)	0.005(7)	0.008(9)	0.003(3)	0.009(10)	0.006(8)	0.048(11)	0.005(6)	0.001(2)	0.004(5)	0.004(4)	0.096(14)	0.058(12)	0.069(13)
Glass	Iter	8.5	27.2	8.9	29.4	18.8	31.4	45.0	13.7	25.7	23.6	24.3	122.0	-	25.3
	Time	0.001(1)	0.006(8)	0.008(10)	0.006(5)	0.009(11)	0.006(7)	0.096(13)	0.002(2)	0.007(9)	0.003(3)	0.004(4)	0.190(14)	0.059(12)	0.006(6)
Ecoli	Iter	14.1	42.9	18.2	23.6	32.2	40.5	45.2	16.8	3.0	55.8	56.9	79.1	-	22.9
	Time	0.002(2)	0.019(9)	0.023(10)	0.010(4)	0.029(11)	0.015(8)	0.197(13)	0.005(3)	0.002(1)	0.012(6)	0.014(7)	0.209(17)	0.138(15)	0.011(5)
Htru2	Iter	12.7	33.8	9.2	15.2	21.5	500.0	500.0	53.4	52.5	27.3	486.8	38.0	-	10.8
	Time	0.046(1)	0.175(4)	0.244(5)	0.067(3)	0.357(7)	17.884(12)	27.836(14)	0.450(9)	0.450(8)	0.272(6)	5.275(13)	1.411(10)	1.569(11)	0.059(2)
Shill Bidding	Iter	12.0	15.0	14.7	24.3	22.7	36.8	40.4	13.8	25.6	54.1	38.0	172.0	-	22.3
	Time	0.011(1)	0.026(3)	0.046(6)	0.029(4)	0.064(10)	0.046(7)	0.794(13)	0.023(2)	0.056(8)	0.073(11)	0.060(9)	2.574(14)	0.287(12)	0.033(5)
Anuran Calls	Iter	22.5	35.7	13.1	24.1	39.3	42.6	45.4	32.5	24.0	65.7	79.3	247.2	-	44.3
	Time	0.339(1)	0.527(3)	1.117(6)	0.377(2)	1.502(7)	0.614(4)	5.119(12)	1.710(8)	2.089(9)	3.473(10)	4.381(11)	31.74(13)	31.874(14)	0.719(5)
Occupancy Detection	Iter	11.5	25.6	18.7	28.6	24.6	28.0	194.9	14.3	19.8	23.5	28.3	74.0	-	14.8
	Time	0.032(1)	0.132(6)	0.213(9)	0.097(4)	0.291(11)	0.109(5)	12.090(14)	0.091(3)	0.163(8)	0.144(7)	0.214(10)	2.060(13)	0.603(12)	0.067(2)
Machine Failure	Iter	13.4	22.0	11.7	11.9	127.4	422.1	34.5	29.8	122.6	100.8	272.5	72.0	-	22.5
	Time	0.015(1)	0.055(4)	0.077(6)	0.022(2)	0.442(10)	0.724(12)	1.058(13)	0.076(5)	0.331(9)	0.230(7)	0.778(11)	1.338(14)	0.328(8)	0.049(3)
Pulsar Cleaned	Iter	25.3	17.1	10.4	18.4	60.7	500.0	500.0	24.4	50.2	57.4	227.8	78.0	-	21.4
	Time	0.045(1)	0.068(3)	0.114(5)	0.052(2)	0.435(9)	12.560(12)	22.953(14)	0.132(6)	0.268(7)	0.325(8)	1.407(11)	2.231(13)	0.827(10)	0.075(4)
Bert-Embedded Spam	Iter	31.6	500.0	8.6	498.7	21.2	500.0	500.0	35.9	28.5	500.0	53.4	500.0	-	41.6
	Time	0.015(1)	0.618(10)	0.630(11)	0.388(5)	0.642(12)	4.866(9)	8.520(14)	0.028(2)	0.030(3)	0.322(6)	0.039(4)	2.008(13)	0.441(8)	0.045(5)
AVK	Iter	13.5	58.5	18.2	44.7	34.4	166.4	158.3	21.7	33.9	76.8	113.6	107.9	-	44.0
	Time	0.029(1)	0.098(4)	0.152(6)	0.063(2)	0.230(8)	2.078(10)	4.266(14)	0.138(5)	0.187(7)	0.262(9)	0.635(11)	2.520(13)	1.961(12)	0.066(3)
MDK	Iter	12.4	22.8	17.7	23.7	27.1	35.9	41.3	16.9	25.6	34.2	55.1	76.0	-	20.0
	Time	0.007(1)	0.023(4)	0.037(9)	0.018(3)	0.056(10)	0.075(11)	0.809(14)	0.008(2)	0.030(6)	0.035(7)	0.036(8)	0.410(13)	0.198(12)	0.025(5)

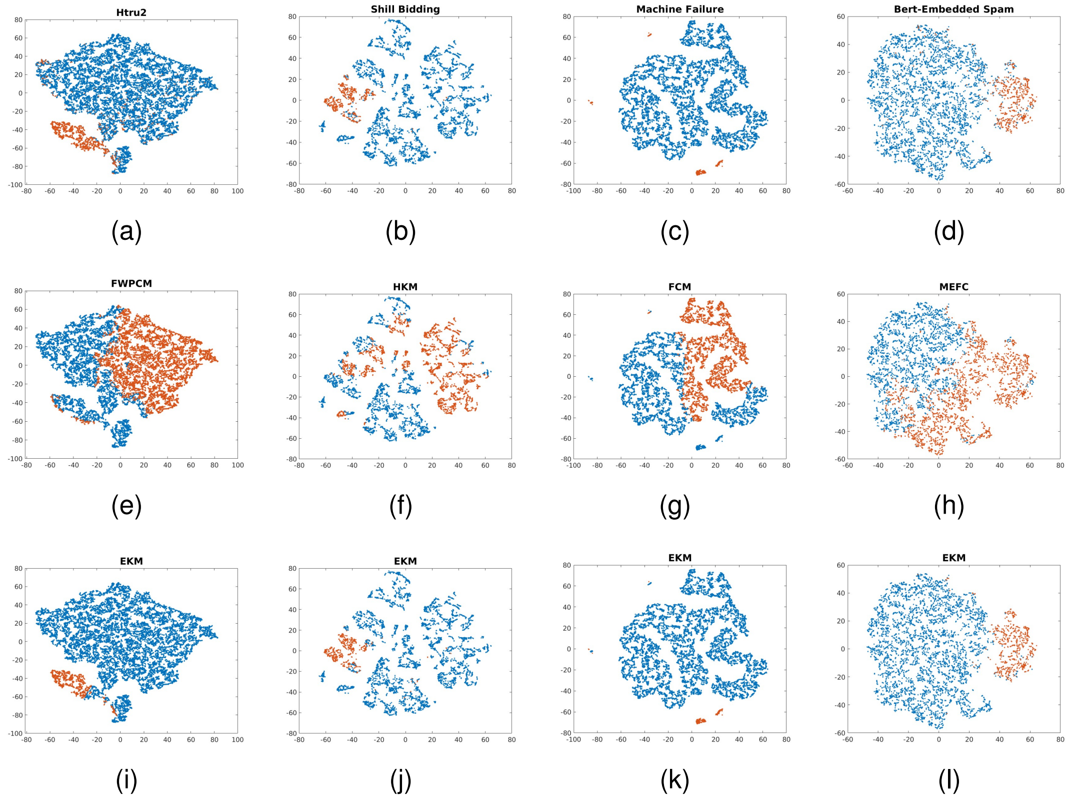

Fig. 5. Scatter diagrams of some selected real imbalanced datasets (from left to right: Htru2, Shill Bidding, Machine Failure, and Bert-Embedded Spam). The t-SNE technique [55] has been applied to reduce the original feature dimensions to 2D for easy visualization. Different colors represent labels. (a)-(d) Primitive scatter diagrams with reference class labels. (e)-(f) Clustering results of some benchmark algorithms (from left to right: FWPCM [19], HKM [6], FCM [5], and MEFC [10]). (i)-(l) Clustering results of the proposed EKM.

V. Conclusion

This paper presents equilibrium K-means (EKM), a novel fuzzy clustering algorithm robust to imbalanced data. EKM is simple, interpretable, and scalable to large datasets. Theoretical analysis regarding the effectiveness and convergence of EKM is given, and compatible with the empirical study. Experimental results on datasets from various domains show that EKM greatly outperforms hard K-means, fuzzy C-means, and other state-of-the-art fuzzy clustering algorithms on imbalanced datasets and performs comparably on balanced datasets. However, the current EKM uses Euclidean distance as a similarity measure, and its performance may degrade on high-dimensional or non-spherical clusters. Future research will focus on integrating EKM with deep representation learning to address high-dimensional challenges, as well as developing kernelized variants to handle arbitrary cluster shapes. These extensions aim to preserve EKM's ability to handle imbalanced data clustering while expanding its applicability to broader data scenarios.

References

[1] B. Krawczyk, "Learning from imbalanced data: open challenges and future directions," *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 221–232, 2016.

[2] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[3] D. Ramyachitra and P. Manikandan, "Imbalanced dataset classification and solutions: a review," *International Journal of Computing and Business Research (IJCBR)*, vol. 5, no. 4, pp. 1–29, 2014.

[4] J. Tanha, Y. Abdi, N. Samadi, N. Razzaghi, and M. Asadpour, "Boosting methods for multi-class imbalanced data classification: an experimental review," *Journal of Big Data*, vol. 7, pp. 1–47, 2020.

[5] J. C. Bezdek, *Pattern recognition with fuzzy objective function algorithms*. Springer Science & Business Media, 2013.

[6] S. Lloyd, "Least squares quantization in pcm," *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982.

[7] R. Krishnapuram and J. M. Keller, "A possibilistic approach to clustering," *IEEE transactions on fuzzy systems*, vol. 1, no. 2, pp. 98–110, 1993.

[8] 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.

[9] H. Yu and J. Fan, "Cutset-type possibilistic c-means clustering algorithm," *Applied Soft Computing*, vol. 64, pp. 401–422, 2018.

[10] N. B. Karayiannis, "Meca: Maximum entropy clustering algorithm," in *Proceedings of 1994 IEEE 3rd international fuzzy systems conference*. IEEE, 1994, pp. 630–635.

[11] R.-P. Li and M. Mukaidono, "A maximum-entropy approach to fuzzy clustering," in *Proceedings of 1995 IEEE International Conference on Fuzzy Systems.*, vol. 4. IEEE, 1995, pp. 2227–2232.

[12] D.-M. Tsai and C.-C. Lin, "Fuzzy c-means based clustering for linearly and nonlinearly separable data," *Pattern recognition*, vol. 44, no. 8, pp. 1750–1760, 2011.

[13] M. Javadian, R. Vaziri, S. Haghzad Klidbary, and A. Malekzadeh, "Refining membership degrees obtained from fuzzy c-means by re-fuzzification," *Iranian Journal of Fuzzy Systems*, vol. 17, no. 4, pp. 85–104, 2020.

[14] M. Javadian, A. Hejazi, and S. H. Klidbary, "Obtaining fuzzy membership function of clusters with the memristor hardware implementation and on-chip learning," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 6, no. 4, pp. 1008–1025, 2022.

[15] S. Krinidis and V. Chatzis, "A robust fuzzy local information c-means clustering algorithm," *IEEE transactions on image processing*, vol. 19, no. 5, pp. 1328–1337, 2010.

[16] Y. Chen, S. Zhou, X. Zhang, D. Li, and C. Fu, "Improved fuzzy c-means clustering by varying the fuzziness parameter," *Pattern Recognition Letters*, vol. 157, pp. 60–66, 2022.

[17] H. Lohit and D. Kumar, "Modified total bregman divergence driven picture fuzzy clustering with local information for brain mri image segmentation," *Applied Soft Computing*, vol. 144, p. 110460, 2023.

[18] A. Bose, U. Maulik, and A. Sarkar, "An entropy-based membership approach on type-ii fuzzy set (emt2fcm) for biomedical image segmentation," *Engineering Applications of Artificial Intelligence*, vol. 127, p. 107267, 2024.

[19] M.-S. Yang and J. B. Benjamin, "Feature-weighted possibilistic c-means clustering with a feature-reduction framework," *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 5, pp. 1093–1106, 2020.

[20] M.-S. Yang and K. P. Sinaga, "Collaborative feature-weighted multi-view fuzzy c-means clustering," *Pattern Recognition*, vol. 119, p. 108064, 2021.

[21] D.-Q. Zhang and S.-C. Chen, "Clustering incomplete data using kernel-based fuzzy c-means algorithm," *Neural processing letters*, vol. 18, pp. 155–162, 2003.

[22] H.-C. Huang, Y.-Y. Chuang, and C.-S. Chen, "Multiple kernel fuzzy clustering," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 1, pp. 120–134, 2011.

[23] Y. Tang, Z. Pan, X. Hu, W. Pedrycz, and R. Chen, "Knowledge-induced multiple kernel fuzzy clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[24] A. Coates and A. Y. Ng, "Learning feature representations with k-means," in *Neural Networks: Tricks of the Trade: Second Edition*. Springer, 2012, pp. 561–580.

[25] B. Yang, X. Fu, N. D. Sidiropoulos, and M. Hong, "Towards k-means-friendly spaces: Simultaneous deep learning and clustering," in *international conference on machine learning*. PMLR, 2017, pp. 3861–3870.

[26] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 132–149.

[27] M. M. Fard, T. Thonet, and E. Gaussier, "Deep k-means: Jointly clustering with k-means and learning representations," *Pattern Recognition Letters*, vol. 138, pp. 185–192, 2020.

[28] S. Haghzad Klidbary and M. Javadian, "A novel hierarchical high-dimensional unsupervised active learning method," *International Journal of Computational Intelligence Systems*, vol. 17, no. 1, p. 193, 2024.

[29] H. Xiong, J. Wu, and J. Chen, "K-means clustering versus validation measures: a data distribution perspective," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006, pp. 779–784.

[30] K. Zhou and S. Yang, "Exploring the uniform effect of fcm clustering: A data distribution perspective," *Knowledge-Based Systems*, vol. 96, pp. 76–83, 2016.

[31] J. Noordam, W. Van Den Broek, and L. Buydens, "Multivariate image segmentation with cluster size insensitive fuzzy c-means," *Chemometrics and intelligent laboratory systems*, vol. 64, no. 1, pp. 65–78, 2002.

[32] P.-L. Lin, P.-W. Huang, C.-H. Kuo, and Y. Lai, "A size-insensitive integrity-based fuzzy c-means method for data clustering," *Pattern Recognition*, vol. 47, no. 5, pp. 2042–2056, 2014.

[33] Y. Liu, T. Hou, and F. Liu, "Improving fuzzy c-means method for unbalanced dataset," *Electronics Letters*, vol. 51, no. 23, pp. 1880–1882, 2015.

[34] Y. Pu, W. Yao, and X. Li, "Em-ifcm: Fuzzy c-means clustering algorithm based on edge modification for imbalanced data," *Information Sciences*, vol. 659, p. 120029, 2024.

[35] A. Priya, R. Agrawal, and B. Rana, "Noise and cluster size insensitive robust weighted fuzzy clustering for medical image segmentation," *Soft Computing*, vol. 28, no. 23, pp. 13 615–13 638, 2024.

[36] 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.

[37] F. Liu, J. Wang, and Y. Liu, "Imi2: A fuzzy clustering validity index for multiple imbalanced clusters," *Expert Systems with Applications*, vol. 238, p. 122231, 2024.

[38] D. Li, S. Zhou, T. Zeng, and R. H. Chan, "Multi-prototypes convex merging based k-means clustering algorithm," *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 11, pp. 6653–6666, 2023.

[39] J. Liang, L. Bai, C. Dang, and F. Cao, "The k-means-type algorithms versus imbalanced data distributions," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 4, pp. 728–745, 2012.

[40] Y. Lu, Y.-M. Cheung, and Y. Y. Tang, "Self-adaptive multiprototype-based competitive learning approach: A k-means-type algorithm for imbalanced data clustering," *IEEE transactions on cybernetics*, vol. 51, no. 3, pp. 1598–1612, 2019.

[41] S. Zeng, X. Duan, J. Bai, W. Tao, K. Hu, and Y. Tang, "Soft multi-prototype clustering algorithm via two-layer semi-nmf," *IEEE Transactions on Fuzzy Systems*, 2023.

[42] L. Bottou and Y. Bengio, "Convergence properties of the k-means algorithms," *Advances in neural information processing systems*, vol. 7, 1994.

[43] A. Gupta, S. Datta, and S. Das, "Fuzzy clustering to identify clusters at different levels of fuzziness: An evolutionary multiobjective optimization approach," *IEEE transactions on cybernetics*, vol. 51, no. 5, pp. 2601–2611, 2019.

[44] A. Asuncion and D. Newman, "Uci machine learning repository," 2007.

[45] J. Wu, *Advances in K-means clustering: a data mining thinking*. Springer Science & Business Media, 2012.

[46] X. L. Xie and G. Beni, "A validity measure for fuzzy clustering," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 13, no. 08, pp. 841–847, 1991.

[47] Y. Liu, Y. Jiang, T. Hou, and F. Liu, "A new robust fuzzy clustering validity index for imbalanced data sets," *Information Sciences*, vol. 547, pp. 579–591, 2021.

[48] C.-H. Wu, C.-S. Ouyang, L.-W. Chen, and L.-W. Lu, "A new fuzzy clustering validity index with a median factor for centroid-based clustering," *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 3, pp. 701–718, 2014.

[49] D. Arthur and S. Vassilvitskii, "K-means++ the advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 2007, pp. 1027–1035.

[50] A. Strehl and J. Ghosh, "Cluster ensembles—a knowledge reuse framework for combining multiple partitions," *Journal of machine learning research*, vol. 3, no. Dec, pp. 583–617, 2002.

[51] K. Y. Yeung and W. L. Ruzzo, "Details of the adjusted rand index and clustering algorithms, supplement to the paper an empirical study on principal component analysis for clustering gene expression data," *Bioinformatics*, vol. 17, no. 9, pp. 763–774, 2001.

[52] D. Graves and W. Pedrycz, "Kernel-based fuzzy clustering and fuzzy clustering: A comparative experimental study," *Fuzzy sets and systems*, vol. 161, no. 4, pp. 522–543, 2010.

[53] M. Huang, Z. Xia, H. Wang, Q. Zeng, and Q. Wang, "The range of the value for the fuzzifier of the fuzzy c-means algorithm," *Pattern Recognition Letters*, vol. 33, no. 16, pp. 2280–2284, 2012.

[54] F. Salehi, M. R. Keyvanpour, and A. Sharifi, "Smkfc-er: Semi-supervised multiple kernel fuzzy clustering based on entropy and relative entropy," *Information Sciences*, vol. 547, pp. 667–688, 2021.

[55] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

Yudong He received a B.S. degree from the University of Science and Technology of China in 2017 and the Ph.D. degree from the Hong Kong University of Science and Technology in 2022. From 2022 to 2025, he was a Postdoctoral Researcher with the Hong Kong University of Science and Technology.

His research interests include machine learning and signal processing.