

Incremental Face Clustering with Optimal Summary Learning Via Graph Convolutional Network

Xuan Zhao, Zhongdao Wang, Lei Gao, Yali Li, and Shengjin Wang*

Abstract: In this study, we address the problems encountered by incremental face clustering. Without the benefit of having observed the entire data distribution, incremental face clustering is more challenging than static dataset clustering. Conventional methods rely on the statistical information of previous clusters to improve the efficiency of incremental clustering; thus, error accumulation may occur. Therefore, this study proposes to predict the summaries of previous data directly from data distribution via supervised learning. Moreover, an efficient framework to cluster previous summaries with new data is explored. Although learning summaries from original data costs more than those from previous clusters, the entire framework consumes just a little bit more time because clustering current data and generating summaries for new data share most of the calculations. Experiments show that the proposed approach significantly outperforms the existing incremental face clustering methods, as evidenced by the improvement of average F -score from 0.644 to 0.762. Compared with state-of-the-art static face clustering methods, our method can yield comparable accuracy while consuming much less time.

Key words: incremental face clustering; supervised learning; Graph Convolutional Network (GCN); optimal summary learning

1 Introduction

In this study, we explore the following problem: clustering faces into identities with new faces arrives in batches, as shown in Fig. 1a. As a vital task in face analysis, face clustering is useful in several applications, such as the management and automatic cleaning of large-scale face datasets, summarizing and indexing of videos, and content-based face retrieval in law enforcement. In most of these applications, new faces are generated over time; thus, the amount of data continuously increases. As the scale of data in the system becomes larger, data clustering has become a prominent problem,

especially because of limited computation and memory resources. Without the benefit of having observed the entire data distribution, incremental face clustering is more challenging than clustering static face datasets.

Literature on data clustering abounds^[1–6]; however, only a few of them focus on incremental data. For example, recent state-of-the-art learning based methods, such as linkage graph^[2] and affinity graph^[1], do not have any strategies for dynamic face data. Although people can use these methods to cope with incremental photos by periodically scanning the entire dataset, this method is inefficient. Conventional approaches^[7–12] rely on statistical information of previous clusters to improve the efficiency of incremental clustering, as shown in Fig. 1b. For instance, incremental K -means^[10] uses the mean value of all nodes in a cluster to compare with new data. LINKS^[12] calculates cosine similarity between the new vectors and subcluster centroids of existing clusters. In these methods, inaccurate statistical information may result in further errors when clustering

• Xuan Zhao, Zhongdao Wang, Yali Li, and Shengjin Wang are with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China. E-mail: wsgsj@tsinghua.edu.cn.

• Xuan Zhao and Lei Gao are with the First Research Institute of Ministry of Public Security, Beijing 100084, China.

* To whom correspondence should be addressed.

Manuscript received: 2020-07-13; accepted: 2020-07-29

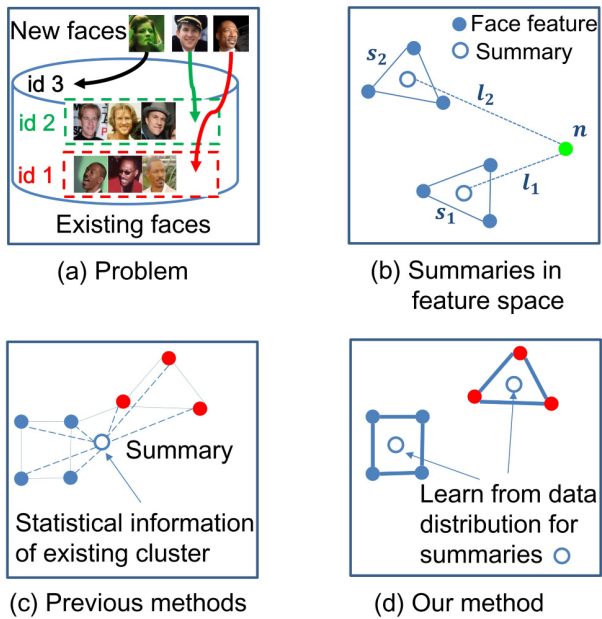


Fig. 1 High-level idea of the proposed method. (a) We address the problem of incremental face clustering. (b) Estimating summaries is a fundamental approach for incremental clustering. Summaries are a small number of features that represent existing data in feature space. In (b), n denotes a new face feature, s_1 is the summary of cluster 1, and s_2 is the summary of cluster 2. If $l_1 < l_2$, then n should be grouped into cluster 1; thus, we do not need to compare n with each existing feature. In (c) and (d), a situation wherein existing clusters have errors, that is, features of the blue and red persons are grouped into one cluster, is analyzed. To estimate summaries, (c) previous methods mainly use the statistical information of existing clusters, such as cluster centers. (d) Our method directly learns summaries from original data distribution using a graph convolutional network. In addition, our method generates improved summaries when previous clusters contain errors (best viewed in color).

new data, especially when previous clusters contain some errors. As shown in Fig. 1c, the features of blue and red persons are connected and form a wrong cluster. In this situation, conventional methods use the center of the wrong cluster that includes red and blue persons for clustering with new data.

The main idea of the proposed method is shown in Fig. 1d. In contrast to previous methods, our method does not rely on previous clustering results, but learns summaries directly from original data distribution. We learn the pattern of “ideal” graphs using a Graph Convolutional Network (GCN) in a supervised manner and aggregate the nodes of graphs with high confidence score to generate summaries. The advantage of this strategy is that the generation of summaries does not rely

on previous clusters nor on any simple assumption of special shape or uniform density.

The pipeline of our incremental method is as follows: Firstly, we utilize previous summaries to generate a set of overlapped candidate graphs with new face features. Secondly, we use learned GCN to predict the confidence of each candidate graph. Thirdly, we merge the candidate graphs to current clustering results. Lastly, we generate summaries of the current full data and store them for clustering with new face features. The approach is efficient because the results of candidate graph generation and GCN inference process are shared by the clustering and summary generating steps.

We experiment on 10 incremental batches of faces. The results show that the proposed approach significantly outperforms the existing incremental face clustering methods, as evidenced by an improvement in the Average F -score (AF) from 0.644 to 0.762. Compared with state-of-the-art static face clustering methods, our method can yield comparable accuracy while consuming much less time.

2 Related Work

2.1 Face clustering

As a fundamental task in face analysis and video application, face clustering has been widely studied for decades. Compared with other face analysis tasks, such as face verification and face recognition, face clustering is a more challenging task due to its unsupervised characteristics.

Most conventional methods make simple assumptions on data distribution^[4–7,9]. For example, K -means^[4] assumes that the clusters are sphere shaped, spectral clustering^[5] requires that each cluster has approximately the same number of nodes, and DBSCAN^[6] needs uniform density for all the clusters. However, in real-life applications, photos of the same person considerably vary because illumination, pose, resolution, occlusion, and age may differ. These differences result in arbitrary shapes and density in the feature space. These traditional clustering methods perform not well for unconstrained faces in real application because the assumptions are not met.

Recent studies have used rich information in the local context to cope with complex data distribution without any assumptions. Approximate Rank-Order clustering (ARO)^[3] updates distances of photo pairs by the rank-order of their mutual nearest neighbors. The linkage

graph^[2] constructs an instance pivot graph for each photo and its neighbors and then utilizes a GCN to learn and infer the graph and update linkage likelihood between each photo and its neighbors. The authors then merged the links to form clusters^[2]. The affinity graph^[1] adopts a pipeline similar to object detection and uses GCN to find and refine the desired clusters on the basis of graph learning. In this work, we also learn to utilize information in the local context to promote clustering accuracy.

2.2 Incremental clustering

Literature on data clustering abounds; however, only a few of them focus on incremental photos. In Ref. [13], the authors argued that without the benefit of having observed the entire data distribution, incremental face clustering was more challenging than clustering with a static dataset; in addition, static clustering methods cannot be directly applied to incremental data.

Most incremental clustering tasks do not require clustering every new photo as soon as it arrives. People usually cluster new photos hourly or daily. To simulate this situation, many incremental clustering approaches process new data in chunks^[13–15], that is, in batches.

Most existing incremental clustering methods^[10, 12–17] utilize the statistical information of previous clusters to reduce computation cost. For instance, incremental *K*-means^[10] uses the mean of all nodes in a cluster to summarize each previous cluster. LINKS^[12] calculates cosine similarity between each new vector and the subcluster centroids of existing clusters. OACFC^[13] computes each new face track in video with previous cluster centers and determines whether to merge it with a previous cluster or to create a new cluster. When new data come, new data are clustered firstly. Then, new clusters are merged with previous clusters if their similarities are above a certain threshold.

These methods mainly rely on previous clustering results, and thus may result in error accumulation, especially when previous clusters contain errors. Moreover, most existing incremental methods make impractical assumptions on the feature distributions. In contrast to the above methods, our proposed method learns directly from the original data distribution in a supervised manner.

2.3 Graph convolutional network

In recent years, GCNs^[18, 19] have attracted considerable attention because of their strong ability to learn space

features on non-Euclidean-structured data, which are also known as graph-structured data. In this type of data, the number of adjacent vertices for each vertex may be different in the topology. Non-Euclidean-structured data exist widely in real world application, such as social networks and relationship analyses. Existing research^[20, 21] has shown that GCN can lead to a remarkable performance gain on various tasks. In this study, we utilize GCN to learn the summaries of previous data distribution for incremental clustering.

3 Proposed Approach

3.1 Overview

Problem definition. Similar to previous research, we assume that new face photos with embedded features $\{F_1, F_2, \dots, F_{i-1}\}$ arrive in batches. Moreover, we assume that $i - 1$ batches of face photos have been clustered. Now, we obtain the i -th batch of face features $F_i = \{f_i\}^{N_i}$ which has N_i images; here, $f_i \in \mathbb{R}^d$ is a feature vector. Our task is to cluster F_i with previous face features in $\{F_1, F_2, \dots, F_{i-1}\}$. The goal is to promote accuracy and efficiency for each batch, not only the last batch.

Motivation. To reduce computation cost, new data must be clustered with summaries of previous data for incremental clustering. However, generating summaries simply by statistical information of previous clusters may result in error accumulation. To avoid error accumulation, we try to learn summaries directly from the original data distribution. We learn the pattern of an “ideal” graph through a GCN in a supervised manner and aggregate the nodes of graphs with high confidence score to generate summaries. The advantage of this strategy is that the generation of summaries does not rely on previous clusters nor on any simple assumption of special shape or uniform density.

3.2 Framework

We propose an incremental face clustering method considering accuracy and efficiency. The framework consists of four main steps (shown in Fig. 2).

Step 1: Generating a set of overlapped candidate graphs by using new features and previous summaries.

Step 2: Using learned GCN to predict the confidence of each candidate graph.

Step 3: Merging the candidate graphs with current clustering results.

Step 4: Generating summaries of the current full data

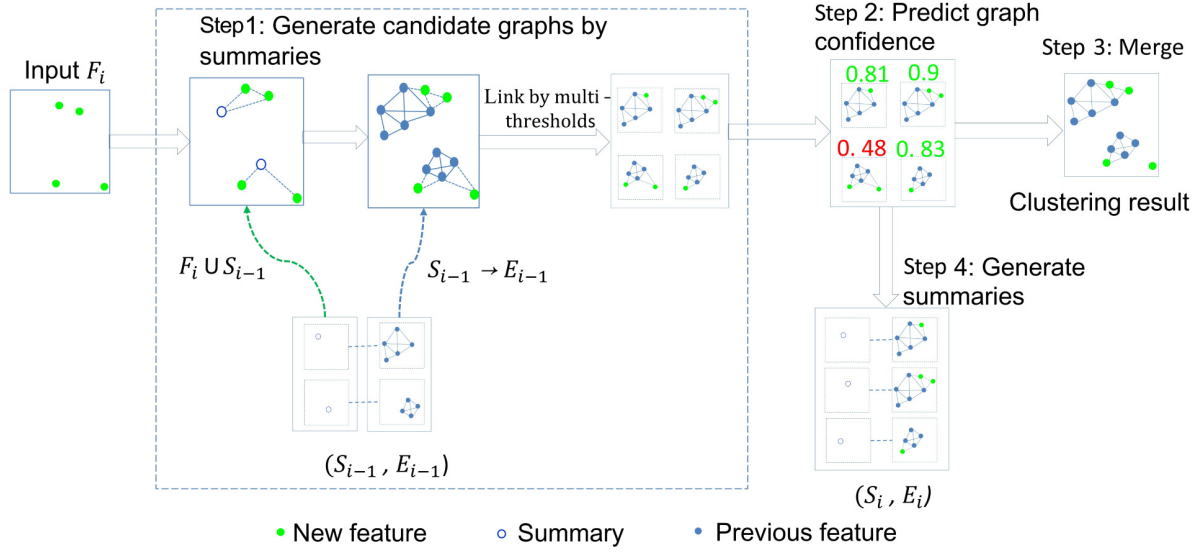


Fig. 2 Main framework of our method for incremental face clustering. When a new batch of face embedding features F_i arrives, we take four steps. In Step 1, we generate candidate graphs by summaries. We cluster F_i with previous summaries S_{i-1} and then change each summary in S_{i-1} to its corresponding features in E_{i-1} , thereby forming a set of clusters. For each cluster, we generate several graphs by multi-thresholds. In Step 2, we predict the confidence score for each graph through a learned GCN. In Step 3, as the candidate graphs overlap, we merge them for clustering results by confidence scores. In Step 4, we generate current summaries S_i and their corresponding features E_i and store them for new data.

using graph confidence and storing them for clustering with the new faces.

The proposed approach is very efficient because the results of candidate graph generation and GCN inference are shared by the clustering and summary generating steps. Algorithm 1 describes the entire process.

3.2.1 Generating candidate graphs by using summaries of previous features

Candidate graphs $G = \{g_i\}_{i=1}^{N_g}$ are a set of graphs in which $g_i = (V, E)$ is a graph. In this study, a graph g is desired to contain as many embedding features of the same person and as less embedding features of other persons as possible. The candidate graphs are utilized to generate clustering results R and estimate the current summaries S_i .

We efficiently generate the candidate graphs by using Algorithm 2. We firstly cluster the new features F_i with the stored previous summaries S_{i-1} . Then, in each cluster, we change each summary in S_{i-1} to its corresponding original embedding features in E_{i-1} and generate a set of candidate graphs via a multi-threshold linking strategy. Step 1 is described in detail in Section 3.3.

Using the stored previous summaries S_{i-1} as a representation of existing features, we avoid calculating global nearest neighbors for all feature nodes to reduce calculations. The process of changing summaries back to their corresponding original embedding features ensures

Algorithm 1 Incremental cluster by learning summaries

Input: new features F_i , previous summaries S_{i-1} and corresponding features E_{i-1} , thresholds for node linkage $\eta = \{\eta_1, \eta_2, \dots\}$, and threshold for graph confidence τ .

Output: current clustering result R , current summaries S_i , and corresponding features E_i

```

1: % generate candidate graphs  $G$  by summaries
2:  $G = \text{Algorithm 2}(F_i, S_{i-1}, E_{i-1}, \eta)$ 
3: for all  $g$  in  $G$  do
4:   predict confidence of graph  $g$  by GCN
5:  $R = \text{MERGE\_GRAPHS\_TO\_CLUSTERS}(G, \eta)$ 
6: % generate summaries
7:  $S_i, E_i = \text{Algorithm 3}(G, \tau)$ 
8: return  $R, S_i, E_i$ 
9:
10: function MERGE_GRAPHS_TO_CLUSTERS( $G$ )
11:   label = 0,  $R = \emptyset$ 
12:   for all  $g$  in  $G$  do
13:     set label of every node in each graph  $g$  to be -1
14:   while  $G \neq \emptyset$  do
15:     select a graph  $g$  with highest confidence from  $G$ 
16:      $G = G \setminus g$ 
17:      $c = \emptyset$ 
18:     for all node  $n$  in  $g$  do
19:       if  $n.\text{label} < 0$  then
20:         add  $n$  to  $c$ 
21:         set  $n.\text{label}$  to be label
22:      $R = R \cup c$ 
23:     label = label + 1
24:   return  $R$ 

```

Algorithm 2 Generate candidate graphs by summaries

Input: new features F_i , previous summaries S_{i-1} and corresponding features E_{i-1} , thresholds for node linkage $\eta = \{\eta_1, \eta_2, \dots\}$.

Output: candidate graphs $G = \{g_i\}_{i=1}^{N_g}$, g_i is a graph

```

1:  $G = \emptyset, \check{\eta} = \text{MIN}(\eta)$ 
2:  $C = \text{CLUSTER\_WITH\_SUMMARIES}(F_i, S_{i-1}, \check{\eta})$ 
3: for all  $c$  in  $C$  do
4:    $\tilde{c} = \text{CHANGE\_SUMMARY\_TO\_FEATURES}(c, S_{i-1}, E_{i-1})$ 
5:    $G = G \cup \text{CONSTRUCT\_GRAPHS}(\tilde{c}, \eta)$ 
6:   return  $G$ 
7: function  $\text{CLUSTER\_WITH\_SUMMARIES}(F_i, S_{i-1}, \check{\eta})$ 
8:    $S_0 = \emptyset$ 
9:    $\tilde{F}_i = F_i \cup S_{i-1}$ 
10:   $\text{Neighbour\_Matrix} = \text{ANN}(\tilde{F}_i)$ 
11:  for all  $(p, q)$  in  $\text{Neighbour\_Matrix}$  do
12:    if  $\text{DISTANCE}(p, q) \leq \check{\eta}$  then
13:      link  $p$  and  $q$ 
14:   $C \leftarrow$  generate clusters by link
15:  return  $C$ 
16:
17: function  $\text{CHANGE\_SUMMARY\_TO\_FEATURES}(\text{Cluster } c, S_{i-1}, E_{i-1})$ 
18:  for all feature  $f$  in  $c$  do
19:    index  $\leftarrow$  find index of  $f$  in  $S_{i-1}$ 
20:    if index  $\geq 0$  then
21:       $c = c \setminus f$ 
22:       $c = c \cup (E_{i-1}[\text{index}])$ 
23:  return  $c$ 
24:
25: function  $\text{CONSTRUCT\_GRAPHS}(\text{Cluster } C_j, \eta)$ 
26:   $G = \emptyset$ 
27:  for all  $\eta_i$  in  $\eta$  do
28:    for all  $(p, q)$  in  $C_j$  do
29:      if  $\text{DISTANCE}(p, q) \leq \eta_i$  then
30:        link  $p$  and  $q$ 
31:   $G = G \cup$  (generate graphs by link)
32:  return  $G$ 

```

the consistency of node density in future clustering. The multi-threshold linking strategy may generate several overlapped candidate graphs for person's identity by a set of different thresholds $\eta = \{\eta_1, \eta_2, \dots\}$. A high linking threshold may generate a tight graph with less neighbor nodes, whereas a low threshold may generate a loose graph with more neighbor nodes.

3.2.2 Estimating confidence for each candidate graph

The confidence score is a float point number ranging from 0 to 1 and is designed to indicate how correct a candidate graph is. A high confidence score is supposed to indicate that the graph is close to the shape of a real

Algorithm 3 Generating summaries

Input: candidate graphs G , threshold for graph confidence τ .

Output: current summaries S and corresponding features E .

```

1: Initial index = 0, set  $C_E = \emptyset$ , set  $C_A = \emptyset$ ,
2: Initial  $S$  as an empty node list,  $E$  as an empty cluster list
3: for all  $g$  in  $G$  do
4:    $C_A = C_A \cup g.\text{Nodes}$ 
5:   if  $g.\text{confidence} \geq \tau$  then
6:      $m \leftarrow$  aggregate  $g$ 
7:      $S[\text{index}] \leftarrow m$ 
8:      $E[\text{index}] \leftarrow g.\text{Nodes}$ 
9:      $C_E = C_E \cup g.\text{Nodes}$ 
10:    index += 1
11: for all  $f$  in  $(C_A \setminus C_E)$  do
12:    $S[\text{index}] \leftarrow f$ 
13:    $E[\text{index}] \leftarrow \{f\}$ 
14:   index += 1
15: return  $S, E$ 

```

face cluster. Furthermore, the confidence score is used to merge the candidate graphs and generate summaries in the further steps.

We use a GCN to learn the structure of real clusters in a supervised manner and then use the trained GCN to infer a confidence score for each candidate graph. The detail of this step is described in Section 3.4.

We learn the structure of a correct graph in a supervised manner, which has been proven in Ref. [1] to be able to generate a prominent performance gain in real-world data. Further more, we design a metric to trade-off between purity and coverage for a candidate graph; this metric is supposed to obtain improved results for our approach. With the strong ability of GCN, we can learn high-dimensional spatial features from graph-structured data and utilize the local context information to help determine the confidence of a graph.

3.2.3 Merging for clustering results

With the estimated graph confidence, we can determine which candidate graphs are more likely to be real clusters and merge the overlapped graphs to become clustering results. For the clustering task, different clusters are not allowed to share the same node. Therefore, the merging process should also split the shared nodes and eliminate the overlap.

We adopt a pseudo label marking strategy similar to the de-overlapping strategy in Ref. [1] to merge the graphs. By contrast, we utilize the pseudo label to mark the nodes and do not require removing the vertices from the graphs. Each time, we fetch a graph with the highest confidence score from the candidate

graph set. We then assign a new pseudo label for each node of the graph if the node has no pseudo label. After all the graphs are fetched, we group the nodes by individual pseudo labels and form the clustering result. The detailed algorithm is described in the function `MERGE_GRAPHS_TO_CLUSTERS` of Algorithm 1.

As analyzed by Ref. [1], the strategy is simple and fast. Compared with the Non-Maximum Suppression (NMS) for de-overlapping the bounding box in object detection (i.e., $O(N^2)$), our merging process has a complexity of $O(N)$.

3.2.4 Generating summaries

With much less number of nodes, the summaries must represent the approximate location of the original embedding features. The summaries of current data $\{F_1, F_2, \dots, F_i\}$ are stored and then used when new data F_{i+1} come.

From candidate graphs G , we select the graphs with the confidence above threshold τ . Then, for each selected graph g , we aggregate all the vertices to obtain a summary by using an aggregation block and add it to the list of S_i . Meanwhile, we add the set of all the features in g to the list of E_i . S_i and E_i are in the data type of list, with $S_i[j]$ as the summary of $E_i[j]$ for each index j . As in Ref. [2], the aggregation block can adopt the following strategies: mean, weighted, and attention aggregation. After the above process, some singleton features in G may not be covered. As a compensation, we set these singleton features as summaries and add them to S_i and E_i . The detailed algorithm is described in Algorithm 3.

To the best of our knowledge, this work is the first to estimate data summaries via supervised learning for incremental clustering. In contrast to traditional methods, our method does not depend on any simple assumptions, such as sphere shape or uniform density. The learned summaries can be more adaptive to arbitrary cluster shapes in the real world. Further more, our summaries are estimated directly from original data rather than the clustering results. This strategy can help avoid error accumulation as new data arrive.

3.3 Generating candidate graphs

In this section, we propose a method to generate candidate graphs through summaries in three steps. Algorithm 2 shows the detailed procedure.

(1) Clustering new features with previous summaries. We firstly merge the new features F_i with the summaries S_{i-1} and form the combined set $\tilde{F}_i$.

Then, we use approximate nearest neighbor^[22] to find the k nearest neighbors for each node in F_i efficiently. We link the neighbor pairs with the distances below threshold $\tilde{\eta}$. Then, we assign a same pseudo label to the linked nodes and form a set of separated clusters C . The detailed procedure is described in the function `CLUSTER_WITH_SUMMARIES` in Algorithm 2.

(2) Changing summaries to their corresponding original features. Now, we have a set of clusters C , with each cluster containing several new features and previous summaries that are adjacent to each other. For every cluster in C , we search each node to find if it is in S_{i-1} . If yes, we obtain the index, find the corresponding features in E_{i-1} , and perform the change operation. The detailed procedure is described in the function `CHANGE_SUMMARY_TO_FEATURES` in Algorithm 2.

(3) Constructing candidate graphs. For every cluster in C , we calculate the distance between any two nodes, use different thresholds to link nodes, and construct a set of overlapped candidate graphs. The detailed procedure is described in the function `CONSTRUCT_GRAPH` in Algorithm 2.

3.4 Estimating confidence for a graph

In this section, we attempt to estimate if the nodes of a graph form a reliable cluster. The pattern contained in a graph is crucial for determining whether its nodes are from the same person. To leverage this, we use GCN to learn and infer the structural pattern in a supervised manner.

3.4.1 Design of confidence metric

Purity^[1] and coverage^[3] are two important metrics for evaluating the confidence of a cluster. These two metrics can be calculated by Eq. (1). Purity encourages a cluster to reject nodes from different identities and thus may result in high precision and low recall. By contrast, coverage encourages a cluster to include as many nodes of the same identities as possible and thus may result in high recall and low precision.

$$\text{purity} = \frac{c \cap \hat{c}}{c}, \quad \text{coverage} = \frac{c \cap \hat{c}}{\hat{c}} \quad (1)$$

where c is the set of nodes in a training graph and $\hat{c}$ is the set of nodes in the ground-truth cluster with the same majority label as c .

Inspired by the F -score^[2] for evaluating the clustering result, we design a fusion metric in Eq. (2) to calculate the confidence metric for each graph and thus balance purity and coverage.

$$\text{fusion} = \frac{2 \times \text{purity} \times \text{coverage}}{\text{purity} + \text{coverage}} = \frac{2 \times (c \cap \hat{c})}{c + \hat{c}} \quad (2)$$

The toy example in Fig. 3 illustrates the effectiveness of the proposed metric. The purity metric is observed to be more sensitive to the error nodes of other identities, whereas the coverage metric is more sensitive to missing nodes of the same identity. The proposed fusion metric, which is a trade-off between the two metrics, is sensitive to missing and error nodes.

3.4.2 Training and inferring network

In our problem, each vertex in a graph may have different numbers of adjacent vertices. This phenomenon is known as non-Euclidean-structured data or graph-structured data, which are not suitable for traditional Convolutional Neural Networks (CNNs). Recently, GCNs have extended CNNs^[23,24] to handle non-Euclidean-structured data. Therefore, we choose the GCN to train and infer the confidence score of candidate graphs.

For a graph g , our GCN network takes the features of all vertices of g as the input of the first layer. Then, we apply L layers of graph convolution. Lastly, a max-pooling layer and a fully connected layer are applied to gather the features of all nodes and output a confidence score.

For each graph convolutional layer with input X , the output Y can be calculated by Eq. (3):

$$Y = \sigma(\tilde{D}^{-1} \tilde{A} X W) \quad (3)$$

where W is a learnable weight matrix to transform the



Fig. 3 Toy example to illustrate the effectiveness of the confidence metric: (a) perfect cluster, (b) an error node, (c) a missing node, and (d) an error node and a missing node. The proposed fusion metric is sensitive to error and missing nodes.

embeddings; $\sigma(\cdot)$ is a non-linear activation function such as Rectified Linear Unit (ReLU); $\tilde{A} = A + \text{IM}$, A is the adjacency matrix of the graph and IM is an identity matrix. Multiplying with $\tilde{A}$ helps each vertex integrate information with neighbor nodes, including itself. $\tilde{D}$ is a diagonal degree matrix, which is used to keep the scale of features to be 1 after convolution.

At the training stage, we optimize the mean square error between the predicted scores and the proposed fusion scores. At the testing stage, we predict the confidence score of a graph by using the trained model.

4 Experiment

4.1 Dataset and evaluation metric

4.1.1 Dataset

MS-Celeb-1M. We construct separate training and testing sets from the MS-Celeb-1M^[25] dataset. MS-Celeb-1M is a well-known large-scale face dataset. The dataset is gathered from webpages and contains approximately 10 000 000 photos from 1 000 000 celebrities. As the dataset is somewhat noisy, we use a cleaned version^[1] of MS-Celeb-1M and select two separated parts for training and testing, respectively. **The training dataset** contains 8600 persons with 58 000 images. We then construct two testing datasets. **The MS small testing dataset** contains 8600 persons with 580 000 images, whereas **the MS big testing dataset** contains 42 800 persons with 2 890 517 images. We split the images and guarantee that no person overlaps between the training and the testing sets. In real-world applications, the face systems usually process new data periodically, such as hourly or daily. To simulate the situation, we randomly split each testing set into 10 batches. In the testing stage, the algorithm sequentially obtains the 10 batches of data. That is, the algorithm firstly obtains the photos in the first batch. After clustering, the algorithm obtains the second batch then the third until the 10th batch.

IJB-B. In this test, we construct training and testing set from different datasets. **The training dataset** contains 5000 persons with approximately 200 000 images, which are randomly selected from the CASIA dataset^[26]. **The testing dataset** contains 1844 persons with 63 766 images from the largest clustering subtask of the IJB-B^[27] dataset. No person overlaps between the training and the testing sets. We also randomly split the testing set into 10 batches to simulate the situation of the increasing data clustering.

4.1.2 Evaluation metric

To evaluate the average accuracy of the incremental clusters in all batches, we adopt two evaluation metrics: AF and Average Normalized Mutual Information (ANMI). AF is defined as

$$AF = \frac{2}{M} \sum_{i=1}^M \frac{\text{Precision}_i \times \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i} \quad (4)$$

where Precision_i denotes the pairwise precision of the i -th batch, Recall_i denotes the pairwise recall of the i -th batch, and $M = 10$ denotes the total batches.

ANMI is defined as

$$ANMI = \sum_{i=1}^M \frac{2 \times I(c, \hat{c})}{M \times \sqrt{H(c)H(\hat{c})}} \quad (5)$$

where $I(\cdot)$ calculates the mutual information and $H(\cdot)$ represents the entropy.

To evaluate the computation efficiency, we use the Average Time (AT) taken for incremental M batches.

$$AT = \frac{1}{M} \sum_{i=1}^M t_i \quad (6)$$

where t_i is the time consumed for the incremental clustering of photos of the i -th batch.

4.2 Method comparison

We compare the proposed method with two categories of clustering methods. One category comprises methods that support incremental clustering: incremental K -means (i- K means)^[10], incremental Density-Based Spatial Clustering of Applications with Noise (i-DBSCAN)^[11], Basic Sequential Algorithm Scheme (BSAS)^[28], and Modified Basic Sequential Algorithm Scheme (MBSAS)^[29]. The other category consists of the methods that do not have incremental strategies: Hierarchical Agglomerative Clustering (HAC)^[7], ARO^[3], Xmeans^[30], affinity graph^[1], and linkage graph^[2]. For these static clustering methods, we merge new data with existing data for full data clustering when a new batch of photos arrives.

Implementation details. We use Python 3.6 and PyTorch 0.4.0 to implement our method. The experiments are run on a single Intel Xeon Gold 6128 CPU clocked at 3.4 GHz, with a TiTAN XP GPU. The system memory is 128 GB. We use Faiss implementation of k NN, with $k = 5$. For the MS small and big testing datasets, we set $\eta = (0.65, 0.7)$ and $\tau = 0.6$. For the IJB-B testing set, we set $\eta = (0.4, 0.45)$ and $\tau = 0.45$. We carefully tune the hyperparameters for each method and report the best results. We use constant

hyperparameters for each batch because adjusting hyperparameters frequently for new data is difficult in real-world applications.

Comparison with incremental methods. From the result of several incremental methods shown in Table 1, we can observe that the algorithms making less assumptions on cluster shape or uniform density can obtain better results. Furthermore, the algorithms that need dynamical parameters as the data distribution and scale change are not suitable for this task. For instance, i- K means needs to appoint the number of clusters, which is difficult to estimate accurately and changes as the data increase. BSAS uses a static clustering threshold to design if a new photo should be assigned to an existing cluster or a new cluster. As a result, the algorithm requires all the distance between the photos to be aligned; moreover, the clustering results highly depend on the order of input data. MBSAS improves BSAS by setting two different thresholds. The lower threshold is for assigning to a previous cluster, whereas the higher one is for creating a new cluster. If the distance is between the two thresholds, the algorithm defers the decision. All the above three methods assume that the clusters are sphere shaped, thus limiting the performance for unrestrained face clustering. For i-DBSCAN, the algorithm is fast, and the number of clusters does not need to be set. However, the algorithm is not good at handling clusters that vary in density. In contrast to the above methods, our method does not make any assumption on cluster shape or density when learning summaries of the data distribution. Therefore, our method outperforms previous incremental clustering methods.

Table 1 Results on the MS small testing dataset. Comparison with baseline and state-of-the-art methods in terms of the AF, ANMI, and AT of 10 batches. For all methods, we tune the corresponding hyperparameters and report the best result.

Method	AF	ANMI	AT (s)
i- K means ^[10]	0.375	0.856	849
BSAS ^[28]	0.462	0.859	342
MBSAS ^[29]	0.470	0.898	1383
i-DBSCAN ^[10]	0.644	0.925	109
Xmeans ^[30]	0.540	0.883	136 800
HAC ^[7]	0.637	0.934	20 877
ARO ^[3]	0.192	0.862	367
Affinity ^[1]	0.740	0.951	339
Linkage ^[2]	0.692	0.957	1099
Our method	0.762	0.955	257

Comparison with static methods. Table 1 also shows the results of several static clustering methods on the MS small testing dataset. Table 2 shows the results of two state-of-the-art static clustering methods on the MS big testing dataset. Table 3 shows the results of two state-of-the-art static clustering methods on the IJB-B testing dataset. Naturally, the incremental face clustering task is much harder than clustering with fixed data size, because the data distribution is more unbalanced and varies in each batch. Xmeans makes an improvement on Kmeans by iteratively searching for an optimum K . However, the algorithm still relies on the assumption of a sphere-shaped cluster, and the strategy for search K considerably increases the computation cost. The accuracy of HAC is higher than that of Xmeans because the algorithm is more robust on data distribution. Although ARO is efficient and does not make any simple assumption on data shape or density, the results are low probably because of its weak generalization ability. As state-of-the-art face clustering algorithms based on supervised graph learning, the affinity and linkage graphs outperform the previous methods. However, as static methods, these two methods do not have special strategy for incremental clustering. The linkage graph consumes a considerable amount of time because it needs to perform graph inference for each face image. Our proposed method is comparable to state-of-the-art, in terms of the support for incremental clustering and less computation cost.

4.3 Analysis and discussion

4.3.1 Accuracy vs. efficiency

Figure 4 shows AF vs. AT of 10 batches on the MS

Table 2 Results on the MS big testing dataset. We compare the proposed method with state-of-the-art methods in terms of the AF, ANMI, and AT of 10 batches. For all methods, we tune the corresponding hyperparameters and report the best result.

Method	AF	ANMI	AT (min)
Affinity ^[1]	0.660	0.950	237
Linkage ^[2]	0.661	0.952	554
Our method	0.672	0.951	203

Table 3 Results on the IJB-B testing dataset. We compare our method with state-of-the-art methods in terms of the AF, ANMI, and AT of 10 batches. For all methods, we tune the corresponding hyperparameters and report the best result.

Method	AF	ANMI	AT (s)
Affinity ^[1]	0.677	0.934	41
Linkage ^[2]	0.669	0.937	126
Our method	0.690	0.936	31

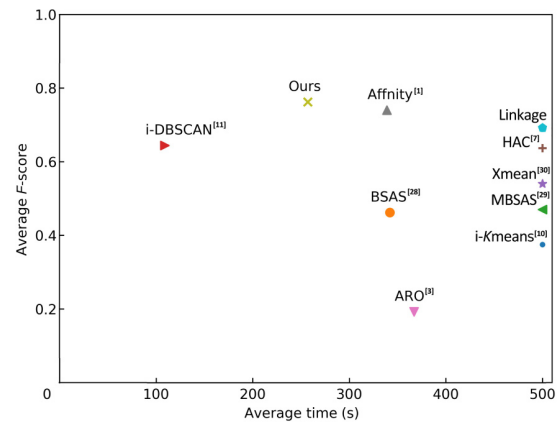


Fig. 4 AF of 10 batches vs. AT (s). The closer to the top left, the better. For easy display, we set a maximum AT of 500 s. AT for i -Kmeans, BSAS, MBSAS, Xmeans, and HAC is 849, 1876, 1383, 136 860, and 20 877 s, respectively.

small testing dataset, including incremental and static clustering methods. Incremental clustering methods, such as i -DBSCAN, BSAS, and our method, usually have less computation cost. The affinity graph, linkage graph, and the proposed method make no assumption on data distribution and thus can obtain higher accuracy. Compared with affinity graph, linkage graph, and i -DBSCAN, our method can trade-off between efficiency and accuracy.

In our experiments, the proposed method reaches comparable accuracy with state-of-the-art full clustering methods. The possible reason may be as follows: (1) We utilize the incremental clustering test protocol in our experiments. In most incremental clustering tasks, people usually cluster new photos hourly or daily, especially with limited computation and memory resources. To simulate this situation, many incremental clustering approaches process new data in chunks^[13–15], that is, in batches. (2) For full clustering methods, we also apply the incremental clustering test protocol. In the testing stage, the algorithm sequentially obtains the 10 batches of data. Without the benefit of having observed the entire data distribution, incremental face clustering is more challenging than clustering static face datasets. We use constant hyperparameters for all batches because adjusting hyperparameters frequently for new data is difficult in real-world applications. The results presented are the average of 10 batches. (3) We design our algorithm aiming at incremental face clustering applications. We predict the summaries of previous data directly from data distribution via supervised learning and design a confidence metric that can trade off well between precision and recall.

4.3.2 Choice of graph confidence metrics

To compare the designed choices of metrics for graph confidence, we train and test GCN with different metrics; the results are shown in Table 4. When the coverage metric is chosen, the method reaches the highest recall because this metric encourages the graph to cover the ground truth as much as possible. However, coverage does not punish the photos of other identities in the graph, thus resulting in low precision. By contrast, purity encourages the graph to obtain only “pure” photos of the same identity but does not encourage to cover the ground truth. As a result, purity metric obtains the highest precision but lowest recall. The proposed metric takes a similar fusion function as the evaluation metric F -score and can trade off well between coverage and purity. Our method obtains a moderate score in precision and recall but reaches the highest value in AF and ANMI.

4.3.3 Analysis of hyperparameter

Influence of η . Table 5 shows the results on the IJB-B testing dataset with different thresholds η for node linkage. AF and ANMI reach the maximum value when $\eta = (0.40, 0.45)$. The computational time decreases as η increases.

Influence of τ . Figure 5 shows the results on the IJB-B testing dataset with different graph confidence threshold τ for generating summaries. During the first stage, about $\tau \leq 0.45$, AF and ANMI increase as AT declines probably because of the removal of low-quality summaries. During the second stage, about $0.45 < \tau \leq 0.55$, AF and ANMI decrease probably because of the removal of summaries with high quality. During this stage, AT decreases firstly and then increases probably because our strategy sets singletons as summaries. As τ rises, additional singletons are released from candidate graphs. The increase in summaries causes the rise of

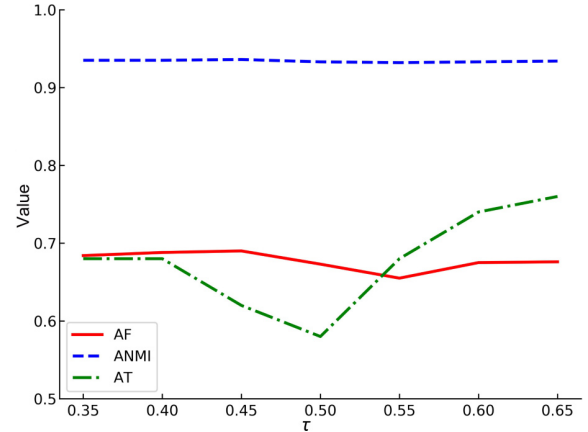


Fig. 5 AF, ANMI, and AT of 10 batches vs τ on the IJB-B testing set. For improved view, we divide AT by 50 s in the figure.

AT. During the third stage, about $\tau > 0.55$, AF, ANMI, and AT increase. During this stage, additional singletons of high-quality faces are released and set as summaries. The increase in summaries generated by high-quality singletons may improve accuracy. In all these stages, the changes of AF and ANMI are not large probably because learning-based adjustment mechanisms occur in the subsequent batches.

5 Conclusion

In this study, the following problem is investigated: clustering faces into identities with new faces arrive in batches. Compared with clustering with a fixed data size, incremental face clustering is a more challenging task because the data distribution is more unbalanced and varies in each batch. Conventional approaches rely on the statistical information of previous clusters to improve the efficiency of incremental clustering; thus, errors accumulate, especially when previous clusters contain errors. Our motivation is to learn cluster shapes in a supervised manner and utilize the learned model to predict summaries. The advantage of this strategy is that the generation of summaries does not rely on any simple assumptions of special shape or uniform density. Furthermore, the summaries are estimated directly from original data distribution and not from the previous clustering results. Therefore, the strategy can avoid error accumulation to some extent. Experiments show that the proposed approach significantly outperforms the existing incremental face clustering methods, with an AF improved from 0.644 to 0.762. Compared with state-of-the-art static face clustering methods, our method can yield comparable

Table 4 Comparison with different metrics for learning graph confidence.

Metric	Precision	Recall	AF	ANMI
Coverage	0.801	0.706	0.750	0.954
Purity	0.912	0.616	0.735	0.946
Fusion	0.839	0.699	0.762	0.955

Table 5 AF, ANMI, and AT of 10 batches vs. η on the IJB-B testing set.

η	AF	ANMI	AT (s)
(0.30, 0.35)	0.624	0.918	44
(0.35, 0.40)	0.670	0.929	35
(0.40, 0.45)	0.690	0.936	31
(0.45, 0.50)	0.672	0.934	30
(0.50, 0.55)	0.652	0.927	29

accuracy while consuming much less time.

The main contributions of this study are threefold. (1) To the best of our knowledge, this work is the first to estimate data summaries via supervised learning for incremental clustering. (2) An efficient framework is proposed to cluster previous summaries with new data. (3) A metric to indicate graph confidence, which can trade-off well between precision and recall, is presented.

In future research, the proposed method must be optimized in spite of it working reasonably well for incremental face clustering. The memory cost is relatively high, especially when the number of images for clustering is large. Compared with traditional incremental clustering methods, such as incremental DBSCAN, our method still consumes more run time. In addition, low-quality faces (i.e., low resolution, profile, blurred, and occlusion) prevent the improvement of clustering accuracy. We will try to evaluate and utilize face quality to generate initial candidate clusters quickly and then perform weighted graph reasoning in accordance with face quality.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Nos. 61701277 and 61771288) and the State Key Development Program in 13th Five-Year (Nos. 2016YFB0801301, 044007008, and 2016YFB1001005). This work was also supported by the National Engineering Laboratory for Intelligent Video Analysis and Application of China.

References

- [1] L. Yang, X. H. Zhan, D. P. Chen, J. J. Yan, C. C. Loy, and D. H. Lin, Learning to cluster faces on an affinity graph, in *Proc. 2019 IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, pp. 2293–2301.
- [2] Z. D. Wang, L. Zheng, Y. L. Li, and S. J. Wang, Linkage based face clustering via graph convolution network, in *Proc. 2019 IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, pp. 1117–1125.
- [3] C. Otto, D. Y. Wang, and A. K. Jain, Clustering millions of faces by identity, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 2, pp. 289–303, 2018.
- [4] S. Lloyd, Least squares quantization in PCM, *IEEE Trans. Inf. Theory*, vol. 28, no. 2, pp. 129–137, 1982.
- [5] J. Shi and J. Malik, Normalized cuts and image segmentation, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, 2000.
- [6] M. Ester, H. P. Kriegel, J. Sander, and X. W. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in *KDD-96 Proc.*, Menlo Park, CA, USA, 1996, pp. 226–231.
- [7] R. Sibson, SLINK: An optimally efficient algorithm for the single-link cluster method, *Comput. J.*, vol. 16, no. 1, pp. 30–34, 1973.
- [8] N. Ailon, R. Jaiswal, and C. Monteleoni, Streaming k -means approximation, presented at Neural Processing Systems 22: 23rd Annu. Conf. Neural Information Processing Systems, Vancouver, Canada, 2009, pp. 10–18.
- [9] M. Shindler, A. Wong, and A. Meyerson, Fast and accurate k -means for large datasets, in *Proc. of the 24th International Conference on Neural Information Processing Systems*, Granada, Spain, 2011, pp. 2375–2383.
- [10] A. M. Bagirov, J. Ugon, and D. Webb, Fast modified global k -means algorithm for incremental cluster construction, *Pattern Recognit.*, vol. 44, no. 3, pp. 866–876, 2011.
- [11] M. Ester, H. P. Kriegel, J. Sander, M. Wimmer, and X. W. Xu, Incremental clustering for mining in a data warehousing environment, in *Proc. 24th Int. Conf. Very Large Data Bases*, New York, NY, USA, 1998, pp. 323–333.
- [12] P. A. Mansfield, Q. Wang, C. Downey, L. Wan, and I. L. Moreno, Links: A high-dimensional online clustering method, arXiv preprint arXiv: 1801.10123, 2018.
- [13] P. Kulshreshtha and T. Guha, An online algorithm for constrained face clustering in videos, in *Proc. 25th IEEE Int. Conf. Image Proc. (ICIP)*, Athens, Greece, 2018, pp. 2670–2674.
- [14] D. Arthur and S. Vassilvitskii, k -means++: The advantages of careful seeding, in *Proc. 18th Annu. ACM-SIAM Symp. on Discrete Algorithms*, Philadelphia, PA, USA, 2007.
- [15] Y. F. Sun and Y. S. Lu, A grid-based subspace clustering algorithm for high-dimensional data streams, in *Proc. Int. Conf. Web Information Systems*, Wuhan, China, 2006.
- [16] A. Y. Zhou, F. Cao, W. N. Qian, and C. Q. Jin, Tracking clusters in evolving data streams over sliding windows, *Knowl. Inf. Syst.*, vol. 15, no. 2, pp. 181–214, 2008.
- [17] A. Namadchian and G. Esfandani, DSCLU: A new data stream clustering algorithm for multi density environments, in *Proc. 13th ACIS Int. Conf. Software Engineering, Artificial Intelligence, Networking and Parallel Distributed Computing (SNPD)*, Kyoto, Japan, 2012.
- [18] W. L. Hamilton, Z. Ying, and J. Leskovec, Inductive representation learning on large graphs, arXiv preprint arXiv: 1706.02216, 2017.
- [19] T. N. Kipf and M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint arXiv: 1609.02907, 2017.
- [20] S. J. Yan, Y. J. Xiong, and D. H. Lin, Spatial temporal graph convolutional networks for skeleton-based action recognition, arXiv preprint arXiv: 1801.07455, 2018.
- [21] Z. M. Chen, X. S. Wei, P. Wang, and Y. W. Guo, Multi-label image recognition with graph convolutional networks, in *Proc. 2019 IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, pp. 5172–5181.
- [22] Y. C. Shi, C. Otto, and A. K. Jain, Face clustering: Representation and pairwise constraints, *IEEE Trans. Inf. Forens. Secur.*, vol. 13, no. 7, pp. 1626–1640, 2018.
- [23] Y. L. Li, S. J. Wang, Q. Tian, and X. Q. Ding, A

survey of recent advances in visual feature detection, *Neurocomputing*, vol. 149, pp. 736–751, 2015.

- [24] Y. L. Li, S. J. Wang, Q. Tian, and X. Q. Ding, Feature representation for statistical-learning-based object detection: A review, *Pattern Recognit.*, vol. 48, no. 11, pp. 3542–3559, 2015.
- [25] Y. D. Guo, L. Zhang, Y. X. Hu, X. D. He, and J. F. Gao, MS-celeb-1m: A dataset and benchmark for large-scale face recognition, in *Proc. 14th European Conf. Computer Vision*, Amsterdam, the Netherlands, 2016.
- [26] D. Yi, Z. Lei, S. C. Liao, and S. Z. Li, Learning face representation from scratch, arXiv preprint arXiv: 1411.7923, 2014.
- [27] C. Whitelam, E. Taborsky, A. Blanton, B. Maze, J. C. Adams, T. Miller, N. D. Kalka, A. K. Jain, J. A. Duncan, K. Allen, et al., IARPA Janus benchmark-B face dataset, in *Proc. 2017 IEEE Conf. Computer Vision and Pattern Recognition Workshops (CVPRW)*, Honolulu, HI, USA, 2017, pp. 592–600.
- [28] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. 4th ed. San Diego, CA, USA: Elsevier, 2009.
- [29] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. 2nd ed. New York, NY, USA: John Wiley & Sons, 2001.
- [30] D. Pelleg and A. Moore, *X-means: Extending k-means with efficient estimation of the number of clusters*, in *Proc. 17th Int. Conf. Machine Learning*, Pattsburgh, PA, USA, 2000.



computer vision, video analysis, etc.

Xuan Zhao is a PhD candidate at the Department of Electronic Engineering, Tsinghua University, advised by Prof. Shengjin Wang. Before that, he received the bachelor and master degrees from Tsinghua University in 2004 and 2007, respectively. His research interests include image processing, pattern recognition,



computer vision, video analysis, etc.

Yali Li received the BE degree with excellent graduates award from Nanjing University in 2007 and the PhD degree from Tsinghua University in 2013. Currently, she is a research assistant at Department of Electronic Engineering, Tsinghua University. Her research interests include image processing, pattern recognition,



Zhongdao Wang is a PhD candidate at the Department of Electronic Engineering, Tsinghua University, advised by Prof. Shengjin Wang. Before that, he received the bachelor degree in mathematics and physics from Tsinghua University in 2017. His research interests are computer vision and machine learning.



Shengjin Wang received the BEng from Tsinghua University in 1985 and the PhD degree from Tokyo Institute of Technology, Tokyo, Japan in 1997. From May 1997 to August 2003, he was a member of the research staff at Internet System Research Laboratories, NEC Corporation, Japan. Since September 2003, he has been a professor at Department of Electronic Engineering, Tsinghua University, where he is currently also a director of the Research Institute of Image and Graphics (RIIG). He has published more than 50 papers on image processing. He is the holder of ten patents. His current research interests include image processing, computer vision, video surveillance, and pattern recognition.



Lei Gao received the PhD degree in computer science from Beihang University, Beijing, China in 2008. He is an associate professor at the First Research Institute of Ministry of Public Security. His current research interests include computer vision and artificial intelligence.