Knowledge and Information Systems
https://doi.org/10.1007/s10115-024-02324-y

REGULAR PAPER

An outlier detection algorithm based on local density feedback

Zhongping Zhang[1,2] · Yuehan Hou[1] · Yin Jia[1] · Ruibo Zhang[3]

Received: 23 April 2024 / Revised: 4 November 2024 / Accepted: 17 December 2024

Abstract
Outlier detection is very important in the field of data mining and is applied to various scenarios, such as financial fraud detection and network intrusion. Traditional outlier detection methods usually detect outliers based on the local density of objects, and have achieved some results. However, some challenges still exist: (1) traditional methods only consider neighbor information when calculating the density of an object and ignore the global information embedded in the dataset, leading to the wrong detection of outliers and normal points that are only detected in the global view; (2) traditional methods focus only on comparing the density of an object to its neighbors, ignoring the similarity to its neighbors, leading to incorrectly detecting normal points in sparse regions as outliers, even if all the neighbors of a given point are normal ones. To address these issues, we propose a novel outlier detection algorithm based on the local density feedback (LDF). Our method utilizes principal component analysis (PCA) and a natural neighbor search for initial density estimation. A feedback mechanism is designed to refine the density iteratively by leveraging neighborhood similarity, and to aggregate global information for a more accurate outlierness depiction. By integrating local and global data characteristics, our method reliably detects outliers across diverse datasets. Experimental results on ten datasets show that the LDF algorithm outperforms the existing methods by 10.7% and 1.89% on average in terms of precision and AUC, respectively.

Keywords Outlier detection · Neighborhood similarity · Density feedback mechanism · Feedback outlier factor

✉ Yuehan Hou
houyh567@163.com

Zhongping Zhang
zpzhang@ysu.edu.cn

Yin Jia
yjia_ysu@163.com

Ruibo Zhang
zhangruibo@whut.edu.cn

1 School of Information Science and Engineering, Yanshan University, Qinhuangdao 066004, China

2 The Key Laboratory for Computer Virtual Technology and System Integration of Hebei Province, Yanshan University, Qinhuangdao 066004, China

3 School of International Education, Wuhan University of Technology, Wuhan 430070, China

Published online: 13 January 2025

1 Introduction

Outlier detection stands as a pivotal research topic in data mining. In various application scenarios, it is also referred to as anomaly detection. Outliers represent entities that stray from the bulk of the data and may stem from unusual mechanisms [1]. Thus, the task of outlier detection aids in unraveling these unusual data generation mechanisms and prompts relevant staffs to address abnormal data. It has applied to various scenarios, including network intrusion detection [2], financial fraud detection [3], natural disaster monitoring [4], and industrial fault detection [5].

Compared with supervised methods and semi-supervised methods, unsupervised methods [6] stand out as the most prevalent and demanding approaches in outlier detection field. From a technical point of view, outlier detection algorithms are mainly categorized into statistical-based methods, clustering-based methods and density-based methods. Statistics-based methods [7–10] typically assume a specific distribution model for the data, and treat points in low-probability areas as outliers, such as the Gaussian mixture model [7] and ECOD [8]. Clustering-based methods typically employ clustering techniques to partition dense points into clusters, identifying isolated points as outliers [11], such as DBSCAN [12], k-means [13] and FDPC [14] proposed in recent years.

Density-based outlier detection methods are the most widely used. They rely on density estimation to gauge the sparsity or deviation of an object. A point is typically identified as an outlier if its density is significantly lower than that of its neighbors [15]. The LOF [16] algorithm was first proposed for solving the problem of detecting local outliers near dense clusters. It measures sparsity by local reachable density and defines local outlier factor as the density of a data point relative to its k-nearest neighbors (KNN) [17]. However, LOF is ineffective when there are clusters of different densities. The COF [18] is a improvement of LOF, and solves the above problem by using the connectivity distance. But it fails when outliers are located in areas where neighborhood density varies substantially. The RDOF [19] uses the reciprocal of the average distance from an object to its KNN as a measure of density, and determines the outlier factor as the density of the kth neighbor relative to the object. In addition, KDF-IF [20] and adaptive-KD [21] utilize the kernel density estimation (KDE) [22] method to calculate the density of objects.

Although existing methods [16, 18–21] have achieved good results, several problems remain:

- Traditional methods only consider neighborhood information when calculating an object's density and ignore the global information within the dataset. Global information means the overall structure of the entire dataset, rather than just local, isolated relationships. Due to the neglect of global information, existing methods fail to detect outliers that can only be identified from a global perspective.
- Traditional methods focus solely on relative comparisons with neighbors, identifying an object as an outlier if its density is lower relative to its neighbors. However, they overlook the significant feature of neighbor similarity, which posits that objects in the same neighborhood exhibit consistent or similar outlier characteristics. Existing methods fail to leverage neighbor similarity, and mistakenly detect normal points in sparse regions as outliers, even if all neighbors of a given point are normal points.

The following example is utilized to elucidate issues inherent in previous algorithms. Figure 1 depicts a two-dimensional scatter plot of the Wine dataset after dimension reduction by PCA method. We focus on normal points q_1, q_2, q_3, and outlier point p to highlight problems existing in prior algorithms. It is clear that normal point q_1 resides in a dense

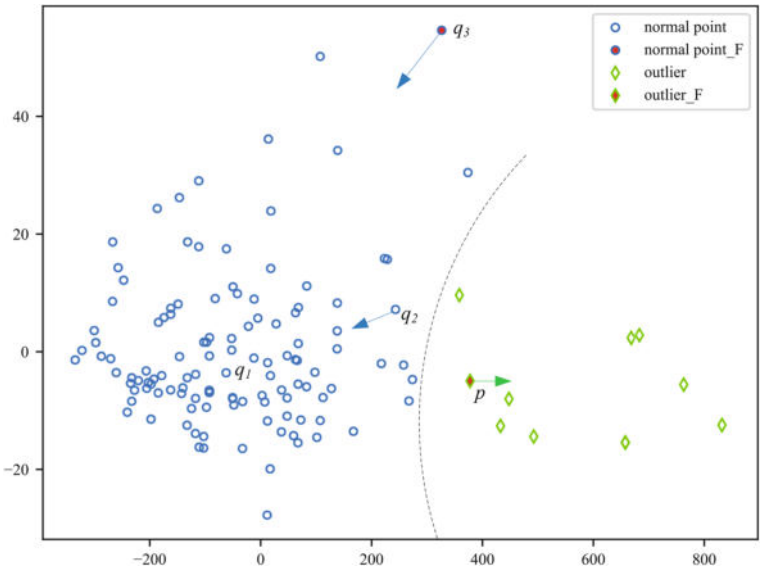

Fig. 1 Distribution of Wine dataset. In the legend, normal point_F denotes a normal point that is likely to be falsely detected as an outlier due to low density, and outlier_F denotes an outlier that is likely to be falsely detected as a normal point due to high density. The values on the horizontal and vertical axes represent the values of the first two dimensions of the objectives

area, q_2 is situated on the boundary of normal points, and q_3 exists in a sparse area. For the outlier p, we can see it lies on the boundary of normal points, with a density akin to q_2 and higher than q_3. If outliers are detected based on the neighborhood density without considering neighborhood similarity and global information, q_3 will be incorrectly identified as an outlier due to its location in a sparse region. Conversely, p will be wrongly identified as a normal point because it is situated in a relatively dense region from the local perspective. Clearly, these results are contrary to the actual facts. If considering all neighbors of q_3 are normal points and strengthening the similarity between q_3 and its neighbors, q_3 can be detected normal just like its neighbors. Similarly, the performance of boundary point q_2 tends toward normal points, while boundary point p inclines toward outlier points, as indicated by the arrows direction ("$\longrightarrow$") in Fig. 1. In addition, since the number of outlier points in the static dataset is known to be l, we argue that avoiding incorrectly detecting normal points as outliers could enhance the algorithm's outlier detection ability.

To address the above problems, this paper introduces an outlier detection algorithm based on local density feedback (LDF). In the beginning, the LDF algorithm utilizes the natural neighbor search algorithm for neighborhood selecting and initial density estimation. We further introduced a density post-processing technique to adjust the density through a density feedback mechanism. Through multiple feedback iterations, the global information is aggregated and eventually committed to yield the degree of outliers. The specific contributions of this study are as follows:

- We develop a tailored density post-processing technique for outlier detection, enhancing the accuracy of outlier degree depiction.
- We propose a density feedback mechanism that uses neighborhood similarity to update the density. This ensures that points within the same neighborhood have similar density

feedback values, while also allowing global information to be intelligently aggregated into each object.
- We introduce the concept of feedback outlier factor to quantify the outlier degree of data objects.
- We conduct extensive experiments on ten datasets and compared results with eight representative algorithms, and demonstrate that the proposed LDF algorithm effectively improves the performance of outlier detection.

The remainder of this article is organized as follows. Section 2 gives an overview of previous works related to our study. Section 3 proposes an outlier detection algorithm based on local density feedback. Section 4 reports the results of experiments and gives comparisons with existing methods. Section 5 concludes this article.

2 Related work

Many studies have delved into the complexities of outlier detection with exhaustive studies. In this section, we describe previous works related to this article, especially density-based algorithms.

The local outlier factor (LOF) algorithm [16], first introduced by Breunig et al., pioneered the notions of local outlier factor. LOF algorithm first searches KNN, then calculates the reachable distance and density of each object, and further define the outlier factor or score by comparing the density of each object with its surrounding neighbors, to quantify the degree of outlierness. Objects with lower density exhibit higher outlier factor, indicating a greater degree of outlierness. Building upon LOF, researchers have proposed numerous variants such as the connectivity-based outlier factor (COF) algorithm [18]. Zhou et al. [23] proposed a algorithm based on high-density iteration and extended k-neighbors to detect outliers. SPAD [24] relies on histograms and utilizes a probability density-based measure. SPAD+ [25] addresses SPAD's limitations by incorporating principal component analysis [26]. Liu et al. [27] employed the concept of potential energy in physics to represent local densities of data and exploited the notion of hubness in network science, to further capture global structural of data.

Abdul et al. [19] proposed RDOF algorithm using a simple density estimation, and the density is defined as the reciprocal of the distance to neighbors. Then RDOF measured the outlier' degree by comparing density with respect to the kth neighbor, addressing the challenge of detecting outliers between clusters of differing densities. Latecki et al. [22] introduced a new density estimation method, KDE, which utilized Gaussian kernel function to smooth the density estimation in outlier detection. Zhang et al. [21] adopt adaptive bandwidth parameters in Gaussian function to enhance the discriminating power of KDE-based outlier detection methods. Wang et al. [28] designed an outlying score based on the two-stage multi-kernel k-nearest neighbors to detect outliers. Dong et al. [20] focused on the density fluctuation of each instance neighborhood and defined two kinds of outlier factors to distinguish local and global anomalies. Based on the assumption that similar objects should have similar outlier scores, Yang et al. [29] designed a framework called neighborhood averaging. It post-processed the outlier scores produced by existing algorithms such as LOF, utilized the average scores of neighborhoods as the new scores of the objects, and demonstrated its effectiveness.

Considering that the parameter needs to be set manually to obtain the neighborhood of an object, which causes the algorithm to be very sensitive to the selection of the number of neighbors k, Zhu et al. [30] proposed the concept of natural neighborhood, which iteratively

determines the appropriate k value for the dataset, thus alleviating the requirement of manually tuning the parameter in the KNN-based algorithm. Wahid et al. [31] introduced a natural neighborhood-based outlier detection (NaNOD) algorithm to derive the k parameters with the natural neighborhood search algorithm, and used a weighted KDE [22] method to estimate the object density. Based on the natural neighbor search algorithm, Xiong et al. [32] proposed a weighted nearest-neighbor graph for calculating the density for the objects. Huang et al. [33] also used the natural neighborhood search algorithm to determine neighborhood, and introduced outlier turning points, considering turning points and their sparse neighborhoods as outliers.

To address the challenge that detection performance degrades as data dimensionality increases [34], Mahboobeh et al. [35] employed principal component analysis (PCA) [26] to partition data dimensions into three subspaces, subsequently comprehensively assessing outlier degrees across these subspaces. PCA can extract significant features from the original dataset, not only eliminating correlations and noise among dimensions but also accelerating subsequent tasks. It has been applied across various domains, including image processing [36] and signal processing [37]. To handle high-dimensional data, Li et al. [38] developed an attribute-weighted outlier detection method based on mixed data by classifying attributes into categorical and numerical attributes.

Inspired by previous methods, we employ PCA and the natural neighbor search algorithm as preprocessing steps in our proposed LDF method. While neighborhood averaging only performs simple score averaging, we introduce the concept of neighborhood similarity, which operates on density to more precisely capture outlier properties. Unlike local outlier detection approaches that focus solely on the surrounding neighborhood and miss global context, we incorporate a feedback mechanism that aggregates global information based on neighborhood similarity, allowing for more accurate outlier characterization.

3 Method

3.1 Density estimation

The LDF algorithm first employs the PCA method to preprocess data, which not only helps eliminate correlations among features but also accelerates the speed of subsequent tasks.

Given a dataset $D = \{a_1, a_2, ..., a_n\}, p = 1, \cdots, n, a_p \in R^d$, where d represents the number of dimensions of data objects, and n represents the number of objects in the dataset. In our study, we fixed the parameter $n\_components$ in PCA to 0.9 to retain 90% of the total variance of dimensions in the data, which is based on empirical evidence and performs well in the vast majority of applications. This setting allows PCA to automatically determine the number of dimensions after reduction. For a more detailed explanation of PCA method, refer to the seminal work by A Maćkiewicz and W Ratajczak [26]. After preprocessing with the PCA method, the dataset D is transformed into a counterpart $X = \{x_1, x_2, ..., x_n\}, p = 1, \cdots, n, x_p \in R^\alpha$ with α dimensions, and $\alpha < d$.

Subsequently, to address the challenge of determining the neighborhood parameter k, the LDF algorithm incorporates natural neighbor search algorithm, adaptively acquiring parameter k and KNN. The preliminary to the natural neighbor search algorithm are as follows.

We denote the Euclidean distance between x_p and x_q by $\mathrm{dist}(x_p, x_q)$, and the Euclidean distance between x_p and its kth nearest object by $\mathrm{dist_k}(x_p)$. The k-nearest neighbor set

$\mathrm{KNN}_k(x_p)$ is the collection of k objects closest to x_p, also referred to as the direct neighbors of x_p, as shown in Equation (1).

$$\mathrm{KNN}_k(x_p) = \{x_q | \mathrm{dist}(x_p, x_q) \leq \mathrm{dist}_k(x_p), x_p \neq x_q\} \tag{1}$$

where the neighborhood of x_p is $\mathrm{KNN}_k(x_p)$.

The reverse k-nearest neighbor set $\mathrm{RNN}_k(x_p)$ is the collection of objects who are neighbors of x_p, as shown in Equation (2).

$$\mathrm{RNN}_k(x_p) = \{x_q | x_p \in \mathrm{KNN}_k(x_q)\} \tag{2}$$

To automatically determine the number of nearest neighbors, the natural neighbor search algorithm expands the neighborhood range r from 1, stabilizing the count of objects with reverse neighbors at a certain value. This value, referred to as the natural eigenvalue k, represents the number of nearest neighbors. To improve the efficiency of KNN and RNN searches, the algorithm utilizes the Ball-Tree [39] structure, which significantly enhances spatial partitioning and search speed.

Algorithm 1 is the description of natural neighbor search algorithm.

Algorithm 1 Natural Neighbor Search Algorithm[31]

Input: Dataset B
Output: Natural eigenvalue k, k-nearest neighbor set KNN_k

Initialize $r = 1$, flag=0
Build a ball-tree based on B
while flag=0 **do**
  for each $x \in B$ **do**
    Find y(the rth neighbor of x)
    $\mathrm{NaN}(y) = NaN(y) + 1$
    $\mathrm{KNN}_r(x) = \mathrm{KNN}_{r-1}(x) \cup \{y\}$
    $\mathrm{RNN}_r(y) = \mathrm{KNN}_{r-1}(y) \cup \{x\}$
  end for
  if num does not change **then**
    flag=1
  end if
  $r = r + 1$
end while
$k = r - 1$
return k, KNN_k

In Algorithm 1, r represents the neighborhood searching range, $NaN(x)$ denotes the reverse neighbors of object x, and $num = count(NaN(x) == 0)$ signifies the count of objects without reverse neighbors. Besides yielding the natural eigenvalue k, Algorithm 1 also produces the k-nearest neighbor set KNN_k, which is utilized for subsequent outlier detection tasks.

Next, LDF utilizes KNN_k to estimate the density as follows.

Definition 1 k Average Distance, $\mathrm{k\_dist}(x_i)$: represents the average distance from the object x_i to its k-nearest neighbors. Its calculation method is shown in Equation (3).

$$\mathrm{k\_dist}(x_i) = \frac{1}{k} \sum_{x_j \in \mathrm{KNN}_k(x_i)} \mathrm{dist}(x_i, x_j) \tag{3}$$

In Equation (3), $\mathrm{KNN}_k(x_i)$ denotes the set of k-nearest neighbors of object x_i, where x_j represents one of the k-nearest neighbors of object x_i. The $\mathrm{dist}(x_i, x_j)$ signifies the Euclidean distance between object x_i and object x_j. The $\mathrm{k\_dist}(x_i)$ serves as a measure of the proximity of the object x_i to its neighbors.

Definition 2 Local Density, $\mathrm{ld}(x_i)$: The local density of object x_i is defined as the reciprocal of the average distance to its k-nearest neighbors, computed as depicted in Equation (4).

$$\mathrm{ld}(x_i) = \frac{1}{\mathrm{k\_dist}(x_i)} \tag{4}$$

In dense regions, objects exhibit smaller distances to their k-nearest neighbors, whereas in sparse regions, objects tend to have relatively larger distances to their k-nearest neighbors. The definition of local density in Equation (4) aligns with the concept that objects in dense regions have high local density, while objects in sparse regions have low local density.

Due to the varying scales of data in different datasets, to prevent the density values becoming excessively large or small, Equation (5) is employed for normalizing the density.

$$\mathrm{ld}_{\mathrm{norm}}(x_i) = \frac{\mathrm{ld}(x_i) - \mathrm{min\_ld}(B)}{\mathrm{max\_ld}(B) - \mathrm{min\_ld}(B)} \tag{5}$$

In Equation (5), $\mathrm{max\_ld}(B)$ stands for the highest local density value among the objects in dataset B, whereas $\mathrm{min\_ld}(B)$ represents the lowest density value among objects in dataset B. After normalization, the local density is scaled to fall within the range [0,1].

3.2 Density feedback mechanism

Due to the exclusive reliance on neighborhood information in local density estimation and the lack of neighborhood similarity and global context, the extracted density features fail to effectively distinguish between low-density normal points and outliers. To address this, we propose a density feedback mechanism that uses local density as an initial input and iteratively adjusts it to reduce discrepancies among neighboring objects, thereby enhancing their similarity. Based on the adjusted density, low-density normal points converge with normal data, while high-density outliers converge with other outliers, leading to a clearer distinction between the two groups. Below are the relevant definitions for the density feedback mechanism.

Definition 3 Density Feedback Value, dfv: dfv represents the dynamically adjusted value within the feedback mechanism, which tends to stabilize at the end of the feedback process.

The initial values of dfv are the local density values of objects, denoted as $\mathrm{dfv}^0(x_i) = \mathrm{ld}_{\mathrm{norm}}x_i, x_i \in B$. Subsequent steps gradually unveil the updating method of dfv.

Definition 4 Feedback Signal, $\Delta\mathrm{dfv}$: $\Delta\mathrm{dfv}(x_i)$ is the average dfv difference between object x_i and its k-nearest neighbors.

Before the first feedback, the feedback signal of object x_i is initialized as $\Delta\mathrm{dfv}^0(x_i)$, and it is calculated as Equation (6).

$$\Delta\mathrm{dfv}^0(x_i) = \frac{\sum_{x_j \in \mathrm{KNN}_k(x_i)} (\mathrm{dfv}^0(x_j) - \mathrm{dfv}^0(x_i))}{k} \tag{6}$$

The feedback signal reflects the difference in density between an object and its neighbors, and can be either positive or negative.

Based on the principle of neighborhood similarity, which suggests that objects within the same neighborhood exhibit similar outlier degrees, the LDF algorithm uses density feedback values to characterize outlier degrees. Therefore, neighborhood similarity implies that objects in the same neighborhood should have comparable density feedback values. The LDF algorithm enhances this similarity by integrating density feedback values and feedback signals.

Based on $\mathrm{dfv}^0(x_i)$ and $\Delta\mathrm{dfv}^0(x_i)$, the first updation for $\mathrm{dfv}(x_i)$ is depicted in Equation (7), and $\mathrm{dfv}^0(x_i)$ is updated to $\mathrm{dfv}^1(x_i)$.

$$\mathrm{dfv}^1(x_i) = \mathrm{dfv}^0(x_i) + \eta \cdot \Delta\ \mathrm{dfv}^0(x_i) \tag{7}$$

In Equation (7), η is a tunable hyperparameter denoting the rate of feedback, and $0 < \eta < 1$.

Within one feedback iteration, object x_i can aggregate information from its neighbors. By employing multiple feedback iterations, object x_i can aggregate global information.

The iterative updating process can be standardized using Equations (8) and (9).

$$\Delta\ \mathrm{dfv}^h\ (x_i) = \frac{\sum_{x_j \in \mathrm{KNN}_k(x_i)} \left(\mathrm{dfv}^h\left(x_j\right) - \mathrm{dfv}^h\ (x_i)\right)}{k}, h = 0, 1, 2, \cdots \tag{8}$$

$$\mathrm{dfv}^h\ (x_i) = \mathrm{dfv}^{h-1}\ (x_i) + \eta \cdot \Delta\ \mathrm{dfv}^{h-1}\ (x_i)\ , h = 1, 2, \cdots \tag{9}$$

It is worth noting: Equation (8) is utilized to compute the feedback signal, commencing from $h = 0$. At $h = 0$, Equation (8) corresponds to the previously mentioned Equation (6). Meanwhile, Equation (9) serves to update the density feedback values, with h starting from 1 to signify the feedback iterations. At $h = 1$, Equation (9) corresponds to the earlier discussed Equation (7). This process alternates between Equation (8) and Equation (9), with each alternation constituting one iteration, also referred to as one feedback cycle.

Combining Equations (8) and (9), we can analyze the feedback mechanism. From a local perspective, if the average density feedback value of x_i's neighbors exceeds that of x_i itself, i.e., $\Delta\mathrm{dfv}(x_i) > 0$, according to Equation (9), the density feedback value of x_i will increase. Conversely, if the average density feedback value of neighbors is less than that of x_i, i.e., $\Delta\mathrm{dfv}(x_i) < 0$, according to Equation (9), the density feedback value of x_i will decrease. Therefore, the feedback mechanism effectively reduces differences between neighboring objects, enhancing their similarity. From a global perspective, density feedback values exhibit a propagative property. An object utilizes neighbors' information to update itself, we can say it aggregates the information of neighbors.

In Fig. 2, big circles are used to draw the neighborhood of point s and point t with $k = 3$. The arrow " $\longrightarrow$" represents the neighbor relationship. For example, "o $\longrightarrow$ s" means that point o is a neighbor of point s, and "s $\longrightarrow$ t" indicates that point s is a neighbor of point t. In the first feedback iteration, point s can aggregate the information of point o; in the second feedback iteration, point t can aggregate information directly from point s. Additionally, point t can indirectly gather information from point o, which has already been aggregated into point s. With increasing feedback iterations, each point in the dataset can aggregate the information from all points, thereby possessing global information. Additionally, the process of enhancing neighborhood similarity also serves to smooth out density.

3.3 Feedback stopping condition

As the number of feedback iterations increases, the magnitude of change in density feedback values gradually diminishes, and density feedback values will reach a convergence state,

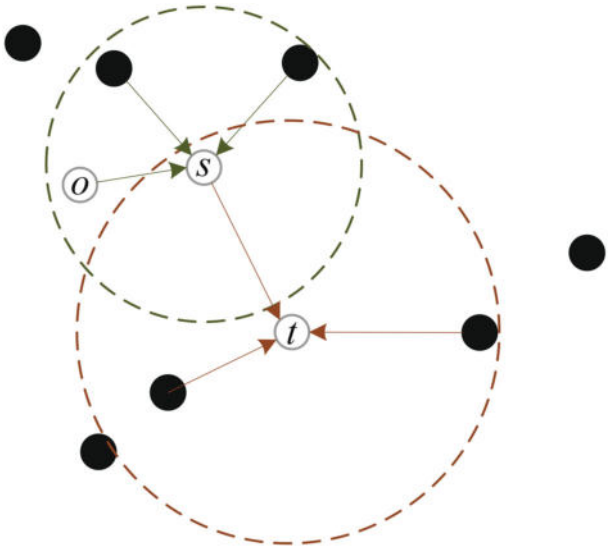

Fig. 2 Neighborhood of objects s and t

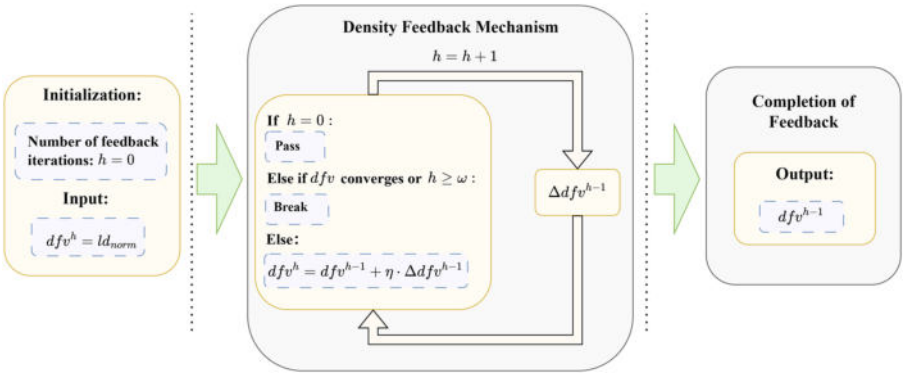

Fig. 3 Diagram of the feedback mechanism of LDF algorithm

where no significant changes occur. To enhance the algorithm's efficiency, a sufficiently small value is specified as the convergence tolerance: ϵ. When the change in density feedback values falls below this tolerance ϵ, denoted as $\forall x_i \in B, \mathrm{dfv}^h(x_i) - \mathrm{dfv}^{h-1}(x_i) \leq \varepsilon$, we regard that it achieves the convergence state. Equation (9) shows that the change in the density feedback value of x_i between the two adjacent iterations is $\eta \cdot \Delta\mathrm{dfv}^{h-1}(x_i)$. Hence, if $\forall x_i \in B, \left|\eta \cdot \Delta\mathrm{dfv}^{h-1}(x_i)\right| \leq \varepsilon$, it is deemed that dfv has converged. Additionally, by setting a maximum number of feedback iterations $max\_iterations$, feedback ceases once this limit is reached. The feedback rate η also plays a role in convergence; if too small, feedback may be sluggish, and if too large, significant fluctuations in density feedback values may prevent convergence. Therefore, fine-tuning η is crucial for achieving optimal results.

Figure 3 illustrates the feedback mechanism of the LDF algorithm. This mechanism extracts Δdfv from dfv and uses it to update the dfv automatically.

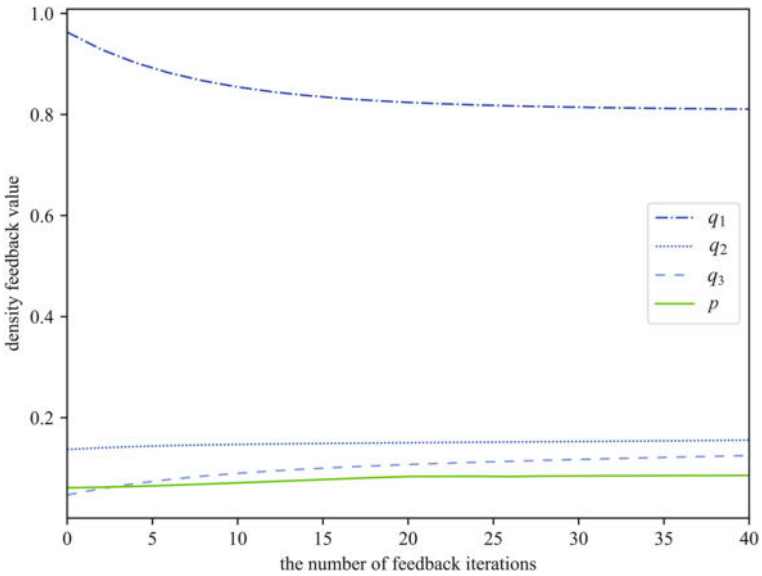

Fig. 4 Curves for density feedback values with varying feedback iterations ($\eta = 0.12$)

Below, we use the Wine dataset in Fig. 1 as an example to explain the effectiveness of the feedback mechanism. Fig. 4 shows the curve representing the evolution of density feedback values for normal points q_1, q_2, q_3, and the outlier p over feedback iterations. With a feedback rate of $\eta = 0.12$, density feedback values approximately stabilize after 40 iterations. Initially, q_3's density feedback value are lower than that of the outlier p. However, as the number of feedback iterations increases, q_3's density feedback value gradually rises, surpassing that of the outlier p by the 4th feedback iteration. Furthermore, the density feedback values of normal points q_1 and q_2 consistently remain higher than that of the outlier p.

In Fig. 5, the points in the Wine dataset are depicted in different colors so that the density feedback values can be visualized intuitively combined with the data distribution. Diamond-shaped points represent outliers, while circle-shaped points represent normal points. Lighter colors indicate higher density feedback values, darker colors indicate lower density feedback values. It is evident that outlier points appear darker compared to normal points, indicating lower density feedback values associated with outliers in contrast to normal points.

To intuitively perceive the changes of density feedback values, Fig. 6 selects the normal point q_3, and the outlier p from Wine dataset, and uses arrows ("$\longrightarrow$") to depict their density feedback values with the increase of feedback iterations (the number of feedback iterations increases in the counterclockwise direction). The length of each arrow corresponds to the density feedback value computed based on Equations (8) and (9). The direction of the arrows accentuates the trend of being more similar to its neighbors, pointing toward the approximate direction of neighbors. The angles between the arrows serve as visual aids to showcase the dynamic changes in density feedback values during feedback iterations. From Fig. 6, it is evident that q_3 undergoes a rapid and substantial variation in density feedback values, while p demonstrates a slower and less pronounced trend. With increasing feedback iterations, the density feedback value of q_3 eventually exceeds that of p.

To further accentuate the distinction between normal and outlier points, Fig. 7 plots the reciprocals of density feedback values for q_1, q_2, q_3, and p over the first 40 iterations. It

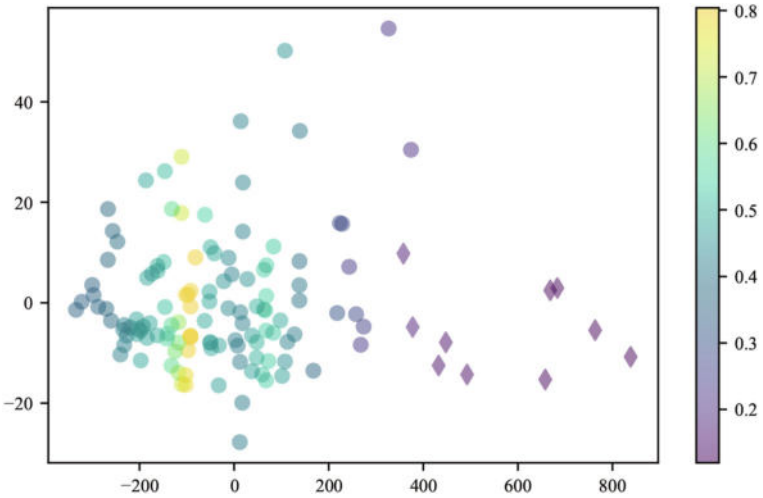

Fig. 5 Wine dataset with colored points. Diamond-shaped points represent outliers, while circle-shaped points represent normal points. The values on the horizontal and vertical axes represent the values of the first two dimensions of the objectives

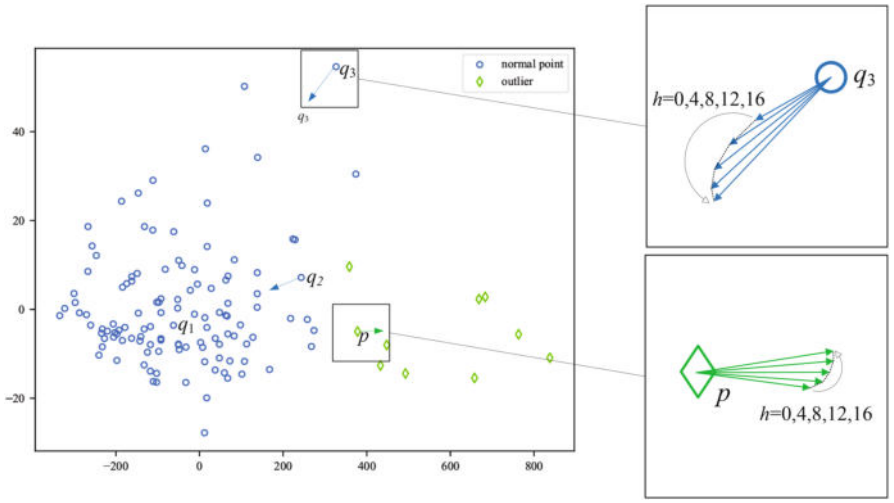

Fig. 6 Changes of density feedback values of points q_3 and p. The values on the horizontal and vertical axes represent the values of the first two dimensions of the objectives

is clear that, the curve of q_3 gradually trends closer to the normal points q_1 and q_2 while trending away from the outlier point p. After the 40th feedback iteration, the density feedback values of the outlier p stabilize consistently above those of normal points q_1, q_2, and q_3. This effectively facilitates the differentiation between the outlier point p and the normal points q_1, q_2, and q_3.

Furthermore, we analyze that normal data situated in sparse regions may exhibit fluctuations due to random noise, which is considered a normal phenomenon. Conversely, outliers are produced by abnormal mechanisms rather than random noise. The introduction of den-

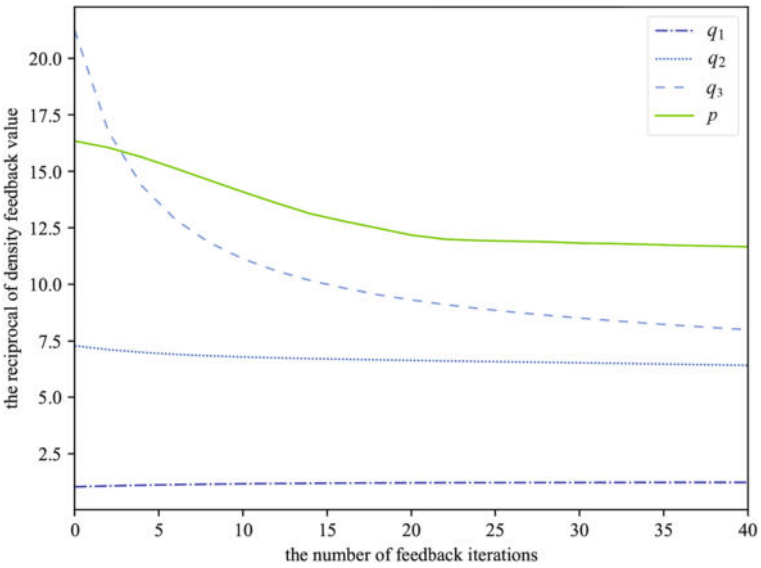

Fig. 7 Curves for reciprocals of density feedback values with varying feedback iterations ($\eta = 0.12$)

sity feedback values and feedback mechanisms helps uncover deeper distribution patterns within the data, reducing sensitivity to noise in density estimation, and thus enhancing the robustness of outlier detection algorithms.

3.4 Feedback outlier factor

After completing the feedback process, objects with smaller density feedback values exhibit a higher degree of outlierliness. To quantify the degree of outliers, we define the feedback outlier factor as a positively correlated representation of outlierliness.

Definition 5 Feedback Outlier Factor, FOF: The FOF for object x_i is defined as the reciprocal of the density feedback value $\mathrm{dfv}(x_i)$, as shown in Equation (10).

$$\mathrm{FOF}(x_i) = \frac{1}{\mathrm{dfv}(x_i)} \tag{10}$$

Objects with a higher feedback outlier factor are more likely to be outliers, whereas objects with a lower feedback outlier factor are more likely to be normal points.

3.5 Algorithm description

The workflow of the LDF algorithm is depicted in Fig. 8. The LDF algorithm first performs dimensionality reduction on the original data using the PCA. Subsequently, it employs the natural neighbor search algorithm to determine the natural feature value k and KNN. Following this, leveraging the idea that objects in dense regions possess higher density while those in sparse regions have lower density, the reciprocal of the average distance to KNN serves as the local density of objects. Then we introduce a density feedback mechanism and density feedback values, aggregating global information using neighborhood similarity.

Density feedback values are updating until they stabilize or the maximum iteration count is reached. The feedback mechanism narrows the differences among neighboring objects, enabling the density feedback values of low-density normal points returning to the normal range, facilitating the differentiation from outliers. Finally, the feedback outlier factor is used to quantify the degree of the outliers, i.e., the larger the feedback outlier factor, the higher the degree of the outliers. Specifically, based on the top-l strategy, the top-l objectives with the highest outlier scores are predicted as outliers.

Algorithm 2 is the description of the LDF. In the LDF algorithm, the time complexity for dimensionality reduction using PCA is $O(nd^2)$, where n is the total number of objects in the dataset and d is the original dimensionality. After the dataset is reduced to α dimensions using PCA, the time complexity for performing the natural neighbor search algorithm with k neighbors using a Ball-Tree structure is $O(\alpha n \log(n))$. The time complexity of the feedback mechanism is $O(\omega\alpha nk)$, in which the max number of feedback iterations ω is a constant value, so it can be simplified to $O(\alpha nk)$. Given that the dimensionality $\alpha << d$ after PCA processing, and the number of neighbors $k << n$, so we can omit them and retain only those related to the data size n. Furthermore, we used the PCA method provided by scikit-learn,[1] which employs various strategies to accelerate the computation, resulting in high efficiency. Hence, the computational complexity is approximately $O(n \log(n))$. Overall, the time complexity of the algorithm is acceptable.

Algorithm 2 Local Density Feedback Outlier Detection Algorithm (LDF)

Input: Raw dataset D, feedback rate η,max feedback iterations $\omega = 300$, convergence tolerance $\varepsilon = 1e^{-4}$
Output: l outliers
1: Use PCA method to reduce the dimensionality of dataset D: $B = \text{PCA}(D)$
2: Use Algorithm 1 to get k and KNN_k
3: **for** each $x_i \in B$ **do**
4: Use Equation (4) and (5) to calculate local density $\text{ld}_{\text{norm}}(x_i)$
5: $\text{dfv}^0(x_i) = \text{ld}_{\text{norm}}(x_i)$
6: **end for**
7: Set the initial number of feedback iterations:$h = 0$
8: **while** dfv unconverging or $h < \omega$ **do**
9: $h = h + 1$
10: **for** each $x_i \in B$ **do**
11: Use Equation (8) to calculate $\Delta\text{dfv}^h(x_i)$
12: Use Equation (9)to update $\text{dfv}^h(x_i)$
13: **end for**
14: **end while**
15: **for** each $x_i \in B$ **do**
16: Use Equation (10) to calculate
17: **end for**
18: Sort FOF in reverse order
19: **return** l outliers

4 Experiment

In this section, experiments were conducted on ten real-world datasets using eight algorithms to demonstrate the superiority of the proposed LDF algorithm. Subsequently, experiments

[1] https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html.

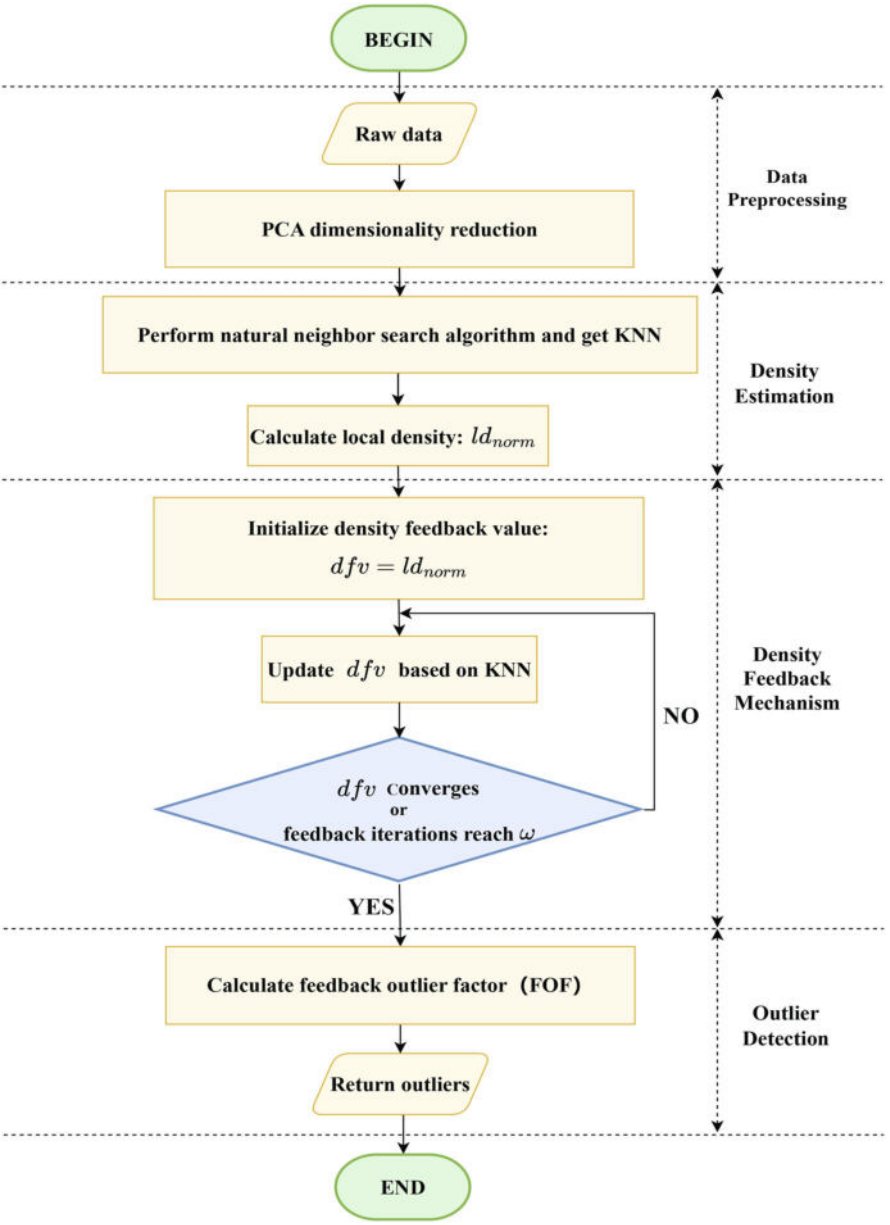

Fig. 8 Flowchart of LDF algorithm

were performed to analyze the effectiveness of the density feedback mechanism, proving its validity from two perspectives: the ablation of the density feedback mechanism and the exclusion of PCA's influence. Next, parameter experiments were conducted to determine the optimal range of parameter η, and finally, time efficiency experiments were carried out.

Table 1 Characteristics of datasets

Dataset	Points	Dimensions	Outliers	Percentage of Outliers
Ecoli	168	7	25	14.8%
Wine	129	13	10	7.7%
Wbc	278	30	21	5.6%
Ionosphere	351	32	126	35.8%
Wdbc	390	30	33	8.4%
Cardio	1831	21	176	9.6%
Arrhythmia	452	274	66	15%
Waveform	3443	21	100	2.9%
Satellite	6435	36	2036	32%
Satimage-2	5803	36	71	1.20%

Table 2 Dataset dimensionality reduction using PCA

Dataset	Initial dimension	Reduced dimension	Percentage of variance retained (%)
Ecoli	7	5	90
Wine	13	2	90
Wbc	30	7	90
Ionosphere	32	28	90
Wdbc	30	2	90
Cardio	21	12	90
Arrhythmia	274	26	90
Waveform	21	15	90
Satellite	36	4	90
Satimage-2	36	4	90

4.1 Experimental setting

4.1.1 Dataset

We adopted real-world datasets sourced from the open-access UCI repository, which included Escherichia coli (Ecoli), Wine, Wisconsin Breast Cancer (Wbc), Ionosphere, Wisconsin Diagnostic Breast Cancer (Wdbc), Cardio, Arrhythmia, Waveform, Satellite and Satimage-2 datasets. The outlier proportion in these datasets ranges from 1.2% to 35.8%, with the number of data instances ranging from 129 to 6435 and dimensions spanning from 7 to 274. These datasets encompass various data scales and distribution types, facilitating a comprehensive evaluation of the performance of the LDF algorithm. Detailed characteristics of the datasets are presented in Table 1.

As the LDF algorithm employs PCA method for dimensionality reduction, Table 2 lists the original dimensions and the number of dimensions after reduction for each dataset. We set the PCA parameter *n_components* to 0.9 in order to retain 90% of the total variance in the dataset.

4.1.2 Comparison algorithms

To evaluate the performance of the LDF algorithm, we compared it with eight algorithms include classic methods: LOF [16], COF [18], and IForest [40], along with state-of-art approaches: NaNOD [20], RDOF [19], ECOD [8], FDPC-OF [14] and SODEP [35]. Among these, LOF, COF, NaNOD and RDOF are density-based algorithms, NaDOD uses the same natural neighbor search algorithm as our LDF, and SODEP use the same PCA method as our LDF, so they serve as horizontal comparison to our LDF method. IForest is a widely recognized isolation-based algorithm, FDPC-OF combines density and clustering methods, and ECOD is a statistics-based methods, offering a vertical comparison to assess the performance of our LDF algorithm.

In the experiments, the feedback rate η for LDF algorithm ranges from 0 to 1. For the comparison algorithm NaNOD, the KDE parameter θ is set to the default value of 0.5. IForest uses default parameters, with $n\_estimators$ set to 100 and $max\_samples$ set to 256, representing the number of iTrees and the sample size, respectively. FDPC-OF utilizes default parameters, with k set to 5 and c set to 2, representing the number of neighbors and the number of clusters, respectively. LOF, COF, RDOF and SODEP require setting the parameter k, which represents the number of neighbors. We test k values ranging from 5 to 100 to determine the optimal k value for LOF, COF, RDOF and SODEP algorithm on each dataset, recording their best performance accordingly. ECOD does not require parameter settings. We used the top-l strategy for all algorithms consistently, and set l = S (the number of true outliers in each dataset), ensuring a fair comparison.

4.1.3 Evaluation measures

We evaluate the performance of algorithms using precision (Pr) and AUC (Area Under the Curve) metrics. The formula for precision is shown in Equation (11).

$$\mathrm{Pr} = \frac{\mathrm{TP}}{\mathrm{TP+FP}} \tag{11}$$

In Equation (11), TP represents the number of instances correctly identified as outliers by the algorithm, while FP represents the number of instances incorrectly identified as outliers by the algorithm. And TP + FP = S, where S is number of outliers contained in the dataset.

The AUC represents the area under the receiver operating characteristic (ROC) curve, where the x-axis and y-axis of the ROC curve represent the false positive rate and true positive rate, respectively. AUC evaluates the overall ranking quality of models, reflecting the model's ability to rank positive instances higher than negative instances. AUC values range from 0 to 1. An AUC value of 0.5 indicates the performance equivalent to random guessing, and has no practical utility. An AUC value of 1 signifies that the model correctly ranks all positive instances ahead of negative instances. Therefore, the closer the AUC value is to 1, the better the model performs.

4.2 Overall performance

We executed our LDF algorithm along with eight comparison algorithms on the ten datasets presented in Table 1. The precision of algorithms are presented in Table 3, and the last row of the table lists the average precision of each algorithm on ten datasets, with the optimal precision for each dataset highlighted in bold.

Table 3 Precision of different algorithms on ten datasets

Dataset	LOF	COF	IForest	RDOF	NaNOD	ECOD	FDPC-OF	SODEP	LDF
Ecoli	0.840	0.880	0.800	0.800	0.840	0.760	0.520	0.840	**0.96**
Wine	0.900	0.900	0.100	0.900	0.900	0.100	0.600	0.900	**1.000**
Wbc	0.571	0.524	0.524	0.571	0.524	0.429	0.096	0.571	**0.619**
Ionosphere	0.825	0.825	0.675	0.667	0.738	0.520	0.857	0.857	**0.865**
Wdbc	0.879	**0.909**	0.515	0.879	0.789	0.455	0.788	0.879	**0.909**
Cardio	0.225	0.210	0.472	0.188	0.534	0.528	0.256	0.460	**0.591**
Arrhythmia	0.515	**0.520**	0.439	0.500	0.510	0.485	0.348	0.510	0.515
Waveform	0.210	0.290	0.090	0.130	0.110	0.050	0.150	0.210	**0.300**
Satellite	0.417	0.432	0.559	0.410	0.583	0.449	0.580	0.541	**0.584**
Satimage-2	0.113	0.169	0.901	0.151	**0.915**	0.620	0.239	0.789	**0.915**
AVG	0.550	0.568	0.508	0.520	0.644	0.440	0.443	0.656	**0.726**

From Table 3, it can be observed that the LDF algorithm achieves the highest precision on nine out of ten datasets. Specifically, for the isolation-based algorithm IForest, the statistics-based algorithm ECOD, and the clustering-based algorithm FDPC-OF, all the neighborhood information-based algorithms—LOF, COF, RDOF, NaNOD, and SODEP—outperform these three algorithms in terms of precision across all datasets except for the Cardio dataset. This illustrates the importance of local information in the outlier detection process. For the six datasets—Ecoli, Wine, Wbc, Ionosphere, Cardio, Waveform and Satellite—the outlier detection precision of LDF surpasses that of the neighborhood information-based algorithms (LOF, COF, RDOF, NaNOD, and SODEP) and exceeds the suboptimal algorithms by 9.09%, 11.1%, 8.41%, 0.933%, 10.7%, 3.45% and 0.172%, respectively. This is mainly because the five aforementioned algorithms only consider local information and neglect the crucial role of global information in the detection process. In contrast, LDF not only utilizes local information but also incorporates global information through the proposed feedback mechanism, enabling it to detect outliers more accurately and effectively solve the issue of low-density normal points being misidentified as outliers. It is worth mentioning that for the COF algorithm, we obtained the best precision results on the Wdbc and Arrhythmia datasets by adjusting the parameter of k. However, the time cost involved did not result in significantly better performance compared to the LDF algorithm, with precision only 0% and 0.971% higher than the LDF algorithm on the Wdbc and Arrhythmia datasets, respectively. Another reason for the slightly lower performance of LDF compared to COF is that Arrhythmia is a high-dimensional dataset, and LDF incorporates too much redundancy in the feedback process. This is also an aspect we are committed to enhancing in the future. Overall, our LDF algorithm produces the highest average precision of 0.726, 10.7% higher than that of the second-ranked SODEP algorithm.

Table 4 presents the AUC results for each algorithm across the ten datasets, with the last row displaying the average AUC of each algorithm on ten datasets, and the best AUC highlighted in bold. It is evident that the LDF algorithm achieves the best AUC on nine out of ten datasets. For the eight datasets—Ecoli, Wine, Wbc, Ionosphere, Arrhythmia, Waveform, Satellite and Satimage-2—the AUC of LDF surpasses that of the neighborhood information-based algorithms (LOF, COF, RDOF, NaNOD, and SODEP) and exceeds the suboptimal algorithms by 0.102%, 0.100%, 0.105%, 0.319%, 0.361%, 2.34%, 6.82% and 0.104%, respectively. This indicates the effective of fully utilization of neighborhood similarity and global informa-

Table 4 AUC of different algorithms on ten datasets

Dataset	LOF	COF	IForest	RDOF	NaNOD	ECOD	FDPC-OF	SODEP	LDF
Ecoli	0.979	0.983	0.949	0.974	0.943	0.936	0.888	0.967	**0.984**
Wine	0.999	0.999	0.766	0.999	0.998	0.733	0.913	0.999	**1.000**
Wbc	0.951	0.929	0.949	0.951	0.894	0.900	0.334	0.951	**0.952**
Ionosphere	0.903	0.920	0.852	0.821	0.858	0.728	0.941	0.925	**0.944**
Wdbc	0.989	**0.993**	0.945	0.983	0.967	0.931	0.953	0.988	**0.993**
Cardio	0.822	0.579	0.818	0.544	0.896	**0.902**	0.588	0.873	0.892
Arrhythmia	0.816	0.832	0.796	0.796	0.809	0.805	0.736	0.817	**0.835**
Waveform	0.742	0.756	0.725	0.657	0.771	0.608	0.696	0.750	**0.789**
Satellite	0.572	0.570	0.704	0.565	0.704	0.583	0.733	0.732	**0.783**
Satimage-2	0.953	0.727	0.994	0.580	0.996	0.965	0.812	0.996	**0.997**
AVG	0.872	0.829	0.850	0.787	0.884	0.809	0.759	0.900	**0.917**

tion in our proposed feedback mechanism. It is worth noting that for the COF algorithm, its AUC value on the Arrhythmia dataset did not reach the highest level, as seen with the precision metric. Instead, our LDF algorithm achieves the highest AUC. The statistics-based ECOD surpasses the LDF on the Cardio dataset because the distribution of the Cardio dataset aligns well with statistical models. Nonetheless, our LDF algorithm remains competitive on the Cardio dataset, closely following the ECOD. Another noteworthy point is that the performance of our LDF algorithm surpasses that of the PCA-based SODEP algorithm. This indicates that when both algorithms utilize PCA, our proposed LDF method is more effective at detecting outliers. Additionally, it further demonstrates that our proposed feedback mechanism can effectively leverage neighborhood similarity and global information to enhance the performance of outlier detection. Overall, our LDF algorithm achieves the highest average AUC of 0.917, 1.89% higher than that of the second-ranked SODEP algorithm. Hence, our LDF algorithm demonstrates higher performance in outlier detection compared to the other algorithms.

Figure 9 utilizes a heatmap to visually depict the average performance enhancement of the LDF algorithm compared to eight comparison algorithms on the datasets mentioned above. Darker shades mean greater improvement in performance, and the values on the blocks indicate the magnitude of the performance improvement. It is evident that the LDF algorithm exhibits varying degrees of improvement over the comparison algorithms. For instance, LDF's precision improves on average by 0.286 compared to ECOD algorithm, while its AUC increases by an average of 0.158 compared to FDPC-OF algorithm.

4.3 Effect of the density feedback mechanism

4.3.1 Ablation study of the density feedback mechanism

We introduce a density feedback mechanism in LDF algorithm for outlier detection, which utilizes neighborhood similarity to aggregate global information. To verify the effectiveness of the density feedback mechanism, we designed one variants: LDOD, with the density feedback mechanism removed. The precisions and AUC of LDF and LDOD on 10 datasets are shown in Table 5, with the best result for each dataset highlighted in bold. By analyzing the

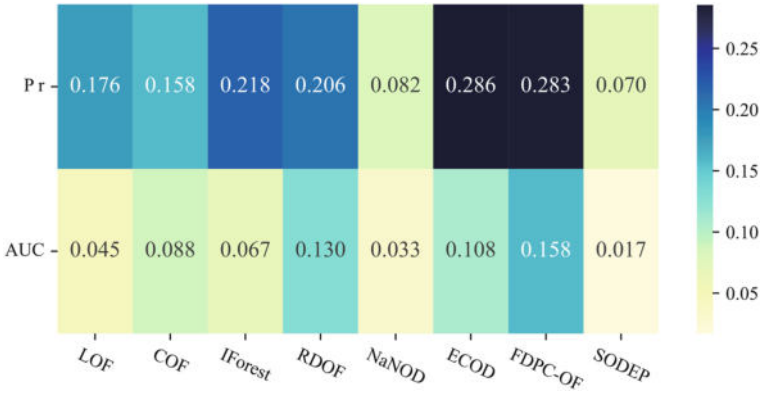

Fig. 9 The average performance enhancement of the LDF algorithm

Table 5 Precision and AUC of LDOD and LDF algorithms

Dataset	LDOD		LDF	
	Pr	AUC	Pr	AUC
Ecoli	0.840	0.956	**0.960**	**0.984**
Wine	0.800	0.996	**1.000**	**1.000**
Wbc	0.571	0.944	**0.619**	**0.952**
Ionosphere	0.857	0.943	**0.865**	**0.944**
Wdbc	0.818	0.973	**0.909**	**0.993**
Cardio	0.290	0.723	**0.591**	**0.892**
Arrhythmia	**0.515**	0.808	**0.515**	**0.835**
Waveform	0.210	0.761	**0.300**	**0.789**
Satellite	0.495	0.686	**0.584**	**0.783**
Satimage-2	0.338	0.974	**0.915**	**0.997**
AVG	0.573	0.876	0.726	0.917

experimental results in Table 5, we draw the following conclusion: LDF consistently outperformed LDOD across ten datasets, which demonstrates the effectiveness of our proposed density feedback mechanism in improving outlier detection performance.

Figure 10 employs a heatmap to illustrate the performance enhancement brought by the feedback mechanism in the LDF algorithm, i.e., the performance improvement of LDF relative to LDOD. Darker shades signify greater performance gains, with the numerical values on the tiles indicating the magnitude of the performance improvement. Clearly, the highest improvement in accuracy was achieved on the Satimage-2 dataset, reaching 0.577, while the greatest AUC improvement was on the Cardio dataset, reaching 0.169.

Figure 11 illustrates the visualization results of the LDOD and LDF algorithms. The subplots in Fig. 11 depict the two main features of the datasets after dimensionality reduction, and subplots on the left side are the visualization results of the LDOD algorithm, while subplots on the right side are those of the LDF algorithm. Taking Fig. 11a as an example, it can be observed that in the scatter plot of the Ecoli dataset, there are some normal objects misclassified as outliers by LDOD, leading to low precision. However, in the LDF algorithm, due to the misclassified normal objects having a plenty of normal neighbors, their similarity

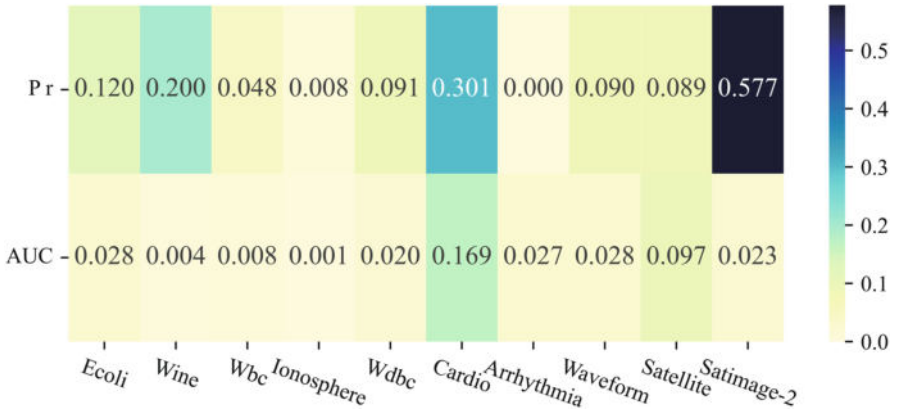

Fig. 10 The performance enhancement brought by the feedback mechanism

with neighbors increases and the degree of outlierness decreases through the density feedback mechanism, thus they can be correctly detected as normal points. Similarly, outliers have outlier neighbors, and their outlier degree increases due to mutual influence in the LDF algorithm.

4.3.2 Analysis to exclude the effects of PCA

To further verify that the proposed density feedback mechanism is the main driver of LDF's improved performance in outlier detection, rather than the introduction of PCA, we conducted experiments on each benchmark by both including and removing PCA. The experimental results are shown in Tables 6 and 7.

By comprehensively analyzing the experimental results in Tables 6 and 7, we can draw the following conclusions: (1) For all methods, there is no direct correlation between the inclusion of PCA and performance improvement. In fact, for some datasets, certain methods experienced a performance drop after the introduction of PCA, such as IForest and ECOD on the Wbc dataset; (2) Compared to the comparison algorithms, LDF demonstrated greater robustness in this set of PCA analysis experiments, with its outlier detection performance remaining stable across all datasets, regardless of whether PCA was included; (3) Even without the inclusion of PCA, LDF still outperformed nearly all comparison algorithms across all datasets. Therefore, based on the analysis above, we can conclude that the core factor driving LDF's performance improvement is the proposed density feedback mechanism, not the inclusion of PCA. The introduction of PCA is not inherently linked to the improvement of outlier detection algorithm performance.

4.4 Experiment of parameter

In this section, we focus on the proposed density feedback mechanism, and analyze the impact of the feedback rate parameter η on LDF's detection performance. To explore the optimal range of η, values ranging from 0 to 1 with an interval of 0.01 were tested on ten datasets. It is worth noting that when η is set to 0, the results produced by the LDF algorithm are identical to those produced by the LDOD algorithm.

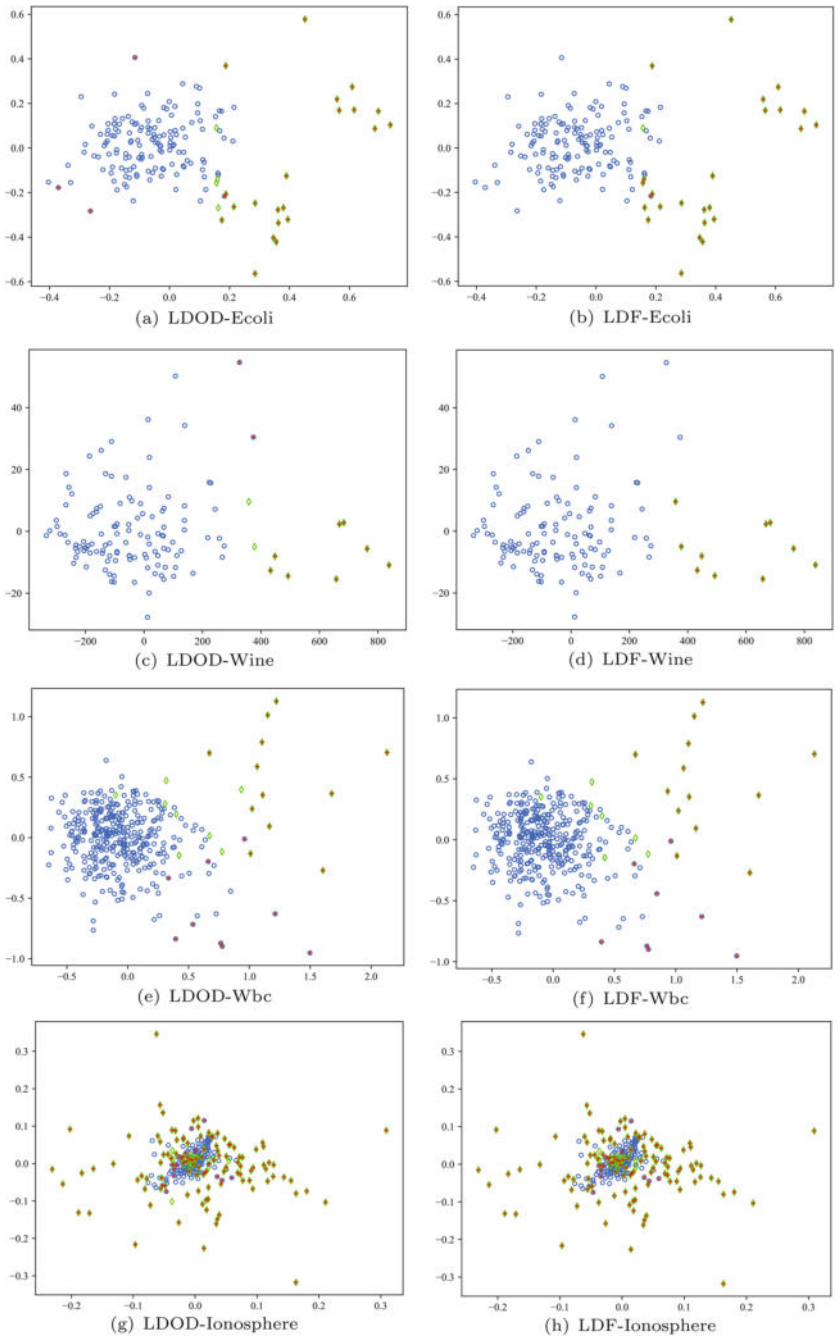

Fig. 11 Visualization results of LDOD and LDF algorithms. Circle-shaped points "∘" represent normal points in the dataset, while diamond-shaped points "◇" denote outliers. Circle-shaped points with symbols "+" indicate outliers incorrectly detected by the algorithm, while diamond-shaped points with symbols "+" represent outliers correctly detected by the algorithm. The values on the horizontal and vertical axes represent the values of the first two dimensions of the objectives. Curves for precision with changing η. Curves for AUC with changing η

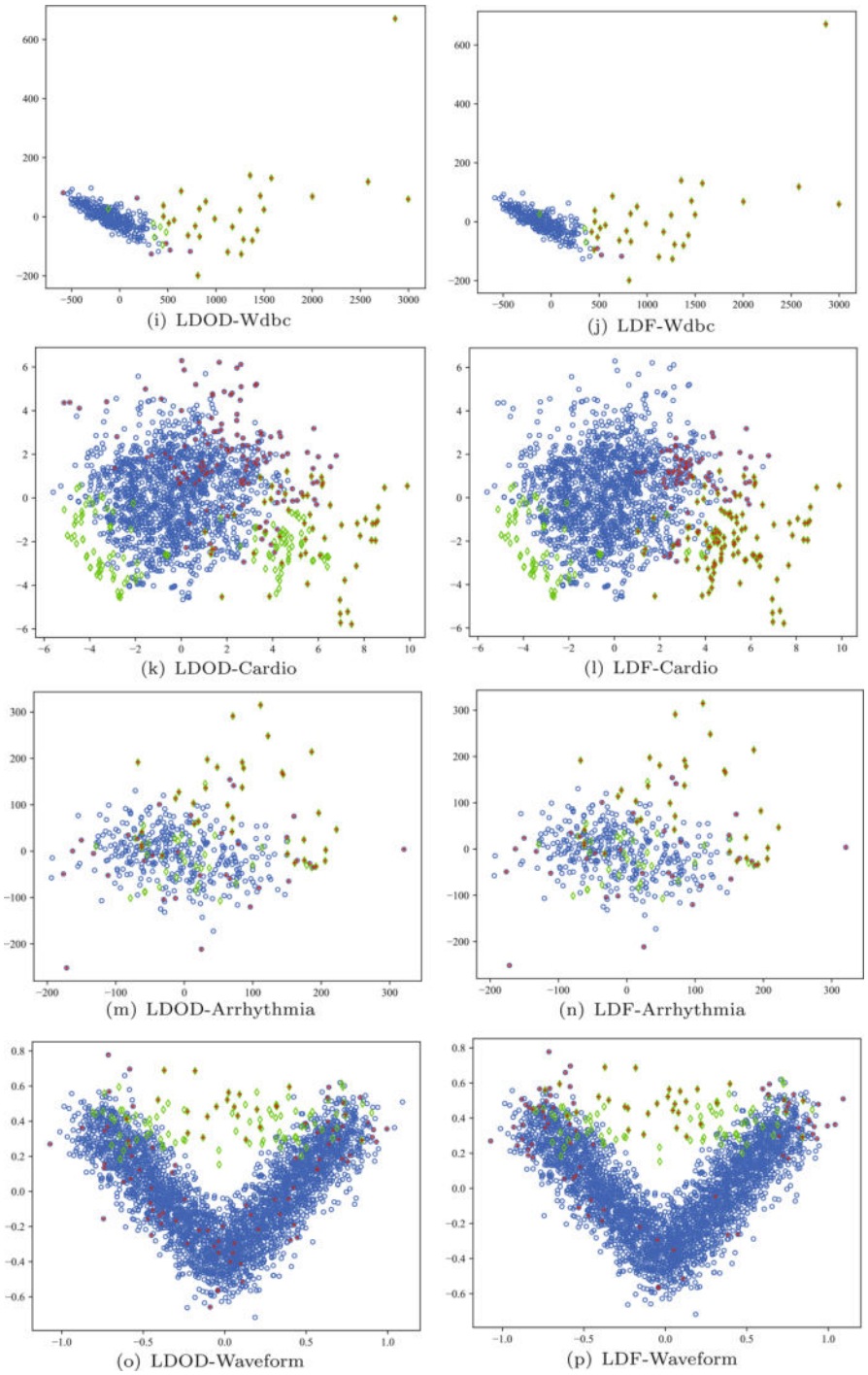

Fig. 11 continued

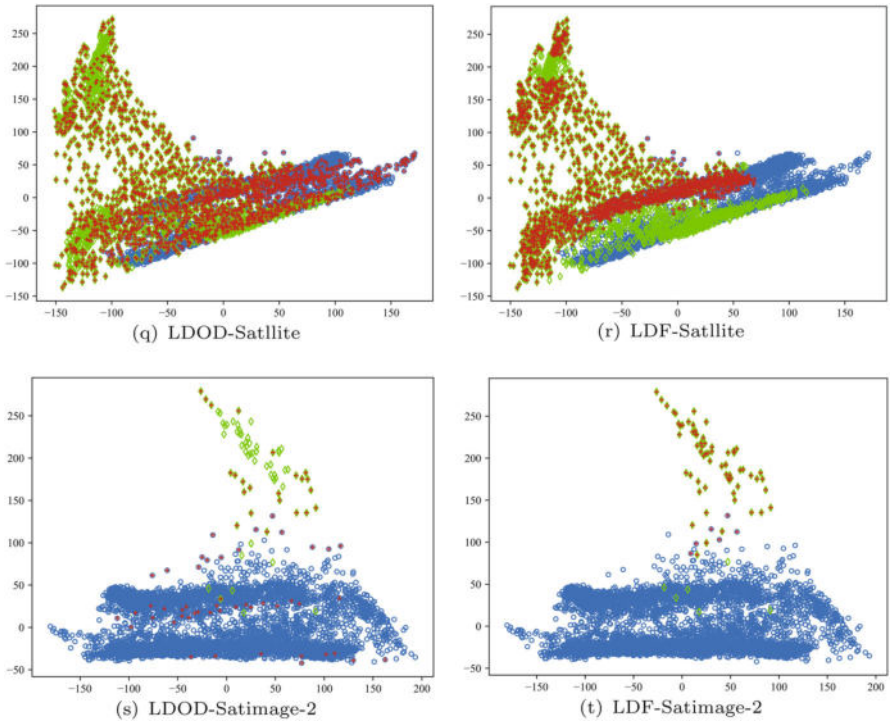

Fig. 11 continued

Figure 11 depicts the curve of precision of the LDF algorithm with varying η. It can be observed that for the Ecoli, Wbc, Wdbc, Cardio, Waveform, Satellite and Satimage-2 datasets, there is a significant increase in precision when the parameter η reaches around 0.02. For the Satellite dataset, precision showed a slight decline after η=0.12, but began to gradually increase again at $\eta = 0.21$. For the Satimage-2 dataset, the curve remained stable from η=0.12 to η=0.23, followed by a slight drop in precision, which then stabilized again after η=0.26. For the Wine dataset, precision reaches 1 when η is around 0.12. The Ionosphere dataset shows a slight increase in precision when η ranges from 0 to 0.02, while the Arrhythmia dataset maintains stability between $\eta = 0$ and 0.15, then declines.

Figure 11 illustrates the curve of AUC of the LDF algorithm with varying η. It reveals a similar trend in AUC as precision. Around $\eta = 0.02$, the LDF algorithm demonstrates commendable performance across all datasets. Therefore, we conclude that the optimal range for parameter η is from 0 to 0.2, with a suggested value of 0.02 for the majority of datasets.

4.5 Analysis of time efficiency

To analyze the time efficiency of LDF compared to the comparison algorithms, we conducted a time efficiency analysis for all methods across all datasets. The experimental results are shown in Table 8, with the unit of measurement in seconds (s).

By analyzing the experimental results in Table 8, we can see that ECOD has the shortest average time for outlier detection across all datasets, with 0.035 s; RDOF has the longest average time, with 46.6 s. Our proposed LDF ranks seventh in time efficiency among all methods,

Table 6 Precision of different algorithms with and without the inclusion of PCA

	Variants	LOF	COF	IForest	RDOF	NaNOD	ECOD	FDPC-OF	SODEP	LDF
Ecoli	NoPCA	0.840	0.880	0.800	0.800	0.840	0.760	0.520	0.840	**0.920**
	PCA	0.840	0.880	0.800	0.800	0.840	0.680	0.480	0.840	**0.960**
Wine	NoPCA	0.900	0.900	0.100	0.900	0.900	0.100	0.600	0.800	**1.000**
	PCA	**1.000**	0.900	0.500	0.900	0.900	0.300	0.700	0.900	**1.000**
Wbc	NoPCA	**0.571**	0.524	0.524	**0.571**	0.524	0.429	0.096	0.524	**0.571**
	PCA	**0.619**	**0.619**	0.381	0.571	0.524	0.381	0.524	0.571	**0.619**
Ionosphere	NoPCA	0.825	0.825	0.675	0.667	0.738	0.520	**0.857**	0.841	0.849
	PCA	0.810	0.825	0.786	0.643	0.675	0.762	0.849	0.857	**0.865**
Wdbc	NoPCA	0.879	**0.909**	0.515	0.879	0.789	0.455	0.788	0.879	**0.909**
	PCA	0.879	**0.909**	0.697	0.879	0.758	0.424	0.394	0.879	**0.909**
Cardio	NoPCA	0.225	0.210	0.472	0.188	0.534	0.528	0.256	0.494	**0.608**
	PCA	0.233	0.210	0.5	0.193	0.483	0.511	0.239	0.460	**0.591**
Arrhythmia	NoPCA	0.515	**0.520**	0.439	0.500	0.510	0.485	0.348	0.500	0.500
	PCA	0.500	0.500	0.409	0.500	0.485	0.409	0.455	0.510	**0.515**
Waveform	NoPCA	0.210	**0.290**	0.090	0.130	0.110	0.050	0.150	0.140	0.200
	PCA	0.220	0.270	0.030	0.150	0.180	0.050	0.210	0.210	**0.300**
Satellite	NoPCA	0.417	0.432	0.559	0.410	0.583	0.449	0.580	0.598	**0.602**
	PCA	0.355	0.359	0.400	0.367	0.532	0.393	0.515	0.541	**0.584**
Satimage-2	NoPCA	0.113	0.169	0.901	0.151	**0.915**	0.620	0.239	**0.915**	**0.915**
	PCA	0.183	0.056	0.859	0.155	0.859	0.704	0.563	0.789	**0.915**

Table 7 AUC of different algorithms with and without the inclusion of PCA

	Variants	LOF	COF	IForest	RDOF	NaNOD	ECOD	FDPC-OF	SODEP	LDF
Ecoli	NoPCA	0.979	0.983	0.949	0.974	0.943	0.936	0.888	0.976	**0.985**
	PCA	0.978	**0.984**	0.944	0.978	0.956	0.942	0.872	0.967	**0.984**
Wine	NoPCA	0.999	0.999	0.766	0.999	0.998	0.733	0.913	0.997	**1.000**
	PCA	**1.000**	0.998	0.939	0.998	0.997	0.897	0.985	0.999	**1.000**
Wbc	NoPCA	0.951	0.929	0.949	0.951	0.894	0.900	0.334	0.948	**0.952**
	PCA	0.950	0.935	0.894	0.945	0.918	0.918	0.859	0.951	**0.952**
Ionosphere	NoPCA	0.903	0.920	0.852	0.821	0.858	0.728	**0.941**	0.912	0.925
	PCA	0.897	0.912	0.899	0.791	0.818	0.909	0.933	0.925	**0.944**
Wdbc	NoPCA	0.989	**0.993**	0.945	0.983	0.967	0.931	0.953	0.978	0.988
	PCA	0.980	0.976	0.953	0.970	0.954	0.890	0.712	0.988	**0.993**
Cardio	NoPCA	0.822	0.579	0.818	0.544	0.896	**0.902**	0.588	0.812	0.797
	PCA	0.840	0.577	0.906	0.586	0.842	**0.911**	0.592	0.873	0.892
Arrhythmia	NoPCA	0.816	**0.832**	0.796	0.796	0.809	0.805	0.736	0.803	0.809
	PCA	0.816	0.834	0.763	0.817	0.810	0.771	0.768	0.817	**0.835**
Waveform	NoPCA	0.742	0.756	0.725	0.657	**0.771**	0.608	0.696	0.729	0.750
	PCA	0.751	0.731	0.608	0.679	0.799	0.631	0.770	0.750	**0.789**
Satellite	NoPCA	0.572	0.570	0.704	0.565	0.704	0.583	0.733	**0.769**	0.754
	PCA	0.527	0.521	0.554	0.526	0.658	0.501	0.673	0.732	**0.783**
Satimage-2	NoPCA	0.953	0.727	0.994	0.580	0.996	0.965	0.812	**0.998**	**0.998**
	PCA	0.957	0.542	0.986	0.584	0.968	0.972	0.760	0.996	**0.997**

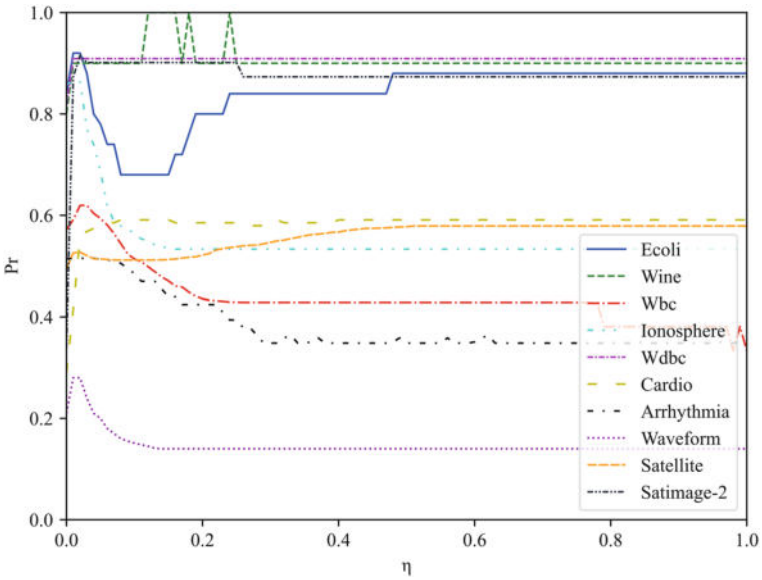

Fig. 12 Curves for precision with changing η

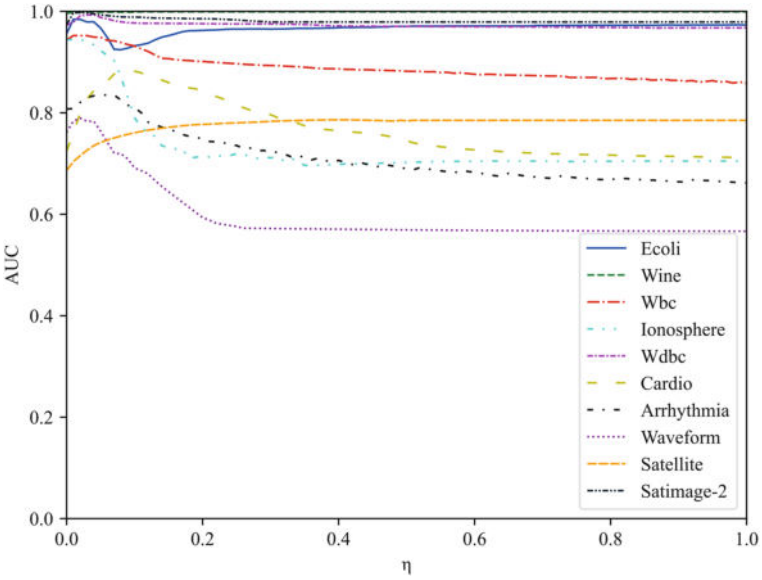

Fig. 13 Curves for AUC with changing η

with an average of 2.79 s. Although LDF does not achieve the best time efficiency, this level of time consumption is acceptable considering the improved outlier detection performance compared to the comparison algorithms.

Table 8 The time efficiency of LDF and comparison algorithms

Datasets	LOF	COF	IForest	RDOF	NaNOD	ECOD	FDPC-OF	SODEP	LDF
	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$	$O(n^3)$	$O(n \log n)$	$O(n)$	$O(n \log n)$	$O(n^2)$	$O(n \log n)$
Ecoli	0.0156s	0.0502s	0.170s	0.185s	0.109s	0.186s	0.008s	0.0827s	0.161s
Wine	0.0130s	0.0386s	0.133s	0.209s	0.111s	0.002s	0.007s	0.710s	0.123s
Wbc	0.0397s	0.122s	0.135s	3.23s	1.95s	0.005s	0.022s	0.0180s	0.360s
Ionosphere	0.034s	0.113s	0.132s	2.70s	1.83s	0.005s	0.0207s	0.0157s	0.0776s
Wdbc	0.0370s	0.131s	0.129s	3.60s	2.10s	0.005s	0.0218s	0.0140s	0.302s
Cardio	0.232s	1.07s	0.193s	38.0s	30.8s	0.0153s	0.228s	0.136s	1.77s
Arrhythmia	0.0781s	0.452s	0.168s	33.8s	25.1s	0.0510s	0.0926s	0.108s	0.0340s
Waveform	0.880s	4.73s	0.390s	147 s	118 s	0.054s	0.501s	0.361s	3.55s
Satellite	0.793s	11.4s	0.354s	127.9s	105.3s	0.014s	0.968s	1.33s	11.9s
Satimage-2	0.649s	9.22s	0.331s	109.8s	82.3s	0.0121s	0.870s	1.30s	9.59s
AVG	0.277s	2.73s	0.214s	46.6s	36.8s	0.035s	0.274s	0.408s	2.79s

5 Conclusion

In this paper, we introduce an outlier detection algorithm based on local density feedback (LDF), and address the issue of that previous density-based algorithms underutilizes neighborhood similarity and global information. The LDF algorithm introduces a density feedback mechanism to further process the local density of objects, which iteratively aggregates global information based on neighborhood similarity, and can better distinguish outliers from normal points. Experiment results show that LDF significantly improves the overall performance in terms of average accuracy by 10.7% and average AUC by 1.89% as compared to the second-best method. In the future, we aim to develop a method that dynamically adjusts the feedback rate parameter based on convergence speed during iterations, and extend the post-processing technique as a general approach, allowing it applicable to other classical algorithms to further enhance performance.

Acknowledgements This work was supported by the National Natural Science Foundation of China (No.61972334), the Innovation Capability Improvement Plan Project of Hebei Province (No.22567626H), the Local Science and Technology Development Fund Project guided by the Central Government (No.226Z1707G), the Intelligent image workpiece recognition of Sida Railway (No.x2021134), and the Performance Appraisal System Qinghuangdao Urban and Health Industry Development CO., LTD (No.x2022247).

Author Contributions Z-ZP presented the core idea of the method. H-YH implemented the model, verified its validity and wrote the paper. J-Y and Z-RB provided guidance and revised the paper.

Data Availability Data will be made available on request.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Smiti A (2020) A critical overview of outlier detection methods. Computer Science Review 38:100306. https://doi.org/10.1016/j.cosrev.2020.100306
2. Ragab M, Sabir MFS (2022) Outlier detection with optimal hybrid deep learning enabled intrusion detection system for ubiquitous and smart environment. Sustainable Energy Technol Assess 52:102311. https://doi.org/10.1016/j.seta.2022.102311
3. Negi K, Kumar GP, Raj G et al (2022) Degree of accuracy in credit card fraud detection using local outlier factor and isolation forest algorithm. In: 2022 12th International Conference on Cloud Computing, Data Science & Engineering (Confluence), IEEE, pp 240–245. https://doi.org/10.1109/Confluence52989.2022.9734123
4. Suhaila J (2021) Functional data visualization and outlier detection on the anomaly of el niño southern oscillation. Climate 9(7):118. https://doi.org/10.3390/cli9070118
5. Ma H, Wang Y, Yuan J et al (2023) Kernel-based KPI-oriented fault detection approach in the presence of outliers. IEEE Trans Industr Inf 19(10):10366–10378. https://doi.org/10.1109/TII.2022.3231934
6. ur Rehman A, Belhaouari SB (2021) Unsupervised outlier detection in multidimensional data. J Big Data 8(1):80. https://doi.org/10.1186/s40537-021-00469-z
7. Reddy A, Ordway-West M, Lee M et al (2017) Using gaussian mixture models to detect outliers in seasonal univariate network traffic. In: 2017 IEEE Security and Privacy Workshops (SPW), pp 229–234, https://doi.org/10.1109/SPW.2017.9
8. Li Z, Zhao Y, Hu X et al (2023) ECOD: unsupervised outlier detection using empirical cumulative distribution functions. IEEE Trans Knowl Data Eng 35(12):12181–12193. https://doi.org/10.1109/TKDE.2022.3159580
9. Li Z, Zhao Y, Botta N et al (2020) Copod: Copula-based outlier detection. In: 2020 IEEE International Conference on Data Mining (ICDM), pp 1118–1123. https://doi.org/10.1109/ICDM50108.2020.00135

10. Aydın F (2023) Boundary-aware local density-based outlier detection. Inf Sci 647:119520. https://doi.org/10.1016/j.ins.2023.119520
11. Evans K, Love T, Thurston SW (2015) Outlier identification in model-based cluster analysis. J Classif 32(1):63–84. https://doi.org/10.1007/s00357-015-9171-5
12. Ester M, Kriegel HP, Sander J et al (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD, pp 226–231. https://doi.org/10.5555/3001460.3001507
13. Chen C, Wang Y, Hu W et al (2020) Robust multi-view k-means clustering with outlier removal. Knowl-Based Syst 210:106518. https://doi.org/10.1016/j.knosys.2020.106518
14. Zhang Z, Li S, Liu W et al (2023) A new outlier detection algorithm based on fast density peak clustering outlier factor. Int J Data Warehous Min 19(2):1–19. https://doi.org/10.4018/IJDWM.316534
15. Riahi-Madvar M, Akbari Azirani A, Nasersharif B et al (2021) A new density-based subspace selection method using mutual information for high dimensional outlier detection. Knowl-Based Syst 216:106733. https://doi.org/10.1016/j.knosys.2020.106733
16. Breunig MM, Kriegel HP, Ng RT et al (2000) Lof: identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD international conference on Management of data, pp 93–104. https://doi.org/10.1145/335191.33538
17. Wahid A, Rao ACS (2022) RDOF: an outlier detection algorithm based on relative density. Expert Syst 39(2):e12859. https://doi.org/10.1111/exsy.12859
18. Tang J, Chen Z, Fu AWC et al (2002) Enhancing effectiveness of outlier detections for low density patterns. In: Advances in Knowledge Discovery and Data Mining: 6th Pacific-Asia Conference, PAKDD 2002 Taipei, 2002 Proceedings 6, Springer, pp 535–548. https://doi.org/10.1007/3-540-47887-6_53
19. Taunk K, De S, Verma S et al (2019) A brief review of nearest neighbor algorithm for learning and classification. In: 2019 international conference on intelligent computing and control systems (ICCS), IEEE, pp 1255–1260. https://doi.org/10.1109/ICCS45141.2019.9065747
20. Dong N, Ren B, Li H et al (2023) A novel anomaly score based on kernel density fluctuation factor for improving the local and clustered anomalies detection of isolation forests. Inf Sci 637:118979. https://doi.org/10.1016/j.ins.2023.118979
21. Zhang L, Lin J, Karim R (2018) Adaptive kernel density-based anomaly detection for nonlinear systems. Knowl Based Syst 139:50–63. https://doi.org/10.1016/j.knosys.2017.10.009
22. Latecki LJ, Lazarevic A, Pokrajac D (2007) Outlier detection with kernel density functions. In: International Workshop on Machine Learning and Data Mining in Pattern Recognition, Springer, pp 61–75. https://doi.org/10.1007/978-3-540-73499-4_6
23. Zhou Y, Xia H, Yu D et al (2024) Outlier detection method based on high-density iteration. Inf Sci 662:120286. https://doi.org/10.1016/j.ins.2024.120286
24. Aryal S, Ting KM, Haffari G (2016) Revisiting attribute independence assumption in probabilistic unsupervised anomaly detection. In: Intelligence and Security Informatics: 11th Pacific Asia Workshop. PAISI 2016, Auckland, New Zealand, April 19, 2016, Proceedings 11, Springer, pp 73–86. https://doi.org/10.1007/978-3-319-31863-9_6
25. Aryal S, Baniya AA, Santosh K (2019) Improved histogram-based anomaly detector with the extended principal component features. arXiv preprint arXiv:1909.12702 https://doi.org/10.48550/arXiv.1909.12702
26. Maćkiewicz A, Ratajczak W (1993) Principal components analysis (PCA). Comput Geosci 19(3):303–342. https://doi.org/10.1016/0098-3004(93)90090-r
27. Liu H, Zhang S, Wu Z et al (2025) Outlier detection using local density and global structure. Pattern Recogn 157:110947. https://doi.org/10.1016/j.patcog.2024.110947
28. Wang X, Duan L, Yu Z et al (2024) Robust multi-kernel nearest neighborhood for outlier detection. IEEE Trans Knowl Data Eng 36(8):4220–4231. https://doi.org/10.1109/TKDE.2024.3364179
29. Yang J, Rahardja S, Fränti P (2024) Smoothing outlier scores is all you need to improve outlier detectors. IEEE Trans Knowl Data Eng 36(11):7044–7057. https://doi.org/10.1109/TKDE.2023.3332757
30. Zhu Q, Feng J, Huang J (2016) Natural neighbor: a self-adaptive neighborhood method without parameter k. Pattern Recogn Lett 80:30–36. https://doi.org/10.1016/j.patrec.2016.05.007
31. Wahid A, Annavarapu CSR (2021) NANOD: a natural neighbour-based outlier detection algorithm. Neural Comput Appl 33(6):2107–2123. https://doi.org/10.1007/s00521-020-05068-2
32. Xiong ZY, Long H, Zhang YF et al (2023) A neighborhood weighted-based method for the detection of outliers. Appl Intell 53(9):9897–9915. https://doi.org/10.1007/s10489-022-03258-0
33. Huang J, Cheng D, Zhang S (2023) A novel outlier detecting algorithm based on the outlier turning points. Expert Syst Appl 231:120799. https://doi.org/10.1016/j.eswa.2023.120799
34. Thudumu S, Branch P, Jin J et al (2020) A comprehensive survey of anomaly detection techniques for high dimensional big data. J Big Data 7(1):42. https://doi.org/10.1186/s40537-020-00320-x

35. Riahi-Madvar M, Nasersharif B, Azirani AA (2021) Subspace outlier detection in high dimensional data using ensemble of pca-based subspaces. In: 2021 26th International Computer Conference, Computer Society of Iran (CSICC), IEEE, pp 1–5. https://doi.org/10.1109/CSICC52343.2021.9420589
36. Malladi SRS, Ram S, Rodríguez JJ (2021) Image denoising using superpixel-based PCA. IEEE Trans Multimedia 23:2297–2309. https://doi.org/10.1109/TMM.2020.3009502
37. Siddiqui F, Sargent P, Montague G (2020) The use of PCA and signal processing techniques for processing time-based construction settlement data of road embankments. Adv Eng Inform 46:101181. https://doi.org/10.1016/j.aei.2020.101181
38. Li J, Liu Z (2024) Attribute-weighted outlier detection for mixed data based on parallel mutual information. Expert Syst Appl 236:121304. https://doi.org/10.1016/j.eswa.2023.121304
39. Omohundro SM (2009) Five balltree construction algorithms. https://api.semanticscholar.org/CorpusID:61067117
40. Liu FT, Ting KM, Zhou ZH (2008) Isolation forest. In: 2008 Eighth IEEE International Conference on Data Mining, IEEE, pp 413–422. https://doi.org/10.1109/ICDM.2008.17

Zhongping Zhang received the Ph.D. in computer software and theory from Fudan University, China, in 2004. Currently, he is a professor at Yanshan University, China. His research interests include big data, data mining, and semi-structured data.

Yuehan Hou received the B.S. degree from Henan University, China, in 2018. She is currently pursuing the M.S. degree in Yanshan University, China. Her research interests include Machine learning and data mining.

Yin Jia received the M.S. degree from Yanshan University, China, in 2024. He is currently pursuing the D.R. degree in Beijing Jiaotong University. His research interests include deep learning and data mining.

Ruibo Zhang is a B.S. candidate in Wuhan University of Technology, China. His research interests include big data, finite element analysis, and topology optimization.