IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 19 July 2024, accepted 1 August 2024, date of publication 9 August 2024, date of current version 19 August 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3440915

RESEARCH ARTICLE

Self-Training Algorithm With Block Similar Neighbor Editing

WENWANG BAI[1], CUIHONG ZHANG[2], ZHENGGUO YANG[2], AND HE YANG[1]
[1]College of Mathematics and Statistics, Northwest Normal University, Lanzhou, Gansu 730070, China
[2]School of Information Engineering and Artificial Intelligence, Lanzhou University of Finance and Economics, Lanzhou, Gansu 730000, China

Corresponding author: Zhengguo Yang (yangzg@lzufe.edu.cn)

This work was supported by the Natural Science Foundation of Gansu Province under Grant 22JR5RA554.

ABSTRACT In the real world, there are only a small amount of data with labels. To make full use of the potential structural information of unlabeled data to train a better classifier, researchers have proposed many semi-supervised learning algorithms. Among these algorithms, self-training is one of the most widely used semi-supervised learning frameworks due to its simplicity. How to select high-confidence samples is a crucial step for self-training. If the misclassified samples are selected as high-confidence samples, this error will be amplified in the iterative process, which affects the performance of the final classifier. To alleviate the impact of this problem, this paper proposes a self-training algorithm with block-similar neighbor editing (STBSNE). STBSNE calculates the distance between samples by the block-based dissimilarity measure, which improves the classification performance on high-dimensional data sets. STBSNE defines the block-estimated neighbor relationship, builds the block-estimated neighbor relationship graph, and proposes the block estimated neighbor editing method to identify outliers and noise points, and edits them to improve the quality of the high-confidence sample selected. Experimental results on 16 benchmark data sets verify the superior performance of the proposed STBSNE compared with seven state-of-the-art algorithms.

INDEX TERMS Semi-supervised learning, self-training, classification, block similar neighbor, data editing.

I. INTRODUCTION

With the development of science and technology, the world has entered the era of big data. Among these huge amounts of data, labeled data accounts for a small proportion. Obtaining all labels of data requires a lot of resources, and in some cases, it is even impossible. To use the potential information of unlabeled data, researchers proposed semi-supervised learning [1], [2], [3] and self-supervised learning [4], [5], [6]. Semi-supervised classification algorithms mainly include outlier detection, graph-based, generative, and discriminative methods [7], [8], [9], [10], [11], [12], [13], [14], [15], [16].

The authors summarize 29 semi-supervised outlier detection algorithms and conduct experiments on 95 imbalanced data sets [7]. The experiments show that the BRM [8] (Bagging-Random Miner) classifier performs better than the other 28 algorithms. Semi-supervised learning algorithms based on graphs [9], [10], [11] mostly construct graphs using k-nearest Neighbor. Yuan et al. [12] proposed a semi-supervised learning algorithm via an adaptive Laplacian graph termed ALGSSL, which reconstructs the sparse graph by constructing the laplacian graph instead of directly using the initial graph. Ma et al. [13] proposed FLGSS, which extracts global and local feature information together. Cappozzo et al. [14] proposed a discriminative method termed AMDA, which employs a Gaussian mixture model to make the classifier more robust. Representative generative methods include SSFCM (semi-supervised fuzzy c-means clustering) [15] and ESFCM [16] (semi-supervised entropy regularized fuzzy c-means clustering).

Among these semi-supervised classification methods, self-training [17] has become one of the most widely used frameworks because of its simplicity. In this case, the performance of the learned classifier depends on the quality of the selected high-confidence samples. Once the misclassified samples are chosen as high-confidence samples, they will always affect

The associate editor coordinating the review of this manuscript and approving it for publication was Chao Tong.

the subsequent iteration and ultimately affect the performance of the learned classifier. Therefore, improving the quality of the selected high-confidence samples is a critical step for self-training-based algorithms.

To improve the quality of the selected high-confidence samples, researchers have proposed many methods. Li and Zhu [18] proposed a boosting self-training framework based on instance generation with natural neighbors for k nearest neighbor termed BoostSTIG, which expands the labeled set using natural neighbors to assign labels for unlabeled samples and then generates a self-training classifier using an ensemble method. Piroonsup and Sinthupinyo [19] proposed a semi-supervised self-training method, which uses semi-supervised clustering techniques to analyze the sufficiency of labeled data to improve the performance of the learned classifiers. Li et al. [20] proposed a framework based on local cores for self-labeled semi-supervised classification (LCSSC), which uses the idea of local cores [21] to improve the quality of the selected high-confidence samples in the self-training iterative process. However, these self-training algorithms do not consider the influence of the mislabeled high-confidence samples.

On the other hand, the state-of-the-art STDP [22], STDP-DE [23], SNNRCE [24], ENaN [25], ELS [26], STDPNF [27], STSFCM [28] and MLSTE [29] calculate the distance between samples using Euclidean distance, which works well in low-dimensional space, but there will be a "curse of dimensionality" in high-dimensional data sets. SDTC [30] uses the Mahalanobis distance to calculate the distance between samples. Compared with the Euclidean distance, the Mahalanobis distance takes into account the interrelationships between various characteristics of the data, but the calculation of the Mahalanobis distance is time-consuming. However, researchers have proposed that distance measures lack the critical factor dissimilarity, whereby two samples in a denser region are less similar than two samples in a lower-density region. Existing distance measurement methods such as Euclidean distance and Mahalanobis distance do not consider the critical factor of difference in the calculation process. In this article, we employ the block-based dissimilarity measure to overcome the shortcomings of Euclidean distance and Mahalanobis distance.

Through the above analysis, we propose a self-training algorithm with block-similar neighbor editing termed STBSNE, which employs a block-based dissimilarity measure to calculate the distance between samples. Especially, we propose a block-similar neighbor editing algorithm to improve the quality of the selected high-confidence samples. The main contributions of the STBSNE algorithm are as follows:

(1) We use dissimilarity measure to calculate the distance between samples and propose a novel block similar neighbors search algorithm to find the block similar neighbors of all samples.

(2) Based on the block-similar neighbors, we develop a novel Block Similar Neighbors Graph to improve the quality of the selected high-confidence sample.

(3) We propose a novel data editing method to improve the performance of a self-training algorithm, which obtains a better classifier.

(4) A large number of experiments confirm the effectiveness of the proposed algorithm.

The remainder of this paper is described as follows. Some related works are reviewed in the Section II, Section III describes the details of the proposed algorithm STBSNE, Section IV includes the experimental setting, and Section V discusses the experimental results. Section VI provides the conclusion of this paper.

II. RELATED WORK

In this section, we mainly introduce several state-of-the-art semi-supervised algorithms based on self-training. Assume that $X = \{x_1, x_2, , \cdots, x_n\}$ denotes the data set containing n samples, $x_i \in \Re^{d\times 1}$ denotes the i-th sample and d denotes the number of features. $Y = \{y_1, y_2, \cdots, y_K\}$ denotes the label set with K possible classes, y_i represents the i-th label. L denotes the labeled samples set. U represents the unlabeled samples set.

A. SELF-TRAINING

Semi-supervised learning is performed by combining information from unlabeled and labeled data. Self-training [17] is one of the typical semi-supervised learning frameworks. First, a base classifier is trained on the labeled data set, and then the high-confidence samples are selected from the unlabeled set and added to the labeled set for iterative training. The self-training algorithm is described as follows:

Self-training algorithm

Input Labeled set L, unlabeled set U
Output Classifier H
1 Initialize high-confidence set $S = \varnothing$
2 WHILE $U \neq \varnothing$ or classifier H is not stable DO
3 Train the classifier H on the labeled set L
4 Use classifier H to assign labels to the samples in unlabeled set U
5 Select some samples with pseudo-labels assigned by H from the unlabeled set U to form a high-confidence set S
6 Update $L \leftarrow L \cup S$, $U \leftarrow U - S$, $S = \varnothing$
7 END WHILE
8 Return classifier H.

Obviously, how to select high-confidence samples is a key step for self-training algorithm.

B. SETRED

SETRED [31] uses a specific data editing method to eliminate the effect of mislabeled sample points (noise points) during the iterative training. SETRED calls the edges connected between points with different class labels tangent edges by constructing the related adjacency graph. CEW(Cut Edge Weight) is added to each iteration of self-training to evaluate whether the newly labeled samples are high-confidence

samples or not, and then only the samples with high-confidence are added to the labeled data set L, and the optimized classifier H is obtained iteratively.

C. STDP-CEW

STDP-CEW [32] discovers the underlying spatial structure of the data set using the density clustering algorithm DPC to find the "previous" sample set L' and "next" sample set L'' of the labeled data set L. The "previous" and "next" unlabeled samples of all the labeled samples in L are labeled, and the "previous" and "next" samples are evaluated with high-confidence using hypothesis testing with cut-edge weights, and finally the samples with high-confidence are added to the labeled data set L, and the optimized classifier H is iteratively obtained.

D. STDPNAN

STDPNaN [33] uses an integrated classifier to improve the label prediction capability of the self-training algorithm. STDPNaN proposes a parameter-free density peak clustering algorithm DPCNaN by introducing natural nearest neighbors. DPCNaN discovers the spatial structure of the entire data set by making each sample point to its nearest sample with higher local density. STDPNaN labels the "next" and "previous" unlabeled samples of all labeled samples in L based on the data space constructed by DPCNaN, adds the labeled sample points to the set of labeled samples L, and iterates to derive the optimized classifier H.

E. BLOCK-BASED DISSIMILARITY MEASURES

Ting et al. [34] proposed a block-based dissimilarity metric, where F represents the probability density function, D represents the sample data, and $H \in \Psi(D)$ represents a hierarchical partition model that divides the space D into non-overlapping non-spatial domains. Let x_i denotes the ith sample in D, and let $I(\bullet)$ denotes the indicator function. Let $R\left(x_i, x_j \,|H; D\right)$ denotes the most minor field under H and D containing x_i and x_j:

$$R\left(x_i, x_j \,|H; D\right) = \underset{h \subset H, s.t. \{x_i, x_j\} \in h}{\arg\min} \sum_{z \in D} 1\,(z \in h) \tag{1}$$

Let $P_F(\Delta)$ denote the probability of Δ computed using the probability density function F, and let the expected probability $m\left(x_i, x_j \,|H; D\right)$ of $R\left(x_i, x_j \,|H; D\right)$ be the block-based dissimilarity of samples x_i and x_j with respect to F and D.

$$m\left(x_i, x_j \,|H; D\right) = E_{\Psi(D)}\left[P_F\left(R\left(x_i, x_j \,|H; D\right)\right)\right] \tag{2}$$

Let $H_b \in \Psi(D)\,(b = 1, \cdots, B)$ be a finite number of models, and $\widetilde{P}(R) = \frac{1}{|D|}\sum_{z \in D} I\,(z \in R)$. Then the block-based dissimilarity of samples x_i and x_j with respect to F and D is

$$m_e\left(x_i, x_j \,|D\right) = \frac{1}{B}\sum_{b=1}^{B} \widetilde{P}\left(R\left(x_i, x_j \,|H_b; D\right)\right) \tag{3}$$

The state-of-the-art self-training algorithms such as SETRED, STDPCEW and STDPNaN calculate the distance between samples using Euclidean distance. However, Euclidean distance is not suitable for high-dimensional data sets. Thus, we employ the block-based dissimilarity measure to overcome the shortcomings of Euclidean distance.

III. SELF-TRAINING ALGORITHM WITH BLOCK SIMILAR NEIGHBOR EDITING (STBSNE)

How to select high-confidence samples is a crucial step for self-training. If the misclassified samples are selected as high-confidence samples, this error will be amplified in the iterative process, which affects the performance of the final classifier. To alleviate the impact of this problem, we calculate the distance between samples by the block-based dissimilarity measure, which considers the distribution of data. This paper proposes a self-training algorithm with block-similar neighbor editing (STBSNE). STBSNE defines the block estimated neighbor relationship, builds the block-estimated neighbor relationship graph, proposes the block-estimated neighbor editing method to identify outliers and noise points, and edits them to improve the quality of the high-confidence sample selected. Next, we introduce the proposed algorithm STBSNE in detail.

A. DEFINITIONS

Definition 1 (Similar neighbors): If $m_e(x_i, x_j) < \alpha$, then x_j is a neighbor of x_i, where α is the cutoff distance, $\alpha \in [0, 1]$.

Definition 2 (Block similar neighbors): Let $MN_k(x_i)$ represent the k nearest similar neighbors of sample x_i. The block similar neighbors $MDN(x_i)$ of sample x_i is defined as:

$$MDN(x_i) = \{x_j | \left(x_i \in MN_k\left(x_j\right)\right) \&\& \left(x_j \in MN_k(x_i)\right)\} \tag{4}$$

Definition 3 (Outliers): Let $MNb(X) = \{g_1, g_2, \cdots, g_n\}$, where $MNb(x_i) = g_i$ means that there are g_i samples have the same block similar neighbor x_i. If $MNb(x_i) = 0$, then, x_i is called an outlier. The set formed by outliers is called the outlier set,

$$LQ = \{x_i \,|MNb(x_i) = 0\} \tag{5}$$

Definition 4 (Local densities and peaks): The local density of sample x_i is calculated as follows:

$$\rho_i = \sum_{j \neq i} \Delta\left(l_{ij} - \varepsilon\right) \tag{6}$$

$$\Delta(\varphi) = \begin{cases} 1, \varphi < 0 \\ 0, \varphi \geq 0 \end{cases} \tag{7}$$

In formulas (6) and (8), $l_{ij} = m_e\left(x_i, x_j\right)$, and ε represents the cutoff threshold.

The peak of sample x_i is calculated as follows:

$$\delta_i = \begin{cases} \max\left(l_{ij}\right), \forall j \neq i, \rho_j \leq \rho_i \\ \min\limits_{j:\rho_j > \rho_i}\left(l_{ij}\right), others \end{cases} \tag{8}$$

The peak δ_i defined as follows: if the point x_i has the highest density, the sample x_i is the peak value. In addition, the density of the sample x_i is not the maximum, the point with the closest distance to the x_i, whose density is greater than x_i is the peak value.

Definition 5 (Parent node and root node): We take the sample with the highest local density as the root node x_r. The parent node of the point x_i is its prototype P_i. Prototype P_i of the sample x_i is defined as:

$$P_i = x_j, \tag{9}$$

where x_j is the nearest sample with a greater density with respect to x_i.

Definition 6 (Noise sample): The neighbor label nl_i of sample x_i is

$$CM\ (i) = \arg\max_k |F_{Mk}|, \tag{10}$$

$$nl_i = y_{CM(i)}, \tag{11}$$

where F_{Mk} represents the set of samples in $MDN\ (x_i)$ with label y_k. If the label y_i of the sample x_i is different from the neighbor label nl_i, then x_i is considered a noise sample.

B. BLOCK SIMILAR NEIGHBORS SEARCH ALGORITHM

We propose a novel block similar neighbors search algorithm to find the block similar neighbors of all samples. We use the KD tree to improve the search speed of k nearest neighbors [35], [36], [37]. The proposed algorithm is summarized in Algorithm 1, where $MDN(x_i)$ represents the block similar neighbors of x_i, $RMN_k\ (x_i)$ represents the k nearest block similar neighbors of x_i, $MNb(x_i)$ represents the number of times that sample x_i appears in the block similar neighbors of another sample, which is the number of $MDN(x_i)$, $MN_\alpha\ (x_i)$ represents the similar neighbors of sample x_i, M represents the distance matrix, and α is a threshold.

Algorithm 1 generates the k Block Similar Neighbors of each sample.

Algorithm 1 Block Similar Neighbors Search Algorithm (MDNSearch)

Input Data set X, α, Distance matrix M
Output MDN
$k = 1, \forall x_i \in X, MDN(x_i) = \varnothing, MN_k\ (x_i) = \varnothing,$
$MNb(x_i) = 0, RMN_k\ (x_i) = \varnothing$
Creat a KD tree Tr
While not converge
For each x_i in X, find its k-nearest neighbors x_j by Tr
    IF $m\left(x_i, x_j\,|H; D\right) < \alpha$
        $MN_k\ (x_i) = MN_k\ (x_i) \cup \left\{x_j\right\}$
        $MNb(x_i) = MNb(x_i) + 1$
        $RMN_k\left(x_j\right) = RMN_k\left(x_j\right) \cup \{x_i\}$
    End if
End for
Calculate LQ by Equation (5)
If LQ not change
    For each x_i in X
        $MDN\ (x_i) = RMN_k\ (x_i) \cap MN_k\ (x_i)$
    End for
    break
Else
    $k = k + 1$
End if
Return MDN

Algorithm 2 Block Similar Neighbors Graph (MDSG)

Input L, U, ε, M
Output $order$
$X = [L; U], count = 1, order = 0$
For each x_i in X
    $order(x_i) = 0$, calculate the ρ_i and δ_i by Equation (6) and (8)
End for
Calculate the sample prototype with Equation (9) and construct the prototype tree PR
While $U \neq \varnothing$ DO
    For each x_i in U
        For each x_j in L
            If x_j is the prototype of x_i
                $L = L \cup x_i, U = U - x_i, order(x_i) = count,$
                $count = count + 1$
            Else If x_i is the prototype of x_j
                $L = L \cup x_i, U = U - x_i, order(x_i) = count,$
                $count = count + 1$
            End if
        End for
    End for
End While
Return $order$

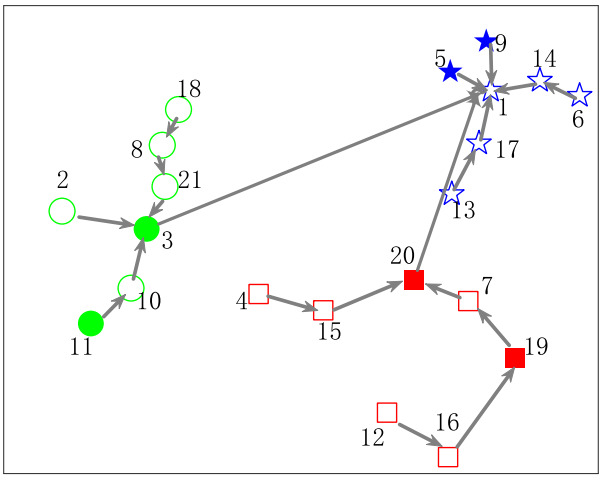
C. HIGH-CONFIDENCE SAMPLES SELECTION ALGORITHM

To improve the quality of the selected high-confidence samples, we develop a novel Block Similar Neighbors Graph algorithm termed MDSG at first. Next, we describe the proposed MDSG in detail. Let the *order* record the shortest path lengths of unlabeled samples to labeled samples with higher densities in the graph constructed by MDSG. Let ε be the cut-off threshold. Obviously, the prototype tree PR can be recursively constructed. MDSG is summarized as follows:

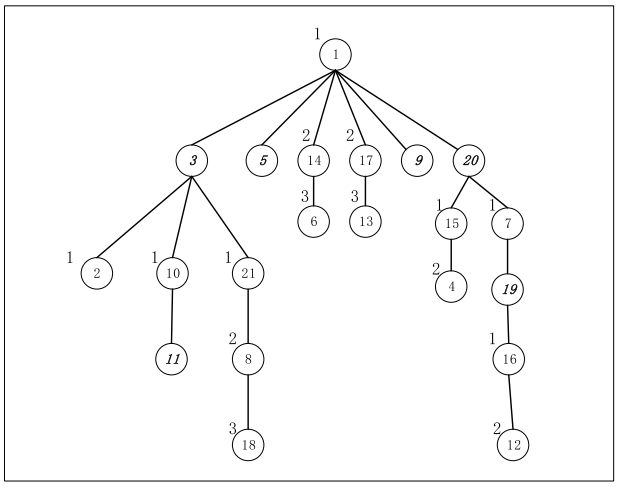
The above MDSG algorithm returns the set $order$. Here, we give an example to illustrate the construction process of Block Similar Neighbors Graph in Figure 1. Squares, circles, and five-pointed stars represent samples in class 1, class 2, and class 3, respectively. The solid ones represent labeled samples, and the hollow ones represent unlabeled samples.

Figure 1(b) is the prototype tree generated using the sample points in Figure 1(a). A circle with italicized and bold numbers represents a labeled sample, and the other circles represent unlabeled samples. The number in the upper left corner of sample x_i is $order(x_i)$.

The performance of the learned classifier by the self-training algorithm mainly depends on the quality of the selected high-confidence samples. We propose an algorithm MDNE, which compares the neighbor label of each sample with its label. If the two labels are the same, the sample is selected as a high-confidence sample, otherwise, the sample is considered as a noise and deleted. In Algorithm 3, we select the first $order(x_i)$ similar neighbors of x_i to generate a new block similar neighbors set, which is to ensure that each



(a) Relational graph



(b) Prototype tree

FIGURE 1. **Block Similar Neighbors Graph.**

unlabeled sample has a block-similar neighbors with greater density.

Algorithm 3 High-confidence samples selection algorithm(MDNE)

Input $X, MDN, order$
Output ES
$ES = \varnothing$
For each x_i in X
    Select the first $order$ block similar neighbors samples
    Calculated nl_i based on MDN by Equation (11)
End for
For each x_i in X
    If $y_i = nl_i$
        $ES = ES \cup x_i$
    End if
End for

From Algorithm 3, we can see that MDNE is used to find the noise points. If a point is identified as a noise, MDNE will delete it immediately. Thus, MDNE guarantees the quality of selected high confidence samples. This is an important highlight of this article.

D. SELF-TRAINING ALGORITHM WITH BLOCK SIMILAR NEIGHBOR EDITING (STBSNE)

As discussed above, STBSNE calculates the distance between samples by using the block-based dissimilarity measure and builds the block-estimated neighbor relationship graph. STBSNE uses the block estimated neighbor editing method to identify outliers and noise points and edits them to improve the quality of the high-confidence sample selected. The details of the STBSNE algorithm are summarized as Algorithm 4:

Algorithm 4 STBSNE algorithm

Input Labeled data L, unlabeled data U, cutoff threshold ε, distance matrix M
Output Classifier H
$X = [L; U]$, $order = \varnothing$, $TU = \varnothing$
$MDN = MDNSearch(X)$
$order = MDSG[L, U, \varepsilon, M]$
Train the classifier H on the labeled data L
$count = 1$
While $count < \max(order)$, DO 7   For each $x_i \in U$
        If $order(x_i) = count$
            $TU = TU \cup x_i$
        End if
    End for
    Use classifier H assign labels to TU
    $ES = MDNE[TU, MDN, order]$
    Update the labeled sample set $L \leftarrow L \cup ES$, unlabeled sample
    set $U \leftarrow U - ES$
    Train the classifier H on the updated labeled set L
    $count = count + 1$
End while
Return H

E. TIME COMPLEXITY ANALYSIS

In the time complexity calculation in this section, we use the same base classifier for all algorithms. Therefore, the time complexity calculation of the base classifier is not considered. Let n denote the number of samples, d denote the number of dimensionality, t denote the number of iterations and k be the number of clusters. The time complexity of the STBSNE algorithm to calculate the sample distance matrix is $O\left(dn^2\right)$. The time complexity of finding block similar neighbors is $O(n \log n)$. And the time complexity of the high-confidence sample selection algorithm is $O(n)$. So the overall time complexity of the STBSNE algorithm is $O\left(dn^2 + tn + n \log n\right)$.

IV. EXPERIMENTAL METHODOLOGY

All the experiments in the paper are conducted with 32G RAM, 64-bit Windows 10, and Inter Core i9 processor. All the codes are implemented with MATLAB 2019b. We use Accuracy as classification evaluation metrics, which can be calculated from the confusion matrix [38]. The related state-

TABLE 1. Data sets details.

Index	data set	Samples	Features	Clusters	Abbreviation
1	AR	1680	1024	120	AR
2	Australian	690	14	2	AUS
3	BUPA	345	6	2	BUP
4	Cars	392	8	3	CAR
5	Cleve	303	13	4	CLE
6	COIL20	1440	1024	20	COI
7	FERET32x32	1400	1024	200	FER
8	Heart	270	13	2	HEA
9	Solar	208	60	2	SOL
10	ORL	400	1024	40	ORL
11	Palm	2000	256	100	PAL
12	Sonar	208	60	2	SON
13	Vehicle	323	12	6	VEH
14	YaleB	2414	1024	38	YAL
15	Yeast	1484	1470	10	YEA
16	Zoo	101	16	8	Zoo

of-the-art SETRED, LSEdit, DE, STDPCEW, STDPNaN, STDP, and ENN [39] are selected as the comparison algorithms.

A. DATA SETS

The data sets used in the experiments are all public [40]. The details of these data sets are shown in Table 1.

Among in the 16 data sets, AR [41], COIL20 [42], ORL[1], Palm[2], FERET32 × 32, Yeast and YaleB [43] are all image data sets. Solar[3] is the solar flare data set. Sona is the sonar data set. The data set Yeast is a data frame of 112 observations of 50. FERET32 × 32 has 1400 samples that is a subset of FERET [44]. We also used six small data sets and the details of each data set can be found from UCI database [4].

B. DATA EDITING TECHNIQUES

This section describes the contrasting data editing techniques used in this paper.

1) DEPURATION DATA EDITING (DE)

Sanchez et al. [45] proposed a clean data editing technique named DE, which first searches the k-nearest neighbors of each labeled sample x_i to form a set N_{x_i}. If there are more than k' samples in N_{x_i} whose labels are y, then let $y_i = y$, where $\frac{(k+1)}{2} \leq k' < k$. DE modify the labels of mislabeled samples and filter noisy data during training.

2) LOCAL SETS EDITION

Li et al. [46] proposed a local set editing technology LSEdit. LSEdit constitutes a local set searching the natural neighbors of each sample, and then uses the noise factor function to evaluate whether each sample is a noise sample, and adds the edited samples with labels to the edit set to filter the noise data.

[1]http://www.uk.research.att.com/facedatabase.html.
[2]https://www.gwern.net/Crops.
[3]https://www.kaggle.com
[4]https://archive.ics.uci.edu/ml/data sets.

3) RELATIVE NEIGHBORHOOD GRAPH EDITION

Reference [47] constructed adjacent undirected graphs $G = (V, E)$, where $V = X$, and E is the set of edges. If $\forall x_k \in X, k \neq i, j$, $(x_i, x_j) \in E \Leftrightarrow \|x_i - x_j\|^2 \leq \|x_i - x_k\|^2 + \|x_j - x_k\|^2$, x_i and x_j are called graph neighbors. RNGE gives the definition of related adjacent graph edges, if $\forall x_k \in X, k \neq i, j$, $(x_i, x_j) \in E \Leftrightarrow \|x_i - x_j\| \leq \max(\|x_i - x_k\|, \|x_j - x_k\|)$.

4) CUT EDGES WEIGHT STATISTIC

Reference [48] constructed a related adjacency graph G, and the edge connected between two points with different class labels in G is called a tangent edge. For each sample x_i in G, the set of sample connected to it is called the nearest neighbor set N_i of x_i, and the samples with the nearest neighbor set N_i have the same class label. If there are multiple cut edges between a sample and its neighbors, the sample is called noise, and finally, the local cut edge weight is used for hypothesis testing.

C. EXPERIMENTAL SETTINGS

In the experiments, we choose KNN as the base classifier. We employ the accuracy as evaluation metric to evaluate the classification performance. Computing the block-based dissimilarity measure, we set the height of each tree to 8 and the total number of trees is 100. We conducted the Wilcoxon signed ranks test at the level of confidence of 95%. The symbol " +", "−", and "∼ " respectively indicate that the algorithm STBSNE proposed is significantly better, worse or equivalent with the comparison algorithms.

We select similar SETRED [31], STDP-CEW [32], STDPNaN [33] and STDP [22] algorithms for comparative experiments. To verify the denoising ability of the proposed algorithm, we conduct comparative experiments with data editing techniques ENN [39], DE [45] and LSEdit [46] under the self-training framework. The operating parameters of the comparison algorithm are set according to the original articles. And the specific conditions are as follows: in the SETRED, STDP-CEW and STDP algorithms, α is the distance interception threshold in the DPC algorithm, θ is the confidence level threshold, we set $\alpha = 2$ and $\theta = 0.1$. In algorithm DE, we set $k = 3, k' = 2$. For the STBSNE algorithm, $\varepsilon = 0.5$ and $\alpha = 0.5$. For ENN, $K = 3$.

V. RESULTS AND DISCUSSION

A. CLASSIFICATION PERFORMANCE AND ANALYSIS

In the real world, labeled data often accounts for a relatively small proportion. Therefore, according to the published papers [19], [27], [28], we randomly select 10% of the samples as training sets for experiments. To avoid the randomness of the experimental results, all experiments in this paper were carried out 50 times. The experimental results are shown in Table 2.

We can draw the following conclusions from the experimental results in Table 2:

TABLE 2. Accuracy of each algorithm on 16 data sets (mean± std).

Accuracy	DE	ENN	LSEdit	SETRED	STDPCEW	STDPNaN	STDP	STBSNE
AR	82.02(5) ±1.27	84.35(3) ±1.01	85.31(2) ±0.81	81.42(7) ±0.77	81.96(6) ±1.34	84.22(4) ±1.95	77.45(8) ±1.14	**98.79(1) ±0.06**
AUS	63.59(8) ±3.96	65.39(3) ±1.88	65.66(2) ±2.61	64.06(6) ±2.07	65.31(4) ±2.64	65.24(5) ±4.08	64.17(6) ±2.76	**70.53(1) ±1.64**
BUP	59.89(5) ±4.89	60.70(3) ±3.14	60.94(2) ±4.04	60.37(4) ±4.06	58.96(7) ±4.05	58.73(8) ±3.86	59.57(6) ±2.80	**67.88(1) ±3.76**
COI	90.45(7) ±0.92	91.22(5) ±1.41	91.64(3) ±0.78	91.61(4) ±0.83	88.23(8) ±1.75	92.14(2) ±1.17	90.96(6) ±0.83	**94.70(1) ±0.23**
FER	88.95(5) ±0.60	89.80(3) ±1.05	90.09(2) ±0.63	86.32(7) ±0.62	87.41(6) ±1.16	89.20(4) ±1.20	83.17(8) ±0.87	**99.14(1) ±0.13**
SOL	79.44(4) ±0.06	79.31(5) ±0.08	81.09(3) ±0.19	78.46(6) ±1.77	81.95(2) ±0.49	76.87(8) ±0.18	78.06(7) ±0.08	**82.95(1) ±3.42**
ORL	85.25(5) ±1.00	86.99(4) ±1.40	87.54(3) ±2.18	85.11(6) ±1.80	84.10(7) ±1.33	89.78(2) ±1.13	83.87(8) ±1.57	**96.47(1) ±0.03**
PAL	86.47(6) ±0.55	88.92(4) ±1.06	90.10(2) ±0.82	87.68(5) ±0.92	85.59(8) ±0.70	89.68(3) ±0.65	85.63(7) ±1.24	**98.66(1) ±0.02**
SON	59.39(8) ±6.32	61.72(6) ±6.73	61.96(4) ±5.43	65.03(2) ±2.53	59.54(7) ±5.29	63.53(3) ±5.08	61.78(5) ±5.35	**69.72(1) ±1.02**
VEH	69.96(7) ±1.98	70.73(3) ±1.99	70.83(2) ±1.65	70.72(4) ±1.52	73.09(2) ±1.84	70.22(6) ±0.97	70.54(5) ±0.89	**78.44(1) ±1.80**
YAL	70.10(7) ±3.18	76.26(2) ±1.10	75.09(4) ±2.26	73.06(5) ±1.51	70.50(6) ±1.20	75.88(3) ±1.78	68.97(8) ±1.33	**95.97(1) ±1.00**
CAR	65.04(8) ±5.22	68.44(4) ±3.35	68.46(3) ±3.46	67.89(6) ±2.41	69.76(2) ±3.65	68.22(5) ±3.47	66.58(7) ±4.59	**78.59(1) ±6.56**
CLE	77.12(6) ±3.67	80.01(2) ±1.37	80.80(2) ±1.39	79.12(3) ±3.48	78.16(4) ±4.64	76.63(7) ±5.65	77.82(5) ±5.81	**83.98(1) ±4.98**
HEA	63.75(2) ±2.12	62.77(4) ±5.80	62.45(6) ±4.73	62.62(5) ±2.49	59.74(8) ±5.73	62.17(7) ±3.97	63.33(3) ±3.94	**66.77(1) ±4.77**
YEA	86.09(7) ±0.46	86.22(5) ±0.17	86.22(5) ±0.05	87.47(3) ±2.34	84.60(8) ±2.75	87.51(2) ±2.46	87.20(4) ±1.97	**89.25(1) ±2.51**
Zoo	84.24(8) ±3.58	87.01(3) ±3.72	84.29(7) ±2.80	87.05(2) ±3.26	85.89(5) ±1.69	86.65(4) ±2.22	85.05(6) ±3.64	**87.13(1) ±2.73**
WSR-test	+	+	∼	+	+	+	+	N/A
AVE.acc	75.73	77.49	77.65	76.75	75.92	77.29	75.26	**84.94**
AVE.std	2.51	2.26	1.12	1.94	2.52	2.53	2.44	2.17
AVE.rank	6.13	3.69	3.25	4.69	5.63	4.56	6.19	1.00

(1) On all 16 data sets, the classification performance of the proposed algorithm STBSNE is higher than that of DE, LSEdit, ENN, SETRED, STDP-CEW, STDPNaN and STDP. The reason is the proposed STBSNE selects higher quality high-confidence samples to train the classifier. As a result, the learned classifier has a better performance.

(2) On the five image data sets of AR, COI, FER, ORL, and YAL, the classification performance of the proposed algorithm STBSNE exceeds 90%, which verifies the good classification performance of the proposed STBSNE for high-dimensional data sets.

(3) Compared with DE, ENN, SETRED, STDP-CEW, STDPNaN, and STDP, the experimental results of statistical testing at the 95% confidence level show that STBSNE can significantly improve the classification performance of the learned classifier.

(4) The proposed STBSNE has achieved good experimental results on data sets with rich types and different sizes, which proves the effectiveness of the algorithm.

B. IMPACT OF LABELED SAMPLE RATIO ON CLASSIFICATION PERFORMANCE

To estimate the impact of the proportion of labeled samples on the classification performance of the algorithm. We randomly selected the proportion of samples with labels from 10% to 90% with the step of 10 %, and conducted experiments on 16 data sets. The experimental results are shown in Figures 2 and 3.

We can draw the following conclusions from Figures 2 and 3:

(1) As the proportion of labeled samples increases, the proposed algorithm STBSNE consistently outperforms the comparison algorithms DE, LSEdit, ENN, SETRED, STDP-CEW, STDPNaN, and STDP on AR, ORL, CAR, YAL, and FER data sets, which shows the excellent classification performance of the proposed algorithm. These five data sets are all image data sets, which reflects the good performance of STBSNE for high-dimensional data sets.

(2) The proposed algorithm STBSNE obtains a higher accuracy when the proportion of data with labels is very small. Therefore, the proposed algorithm STBSNE is more suitable for the situation where there are only a few labels in a large number of data sets. Experimental results show that our proposed algorithm is more suitable for real-life scenes where the proportion of labeled data is relatively small.

(3) The classification performance of the proposed algorithm STBSNE on the ten data sets including AR, COI, HEA, BUP, ORL, PAL, CAR, SON, AUS, and YAL is relatively

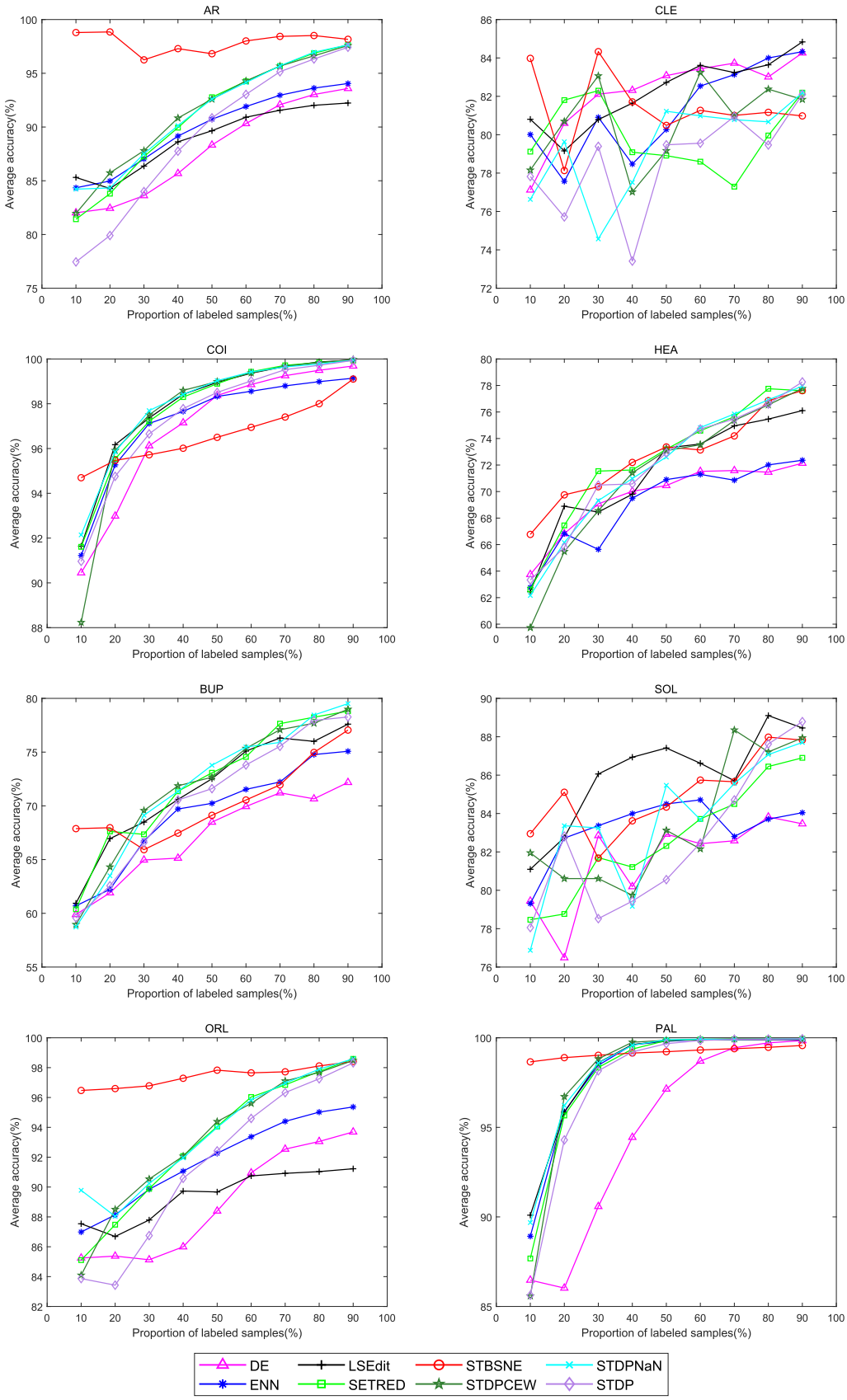


FIGURE 2. Classification performance of each algorithm under different labeled sample ratios.

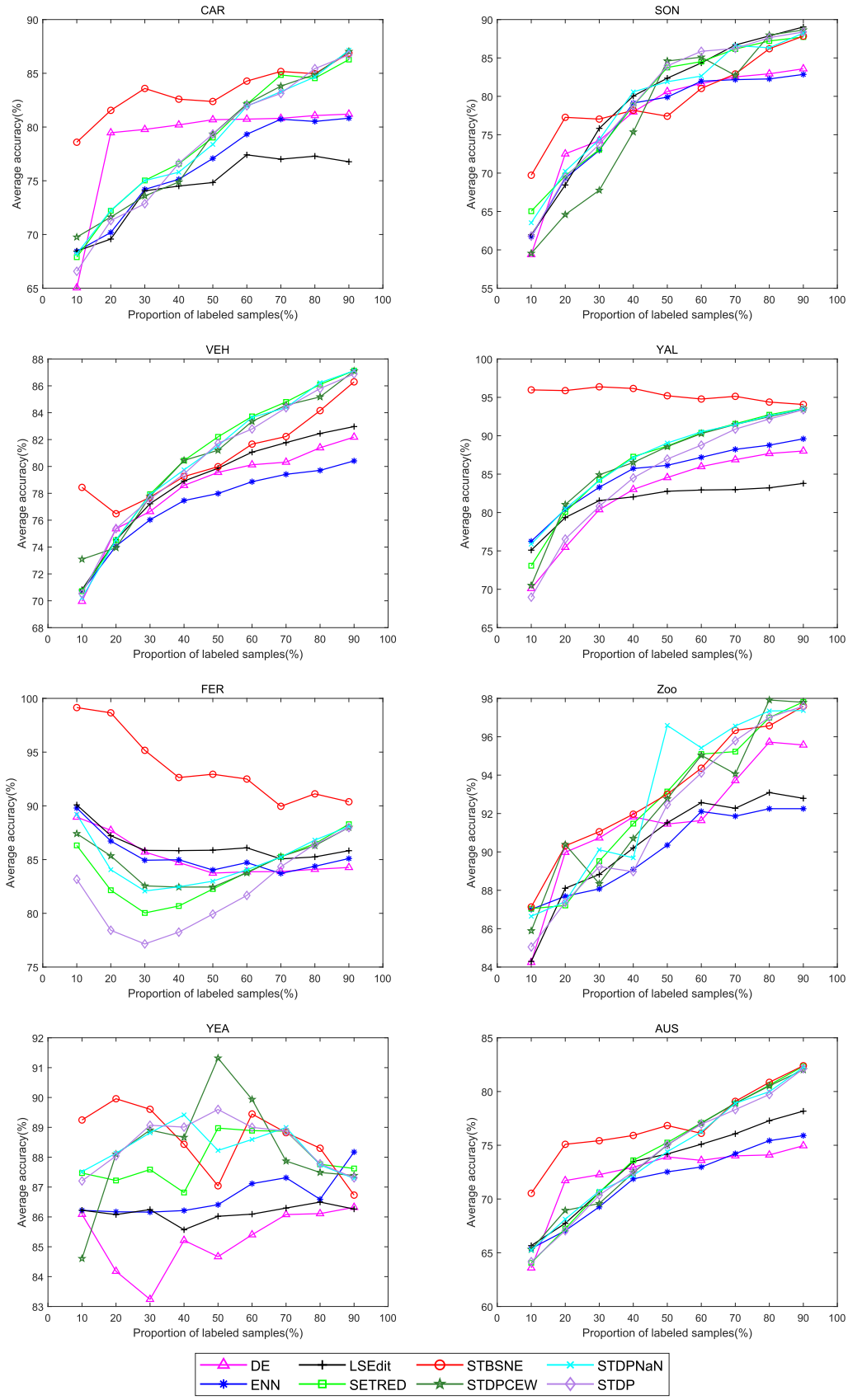


FIGURE 3. Classification performance of each algorithm under different labeled sample ratios.

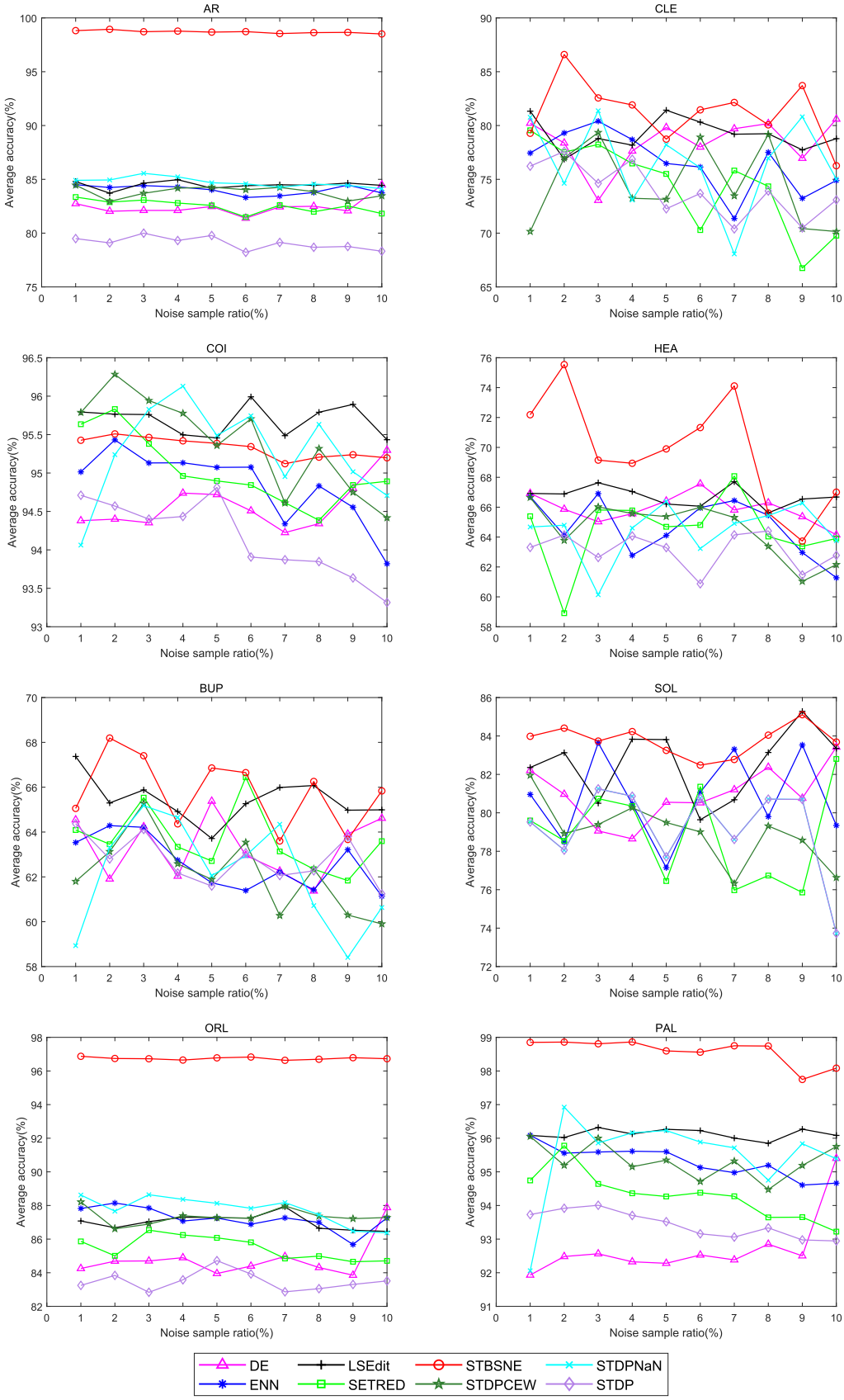


FIGURE 4. The influence of noise ratio on each algorithm.

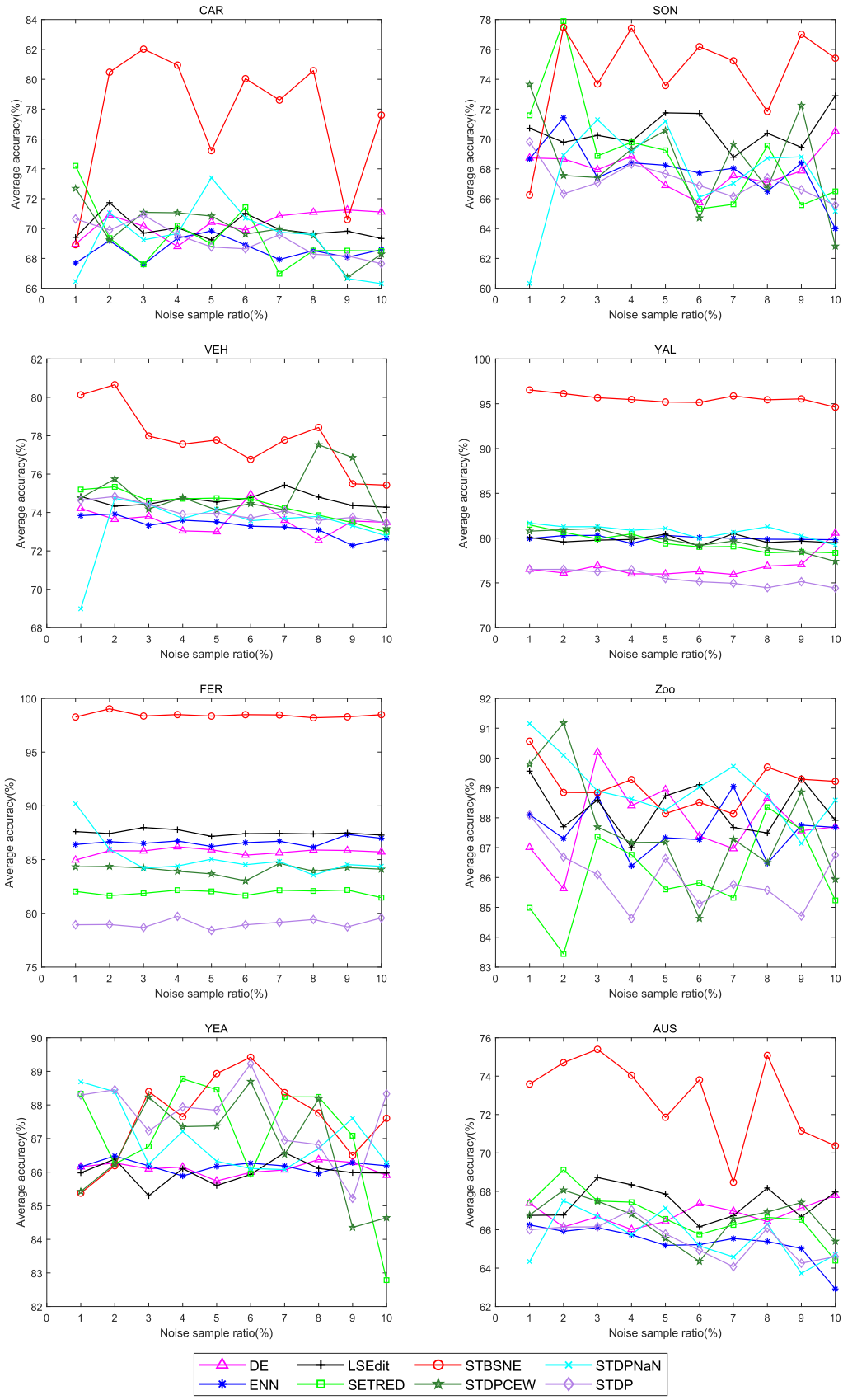


FIGURE 5. The influence of noise ratio on each algorithm.

TABLE 3. The running time of each algorithm on 16 data sets.

time	DE	ENN	LSEdit	SETRED	STDPCEW	STDPNaN	STDP	STBSNE
AR	10.16	5.64	7.89	72.82	211.58	50.73	3.95	22.74
AUS	0.13	0.04	0.18	2.31	3.65	2.83	0.15	1.37
BUP	0.03	0.01	0.05	1.37	0.73	2.24	0.10	0.63
COI	3.46	6.00	7.24	44.80	156.43	62.87	3.51	11.08
FER	4.43	3.66	5.13	55.73	104.5	36.69	2.80	31.60
SOL	0.02	0.03	0.03	1.08	0.24	0.30	0.05	0.60
ORL	0.50	0.33	0.44	7.12	4.83	4.77	0.27	2.02
PAL	7.08	2.88	3.54	45.27	313.00	40.23	1.81	9.06
SON	0.02	0.03	0.03	1.07	0.23	0.54	0.05	0.58
VEH	0.09	0.07	0.09	3.61	7.91	2.28	0.24	1.78
YAL	9.69	11.06	15.49	120.05	458.46	151.97	7.47	131.25
CAR	0.04	0.02	0.03	1.35	0.96	0.63	0.10	0.83
CLE	0.07	0.03	0.06	1.14	0.45	0.48	0.06	0.61
HEA	0.04	0.03	0.03	1.09	0.36	0.15	0.05	0.49
YEA	3.56	9.51	3.62	42.19	33.25	47.62	3.81	21.56
Zoo	0.01	0.01	0.01	1.07	0.05	0.04	0.02	0.07
AVE.time	2.46	2.46	2.74	25.13	81.04	25.27	1.53	14.77

stable as the proportion of labels increases. Under different label ratios, the classification performance curves of the STBSNE algorithm are relatively smooth, which shows the robustness of the proposed STBSNE.

C. NOISE EXPERIMENT ANALYSIS

To verify the denoising ability of STBSNE, we conducted noise experiments. We randomly select 20% of the labeled samples in the 16 data sets. Among them, [1%, 10%] of the data are given the wrong labels. Then, carry out the experiment. The accuracy is chosen as the evaluation metric of classification performance. Each algorithm was run 50 times on each data set. The experimental results are shown in Figures 4 and 5.

As can be seen from Figures 4 and 5:

(1) The classification performance of the proposed algorithm STBSNE is higher than 90% on AR, COI, FER, ORL, PAL,and YAL data sets, and higher than 80% on CLE, Zoo and YEA data sets. This can prove the good denoising ability of the proposed algorithm STBSNE. The experimental results show that the proposed editing algorithm has good denoising ability.

(2) The classification performance of the proposed algorithm STBSNE is far better than DE, LSEdit, ENN, SETRED, STDP, STDPCEW, STDPNaN and STDP algorithms on 11 data sets including AR, CLE, HEA, FER, AUS, ORL, PAL, CAR, SON, VEH and YAL, which proves the denoising performance of the STBSNE algorithm. There are seven image data sets in these 11 data sets, which shows that the STBSNE algorithm has a good performance for high-dimensional data sets.

(3) Once the misclassified samples are chosen as high-confidence samples, they will always affect the subsequent iteration, and ultimately affect the performance of the learned classifier. The result proves that the performance of STBSNE is better than the compared algorithms.

D. RUNTIME ANALYSIS

The overall time complexity of the STBSNE algorithm is $O\left(dn^2 + tn + n\log n\right)$. SETRED needs to construct related adjacency graphs and the overall time complexity is $O\left(tdn^3\right)$. The time complexity of the STDP-CEW algorithm is $O\left(tdn^3\right)$. The STDPNaN algorithm finds the spatial structure and the natural nearest neighbor using DPC with $O\left(n^2\right)$ and $O\left(n\log n\right)$, respectively. Therefore, the overall time complexity of STDPNaN is $O\left(n^2\right)$. The DE and ENN algorithms mainly search for the k-nearest neighbors of the sample, and the time complexity is $O\left(tn^2\right)$. The LSEdit algorithm is mainly to find the natural nearest neighbor NaN, and the time complexity is $O\left(n\log n\right)$. The STDP algorithm mainly uses DPC to discover the spatial structure, and the time complexity is $O\left(tn^2\right)$. We have carried out the experiments on running time, and the experimental results are listed in Table 3.

Table 3 shows that: On 16 data sets, the running time of the proposed algorithm STBSNE is higher than that of STDP, DE, ENN, LSEdit, and lower than that of STDPNaN, SETRED, and STDPCEW, which is consistent with the theoretical analysis. In general, the time complexity of the proposed algorithm STBSNE is lower than that of STDP-CEW and SETRED, and higher than the other five comparison algorithms.

VI. CONCLUSION

Most of the current semi-supervised machine learning algorithms under the self-training framework use Euclidean distance to calculate the distance between samples, which is not suitable for applications in high-dimensional data scenarios. To solve this problem, we propose a self-training algorithm with block-similar neighbor editing method STBSNE based on the block dissimilarity measure. The STBSNE algorithm uses the dissimilarity measure to calculate the distance matrix, which improves the classification performance of the algorithm in high-dimensional space. The STBSNE algorithm searches the block estimation neighbors of the samples, determines the relationship between the samples according to the local density and peak value of the samples, and builds the block estimation neighbor relationship graph. A new high-confidence sample selection

algorithm is proposed, which can not only deal with noisy data conveniently, and high-quality high-confidence samples can be selected for iterative training. Compared with other algorithms based on the self-training framework, the STBSNE algorithm has good classification performance on the data set with a labeled sample ratio of 10%. Compared with other algorithms, the classification performance of the STBSNE algorithm is not significantly improved on data sets with labeled sample data increases. Therefore, STBSNE is more suitable for the situation with less labeled data, and the comparative experiments on 16 data sets can fully demonstrate the good classification performance of the proposed algorithm STBSNE. In the noise experiment, the classification performance of the STBSNE algorithm is better than other comparison algorithms under different noise sample ratios, and the performance is stable, which shows the good data editing ability of the STBSNE algorithm.

The STBSNE algorithm mainly relies on the block to estimate the neighbor relationship graph, so the accuracy of the relationship graph construction affects the classification performance of the STBSNE algorithm. In future work, we plan to study the construction of a higher-accuracy graph and the adaptability of the STBSNE algorithm to multiple classifiers.

REFERENCES

[1] J. Liu, Y. Liu, and X. Luo, "Semi-supervised learning method," *Chin. J. Comput.*, vol. 38, no. 8, pp. 1592–1617, 2015.

[2] X. Zhou and M. Belkin, "Semi-supervised learning," *Academic Press Library Signal Process.*, vol. 1, pp. 1239–1269, Jan. 2014.

[3] X. Zhu and A. B. Goldberg, "Introduction to semi-supervised learning," *Synth. Lectures Artif. Intell. Mach. Learn.*, vol. 3, no. 1, pp. 1–130, 2009.

[4] S. Zhang, X. Zhang, S. Wan, W. Ren, L. Zhao, and L. Shen, "Generative adversarial and self-supervised dehazing network," *IEEE Trans. Ind. Informat.*, vol. 20, no. 3, pp. 4187–4197, Mar. 2024.

[5] S. Zhang, W. Ren, X. Tan, Z.-J. Wang, Y. Liu, J. Zhang, X. Zhang, and X. Cao, "Semantic-aware dehazing network with adaptive feature fusion," *IEEE Trans. Cybern.*, vol. 53, no. 1, pp. 454–467, Jan. 2023.

[6] Y. Liu, Z. Yan, T. Ye, A. Wu, and Y. Li, "Single nighttime image dehazing based on unified variational decomposition model and multi-scale contrast enhancement," *Eng. Appl. Artif. Intell.*, vol. 116, Nov. 2022, Art. no. 105373.

[7] M. E. Villa-Pérez, M. Á. Álvarez-Carmona, O. Loyola-González, M. A. Medina-Pérez, J. C. Velazco-Rossell, and K.-K.-R. Choo, "Semi-supervised anomaly detection algorithms: A comparative summary and future research directions," *Knowl.-Based Syst.*, vol. 218, Apr. 2021, Art. no. 106878, doi: 10.1016/j.knosys.2021.106878.

[8] J. B. Camiña, M. A. Medina-Pérez, R. Monroy, O. Loyola-González, L. A. P. Villanueva, and L. C. G. Gurrola, "Bagging-RandomMiner: A one-class classifier for file access-based masquerade detection," *Mach. Vis. Appl.*, vol. 30, no. 5, pp. 959–974, Jul. 2019, doi: 10.1007/s00138-018-0957-4.

[9] S. Vashishth, P. Yadav, M. Bhandari, and P. Talukdar, "Confidence-based graph convolutional networks for semi-supervised learning," in *Proc. 22nd Int. Conf. Artif. Intell. Statist.*, 2019, pp. 1792–1801.

[10] H. Cai, B. Liu, Y. Xiao, and L. Lin, "Semi-supervised multi-view clustering based on constrained nonnegative matrix factorization," *Knowl.-Based Syst.*, vol. 182, Oct. 2019, Art. no. 104798.

[11] S. Liu, C. Ding, F. Jiang, Y. Wang, and B. Yin, "Auto-weighted multi-view learning for semi-supervised graph clustering," *Neurocomputing*, vol. 362, pp. 19–32, Oct. 2019.

[12] Y. Yuan, X. Li, Q. Wang, and F. Nie, "A semi-supervised learning algorithm via adaptive Laplacian graph," *Neurocomputing*, vol. 426, pp. 162–173, Feb. 2021.

[13] J. Ma, N. Wang, and B. Xiao, "Semi-supervised classification with graph structure similarity and extended label propagation," *IEEE Access*, vol. 7, pp. 58010–58022, 2019.

[14] A. Cappozzo, F. Greselin, and T. B. Murphy, "Anomaly and novelty detection for robust semi-supervised learning," *Statist. Comput.*, vol. 30, no. 5, pp. 1545–1571, Sep. 2020.

[15] E. Yasunori, H. Yukihiro, Y. Makito, and M. Sadaaki, "On semi-supervised fuzzy c-means clustering," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Aug. 2009, pp. 1119–1124.

[16] X. Yin, T. Shu, and Q. Huang, "Semi-supervised fuzzy clustering with metric learning and entropy regularization," *Knowl.-Based Syst.*, vol. 35, pp. 304–311, Nov. 2012.

[17] O. T. Nartey, G. Yang, J. Wu, and S. K. Asare, "Semi-supervised learning for fine-grained classification with self-training," *IEEE Access*, vol. 8, pp. 2109–2121, 2020.

[18] J. Li and Q. Zhu, "A boosting self-training framework based on instance generation with natural neighbors for K nearest neighbor," *Int. J. Speech Technol.*, vol. 50, no. 11, pp. 3535–3553, Nov. 2020.

[19] N. Piroonsup and S. Sinthupinyo, "Analysis of training data using clustering to improve semi-supervised self-training," *Knowl.-Based Syst.*, vol. 143, pp. 65–80, Mar. 2018.

[20] J. Li, Q. Zhu, Q. Wu, and D. Cheng, "An effective framework based on local cores for self-labeled semi-supervised classification," *Knowl.-Based Syst.*, vol. 197, Jun. 2020, Art. no. 105804.

[21] D. Cheng, Q. Zhu, and Q. Wu, "A local cores-based hierarchical clustering algorithm for data sets with complex structures," in *Proc. IEEE 42nd Annu. Comput. Softw. Appl. Conf. (COMPSAC)*, vol. 1, Jul. 2018, pp. 410–419.

[22] D. Wu, M. Shang, X. Luo, J. Xu, H. Yan, W. Deng, and G. Wang, "Self-training semi-supervised classification based on density peaks of data," *Neurocomputing*, vol. 275, pp. 180–191, Jan. 2018.

[23] D. Wu, M. Shang, G. Wang, and L. Li, "A self-training semi-supervised classification algorithm based on density peaks of data and differential evolution," in *Proc. IEEE 15th Int. Conf. Netw., Sens. Control (ICNSC)*, Mar. 2018, pp. 1–6.

[24] Y. Wang, X. Xu, H. Zhao, and Z. Hua, "Semi-supervised learning based on nearest neighbor rule and cut edges," *Knowl.-Based Syst.*, vol. 23, no. 6, pp. 547–554, Aug. 2010.

[25] L. Yang, Q. Zhu, J. Huang, and D. Cheng, "Adaptive edited natural neighbor algorithm," *Neurocomputing*, vol. 230, pp. 427–433, Mar. 2017.

[26] S. Zhao and J. Li, "ELS: A fast parameter-free edition algorithm with natural neighbors-based local sets for K nearest neighbor," *IEEE Access*, vol. 8, pp. 123773–123782, 2020.

[27] J. Li, Q. Zhu, and Q. Wu, "A self-training method based on density peaks and an extended parameter-free local noise filter for K nearest neighbor," *Knowl.-Based Syst.*, vol. 184, Nov. 2019, Art. no. 104895.

[28] H. Gan, N. Sang, R. Huang, X. Tong, and Z. Dan, "Using clustering analysis to improve semi-supervised classification," *Neurocomputing*, vol. 101, pp. 290–298, Feb. 2013.

[29] Z. Wei, H. Wang, and R. Zhao, "Semi-supervised multi-label image classification based on nearest neighbor editing," *Neurocomputing*, vol. 119, pp. 462–468, Nov. 2013.

[30] J. Tanha, M. van Someren, and H. Afsarmanesh, "Semi-supervised self-training for decision tree classifiers," *Int. J. Mach. Learn. Cybern.*, vol. 8, no. 1, pp. 355–370, Feb. 2017.

[31] M. Li and Z.-H. Zhou, "SETRED: Self-training with editing," in *Proc. Pacific–Asia Conf. Adv. Knowl. Discovery Data Mining*, Jan. 2005, pp. 611–621.

[32] Y. Liu, "Self-training algorithm combining density peak and cut edge weight," *J. Vis. Lang. Comput.*, vol. 2020, no. 1, pp. 11–16, Jul. 2020.

[33] S. Zhao and J. Li, "A semi-supervised self-training method based on density peaks and natural neighbors," *J. Ambient Intell. Humanized Comput.*, vol. 12, no. 2, pp. 2939–2953, Feb. 2021.

[34] K. M. Ting, Y. Zhu, M. Carman, Y. Zhu, and Z.-H. Zhou, "Overcoming key weaknesses of distance-based neighbourhood methods using a data dependent dissimilarity measure," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1205–1214.

[35] K. Zhou, Q. Hou, R. Wang, and B. Guo, "Real-time KD-tree construction on graphics hardware," *ACM Trans. Graph.*, vol. 27, no. 5, pp. 1–11, Dec. 2008.

[36] S. Popov, J. Günther, H. Seidel, and P. Slusallek, "Stackless KD-tree traversal for high performance GPU ray tracing," *Comput. Graph. Forum*, vol. 26, no. 3, pp. 415–424, Sep. 2007.

[37] J. Sang, M. A. Akbar, B. Cai, H. Xiang, and H. Hu, "Joint image compression and encryption using IWT with SPIHT, Kd-tree and chaotic maps," *Appl. Sci.*, vol. 8, no. 10, p. 1963, Oct. 2018.
[38] J. T. Townsend, "Theoretical analysis of an alphabetic confusion matrix," *Perception Psychophysics*, vol. 9, no. 1, pp. 40–50, Jan. 1971.
[39] D. L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-2, no. 3, pp. 408–421, Jul. 1972.
[40] N. Asuncion and A. David. (2007). *UCI Machine Learning Repository*. [Online]. Available: https://archive.ics.uci.edu/ml/datasets.php
[41] A. Martinez and R. Benavente, "The AR face database," 1998, vol. 24. [Online]. Available: https://www.researchgate.net/publication/243651904_The_AR_face_database
[42] S. A. Nene, S. K. Nayar and H. Murase, "Columbia image object library (COIL-20)," Tech. Rep. CUCS-005-96, Feb. 1996. [Online]. Available: https://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php
[43] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 6, pp. 643–660, Jun. 2001.
[44] W. Zhao, R. Chellappa, and A. Krishnaswamy, "Discriminant analysis of principal components for face recognition," in *Proc. 3rd IEEE Int. Conf. Autom. Face Gesture Recognit.*, Jun. 1998, pp. 73–85.
[45] J. Sanchez, "Analysis of new techniques to obtain quality training sets," *Pattern Recognit. Lett.*, vol. 24, no. 7, pp. 1015–1022, Apr. 2003.
[46] J. Li, Q. Zhu, and Q. Wu, "A parameter-free hybrid instance selection algorithm based on local sets with natural neighbors," *Int. J. Speech Technol.*, vol. 50, no. 5, pp. 1527–1541, May 2020.
[47] J. S. Sánchez, F. Pla, and F. J. Ferri, "Prototype selection for the nearest neighbour rule through proximity graphs," *Pattern Recognit. Lett.*, vol. 18, no. 6, pp. 507–513, Jun. 1997.
[48] F. Muhlenbach, S. Lallich, and D. A. Zighed, "Identifying and handling mislabelled instances," *J. Intell. Inf. Syst.*, vol. 22, no. 1, pp. 89–109, 2004.

WENWANG BAI is currently pursuing the M.S. degree in statistics with the College of Mathematics and Statistics, Northwest Normal University, Lanzhou, Gansu, China.

His research interests include applied statistics and machine learning.

CUIHONG ZHANG is currently pursuing the M.S. degree in management science and engineering with Lanzhou University of Finance and Economics, Lanzhou, China. Her research interests include machine learning and artificial intelligence.

ZHENGGUO YANG received the B.S. degree in mathematics from Zhengzhou University, Zhengzhou, China, in 2010, and the M.S. and Ph.D. degrees from the School of Mathematics, University of Chinese Academy of Sciences, Beijing, China, in 2013 and 2016, respectively.

He is currently an Associate Professor with the College of Information Engineering, Lanzhou University of Finance and Economics, Lanzhou, China. His research interests include machine learning and artificial intelligence.

HE YANG received the B.S., M.S., and Ph.D. degrees in mathematics and applied mathematics from Northwest Normal University, Lanzhou, China, in 2005, 2008, and 2011, respectively.

From 2014 to 2016, he was a Postdoctoral Researcher with the Department of Mathematics, Shanghai Jiaotong University, Shanghai, China. From 2016 to 2017, he was a Visiting Scholar with the Department of Mathematics, Texas A&M University Kingsville, TX, USA. He is currently a Professor with the School of Mathematics and Statistics, Northwest Normal University. His research interest includes nonlinear analysis and its applications.