Complex & Intelligent Systems (2023) 9:5063–5080
https://doi.org/10.1007/s40747-023-00982-z

ORIGINAL ARTICLE

Structure-enhanced pairwise feature learning for face clustering

Shaoying Li[1,2] · Jie Li[3] · Bincheng Wang[1,2] · Wei Yao[1,2] · Bo Liu[1,2]

Received: 16 October 2022 / Accepted: 22 January 2023 / Published online: 1 March 2023

Abstract
Face clustering groups massive unlabeled face images according to their underlying identities and has proven to be a valuable tool for data analysis. Most recent studies have utilized graph convolutional networks (GCNs) to explore the structural properties of faces, thereby effectively achieving improved clustering performance. However, these methods usually suffer from computational intractability for large-scale graphs and tend to be sensitive to some postprocessing thresholds that serve to purify the clustering results. To address these issues, in this paper, we consider each pairwise relationship between two samples as a learning unit and infer clustering assignments by evaluating a group of pairwise connections. Specifically, we propose a novel clustering framework, named structure-enhanced pairwise feature learning (SEPFL), which mixes neighborhood information to adaptively produce pairwise representations for cluster identification. In addition, we design a combined density strategy to select representative pairs, thus ensuring training effectiveness and inference efficiency. The extensive experimental results show that SEPFL achieves better performance than other advanced face clustering techniques.

Keywords Face clustering · Pairwise relationship · Neighborhood mixing · Density clustering

Introduction

Benefiting from the development of deep learning and the emergence of large-scale face datasets [9,14,16,48], face recognition techniques have made significant progress in recent years [6,15,20,24,35]. However, manually annotating a massive number of face images is time consuming and labor intensive. To alleviate these limitations, several face clustering methods have been proposed [29,36,37,41,42]. These approaches not only provide high-quality pseudo-labels for model learning but also extend their applications from family photo album management [46] to automatic data cleaning [25,45].

Most traditional clustering algorithms have relatively strict assumptions. For example, K-means [22] requires specifying the number of clusters, spectral clustering favors balanced clustering results, and density-based spatial clustering of applications with noise (DBSCAN) [7] may not work in cases with large cluster density variations. While hierarchical clustering methods [18,26,47] generate clusters with arbitrary shapes and numbers, they are not suitable for large-scale face datasets due to the high complexity of the proximity matrix computation in each iteration.

Recently, researchers have attempted to incorporate a small amount of supervised information into the clustering process, resulting in several supervised face clustering methods [19,29,41]. Considering the nature of clustering, those methods do not yield clustering results in a straightforward manner. Instead, they seek to generalize the structural knowledge learned from labels, such as sample density [8,19,41] and inter-sample connectivity [27,29,37], to unknown observations. Thus, the key issue is, given partial labels, how to effectively and efficiently model these structural characteristics. Due to the great power of propagating and aggregat-

✉ Bo Liu
boliu@hebau.edu.cn

Shaoying Li
lishaoying000@gmail.com

Jie Li
lijie_hebau@163.com

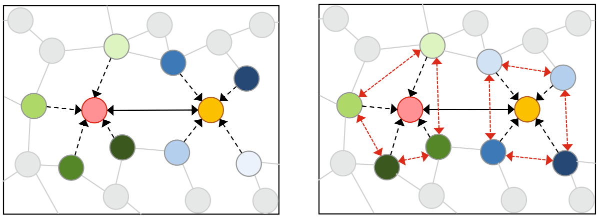
Bincheng Wang
vividbingo37@gmail.com

Wei Yao
yaowei@hebau.edu.cn

[1] Hebei Agricultural University, College of Information Science and Technology, Baoding, Hebei, China

[2] Hebei Key Laboratory of Agricultural Big Data, Hebei Agricultural University, Baoding, Hebei, China

[3] Experiment and Training Center, Hebei Agricultural University, Baoding, Hebei, China



(a) Conventional pairwise features **(b)** Enhanced pairwise features

Fig. 1 The motivation of our approach. **a** Conventional pairwise features mainly focus on neighbor-to-center properties. **b** Our enhanced pairwise features further encode intra-neighbor relationships

ing information over graphs, graph convolutional networks (GCNs) [38] are widely used to describe intra-cluster and inter-cluster relationships [2,13,17], and they significantly improve the performance of face clustering [29,36,37,41]. However, GCN-based approaches bring excessive computational cost and memory consumption. While some sampling strategies [37,40] have been proposed to improve scalability, they also suffer from overlapping subgraphs [37] and a lack of global views [40]. Moreover, one or more thresholds are usually employed in the post-processing phase to remove noisy edges, thus limiting the adaptability of these methods.

Concurrently, following the link prediction approach [3,43,44], some methods uncover clustering patterns from pairwise relationships [19], which can alleviate the influences of redundant and noisy connections in graphs [41]. Specifically, when constructing the pairwise features of two target faces, their k-nearest neighbor (k-NN) features are also involved to improve contextual awareness (as shown in Fig. 1(a)). However, the explicit exchange of information among neighbors is usually ignored during pairwise learning, but this process also plays an essential role in outlining local variations. This fact is highlighted in Fig. 1(b).

To resolve these problems, in this paper, we propose a structure-enhanced pairwise feature learning method (SEPFL) for face clustering. Unlike existing methods that employ some trivial functions to aggregate the neighbors around each sample [19], in SEPFL, the relationships between a sample and its neighbors, as well as the inherent intra-neighbor patterns, are jointly encoded to adaptively weight the candidate neighbors. Thus, the elements of the pair features ensure that the local relationships are sufficiently mixed, which is particularly beneficial for clustering learning.

Moreover, the two samples forming a pair should have some differences to characterize the local structural changes. Thus, in terms of density, a density gap should be present between them. To achieve this goal, a combined density is proposed to enhance the density decay from high-density samples, usually cluster centers, to low-density samples, such as boundary samples or outliers. Guided by the combined density, many redundant pairs are discarded, resulting in both clustering performance and inference efficiency improvements.

The main contributions of this paper are summarized as follows.

1. We propose a novel face clustering framework that performs data grouping at the pair level. Compared to graph-based approaches, our framework incorporates pairwise feature learning for connectivity classification, reducing the computational cost and alleviating the dependence on thresholds in the inference phase.
2. We propose a neighborhood mixing mechanism for learning structure-enhanced pairwise representations, in which information interchange among neighbors is effectively promoted to characterize the local structural variations.
3. We design a combined density strategy to assist in selecting more representative pairs for both training and inference, thus further improving the clustering accuracy of our method.
4. Our method is more competitive than other advanced methods and demonstrates effectiveness in other clustering tasks, such as fashion clustering.

The paper is organized as follows. In Related work, we briefly introduce the related work on face clustering regarding the aspect of whether supervision information is involved. In Methodology, we present the innovation points of this paper in detail. Experiments such as comparisons with different face clustering algorithms are presented in Experiments. Conclusion is the conclusion of this paper.

Related work

Due to their restrictive assumptions regarding data distribution and scalability issues, traditional clustering methods [7,11,22] are unable to be directly applied for large-scale face clustering. Therefore, we briefly review unsupervised and supervised face clustering methods in this section.

Unsupervised face clustering

Hierarchical clustering can handle complex data distributions and does not require the number of clusters to be specified in advance, resulting in the proposal of a series of methods that group faces in an agglomerative manner. Lin et al. [18] proposed a proximity-aware hierarchical clustering (PAHC) method, which separates positive and negative samples in the feature space to simplify the subsequent clustering procedure; however, this approach is inadequate when dealing with unbalanced clustering sizes. By introducing neighborhood structures, Zhu et al. [47] proposed a rand-order (RO) distance metric to alleviate the nonuniform distribution issues that are typically related to face data. As an extended version of the RO metric, Otto et al. [26] adopted an approximation mechanism for faster nearest-neighbor searching, which supports the clustering of millions of faces. The above methods usually work under transductive settings, leading to model retraining when new faces are encountered. Therefore, some recent methods have taken inductive or supervised approaches for face clustering.

Supervised face clustering

Supervised face clustering leverages the structural descriptions learned from labels to guide the clustering process for unknown samples. These methods can be roughly divided into two categories: two-stage methods and one-stage methods.

Two-stage methods

The two-stage approaches usually follow a coarse-to-fine procedure, in which the first stage provides an overall view of the input data, commonly in the form of a full graph or multiple subgraphs. Each vertex in the graph represents a face, and the edges indicate the relationships between pairs of vertices. Ideally, two faces belonging to the same identity are linked by an edge. The second stage gradually refines the relationships to discard the influences of noisy vertices and outliers. For instance, consensus-driven propagation (CDP) [42] first builds a multiview representation of the given data based on k-NN graphs through a base model and several proposed committee models, which are then merged into a single graph for label propagation. Most two-stage methods use GCNs as the base feature extractors due to their powerful representation capabilities. Wang et al. devised an L-GCN method [37] that takes a subgraph as a learning unit. It first utilizes a GCN to reason the linkage likelihood of a pivot and its nearest neighbors and then obtains consistent clustering results by merging the predicted links of multiple subgraphs. Among its variants, Qi et al. proposed a residual GCN (RGCN) [27] method, which cascades multiple GCN blocks with residual connections. Yang also invented a two-stage clustering framework; the first stage, called GCN-D, detects high-quality cluster proposals with high recall and purity, and the second stage, called GCN-S, refines the proposals by removing outliers [40]. Similarly, in a later work [41], the clustering process was divided into two sequential subtasks, GCN-V, which predicted node confidence, and GCN-E, which estimated edge connectivity. Notably, GCN-V was trained on the entire graph rather than subgraphs, resulting in less learning bias. However, only a one-layer GCN was deployed to reduce the computational cost, limiting the representation power of the GCN. Guo et al. designed a density-aware feature embedding network (DA-Net) [8], which consisted of two subnetworks, one based on a GCN and one based on a long short-term memory (LSTM) unit. The first subnetwork is used to enable information propagation over each subgraph, and the second subnetwork covers remote dependencies via a proposed density chain module. To improve the graph quality, Wang et al. [36] transformed the input samples into a structured space with fewer noise edges and identified the candidate neighbors of each vertex with a designed adaptive filter module.

In short, these two-stage-based algorithms not only achieve improved clustering performance but also provide richer structural descriptions, including the quality of graphs [40], the centrality of points [41], the connectivity of edges [27,37], and the more discriminative feature representations [36], which can support further decision making. However, these methods also have the following drawbacks. First, the use of multiple phases or subtasks may lead to suboptimal results, in addition to the fact that they require more hyperparameters to be tuned. Second, although GCNs have shown advantages in aggregating features, they significantly increase the computational burden. Even though the resource usage can be reduced by sampling subgraphs, additional mechanisms need to be designed for subgraph merging.

One-stage methods

Recently, by designing sampling strategies at different scales, some single-stage face clustering methods have been investigated to simplify the training processes of two-stage techniques. The structure-aware face clustering (STAR-FC) algorithm [29] takes clusters as the smallest sampling units instead of points and achieves large-scale GCN training while preserving the important structural information of the entire graph. In addition, STAR-FC takes a full graph as input to ensure inference efficiency [19] performs face clustering at the pair level, and it uses a breadth-first search (BFS) algorithm to deduce the final clustering results from the predicted pairwise relationships. Pair-based approaches have demonstrated competitive performance in face clustering. However, how to learn effective pair representations and how to construct pairwise data that are less subject to structural biases are topics have not been effectively explored.

Methodology

As previously stated, existing methods for learning pairwise embeddings usually ignore the relationships among neighbors. Therefore, we propose an SEPFL method that employs a neighborhood mixing module to better capture structural properties. Moreover, a combined density strategy, which

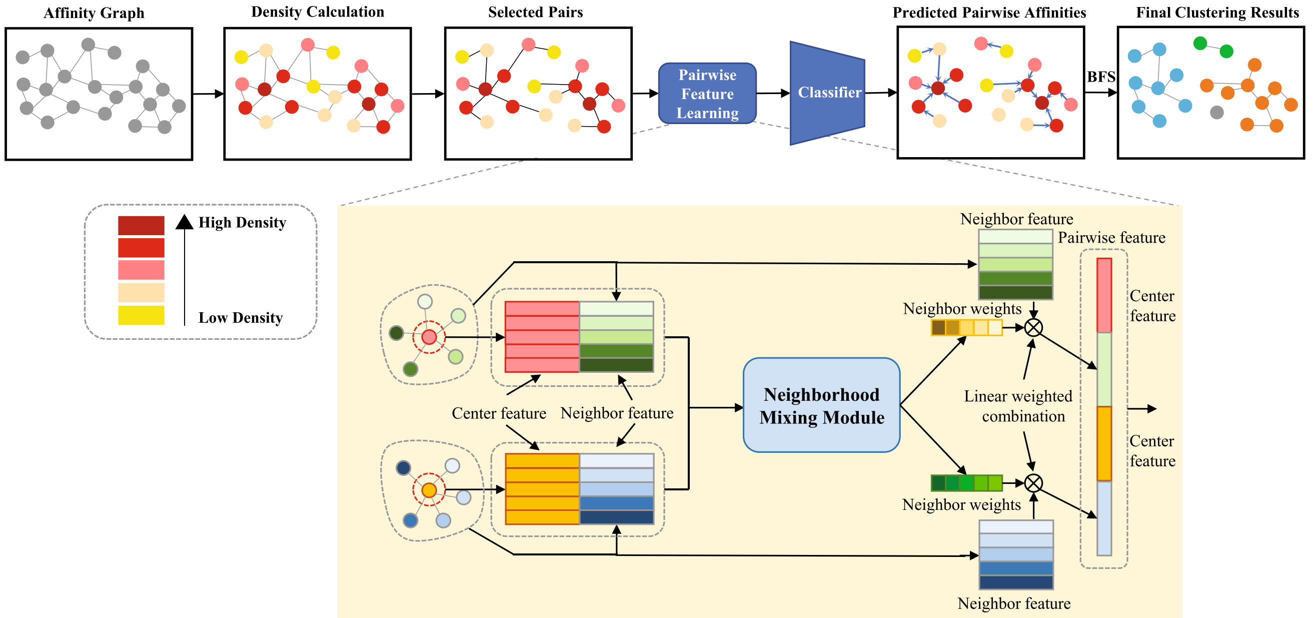


Fig. 2 Overview of the proposed SEPFL framework. First, we calculate the combined density of each sample based on a k-NN graph, which is used to select representative sample pairs. Then, these selected pairs are fed into a neighborhood mixing module where structure-enhanced pairwise representations are derived. Next, by utilizing a binary multilayer perceptron (MLP) classifier, two samples are directly assigned to one or more clusters. Finally, we perform a breadth-first search (BFS) over the selected relationships, which produces the output clustering results

selects more representative pairs for training and testing, is further proposed. The entire framework is shown in Fig. 2.

For a given face dataset with C identities, image features are extracted by a pretrained convolutional neural network (CNN) [6] and normalized to a set $\mathcal{F} = \{f_i\}_{i=1}^{N}$, where $f_i \in \mathbb{R}^D$ is the i-th face, N is the total number of samples, and D denotes the dimensionality of the features. To obtain a broad view of $\mathcal{F}$, a k-NN graph is constructed based on cosine similarity. The goal of face clustering is to assign a unique pseudo-label y' to each cluster, where $y' \in \{1, 2, \ldots, C'\}$ and C' denotes the number of predicted clusters.

Ideally, all faces associated with the same identity should be grouped together. Therefore, a variety of metrics are introduced to measure the gap between the predicted clustering results and the ground-truth labels [1,30].

Structure-enhanced pairwise feature learning

Learning better structural descriptions is essential for face clustering. To handle large-scale clustering problems, most existing methods adopt GCNs to learn the structural patterns of predivided subgraphs, which are typically based on vertex confidence [41], edge or subgraph connectivity [29], and so on. Subsequently, the patterns obtained from the subgraphs are fused to restore the clustering results. However, this multistage cluster generation schema increases the computational cost of the overall method, and each stage may also introduce some hyperparameters. For instance, Table 1 summarizes the main hyperparameters used in the different stages of several face clustering algorithms. The k-NN method provides the initial structure of the input data for all the mentioned methods. In addition, some hyperparameters are applied at different stages to control the sparsity [40], connectivity [36,37] or randomness [29] of the graphs. Generally, at least one cutting threshold is applied during the postprocessing step to eliminate the noisy connections between samples or clusters [29,36,37,41]. As a result, these additional hyperparameters may affect the generalizability and scalability of the associated methods.

Based on the above observations, instead of graphs, we use more primitive structural descriptions, i.e., pairwise relationships, as learnable units. Specifically, a single-stage clustering framework that integrates pairwise feature learning and pairwise relationship classification is proposed. A binary MLP classifier is used to directly predict whether two samples belong to the same cluster. Consequently, according to these filtered positive pairs, each connected component found by a simple search strategy can be taken as a cluster.

The key challenges of the proposed approach are twofold. First, the simplicity of the paired structure comes at the cost of limited representational power. A pairwise relationship is dominated by the two samples contained in it, leading to a lack of local structure perception. To solve the representation problem, we propose a neighborhood mixing approach for pairwise feature learning. Details are provided in “Neighborhood mixing module section”. Second, while pairwise

Table 1 Comparison of the main hyperparameters adopted by different methods

Methods	Preprocessing	Stage 1	Stage 2	Postprocessing
L-GCN [37]	K: Number of neighbors	(K_1, K_2): Sizes of 1-hop and 2-hop neighbors		max_{sz}: Maximum number of samples in each cluster $step$: A threshold for removing weak edges
LTC [40]	K: Number of neighbors	e^{τ}: A threshold for removing weak edges $smax$: Maximum number of samples in each subgraph		
GCN(V+E) [41]	K: Number of neighbors		ρ: Portion of vertices for GCN-E	τ: A threshold for removing weak edges
STAR-FC [29]	K: Number of neighbors	(M,N): Numbers of seeds and clusters, respectively (K_1,K_2): Cluster randomness and sample randomness, respectively		τ_1: A threshold for removing weak edges τ_2: A threshold for removing edges with low node intimacy
Ada-NETS [36]	K: Number of neighbors	δ: A parameter for the Huber loss β: Q-value parameter	β_1,β_2,λ: Parameters for the Hingle loss	θ: A threshold for preserving high-confidence edges
CFPC [19]	K: Number of neighbors	p: A parameter of the weighting function		
Ours	K: Number of neighbors	θ: Spherical density radius		

structures allow for greater flexibility, they give rise to large amounts of redundant relationships. For example, for N samples, at most $N \times (N-1)/2$ pairs can be extracted from them. To ensure the sufficiency of the training data and the efficiency of the inference process, we propose a density-guided pair selection strategy for the construction of candidate pairs. The mechanism of this strategy is presented in "Density-guided pair selection section".

Neighborhood mixing module

As a key component of the GCN, neighborhood aggregation captures each center node's contextual information to enhance its local structure awareness. For the same reason, combining pairwise features and their neighborhood embeddings is necessary to classify pairwise relationships. Specifically, given a pair of feature vectors, f_a and f_b, as well as their neighborhood vectors, $f_{\mathcal{N}_a}$ and $f_{\mathcal{N}_b}$, the combined pair representation is defined as:

$$f_{ab} = [f_a, f_{\mathcal{N}_a}, f_b, f_{\mathcal{N}_b}] \tag{1}$$

where [,] is the concatenation operation, and $\mathcal{N}_a$ is the set of neighbors of f_a.

Leaving linear transformations and nonlinear activations aside, a common aggregation mechanism is to calculate the weighted combination of the neighboring features of f_a:

$$f_{\mathcal{N}_a} = \sum_{i \in \mathcal{N}_a} w_{ai} f_i \tag{2}$$

where w_{ai} denotes the contribution of the ith neighbor in $\mathcal{N}_a$ to f_a. Therefore, the effectiveness of the aggregator is mainly dependent on the weighting scheme and the representation power of the feature vectors of the neighboring sample. Table 2 summarizes some typical weight setting methods, which are based on neighborhood sizes [37], attention coefficients [34], the distance decay between two samples [19], etc. These weight values are only learned through center-neighbor pairs, which may fail to reflect complex similarity relationships. While some approaches perform multiplication pooling over each neighbor pair [32], more collaborative relationships among the neighbors should be explored. To leverage local interactions for weight learning, we design a module for neighborhood mixing. It consists of three steps: center-neighbor relation embedding, neighborhood information mixing and neighboring weight generation (as shown in Fig. 3). Each of these three steps is elaborated below.

Center-neighbor relation embedding: In this step, we focus on learning the relations between a center and its first-order neighbors. First, a feature matrix $X_i \in \mathbb{R}^{K \times 2D}$ is built, where the jth row is the concatenated result of f_i and its jth neighbor of size K. To obtain primitive relational representations, X_i is passed to an MLP consisting of two fully connected layers and a nonlinear activation function, which is followed

Table 2 List of different weight setting methods

Methods	Weight formula	Aggregation formula
GCN [37]	$w_j = \frac{1}{deg(i)}$ or $w_j = \frac{1}{\sqrt{deg(i)}\sqrt{deg(j)}}$	$f_i^{agg} = \sum_{j=0}^{K} w_j f_j$
Similarity aggregation [41]	$w_j = Similarity(f_i, f_j)$	
GAT [34]	$w_j = softmax(\sigma\left(\mathrm{a}^T\left[Wf_i \vert\vert Wf_j\right]\right))$	
NIA-GCN [32]	$w = \frac{1}{K}$	$f_i^{agg} = \sum_{j=0}^{K} \sum_{k=0}^{K} w_{jk} f_j \odot f_k$ *
Rank weight aggregation [19]	$w_j = (k - j)^p$	$f_i^{agg} = \sum_{j=0}^{K} w_j f_j$
Ours	$w_j = NMM(f_i, f_j)$**	

* $\odot$ is the element-wise multiplication operator
** NMM stands for the proposed neighborhood mixing module

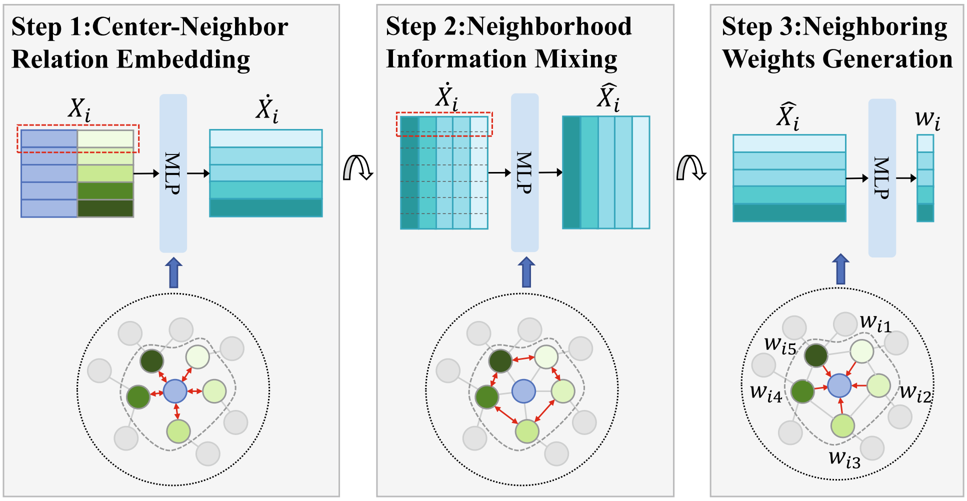

Step 1:Center-Neighbor Relation Embedding
X_i
MLP
$\dot{X}_i$
Step 2:Neighborhood Information Mixing
$\dot{X}_i$
MLP
$\hat{X}_i$
Step 3:Neighboring Weights Generation
$\hat{X}_i$
MLP
w_i
w_{i1}
w_{i2}
w_{i3}
w_{i4}
w_{i5}

Fig. 3 Neighbor weight learning procedure via the neighborhood mixing module

by a batch normalization (BatchNorm) layer for stable optimization. In addition, a skip connection is added between the input and the output of the MLP. The whole process can be written as follows:

$$\dot{X}_i = X_i + BatchNorm(\sigma(X_i W_1)W_2) \tag{3}$$

where σ is the Gaussian error linear unit (GELU) activation function [10], and $W_1 \in \mathbb{R}^{2D\times 4D}$ and $W_2 \in \mathbb{R}^{4D\times 2D}$ are two learnable weight matrices.

Neighborhood information mixing: When measuring the importance of one neighbor point $j \in \mathcal{N}_i$ to the center point i, it is not sufficient to exchange information between the two members of the pair. Indeed, all the neighbors in $\mathcal{N}_i$ should be considered. To achieve this goal, we devise a neighborhood mixing mechanism, which was initially introduced in [33] to support information communication along the spatial dimension. Specifically, we assign a similar MLP that is fed with the output feature matrix $\dot{X}_i$ of the previous step and aggregate features across neighbors as follows:

$$\hat{X}_i = \dot{X}_i + BatchNorm(W_4\sigma(W_3\dot{X}_i)) \tag{4}$$

where $W_3 \in \mathbb{R}^{2K\times K}$ and $W_4 \in \mathbb{R}^{K\times 2K}$. As shown in 3, the neighborhood interactions are encoded along each row of $\dot{X}_i$.

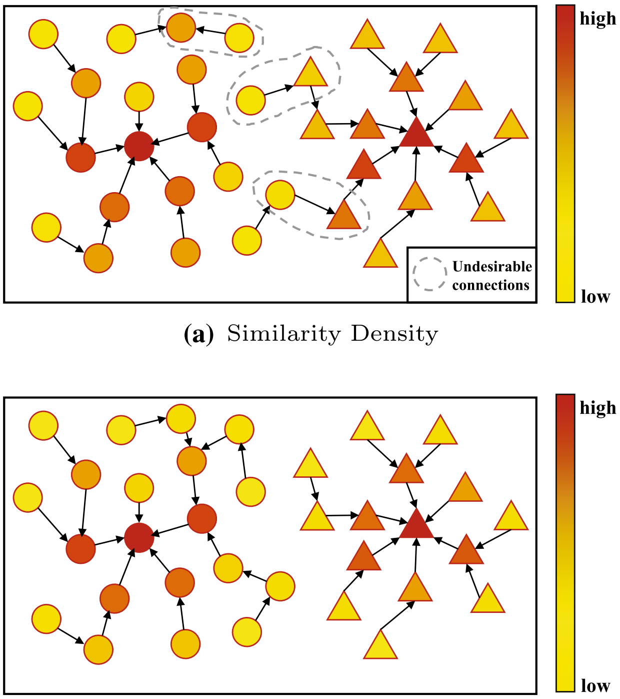
Neighboring weight generation: Finally, a two-layer MLP is used to obtain the weight assignment of the jth neighbor of point i:

$$w_{ij} = \sigma(\hat{X}_i[j,:]W_5)W_6 \tag{5}$$

where $\hat{X}_i[j,:]$ denotes the ith row of $\hat{X}_i$, $W_5 \in \mathbb{R}^{2D\times D}$ and $W_6 \in \mathbb{R}^{D\times 1}$.

Density-guided pair selection

Due to the redundancy of pairwise connections, it is unnecessary to evaluate all the possible pairs extracted from a dataset. We assume that a data point should be attracted to some cluster center or to another point that is closer to the cluster center than it is. Consequently, the set of candidate pairs can be constructed by concentrating on such directed relationships. Since cluster centers usually lie in the high-density regions of the feature space [41], the two samples forming a pair are expected to differ in terms of density.



(a) Similarity Density

(b) Combined Density

Fig. 4 Comparison between the similarity density and combined density

```
Algorithm 1 Density-Guided Pair Selection
Require: feature sets $\mathcal{F}$, number of neighbors $k$, threshold $\theta_\rho$.
Ensure: Candidate pairs $\mathbb{E}$
procedure DENSITY- GUIDED  PAIR  SELECTION
    $\mathbb{E} = \emptyset$
    $\rho = $ CALCULATE  DENSITY($\mathcal{F}, k, \theta_\rho$).
    for all samples $i$ in $\mathcal{F}$ do
        Find the first neighbor node $j$ satisfying $\rho_j > \rho_i$.
        if $j$ exists then
            $\mathbb{E} = \mathbb{E} \cup \{(i, j)\}$
        end if
    end for
    return $\mathbb{E}$
end procedure

function CALCULATE  DENSITY($\mathcal{F}, k, \theta_\rho$)
    $\rho$=$\emptyset$;
    for all sample $i$ in $\mathcal{F}$ do
        Calculate the sum of the similarity values of $k$ neighbors $\rho_i^d$ by Eq. (6).
        Calculate the number of neighbors $\rho_i^s$ with similarity values greater than $\theta_\rho$ by Eq. (7).
        Calculate the combined density $\rho_i$ by Eq. (8).
        $\rho = \rho \cup \rho_i$
    end for
    return $\rho$
end function
```

Face images belonging to the same individual should be clustered together. The informative faces with near-frontal views and normal illuminations are found in higher density regions, while the low-quality faces with complex expressions and extreme lighting conditions appear at lower density cluster boundaries. Thus, candidate pairs can be collected along a path that starts from one sample and ends at a density peak.

Most existing face clustering methods adopt density settings based on neighborhood distances or similarities [8,41]. Given a sample f_i, its density ρ_i^d can be written as:

$$\rho_i^d = \sum_{j \in \mathcal{N}_i} a_{ij} \tag{6}$$

where a_{ij} is the similarity between f_i and its jth nearest neighbors. However, due to the complex distribution of faces, samples with higher density levels may be located at the boundaries of the clusters. As shown in Fig. 4(a), those samples lead to some undesirable connections between two clusters, which degrade the resulting clustering performance. To suppress the densities of the boundary samples, we propose a density fusion mechanism. In particular, a spherical density is added to adjust the initial value of ρ_i^d, which is written as follows:

$$\rho_i^s = \sum_{j \in \mathcal{N}_i} (a_{i,j} \geq \theta_\rho) \cdot 1 \tag{7}$$

where θ_ρ is the threshold value used to identify neighbors with high similarities. The introduction of spherical density is motivated by the fact that samples farther away from the cluster center often exhibit more significant neighborhood differences. Hence, the combined density for sample i is defined as:

$$\rho_i = \frac{\rho_i^d}{2\max(\{\rho_i^d\}_{i=1}^N)} + \frac{\rho_i^s}{2\max(\{\rho_i^s\}_{i=1}^N)} \tag{8}$$

Fig. 4(b) illustrates the results obtained after applying the combined density, whose main steps are summarized in algorithm 1.

Construction of training and testing sets

Unlike graph-based face clustering methods, our method takes a pair of samples as the underlying processing unit, so it becomes crucial to identify pairs of data. As described in "Density-guided pair selection section", a density-guided pair selection strategy is proposed for constructing training and testing sets.

As a result of algorithm 1, a set of candidate pairs can be constructed for model training. Specifically, pairs with

Algorithm 2 Pairwise Clustering

```
Require: candidate pairs $\mathbb{E}$
Ensure: clusters $\mathbb{C}$
procedure CLUSTERING
    for all pairs $e$ in $\mathbb{E}$ do
        Predicting the connectivity of $e$ via the network.
        if the connectivity of $e$ is False then
            $\mathbb{E} = \mathbb{E} \setminus \mathbf{e}$
        end if
    end for
    $\mathbb{C}$ = Use the BFS algorithm to generate clusters according to $\mathbb{E}$.
    return $\mathbb{C}$
end procedure
```

the same label are regarded as positive pairs; otherwise, they are negative pairs. However, this splitting approach tends to cause an imbalance between the positive and negative pairs, leading to an excess of positive pairs. To address this issue, we select pairs of samples and their K-nearest neighbors to augment the training set and keep the ratio of positive and negative pairs at 1:1.

In addition, for the testing set, the experimental results suggest that the aforementioned pair selection strategy is capable of characterizing the structures of clusters. Therefore, to guarantee the efficiency of the inference process, the set of testing pairs is not augmented. The main steps of the proposed framework are summarized in algorithm 2.

Complexity analysis

The computational complexity of algorithm 2 mainly arises from the k-NN graph construction and pair selection steps. The k-NN search is the bottleneck of the algorithm, which has a complexity of $O(n^2)$. With the approximate nearest search technique [39], the time complexity is reduced to $O(n \log n)$. Since the size of the candidate set used for inference does not exceed the number of samples, the time cost of pair selection is $O(n)$. Due to the pair augmentation process employed for training, the pair selection cost is $O(nk)$, where $k \ll n$ is the neighbor size for the pair search. Hence, the overall complexity is approximately $O(n \log n)$.

Experiments

Datasets

MS-Celeb-1 M (MS1M) [9] is a widely used large-scale face dataset. It contains approximately 100K identities and 10M face images, with varying numbers of images associated with each identity. Following the protocol used in [19,29,36,41], we clean the dataset in terms of the annotations from Arcface [6], producing approximately 86K identities and 5.82M face images. Then, we divide the dataset into 10 equal parts and select the first part for training and the rest for testing. In particular, the five testing sets are constructed by selecting 1, 3, 5, 7 and 9 parts, and the numbers of images are 584K, 1.74M, 2.89M, 4.05M, and 5.21M, respectively. Additionally, to verify the feasibility of the proposed method, we test it on the DeepFashion [21] dataset for fashion clustering. We follow the settings in [41], where the training set includes about 26K images and 4K categories, and the testing set includes about 27K images and 4K categories.

Evaluation metrics

Two common metrics are used to assess the performance of clustering algorithms, namely, the Pairwise F-score (F_P) [30] and BCubed F-score (F_B) [1], which are harmonic means of precision and recall.

The Pairwise F-score is calculated based on sample pairs. The pairwise precision indicates the proportion of sample pairs that are correctly predicted among all pairs predicted to belong to the same class, which is written as:

$$Pairwise\ Precision = \frac{TP}{TP + FP} \tag{9}$$

where TP and FP are the abbreviations of true-positive pairs and false-positive pairs, respectively. Similarly, the pairwise recall is written as:

$$Pairwise\ Recall = \frac{TP}{TP + FN} \tag{10}$$

where FN denotes false-negative pairs.

The BCubed F-score measures the difference between the ground-truth labels and the cluster results. Let $G(i)$ and $P(i)$ denote sets of samples that have the same annotation and cluster assignments as sample i, respectively, and let $C(i, j)$ indicate the *consistency* of samples i and j, which is formulated as:

$$C(i, j) = \begin{cases} 1, & G(i) = G(j)\ and\ P(i) = P(j) \\ 0, & otherwise \end{cases} \tag{11}$$

The precision and recall are defined as

$$BCubed Precision = \frac{1}{N} \sum_{i=1}^{N} \sum_{j \in P(i)} \frac{C(i, j)}{|P(i)|} \tag{12}$$

and

$$BCubed Recall = \frac{1}{N} \sum_{i=1}^{N} \sum_{j \in G(i)} \frac{C(i, j)}{|G(i)|} \tag{13}$$

where $|G(i)|$ and $|P(i)|$ denote the sample sizes of sets $G(i)$ and $P(i)$, respectively. The following formula calculates the

$F\text{-}score$ for both metrics:

$$F\text{-}score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{14}$$

Implementation details

In this study, we follow the settings in [19,29,36,40,41] and utilize Arcface [6] as a base feature extractor to obtain 256-dimensional input features. We also use the k-NN algorithm [5] to search K neighbors for each sample, where K is set to 80 and 20 for the MS1M dataset and the DeepFashion dataset, respectively. For the construction of the combined density, we set $\theta_\rho = 0.7$ for the MS1M dataset and $\theta_\rho = 0.9$ for the DeepFashion dataset to properly increase the density gap between the cluster center and boundary. During the training phase, we use the cross-entropy loss to optimize the network. The stochastic gradient descent (SGD) optimizer is used with an initial learning rate of 0.01, a momentum of 0.9 and a weight decay of 1e-4. The batch size is set to 512, and the training process ends after 100 epochs.

Method comparison

To demonstrate the validity of our approach, we compare SEPFL with a series of clustering baselines, including six traditional clustering methods and seven deep learning-based methods. A brief description of each algorithm is given below.

- K-Means [22]: The most commonly used clustering method, K-means produces clustering results with a predefined number of clusters.
- HAC [31]: Hierarchical clustering is a bottom-up approach that iteratively merges samples through various distance metrics.
- DBSCAN [7]: DBSCAN is a density-based method that has demonstrated advantages in handling data with complex distributions.
- MeanShift [4]: The convergence of multiple sets of samples to the same local maximum density constitutes the resultant cluster.
- Spectral Clustering [12]: The similarity matrix of the data is eigen-decomposed and clustered according to the eigenvectors.
- ARO [26]: A new metric is proposed to achieve improved rank-order clustering [47].
- CDP [42]: This is a graph-based clustering method. By fusing the information of multiple samples, better pairwise features can be obtained.
- L-GCN [37]: L-GCN is a supervised clustering algorithm that uses a GCN to learn sample structure information for connection prediction.
- LTC [40]: This is a two-stage clustering method. The input data are processed separately using the ideas of classification and segmentation.
- GCN(V+E) [41]: This is a two-stage clustering method. The whole constructed graph is fed into a GCN to predict the confidence levels of the samples and construct subgraphs. After that, the noise points in the subgraphs are predicted.
- CFPC [19]: This is a pairwise learning-based clustering method. CFPC assists the pairwise learning process by fusing neighbor sample information to obtain structural information, differing from the GCN-based approaches.
- STAR-FC [29]: This method suggests a structure-preserving sampling strategy to build a subgraph that preserves enough structural information to make training on tens of millions of face data possible.
- Ada-NETS [36]: Ada-NETS solves the problem of introducing too many noisy edges when constructing graph-structured data via the adaptive neighbor discovery method.

Result comparison

Table 3 shows the results obtained by our method and other clustering baselines, including four conventional clustering methods and seven supervised clustering methods, on the MS1M dataset. It can be seen that the proposed method obtains the best results on all five testing datasets, which possess varying sizes. Compared to the second-best approach, i.e., Ada-NETS, the proposed SEPFL method achieves greater performance gains as the dataset volume increases. For instance, SEPFL outperforms Ada-NETS by 0.55% and 0.29% on the 584K data in terms of F_P and F_B, and when the data size increases to 5.21 M, the performance gaps are enlarged to 2.28% and 1.77%, respectively. These results suggest that our method is able to learn more generalized representations for clustering. In addition, by introducing the neighborhood learning strategy, SEPFL consistently outperforms CFPC, which also works at the pair level. Note that out of the four conventional clustering methods, only K-means achieves competitive results, because the ground-truth number of clusters is given in advance.

To demonstrate the applicability of our model to nonface images, we conduct experiments on the DeepFashion dataset. As shown in Table 4, our approach produces the best results. In terms of the two utilized metrics, SEPFL is ahead of Ada-NETS by 2.77% and 1.76%, respectively, which indicates the generalization ability of the proposed model on clustering tasks.

Figure 5 compares the efficiency and accuracy of our method with several clustering baselines on the MS1M dataset (part 1). Since ARO does not obtain competitive performance (as shown in Table 3), its results are not included.

Table 3 Comparison among the face clustering results obtained with different numbers of unlabeled images from the MS1M dataset

Number of images Methods/ Metrics	584K F_P	 F_B	1.74 M F_P	 F_B	2.89 M F_P	 F_B	4.05 M F_P	 F_B	5.21 M F_P	 F_B
K-Means [22]	79.21	81.23	73.04	75.20	69.83	72.34	67.90	70.57	66.47	69.42
HAC [31]	70.63	70.46	54.40	69.53	11.08	68.62	1.40	67.69	0.37	66.96
DBSCAN [7]	67.93	67.17	63.41	66.53	52.50	66.26	45.24	44.87	44.94	44.74
ARO [26]	13.60	17.00	8.78	12.42	7.30	10.96	6.86	10.50	6.35	10.01
CDP [42]	75.02	78.70	70.75	75.82	69.51	74.58	68.62	73.62	68.06	72.92
L-GCN [37]	78.68	84.37	75.83	81.61	74.29	80.11	73.70	79.33	72.99	78.60
LTC [40]	85.66	85.52	82.41	83.01	80.32	81.10	78.98	79.84	77.87	78.86
GCN(V+E) [41]	87.93	86.09	84.04	82.84	82.10	81.24	80.45	80.09	79.30	79.25
CFPC [19]	90.67	89.54	86.91	86.25	85.06	84.55	83.51	83.49	82.41	82.40
STAR-FC [29]	91.97	90.21	88.28	86.26	86.17	84.13	84.70	82.63	83.46	81.47
Ada-NETS [36]	92.79	91.40	89.33	87.98	87.50	86.03	85.40	84.48	83.99	83.28
SEPFL	**93.34**	**91.69**	**90.20**	**88.63**	**88.63**	**87.09**	**87.35**	**85.92**	**86.27**	**85.05**

Best results are given in bold

Table 4 Comparison results obtained on DeepFashion

Methods	F_P	F_B
K-means [22]	32.86	53.77
HAC [31]	22.54	48.77
DBSCAN [7]	25.07	53.23
MeanShift [4]	31.61	56.73
Spectral [12]	29.02	46.40
ARO [26]	26.03	53.01
CDP [42]	28.28	57.83
L-GCN [37]	28.85	58.91
GCN(V+E) [41]	38.47	60.06
CFPC [19]	37.67	62.17
STAR-FC [29]	37.07	60.60
Ada-NETS [36]	39.30	61.05
SEPFL	**42.07**	**62.81**

Best results are given in bold

The results indicate that our method achieves the highest accuracy as well as being time-efficient with a faster inference time than most clustering methods.

Parameter analysis

In this section, we explore the effects of different numbers of neighbors K and density thresholds θ on the clustering results.

Influence of the number of neighbors k

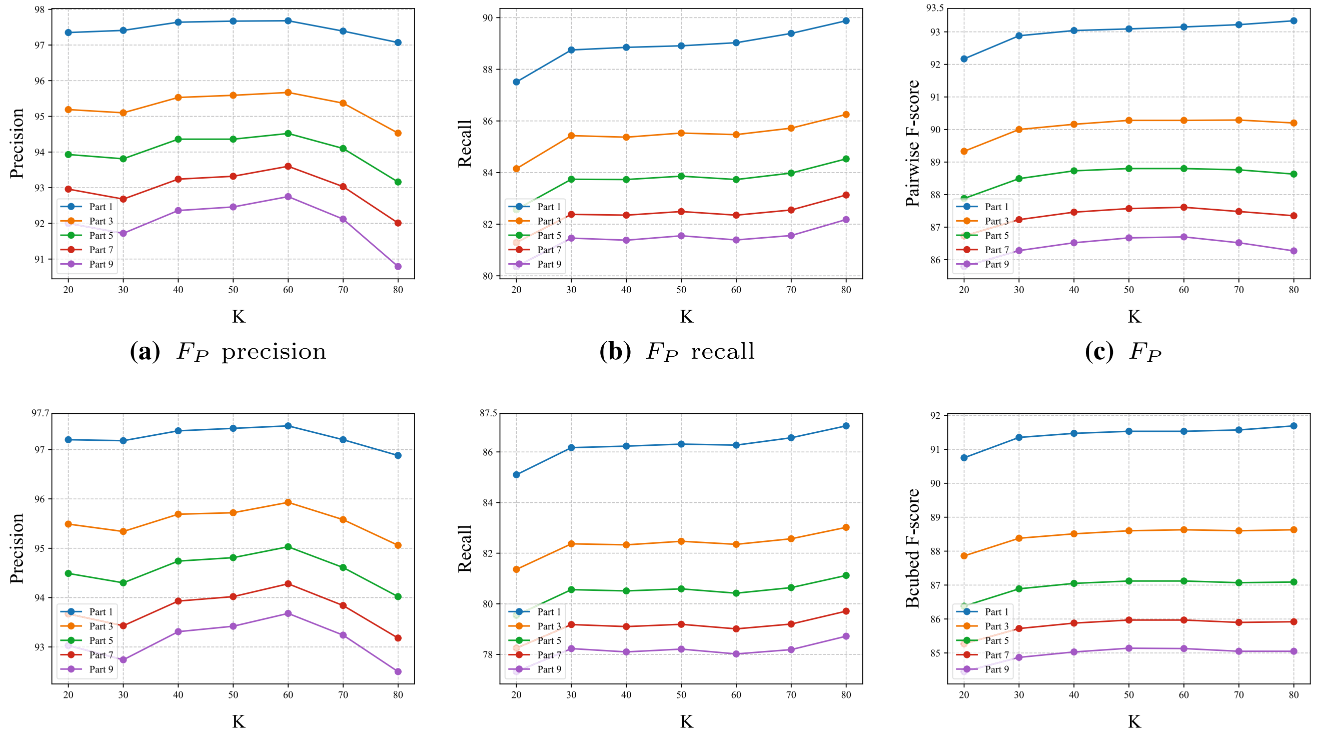
In our model, the number of neighbors k is mainly used to establish the regions for density calculation and pair selection. For simplicity, we set the same k for the above operations. To investigate the influence of different values of k on the clustering performance, we increase the parameter from 20 to 80 with a step size of 10, and the results are shown in Fig. 6. We observe similar trends in the metrics on all testing partitions. More candidate pairs are included as k increases, resulting in a higher recall. A larger k brings more false-positive pairs, which decreases the precision rate. In short, the performance is stable when k is greater than 30, demonstrating that our model is not sensitive to k.

Influence of the density threshold θ

As defined in Eq. (8), the proposed combined density ρ is the average of the similarity density ρ^d and the spherical density ρ^s. The parameter θ is employed to adjust the importance of ρ^d to ρ. Figure 7 presents the influences of θ on the MS1M and DeepFashion datasets. Apparently, the performance of the proposed approach can be improved when θ is set within a certain range (e.g., from 0.6 to 0.75 on the MS1M dataset). This is because as θ tends to 0 or 1, ρ^s approximates a constant, resulting in the degradation from ρ to ρ^d. Moreover, due to the differences in the distributions and sizes of datasets, a larger θ is required to activate ρ^s on DeepFashion than on MS1M.

Influence of postprocessing thresholds

In “Structure-enhanced pairwise feature learning” section, we argue that some manual-setting postprocessing thresholds may limit the applicability of existing clustering methods to different datasets and scenarios. To verify the sensitivity of the clustering results to the threshold values, we take STAR-FC [29] as an example, which includes two postprocessing thresholds, i.e., τ_1 and τ_2. Specifically, τ_1 represents the threshold for removing weak edges and τ_2 represents the threshold for removing edges with low node intimacy. For



(a) F_P precision **(b)** F_P recall **(c)** F_P

(d) F_B precision **(e)** F_B recall **(f)** F_B

Fig. 6 Comparison of the F_P and F_B results obtained with different values of K from the MS1M dataset

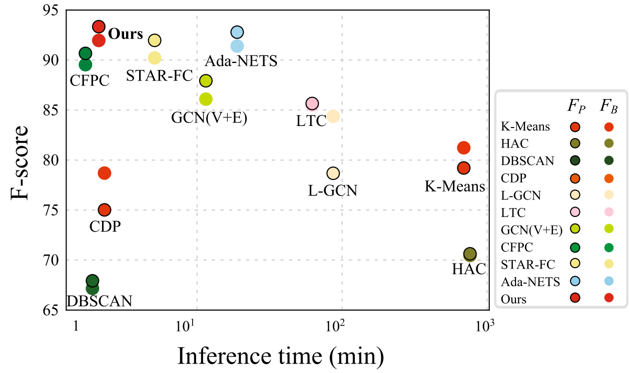


Fig. 5 Comparison of efficiency and accuracy between our method and other clustering methods

comparison, we first adjust them to achieve the best performance, then fix one of them and change the value of the other. The experimental results are illustrated in Fig. 8.

It can be observed that (1) the clustering results are sensitive to the value of the threshold. (2) Some dataset-specific thresholds should be established due to distribution differences between datasets, and (3) the two metrics (F_P and F_B) show different trends with the same group of thresholds. As a result, these inconsistencies increase the difficulty of setting thresholds. Since our method requires no postprocessing threshold, it provides better robustness and stability.

Ablation study

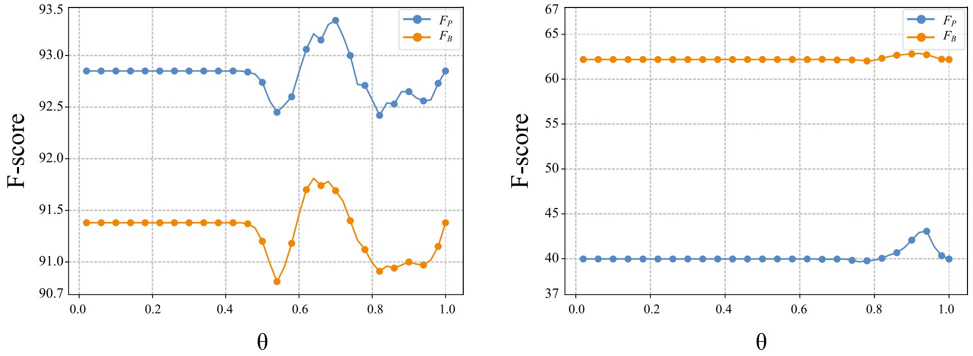
In this section, we further analyze the effectiveness of our algorithm through a large number of ablation experiments on the MS1M dataset.

Design of pairwise features

We explore different designs of pairwise features for clustering purposes. According to Eq. (1), we can formulate a pair representation by directly concatenating the features of two faces without any neighborhood information; this process is denoted as simple concatenation. To investigate the benefits of local relations with respect to pairwise descriptors, we compare several neighborhood aggregation strategies. Mean aggregation computes the average of neighbors [37]. Similarity aggregation obtains the weighted sum of neighbors based on cosine similarity [41]. As discussed in Neighborhood mixing module Section, the proposed SEPFL approach employs a neighborhood mixing schema for weight learning.

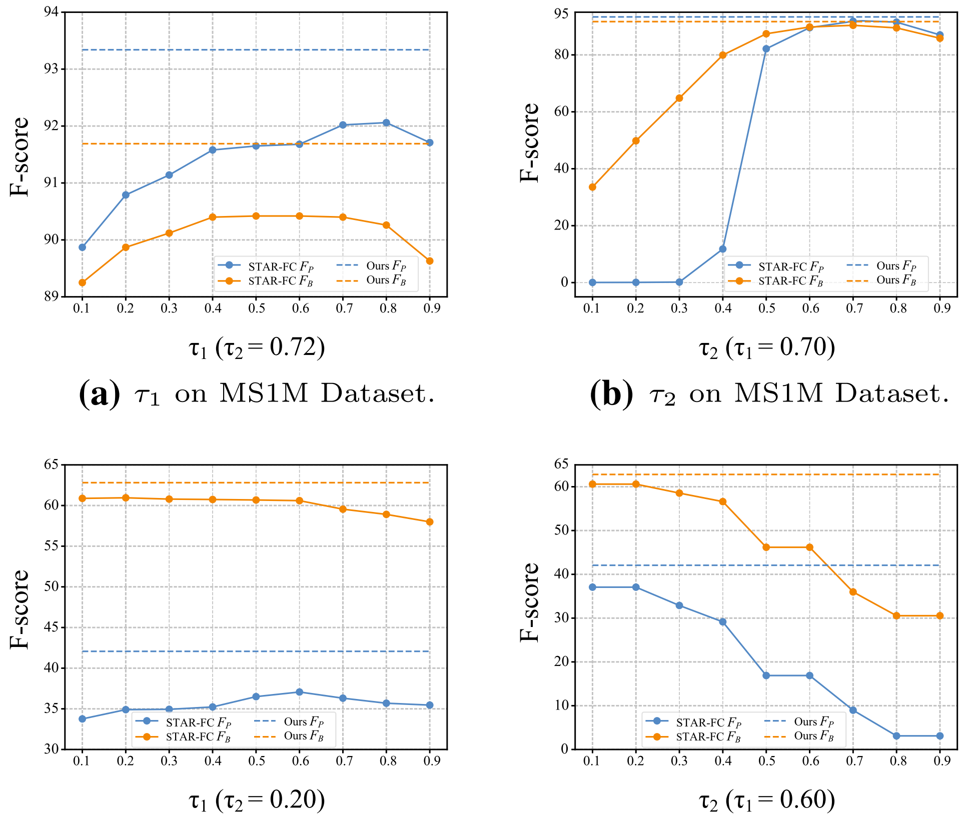
As shown in Fig. 9, the performances are significantly improved when the contextual properties are considered. In addition, these aggregation strategies differ mainly in their weighting procedures. Mean aggregation assigns the same weight to each neighbor. Since this setting does not reveal

Fig. 7 Effects of different density thresholds on the clustering results



(a) MS1M **(b)** DeepFashion

Fig. 8 Effects of different postprocessing thresholds on the clustering results. τ_1 and τ_2 are two edge-cutting thresholds employed in STAR-FC [29]. Our results are shown as two lines



(a) τ_1 on MS1M Dataset. **(b)** τ_2 on MS1M Dataset.

(c) τ_1 on DeepFashion Dataset. **(d)** τ_2 on DeepFashion Dataset.

Table 5 Comparison among the results obtained with different densities

Dataset	MS1M Part 1		MS1M Part 3		MS1M Part 5		MS1M Part 7		MS1M Part 9		DeepFashion	
methods/ metrics	F_P	F_B	F_P	F_B	F_P	F_B	F_P	F_B	F_P	F_B	F_P	F_B
Similarity density	92.85	91.38	89.68	88.15	87.96	86.46	86.61	85.23	85.50	84.30	39.97	62.19
Spherical density	74.12	77.01	70.67	74.18	69.38	72.91	68.28	71.87	67.59	71.17	28.12	53.32
Combined density	**93.34**	**91.69**	**90.20**	**88.63**	**88.63**	**87.09**	**87.35**	**85.92**	**86.27**	**85.05**	**42.07**	**62.81**

Best results are given in bold

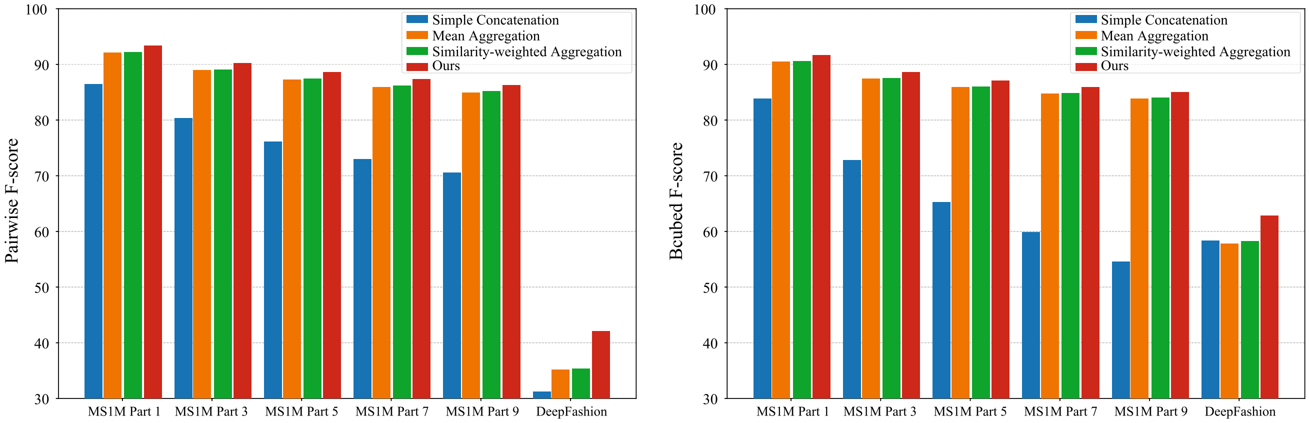
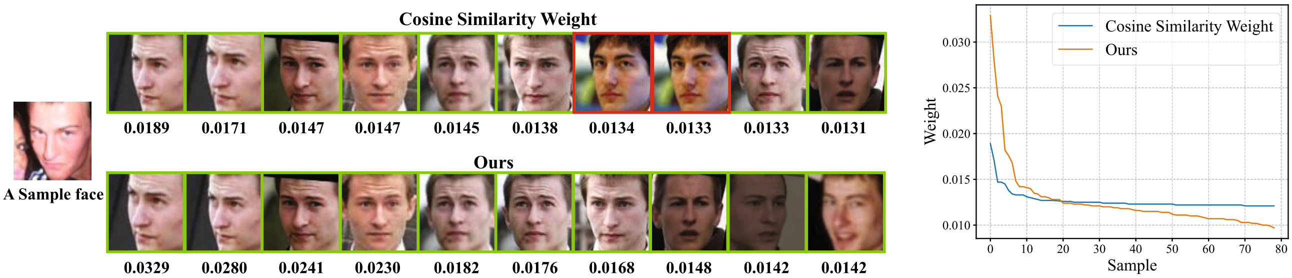


Fig. 9 Compare the results of different feature designs on the F_P and F_B evaluation metrics



(a) Ranking results obtained based on different weighting strategies.

(b) Weight distributions of two weighting strategies.

Fig. 10 Comparison of different weighting strategies. **a** Given a face sample, we list the top ten most similar samples based on the calculated weights (a larger weight indicates a higher similarity value). The faces with the same identity are outlined by green boxes, and the rest are outlined by red boxes. **b** We plot the weight distributions of two weighting strategies. As we can see, our method can generate more discriminative weights to reduce the influence of noisy or unreliable face images

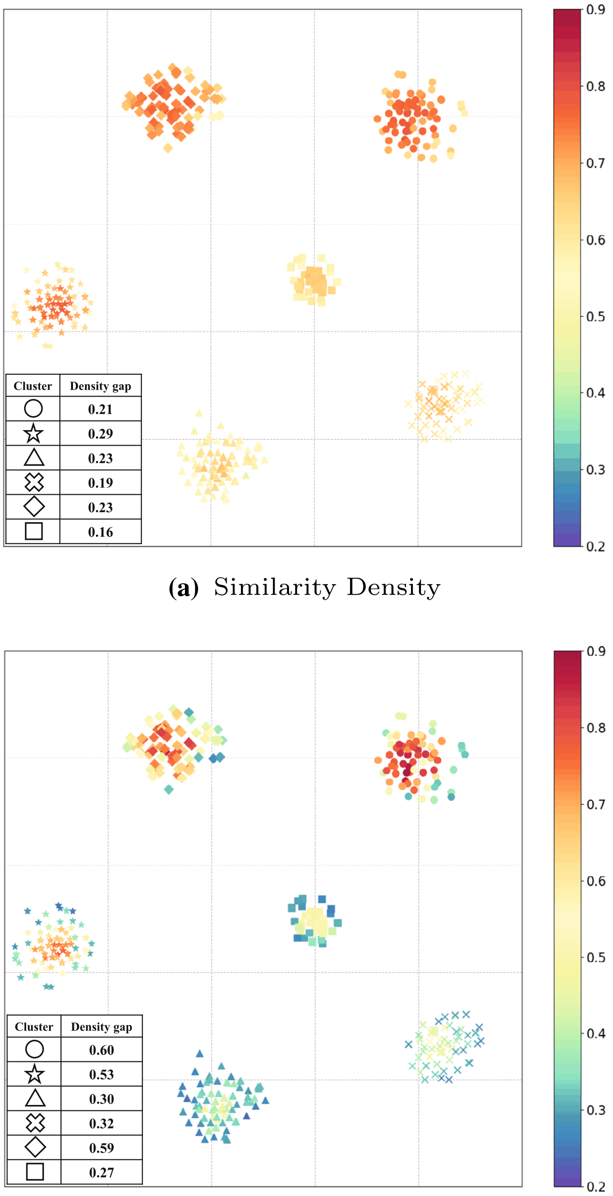
the importance levels of the neighbors, the performance of this technique is inferior to similarity-weighted aggregation. Our model enhances the exchange of information within the neighborhood for weight learning and thus achieves the best results.

Furthermore, to evaluate the quality of the weights produced by different weighting strategies, we illustrate the weights assigned to the neighbors of a given sample in Fig. 10. The results obtained based on cosine similarity weighting are included for comparison. As shown in Fig. 10(a), after ranking the neighbors by their weights, the similarity weighting method yields two false-positive samples (outlined by red boxes). In contrast, SEPFL generates more robust weight assignments. Additionally, Fig. 10(b) shows the distributions of the weight values of the two weighting schemes. The results suggest that our model can assign more discriminative weights to the neighborhood samples, by which the importance of irrelevant samples is effectively suppressed.

Design of the combined density

As discussed in "Density-guided pair selection section", we combine two densities for pair selection. Based on a fixed-size neighborhood centered at a sample f_i, the first density is computed by the sum of the similarities to f_i, which is called the similarity density and defined by Eq. (6), and the second density is derived from the number of samples whose similarities to f_i are higher than a threshold, which is called the spherical density and defined by Eq. (7). Table 5 shows the effects of different density settings. It can be seen that the spherical density performs the worst since it outputs discrete values. However, it can improve the similarity density by increasing the density gap between the cluster center and boundary samples. This enables the combined density to achieve the best results.

In addition, Fig. 11 visualizes the feature distribution of 6 identities sampled from the MS1M dataset using t-distributed stochastic neighbor embedding (t-SNE) [23]. As shown in Fig. 11(a), in terms of the similarity density, the samples in the dense regions share relatively high densities and therefore



(a) Similarity Density

(b) Combined Density

Fig. 11 Visualization of the feature distributions of two types of densities

may result in excessive low-information pairs. We present the updated densities in Fig. 11(b). It is clearly seen that the density variations within clusters are appropriately enlarged. Hence, the combined density can effectively improve the quality of the pairs output by algorithm 1.

Influence of the number of pairs

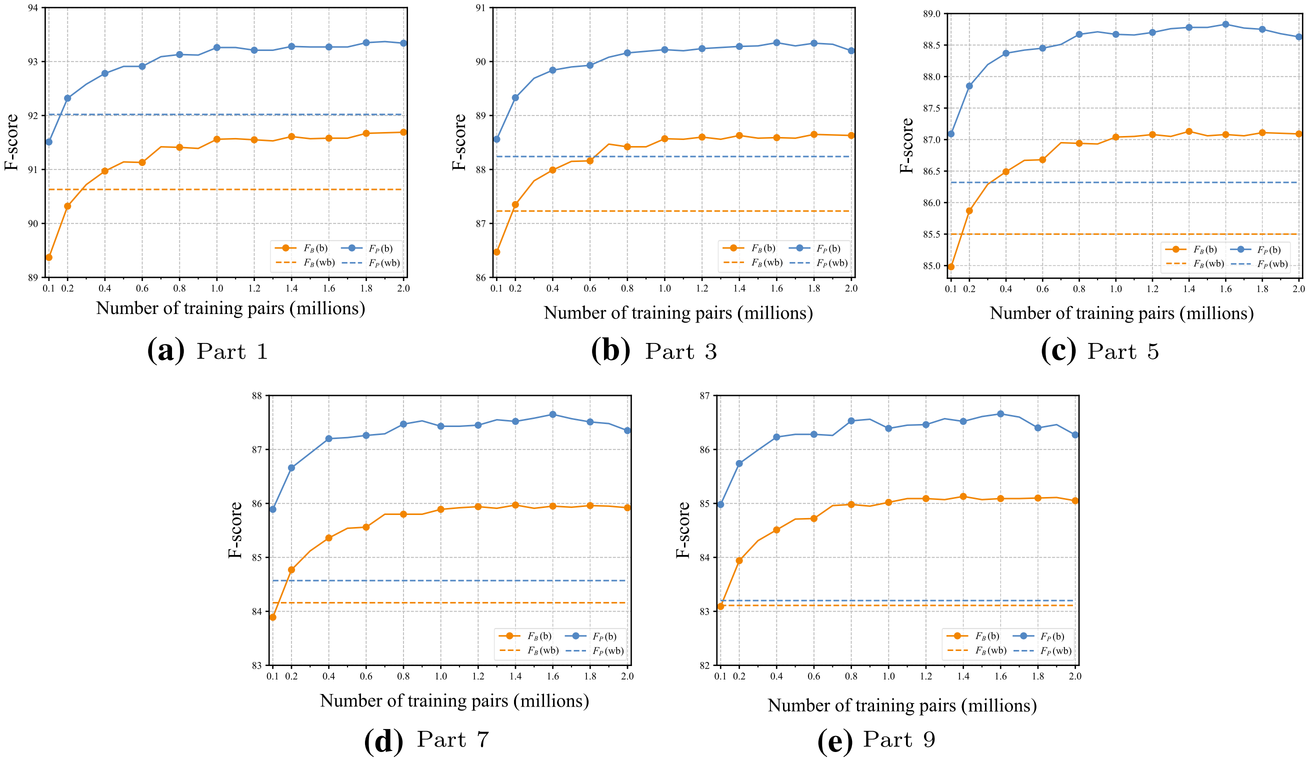
Unlike graph-based face clustering methods, our method uses a sample pair as the underlying processing unit, so it is crucial to identify data pairs. As described in "Construction of training and testing sets", a density-guided pair selection strategy is proposed for constructing training and testing sets. Basically, each sample serves as the starting point for the construction of at least one pair. However, the resulting imbalance between positive and negative pairs may cause training bias, and insufficient data lead to overfitting. To solve these issues, we adjust the numbers of positive and negative pairs to be equal by adding random pairs. To investigate how the number of pairs affects the clustering performance, we vary the parameter over a wide range on the MS1M dataset.

For the training phase, we include the results obtained without the balancing procedure for comparison and report the F-scores in Fig. 12. The experimental results reveal that (1) both the F_P and F_B metrics gradually improve as the training size increases, and when the performance saturates, adding more training data does not yield better results. (2) The balance between positive and negative pairs is essential for achieving performance improvement, and (3) this imbalance also affects the generalizability of the model; for example, as the volume of the testing set increases (i.e., from part 1 to part 9), the performance loss is enlarged.

Given testing pairs with a size of N', we begin with a random selection of samples to construct the corresponding pairs. The process continues until each sample is selected once as the starting point in a pair. After that, the size of the candidate set for testing is further increased to 2.0 millions by adding the remaining pairs. A baseline, by which the positive and negative pairs are equally randomly selected from all possible pairs, is utilized for comparison purposes. As summarized in Fig. 13, we observe that (1) the density-guided pair selection method consistently outperforms its counterpart based on random selection as the number of pairs increases, and (2) its performance peaks when the number of pairs reaches N' and deteriorates thereafter. The reason for this is that our method can identify sufficient representative pairs, while adding extra noise pairs would degrade the achieved performance. According to these observations, the proposed method provides a simple method for pair selection, yielding improved clustering accuracy and inference efficiency.

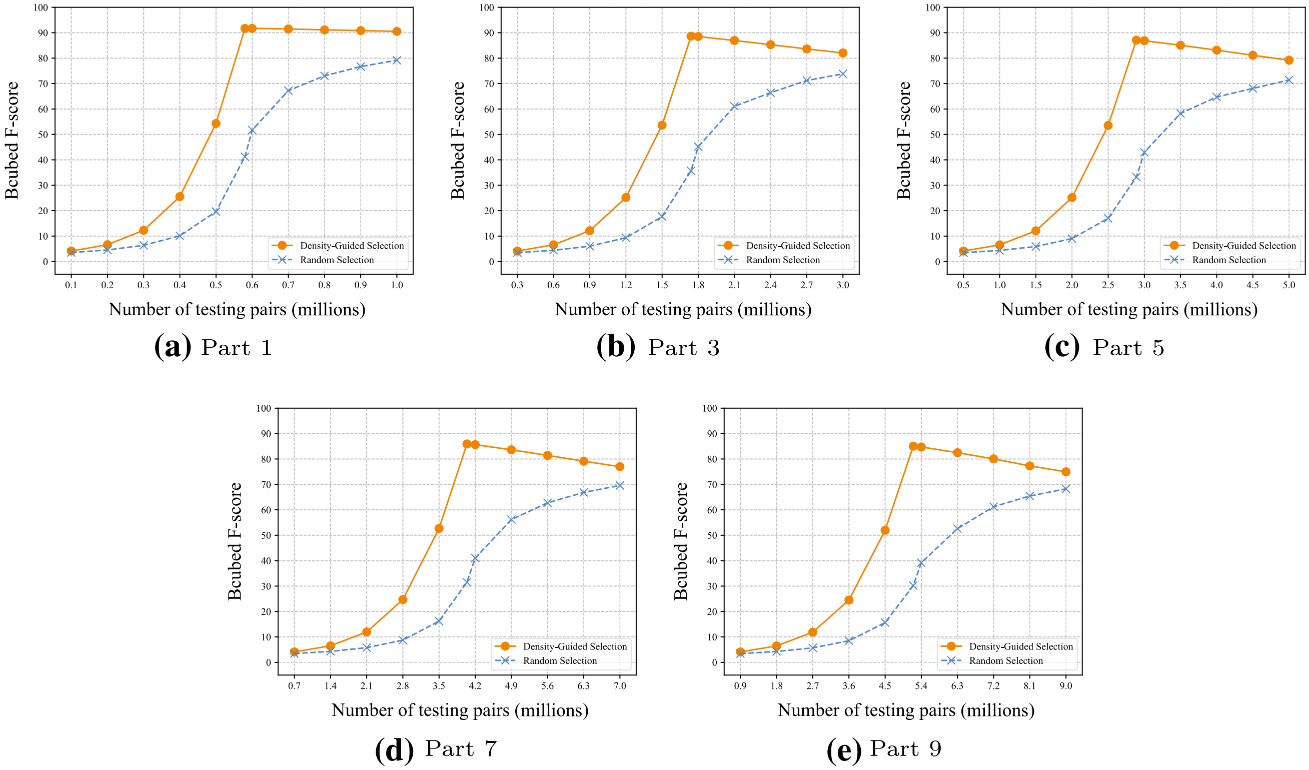
Statistical testing

In this section, we investigate the significant difference between SEPFL and other clustering methods using the Wilcoxon signed-rank test [28], which is a non-parametric statistical hypothesis test method. The null hypothesis H_0 in this experiment indicates that there is no significant difference between the two methods, while the alternative hypothesis H_1 indicates the opposite conclusion. P-value is used to determine whether the null hypothesis holds, and if p-value is less than the significance level α then the null hypothesis is rejected.



(a) Part 1 **(b)** Part 3 **(c)** Part 5 **(d)** Part 7 **(e)** Part 9

Fig. 12 Comparison among the F-scores obtained under different numbers of training pairs from the MS1M dataset. F_B (b) or F_B (wb) stands for the results obtained with or without the balancing procedure (the F_p metric is denoted in the same way)



(a) Part 1 **(b)** Part 3 **(c)** Part 5 **(d)** Part 7 **(e)** Part 9

Fig. 13 Comparison among the Bcubed F-scores obtained under different numbers of testing pairs from the MS1M dataset

Table 6 Statistical analysis results of Wilcoxon signed-rank test

Datasets	Metrics	vs. STAR-FC	vs. ADA-NETS
MS1M Part1	F_P	1.953125×10^{-3}	1.953125×10^{-3}
	F_B	1.953125×10^{-3}	1.953125×10^{-3}
MS1M Part3	F_P	1.953125×10^{-3}	1.953125×10^{-3}
	F_B	1.953125×10^{-3}	1.953125×10^{-3}
MS1M Part5	F_P	1.953125×10^{-3}	1.953125×10^{-3}
	F_B	1.953125×10^{-3}	1.953125×10^{-3}
MS1M Part7	F_P	1.953125×10^{-3}	1.953125×10^{-3}
	F_B	1.953125×10^{-3}	1.953125×10^{-3}
MS1M Part9	F_P	1.953125×10^{-3}	1.953125×10^{-3}
	F_B	1.953125×10^{-3}	1.953125×10^{-3}
DeepFashion	F_P	1.953125×10^{-3}	1.953125×10^{-3}
	F_B	1.953125×10^{-3}	1.953125×10^{-3}

As shown in Tables 3 and 4, there are large gaps between the traditional clustering methods and our method in terms of the two clustering metrics. Therefore, we choose to compare our method with two deep learning-based methods, namely STAR-FC [29] and Ada-NETS [36], which achieve competitive results among the compared methods.

To ensure the robustness of the results, we repeat the experiments 10 times using different random seeds on each dataset, and the results are shown in Table 6. It can be seen that the obtained p-values are all less than the common significance level of 0.05, indicating that our method is significantly different from other clustering methods.

Conclusion

A GCN efficiently obtains sample structure information by aggregating neighboring features but requires large levels of memory and time consumption. This paper proposes a novel pairwise learning method for face clustering, denoted as SEPFL. In particular, we design a neighborhood mixing block to weight the aggregation of neighborhood features as structural information by learning the correlations between samples and neighbors. Unlike other methods, the neighborhood mixing block considers both the relationships between samples and neighbors and the relationships between neighbors to learn more comprehensive structural information. In addition, a density-guided pair selection strategy is used to select candidate pairs, which avoids the influence of excessive redundant pairs on the clustering results.

We conduct extensive experiments on the MS1M and DeepFashion datasets. The experimental analysis proves that (1) SEPFL reduces the computational cost and alleviates the dependence on thresholds. (2) The neighborhood mixing block has a powerful ability to obtain structural information. (3) The density-guided pair selection strategy is capable of selecting representative candidate pairs. (4) SEPFL has higher accuracy than other advanced face clustering methods. As the amount of data increases, SEPFL exhibits better robustness.

Although SEPFL achieves good performance in various experiments, the method performs clustering on the original feature space, which is limited by the utilized feature extraction model. Problems such as complex sample distributions and large numbers of noisy samples or difficult samples may be encountered, which largely affect the clustering effect. In future works, we will explore an efficient feature learning method to transform the given data into an easily distinguishable feature space to assist with data clustering.

Acknowledgements This work was supported by the National Natural Science Foundation of China (No. 61972132), the S&T Program of Hebei (Nos. 20327404D, 21327404D), the Natural Science Foundation of Hebei Province (No. F2020204009).

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Amigó E, Gonzalo J, Artiles J, Verdejo F (2009) A comparison of extrinsic clustering evaluation metrics based on formal constraints. Inf Retr 12(4):461–486
2. Bo D, Wang X, Shi C, Zhu M, Lu E, Cui P (2020) Structural deep clustering network. Proc Web Conf 2020:1400–1410
3. Bu Z, Wang Y, Li HJ, Jiang J, Wu Z, Cao J (2019) Link prediction in temporal networks: integrating survival analysis and game theory. Inf Sci 498:41–61
4. Cheng Y (1995) Mean shift, mode seeking, and clustering. IEEE Trans Pattern Anal Mach Intell 17(8):790–799
5. Cover T, Hart P (1967) Nearest neighbor pattern classification. IEEE Trans Inf Theory 13(1):21–27
6. Deng J, Guo J, Xue N, Zafeiriou S (2019) Arcface: Additive angular margin loss for deep face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 4690–4699
7. Ester M, Kriegel HP, Sander J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, pp 226–231
8. Guo S, Xu J, Chen D, Zhang C, Wang X, Zhao R (2020) Density-aware feature embedding for face clustering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 6698–6706

9. Guo Y, Zhang L, Hu Y, He X, Gao J (2016) Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In: Proceedings of the European Conference on Computer Vision, pp 87–102
10. Hendrycks D, Gimpel K (2017) A baseline for detecting misclassified and out-of-distribution examples in neural networks. In: International Conference on Learning Representations
11. Ho J, Yang MH, Lim J, Lee KC, Kriegman D (2003a) Clustering appearances of objects under varying illumination conditions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, vol 1, pp I–I
12. Ho J, Yang MH, Lim J, Lee KC, Kriegman D (2003b) Clustering appearances of objects under varying illumination conditions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, vol 1, pp I–I
13. Huo G, Zhang Y, Gao J, Wang B, Hu Y, Yin B (2021) Caegcn: Cross-attention fusion based enhanced graph convolutional network for clustering. IEEE Transactions on Knowledge and Data Engineering
14. Kemelmacher-Shlizerman I, Seitz SM, Miller D, Brossard E (2016) The megaface benchmark: 1 million faces for recognition at scale. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 4873–4882
15. Kim Y, Park W, Roh MC, Shin J (2020) Groupface: Learning latent groups and constructing group-based representations for face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 5621–5630
16. Klare BF, Klein B, Taborsky E, Blanton A, Cheney J, Allen K, Grother P, Mah A, Jain AK (2015) Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 1931–1939
17. Li X, Zhang H, Zhang R (2021) Adaptive graph auto-encoder for general data clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence
18. Lin WA, Chen JC, Chellappa R (2017) A proximity-aware hierarchical clustering of faces. In: IEEE International Conference on Automatic Face & Gesture Recognition, pp 294–301
19. Liu J, Qiu D, Yan P, Wei X (2021) Learn to cluster faces via pairwise classification. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 3845–3853
20. Liu W, Wen Y, Yu Z, Li M, Raj B, Song L (2017) Sphereface: Deep hypersphere embedding for face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 212–220
21. Liu Z, Luo P, Qiu S, Wang X, Tang X (2016) Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 1096–1104
22. Lloyd S (1982) Least squares quantization in pcm. IEEE Trans Inform Theory 28(2):129–137
23. Van der Maaten L, Hinton G (2008) Visualizing data using t-sne. J Mach Learn Res 9:2579–2605
24. Meng Q, Zhao S, Huang Z, Zhou F (2021) Magface: A universal representation for face recognition and quality assessment. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 14225–14234
25. Nech A, Kemelmacher-Shlizerman I (2017) Level playing field for million scale face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 7044–7053
26. Otto C, Wang D, Jain AK (2017) Clustering millions of faces by identity. IEEE Trans Pattern Anal Mach Intell 40(2):289–303
27. Qi C, Zhang J, Jia H, Mao Q, Wang L, Song H (2021) Deep face clustering using residual graph convolutional network. Knowl Based Syst 211:106561
28. Richardson Alice M (2015) Nonparametric statistics: A step-by-step approach. Int Stat Rev 83(1):163–164
29. Shen S, Li W, Zhu Z, Huang G, Du D, Lu J, Zhou J (2021) Structure-aware face clustering on a large-scale graph with 107 nodes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 9085–9094
30. Shi Y, Otto C, Jain AK (2018) Face clustering: representation and pairwise constraints. IEEE Trans Inf Forensics Secur 13(7):1626–1640
31. Sibson R (1973) Slink: an optimally efficient algorithm for the single-link cluster method. Comput J 16(1):30–34
32. Sun J, Zhang Y, Guo W, Guo H, Tang R, He X, Ma C, Coates M (2020) Neighbor interaction aware graph convolution networks for recommendation. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp 1289–1298
33. Tolstikhin IO, Houlsby N, Kolesnikov A, Beyer L, Zhai X, Unterthiner T, Yung J, Steiner A, Keysers D, Uszkoreit J, et al. (2021) Mlp-mixer: An all-mlp architecture for vision. Advances in Neural Information Processing Systems 34
34. Velickovic P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y (2017) Graph attention networks. Statistics 1050:20
35. Wang H, Wang Y, Zhou Z, Ji X, Gong D, Zhou J, Li Z, Liu W (2018) Cosface: Large margin cosine loss for deep face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 5265–5274
36. Wang Y, Zhang Y, Zhang F, Wang S, Lin M, Zhang Y, Sun X (2022) Ada-nets: Face clustering via adaptive neighbour discovery in the structure space. In: International Conference on Learning Representations
37. Wang Z, Zheng L, Li Y, Wang S (2019) Linkage based face clustering via graph convolution network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 1117–1125
38. Welling M, Kipf TN (2016) Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations
39. Wieschollek P, Wang O, Sorkine-Hornung A, Lensch H (2016) Efficient large-scale approximate nearest neighbor search on the gpu. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 2027–2035
40. Yang L, Zhan X, Chen D, Yan J, Loy CC, Lin D (2019) Learning to cluster faces on an affinity graph. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 2298–2306
41. Yang L, Chen D, Zhan X, Zhao R, Loy CC, Lin D (2020) Learning to cluster faces via confidence and connectivity estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 13369–13378
42. Zhan X, Liu Z, Yan J, Lin D, Loy CC (2018) Consensus-driven propagation in massive unlabeled data for face recognition. In: Proceedings of the European Conference on Computer Vision, pp 568–583
43. Zhang M, Chen Y (2017) Weisfeiler-lehman neural machine for link prediction. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp 575–583
44. Zhang M, Chen Y (2018) Link prediction based on graph neural networks. Advances in Neural Information Processing Systems 31
45. Zhang Y, Deng W, Wang M, Hu J, Li X, Zhao D, Wen D (2020) Global-local gcn: Large-scale label noise cleansing for face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 7731–7740
46. Zhao M, Teo YW, Liu S, Chua TS, Jain R (2006) Automatic person annotation of family photo album. In: International Conference on Image and Video Retrieval, pp 163–172

47. Zhu C, Wen F, Sun J (2011) A rank-order distance based clustering algorithm for face tagging. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 481–488
48. Zhu Z, Huang G, Deng J, Ye Y, Huang J, Chen X, Zhu J, Yang T, Lu J, Du D, et al. (2021) Webface260m: A benchmark unveiling the power of million-scale deep face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 10492–10502