Structure-Aware Face Clustering on a Large-Scale Graph with 10^7 Nodes

Shuai Shen[1,2], Wanhua Li[1,2], Zheng Zhu[1,2], Guan Huang[3], Dalong Du[3], Jiwen Lu[1,2,*], Jie Zhou[1,2]
[1] Department of Automation, Tsinghua University, China
[2] Beijing National Research Center for Information Science and Technology, China
[3] XForwardAI
{shens19,li-wh17}@mails.tsinghua.edu.cn;zhengzhu@tsinghua.edu.cn;
{guan.huang,dalong.du}@xforwardai.com;{lujiwen,jzhou}@tsinghua.edu.cn

Abstract

Face clustering is a promising method for annotating unlabeled face images. Recent supervised approaches have boosted the face clustering accuracy greatly, however their performance is still far from satisfactory. These methods can be roughly divided into global-based and local-based ones. Global-based methods suffer from the limitation of training data scale, while local-based ones are difficult to grasp the whole graph structure information and usually take a long time for inference. Previous approaches fail to tackle these two challenges simultaneously. To address the dilemma of large-scale training and efficient inference, we propose the STructure-AwaRe Face Clustering (STAR-FC) method. Specifically, we design a structure-preserved subgraph sampling strategy to explore the power of large-scale training data, which can increase the training data scale from 10^5 to 10^7. During inference, the STAR-FC performs efficient full-graph clustering with two steps: graph parsing and graph refinement. And the concept of node intimacy is introduced in the second step to mine the local structural information. The STAR-FC gets 91.97 pairwise F-score on partial MS1M within 310s which surpasses the state-of-the-arts. Furthermore, we are the first to train on very large-scale graph with 20M nodes, and achieve superior inference results on 12M testing data. Overall, as a simple and effective method, the proposed STAR-FC provides a strong baseline for large-scale face clustering. Code is available at https://sstzal.github.io/STAR-FC/.

1. Introduction

Recent years have witnessed the great progress of face recognition [9, 28, 29, 38, 39, 41]. Large-scale datasets are an important factor in the success of face recognition and there is an increasing demand for larger-scale data. Face clustering [22, 30, 42, 48, 50, 51, 52] is a natural way to solve the data annotation problem so as to make better use of massive unlabeled data. Face clustering is also one possible approach to organize and file large volumes of real face images in social media or other application scenarios.

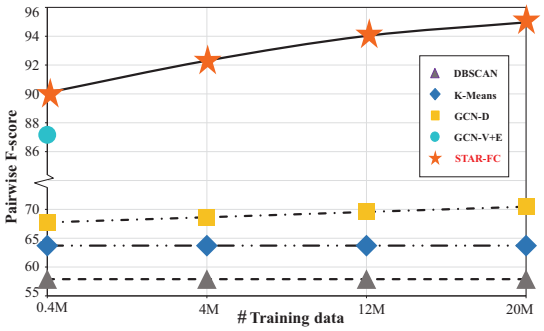


Figure 1: Method comparison when training with different scales of data and testing on 12M data from WebFace42M [55]. The proposed STAR-FC can fully explore the power of large-scale training data. GCN-V+E fails to handle larger training graph while GCN-D's performance is severely restricted due to the less consideration of the global structural information.

Recently a variety of efforts have been devoted to face clustering. Traditional unsupervised methods [16, 53] including K-Means [30] and DBSCAN [10] usually depend on some manually designed clustering strategies. They perform well on small datasets, however they are less effective when dealing with large-scale data as shown in Figure 1. Recent research trends [12, 42, 47, 49, 50] turn to the GCN-based supervised learning. These methods are performed based on the affinity graph and can be roughly divided into global-based and local-based ones *according to whether their GCN input is the whole graph or not*. The representative global-based method GCN-V+E [47] uses the entire graph for GCN training. As shown in Figure 1, it boosts the face clustering performance greatly compared with unsupervised methods, however the training data scale

* Corresponding author

is limited by the GPU memory which makes it difficult to further explore the power of larger-scale training sets. Although local-based methods such as GCN-D [49] shown in Figure 1 don't suffer from memory restrictions, its performance is limited since it lacks the comprehension of global graph structure. Besides, it organizes the data as overlapped local subgraphs which leads to inefficient inference.

For many computer vision tasks [7, 8, 11, 15, 25, 34], large-scale training data is one of the most important engines to promote the performance. With the emergence of some new large-scale benchmarks such as WebFace42M [55] whose data volume is ten times that of MS1M [13], we have more data available for training. Therefore, exploring the power of these rich training data is imperative. For testing, efficiency matters, we are therefore eager to perform full graph inference. Based on the above motivations, we propose a structure-aware face clustering method STAR-FC to address the dilemma of large-scale training and efficient inference. Specifically, we design a GCN [20] based on the KNN [6] affinity graph to estimate the edge confidence. Furthermore, a structure preserved subgraph sampling strategy is proposed for larger-scale GCN training. During inference, we perform face clustering with two steps: graph parsing and graph refinement. In the second step, node intimacy is introduced to mine the local structural information for further refinement. In the inference process, the entire graph is taken as the input for efficiency.

The experiments demonstrate that with these structure-aware designs, the STAR-FC can not only perform sample-based training but also implement full-graph inference. With sample-based training, the training data scale can be increased by two orders of magnitude from 10^5 to 10^7 and beyond. As shown in Figure 1, with the increase of training data, our method has been consistently improved and finally achieves 95.1 *pairwise F-score*. Interestingly, we find that such sampling method does not lead to performance loss and brings some extra accuracy gain since it enhances the generalization of the model. In inference, with full-graph as input, the efficiency can be guaranteed. We achieve state-of-the-art face clustering results on partial MS1M within 310s. What's more, we can complete inference on 12M data within 1.7h thus provide a strong baseline for face clustering. To summarize, we make the following contributions:

- To fully explore the power of large-scale training dataset, we propose a structure-preserved subgraph sampling strategy which can break through the limitation of training data scale from 10^5 to 10^7.
- For inference, we take the entire graph as input to ensure efficiency. We transform face clustering into two steps: graph parsing and graph refinement. Node intimacy is introduced in the second step to explore the local structure for further graph refinement.
- The proposed STAR-FC achieves 91.97 *F-score* on partial MS1M within 310s. Moreover, we are the first to conduct large-scale training on 20M data which provides a strong baseline for large-scale face clustering.

2. Related Work

Face Clustering. Face clustering has been extensively studied as a classic task in machine learning. It provides an alternative way to exploit massive unlabeled data. Traditional algorithms including K-Means [30], spectral clustering [16], hierarchical clustering [53] and DBSCAN [10] laid a good theoretical foundation for clustering. However, they generally rely on simple data distribution assumptions thus are ineffective when dealing with real data. To improve the robustness in complex distributed face clustering, Lin *et al.* [27] proposed the proximity-aware hierarchical clustering. Zhu *et al.* [54] and Otto *et al.* [32] designed the rank-order connection metric. However, since [32, 54] did not establish the graph structure and lacked preliminary analysis for neighbor relations, they achieved poor results. Lin *et al.* [26] tried to measure density affinities between local neighborhoods. [36] modeled face clustering as a structured prediction problem using conditional random field.

Methods above make less use of supervised information in face clustering. More recently, the research trends turn to GCN-based supervised face clustering and have achieved impressive results [12, 42, 47, 48, 49, 50, 52]. These methods can be roughly divided into two categories: local-based face clustering [42, 49, 50] and global-based one [47]. In these local-based methods, Zhan *et al.* [50] designed a mediator network to aggregate information in the local graph. Wang *et al.* [42] predicted the linkage in an instance pivot subgraph. Yang *et al.* [49] generated a series of subgraphs as proposals and detected face clusters thereon. These methods pay more attention to local graph information and rely heavily on redundancy subgraph operations which limit their performance and lead to slow inference. Representative global-based method [47] took the entire graph as input and predicted the confidence and connectivity of all vertices. In [47], the holistic graph structure is better considered, however due to GPU memory limitation, it may be out of memory when dealing with larger training data. We therefore propose the STAR-FC to simultaneously tackle the challenges of large-scale training and efficient inference.

Graph Convolutional Networks. Graph Convolutional Networks (GCNs) [4, 35, 40, 43] extend the convolution idea of CNNs [5, 21, 33] to process non-Euclidean structured data. GCNs have shown impressive capability on various tasks [2, 20, 23, 24, 45, 46]. More recently, to improve GCN's ability in handling larger-scale training graph, some GCN training algorithms have been proposed. GraphSAGE [14] traded off the performance and runtime via sam-

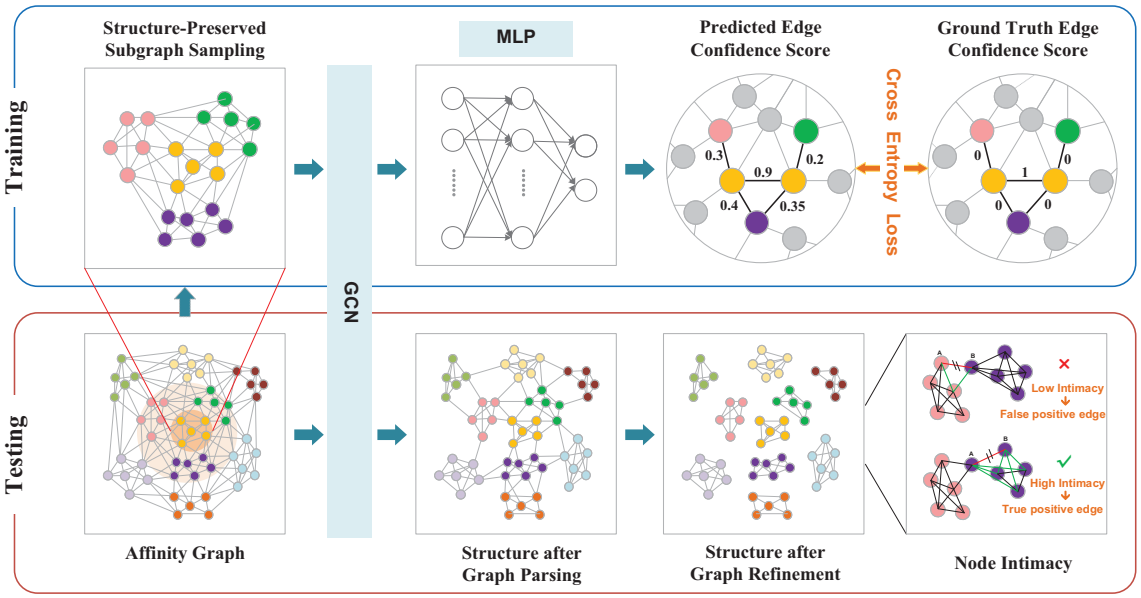


Figure 2: Overview of the proposed STAR-FC framework. In the training process, we use the structure-preserved subgraph sampling strategy to get a variety of subgraphs which are used to train the GCN-based edge confidence estimator. The cross-entropy loss is employed to supervise the training. During inference, based on the built affinity graph, we transform the face clustering into two steps: graph parsing and graph refinement. For the first step, the trained GCN takes the entire graph as input and estimates all edge confidence scores simultaneously. The affinity graph is parsed with these predicted scores. In the second step, node intimacy is employed for further graph refinement. After these two steps, the cluster structure will become clear and the face clusters can be directly read from the graph. For details about the node intimacy, please refer to Figure 4.

pling a fixed number of neighbors for aggregation. FastGCN [3] interpreted graph convolutions as integral transforms of embedding functions under probability measures and proposed to sample vertices rather than neighbors for controllable computation cost. Whereas the key differences between the above methods and the proposed one lie in the sample mode. These previous methods perform graph sampling with nodes as the smallest unit, while the proposed method implements sampling on clusters with near neighbor relationships trying to approximate global structure. Our method can keep most of the inter-cluster edges which can be provided as hard negative samples during training.

3. Methodology

3.1. Overview

To address the dilemma of large-scale training and efficient inference, we propose a structure-aware face clustering method. An overview of the proposed STAR-FC is shown in Figure 2. During training, the GCN-based edge confidence estimator is trained with the structure-preserved subgraph sampling strategy. We aim to approximate the full graph structure with the sampled subgraph and it retains most of the hard negative edges which contribute a lot for training. In this way, the potential of large-scale data can be fully unleashed. We specifically model the edge prediction as a binary classification problem and use the cross-entropy loss for supervising. During inference, we transform the face clustering into two steps: graph parsing and graph refinement. For graph parsing, we take the whole graph as the input of the trained GCN to estimate all edge confidence scores simultaneously. Then edges with low scores are removed and graph structures will become clearer. However there still exist a few wrong connections. They have relatively high scores thus hard to be eliminated. To further refine the graph, we introduce the node intimacy for edge pruning again. After these two steps, face clusters are naturally formed by those connecting groups in the graph.

3.2. Large-Scale GCN Training

In this section, we detail the large-scale training process of the proposed STAR-FC.

Design of the GCN. In this step, we design a GCN-based edge confidence predictor on the basis of KNN affinity graph. We first get the feature matrix $F \in \mathbb{R}^{N \times D}$ with a trained ResNet-50, where N is the number of face images and D is the feature dimension. To build the KNN affinity graph, each sample can be treated as a node in the graph and is linked to its K nearest neighbours [6]. The corresponding sparse symmetric adjacency matrix is $A \in \mathbb{R}^{N \times N}$.

Since the CNN is trained under the strong supervision of classification loss, the extracted features F actually contain rich identity information. However, due to intra-class variance and the fixed K value in KNN algorithm, the affinity graph may contain many wrong edge connections. We therefore try to directly predict the existence of the edges employing a GCN for propagating neighbor information. Following [42, 47], we use the more effective modified GCN as our backbone and the computational process of a L-layer modified GCN can be formulated as:

$$F_{l+1} = \sigma\left([F_l^T, (\widetilde{A}F_l)^T]^T W_l\right), \tag{1}$$

where $\widetilde{A} = \widetilde{D}^{-1}(A + I)$. $\widetilde{D}$ is a diagonal degree matrix with $\widetilde{D} = \sum_j \widetilde{A}_{ij}$. F_l denotes the embeddings at l-th layer and F_0 is the input face features. $W_l \in \mathbb{R}^{D_{in} \times D_{out}}$ is a learnable matrix that maps the embeddings to a new space. σ is a nonlinear activation and we use ReLU [31, 44] in this work. F_L indicates the output features of L-layer.

Since F_L aggregates many messages from the neighborhood and encode graph structural information, it is more suitable for the face clustering task. To predict the existence of the edges in the affinity graph, we design a binary classifier employing a 2-layer MLP [17] with the objective to minimize the cross-entropy loss between the predicted edge confidence and the ground-truth edge labels. Particularly, we take pair features corresponding to the edges in the affinity graph as the input of the MLP and get the two-dimension predicted edge confidence. The ground-truth label of an edge is 1 if the two nodes connected by this edge belong to the same class, otherwise it will be 0.

Under this simple supervision of binary signals, the distribution of the predicted confidence will appear as two sharp peaks approaching 0 and 1 as shown in Figure 3. Therefore, for inference we can use a single threshold τ_1 to effectively eliminate most of the wrong edges. This operation may lead to two types of misjudgments: (1) it may cut off a small number of real edges; (2) it may leave a few wrong connections hard to be identified via confidence. Since the original affinity graph is densely connected, the lost correct edges in the former case have a slight effect on the connectivity of the final graph. Those remaining wrong edges will be processed in the following procedure with node intimacy in Section 3.3.

Structure-Preserved Subgraph Sampling. Previous methods [47, 49] usually use 10% of MS1M (0.5M face images) [13] for GCN training. For global-based method [47] this has approximated to the memory threshold of a typical 1080Ti GPU. Although local-based methods [42, 49] can alleviate the GPU storage burden through local graph operation, they rely heavily on numerous overlapped subgraphs which severely affect their efficiency and accuracy.

Recent years have witnessed the success of large-scale training in many computer vision tasks [7, 8, 11, 15, 25, 34]. To fully explore the power of large-scale training data, we design a structure-preserved subgraph sampling (SPSS) strategy for GCN training. The edges in an affinity graph are mainly composed of two parts: dense intra-cluster connections and sparse connections between near clusters. Try to approximate the dense connections within clusters, our approach treats face clusters as the smallest sampling unit, different from previous methods [3, 14] which perform randomly sampling on nodes. To further model those inter-cluster connections, we extend the subgraph from the selected cluster to its neighbor clusters. On one hand, the sampled subgraphs preserve the important structural information of the full graph, *i.e.* the edge connections within the clusters and the connections between near clusters. On the other hand, many near clusters are sampled in a subgraph, and the edges between these near clusters can be treated as hard negative examples which can contribute a lot to the GCN training. Equipped with such a structure-preserved subgraph sampling strategy, our method can benefit from increasing training data. Interestingly, this sampling strategy does not lead to performance loss since the structural information of the whole graph is fully considered. Besides, the experimetal results in Table 2 show that it brings further performance gain since the enhancement of generalization.

Algorithm 1 Structure-Preserved Subgraph Sampling

Input: Training nodes reorganized in clusters C, cluster seeds number M, parameters N.
Output: Sampled subgraph $\mathcal{S}$

$\mathcal{S} = \emptyset, \mathcal{S}_1 = \emptyset, \mathcal{S}_2 = \emptyset$
Randomly select M clusters $\mathcal{C}_i$ $(i = 1, \cdots, M)$ from C
for $i = 1$ to M **do**
    Sample N nearest neighbor clusters $\mathcal{C}_{ij}$ $(j = 1, \cdots, N)$ of $\mathcal{C}_i$
    $\mathcal{S}_1 = \mathcal{S}_1 \bigcup \mathcal{C}_i \bigcup_{j=1}^{N} \mathcal{C}_{ij}$
end for
Construct $\mathcal{S}_2$ via applying *CR* on $\mathcal{S}_1$
Construct $\mathcal{S}$ via applying *SR* on $\mathcal{S}_2$
return $\mathcal{S}$

Algorithm 1 shows the details of the proposed SPSS. Given the training nodes reorganized in clusters, we randomly select M clusters from them as the sampling seeds. For each seed cluster, we extend to its N nearest neighbor clusters which are measured by the cosine similarity of the center. After this step, we can get a subgraph $\mathcal{S}_1$ consists of $M \times N$ clusters. To further strengthen the generalization, we introduce the *cluster randomness* (CR) strategy by randomly selecting K_1 clusters from $\mathcal{S}_1$, and the *sample randomness* (SR) strategy by randomly selecting K_2 nodes from $\mathcal{S}_2$. Then we rebuild the KNN affinity graph based on these sampled nodes to construct the subgraph $\mathcal{S}$.

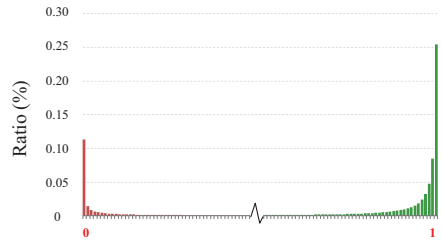


Figure 3: Distribution of the predicted edge scores obtained by the trained GCN. Since the GCN is trained under the supervision of binary signals, its predicted scores appear as two sharp peaks approaching 0 and 1 with high separability.

3.3. Efficient Face Clustering Inference

This subsection shows how to perform efficient inference and get the final face clusters with the proposed STAR-FC in detail. Specifically, we transform the face clustering task into two steps: graph parsing and graph refinement.

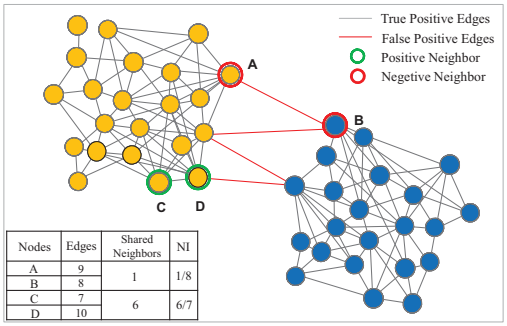
Graph Parsing. In this step, we parse the built affinity graph preliminarily with the trained GCN. We feed the entire graph into the GCN and obtain all edge scores simultaneously. Figure 3 shows that the predicted scores are distributed in nearly two sharp peaks approaching 0 and 1. We can therefore perform simple but effective pruning with a single threshold τ_1. A small number of correct edges may be cut off wrongly in this step. Whereas since the initial affinity graph is densely connected, this has a slight effect on the connectivity of the final clusters. After this step, most wrong connections are removed, and the graph structure has become clearer. However, there still exists a minority of false positive edges. To deal with these edges, we propose node intimacy in the second step trying to mine the local graph structure for further graph refinement.

Graph Refinement. The left false positive edges mistakenly connect different clusters, which may seriously affect the final clustering performance. These edges can not be removed directly using edge scores, we therefore try to identify them with node intimacy (NI).

The concept of node intimacy is inspired by the human familiarity. In human societies, two familiar people usually have many mutual friends. Extending this idea to the graph, we establish the concept of node intimacy. Particularly, given two nodes $\mathcal{N}_1$ and $\mathcal{N}_2$. There are n_1 edges connected to $\mathcal{N}_1$ while n_2 edges connected to $\mathcal{N}_2$ in all. Node $\mathcal{N}_1$ and $\mathcal{N}_2$ have k common neighbor nodes. We then calculate the NI between $\mathcal{N}_1$ and $\mathcal{N}_2$ as follows:

$$NI = Aggregation\left(\frac{k}{n_1}, \frac{k}{n_2}\right). \tag{2}$$

Common aggregation operations include *mean, minimum and maximum*. Comparison in Section 4.2 shows that the *maximum* function has the best performance. Figure 4 illustrates the NI on graph and shows the specific calculation.



Nodes	Edges	Shared Neighbors	NI
A	9	1	1/8
B	8		
C	7	6	6/7
D	10		

Figure 4: Illustration of the proposed node intimacy. We observe that nodes within a cluster are densely connected with each other while there exist sparse wrong connections between clusters. Inspired by such observation, we present the node intimacy which aims to estimate whether two nodes should be linked by measuring their common neighbors. The table at the bottom left shows the specific NI calculation of two pairs of nodes (*i.e.* A&B, C&D). The results indicate that the positive neighbor C&D exactly have high NI while the negative neighbor A&B have low NI.

We further implement the above calculation into a matrix operation. Given adjacency matrix $A \in \mathbb{R}^{N\times N}$, the number of mutual neighbors of all node pairs is $\widetilde{A} = A\,A$, with each element $\widetilde{a}_{ij}$ in $\widetilde{A}$ denoting the number of mutual neighbours for $\mathcal{N}_i$ and $\mathcal{N}_j$. Then the NI is formulated as:

$$NI = max((\widetilde{A}^T\, sum_0)^T, \widetilde{A}\, sum_1), \tag{3}$$

$$sum_0 = vec(\sum_j a_{\cdot j}{}^{-1}), sum_1 = vec(\sum_i a_{i\cdot}{}^{-1}),$$

For inference, we use node intimacy to represent the edge score and remove those edges whose score is below τ_2. After this step, we expect that most wrong connections have been removed. We can thus directly read the face clusters from the affinity graph.

Complexity Analysis. During inference, the main computation lies in the GCN and the node intimacy. Both of these calculations are sparse matrix multiplication, thus the complexity is $O(|\mathscr{E}|)$, where $\mathscr{E}$ denotes the edges in the affinity graph. For a KNN affinity graph with N nodes, we have $|\mathscr{E}| \leqslant |KN|$, thus the complexity increases linearly as the number of nodes in the graph increases.

4. Experiments

4.1. Experimental Settings

Datasets. We use MS1M [13] and a large face benchmark named WebFace42M [55] for training and testing in *face clustering*. We follow the noisy list provided in [9] to clean the MS1M, and the refined MS1M contains about 5.82M images from 85K identities. We follow the setting in [49] to partition MS1M [13] into 10 splits with

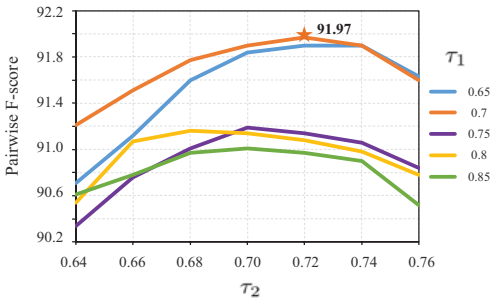


Figure 5: *Pairwise F-score* under different selections of two thresholds τ_1 and τ_2.

Method	Precision	Recall	F_P	F_B
Naive Pruning	92.83	74.24	82.50	80.93
NI (mean)	94.21	84.97	89.35	87.58
NI (min)	95.18	81.93	88.06	86.08
NI (max)	**95.50**	**85.91**	**90.45**	**88.06**

Table 1: Comparison between the naive and the NI-based pruning with different aggregation functions.

almost equal number of identities, while 1 part as labeled data (*part0_train*) for training and the other 9 parts (*part1_test*, $\cdots$, *part9_test*) as unlabeled data for testing. Each part consists of about 0.5M images from 8.6K identities. The WebFace42M is a new million-scale face benchmark including about 42M images and 2M identities which are cleaned from 260M images. It has near 7 times more images than MS1M thus presents a new challenge for face clustering. The MegaFace [19] is used to evaluate the *face recognition* performance of the model trained using pseudo-labeled face images. It consists of a probe set with 3,530 images and a gallery set with over 1M images.

Metrics. We evaluate the performance of our approach on both *face clustering* and *face recognition* tasks. For face clustering, we adopt the commonly used metric *Pairwise F-score* (F_P) and *BCubed F-score* (F_B) [1]. For face recognition, we use different proportions of pseudo-label data along with 1 part labeled data to train the face recognition model and then test the rank-1 face identification accuracy on MegaFace challenge 1 with 1M distractors.

Implementation Details. The affinity graph is built by KNN algorithm [6] with $K = 80$ for MS1M [13] and $K = 30$ for WebFace42M [55]. For structure-preserved subgraph sampling, we set $M = 2$ (the number of cluster seeds), $N = 750$ (the number of the sampled near clusters for each seed), $K_1 = 1300$ (parameter in CR) for MS1M and $M = 4, N = 1100, K_1 = 4000$ for WebFace42M, then set $K_2 = 90\%$ (parameter in SR) for both datasets.

4.2. Ablation Study

All models in this subsection are trained with the *part0_train* and tested on the *part1_test* in MS1M.

Method	SPSS	CR	SR	Nodes per batch	F_P	F_B
a				$\sim 500K$	90.45	88.06
b	✓			$\sim 10K$	91.21	89.19
c	✓	✓		$\sim 10K$	91.80	90.05
d	✓		✓	$\sim 10K$	91.92	90.06
e	✓	✓	✓	$\sim 10K$	**91.97**	90.21

Table 2: Comparison of different sampling strategies.

Selection of Thresholds. In our method, we refine the affinity graph with two steps involving two thresholds τ_1 and τ_2. In the first step, τ_1 is used to process the GCN output edge scores, while in the second step τ_2 is employed to prune the edges with low intimacy. We conduct experiments with different τ_1, τ_2. Results in Figure 5 show that $\tau_1 = 0.7, \tau_2 = 0.72$ is a suitable choice, we therefore adopt this setting in the following experiments.

Design of Node Intimacy. In our method, we transform the face clustering into two steps: graph parsing with a trained GCN and graph refinement with node intimacy (NI). In this subsection, we investigate the impact of NI for the final face clustering performance, and compare three designs of NI. For the naive pruning method in Table 1, face clusters are obtained with dynamic edge pruning [50] based on the edge scores predicted in the graph parsing step. Results in Table 1 show that compared with the naive pruning strategy, using node intimacy for further graph refinement can significantly boost the *pairwise F-score* from 82.5 to 90.45. This reveals the superiority of NI in dealing with face clustering problem. We further compare three different aggregation functions, *i.e. mean, minimum and maximum* for NI. Results in Table 1 show that the *maximum* strategy outperforms the other two methods. Therefore, we choose the *maximum* strategy in the following experiments.

Effect of Sampling Strategy. In the training process, we propose the structure-preserved subgraph sampling (SPSS) strategy. To add more randomness, We further introduce *cluster randomness* (CR) by randomly sampling partial clusters from the subgraph and *sample randomness* (SR) by randomly sampling some nodes from the subgraph. In this subsection, we study the effect of SPSS for GCN training. As shown in Table 2, for the non-sampling method (a), it needs to take the whole graph with about 500K nodes per batch for training which leads to high GPU memory consumption. Nevertheless, equipped with the struct-preserved subgraph sampling strategy, our method only uses a subgraph with about 10K nodes per batch for training, and we achieve 91.21 *pairwise F-score* which is comparable with the non-sampling method with less GPU memory usage. This interseting performance gain demonstrates that our sampling strategy successfully preserves most structure message of the whole graph. Moreover, adding the cluster and sample randomness to the SPSS can further improve the *pairwise F-score* from 91.21 to 91.97. We argue that

#unlabeled	1.74M		2.89M		4.05M		5.21M	
Method / Metrics	F_P	F_B	F_P	F_B	F_P	F_B	F_P	F_B
K-Means [30]	73.04	75.20	69.83	72.34	67.90	70.57	66.47	69.42
HAC [37]	54.40	69.53	11.08	68.62	1.40	67.69	0.37	66.96
DBSCAN [10]	63.41	66.53	52.50	66.26	45.24	44.87	44.94	44.74
ARO [32]	8.78	12.42	7.30	10.96	6.86	10.50	6.35	10.01
CDP [30]	70.75	75.82	69.51	74.58	68.62	73.62	68.06	72.92
L-GCN [42]	75.83	81.61	74.29	80.11	73.70	79.33	72.99	78.60
GCN-D [49]	83.76	83.99	81.62	82.00	80.33	80.72	79.21	79.71
GCN-V+E [47]	84.04	82.84	82.10	81.24	80.45	80.09	79.30	79.25
STAR-FC	**88.28**	**86.26**	**86.17**	**84.13**	**84.70**	**82.63**	**83.46**	**81.47**

Table 3: Comparison on face clustering when training with 0.5M face images and testing with different numbers of unlabeled face images. All results are obtained on the MS1M dataset. The proposed STAR-FC consistently outperforms other face clustering baselines on different scale of testing data.

Method	Precision	Recall	F_P	Time
K-Means [30]	52.52	70.45	60.18	11.5h
DBSCAN [10]	72.88	42.46	53.50	**110s**
HAC [37]	66.84	70.01	68.39	12.7h
ARO [32]	81.10	7.30	13.34	1650s
CDP [50]	80.19	70.47	75.01	140s
L-GCN [42]	74.38	83.51	78.68	5208s
GCN-D+S [49]	**98.24**	75.93	85.66	3700s
GCN-V+E [47]	92.56	83.74	87.93	690s
DA-Net [12]	95.88	85.87	90.60	329s
STAR-FC	96.20	**88.10**	**91.97**	310s

Table 4: Methods comparison on face clustering performance and inference time. All models are trained with part0_train (0.5M images) from MS1M and tested with part1_test (0.5M images) from MS1M. The STAR-FC significantly outperforms the state-of-the-arts, and can control the inference time within 310s.

the introduction of randomness enhances the generalization of the trained model. These experimental results prove the ability of our method to handle large-scale training effectively. With the proposed SPSS, we can break through the limitation of the size of the training set and achieve excellent face clustering performance.

4.3. Face Clustering on MS1M

Table 3 and Table 4 show the comparison on face clustering. All results in Table 4 are obtained on the MS1M dataset with *part0_ train* as the training set and *part1_ test* as the testing set, and the inference time is obtained following the experimental configuration in [47]. In Table 3 we further show the face clustering performance on different numbers of unlabeled data. Results in Table 4 show that the proposed STAR-FC outperforms other clustering baselines consistently. Moreover, since all modules in the STAR-FC use full graph operation and parallel matrix computing, it can perform efficient inference within 310s. For fair comparison with DA-Net [12], this time does not include the time of computing KNN graph, and the total inference time including computing KNN is 435s which can be accelerated with parallel GPUs. Results in Table 3 show that our method can keep superior performance when dealing with larger-scale inference. What's more, compared with the representative clustering method GCN-V+E [47], our method boosts the F_P significantly from 79.3 to 83.46 and improves the F_B from 79.25 to 81.47.

We further use these pseudo-labeled data to train face recognition models and investigate the performance gain brought by these extra pseudo-labeled training data. We follow the experiment setting in [47, 49], and use labeled data and various amounts of unlabeled data with pseudo-label to train the face recognition models. Figure 6 shows the rank-1 face identification accuracy on MegaFace [19] with 1M distractors. As shown in Figure 6, extra unlabeled training data with pseudo-label brings continuous performance gain for face recognition. Owing to the superior performance in face clustering, our method achieves higher recognition accuracies than other face clustering baselines. With extra 5.21M unlabeled data, our method improves the recognition performance on MegaFace from 58.2% to 79.26%.

4.4. Face Clustering on WebFace42M

In this subsection, we first compare the face clustering performance of different methods on the WebFace42M and then explore the training upper bound of the STAR-FC.

Recent years have witnessed the success of large-scale training in many computer vision tasks. Large-scale training data is one of the key engines for the performance gain. To verify the capacity of the proposed method to handle large-scale graph, we conducte more experiments on the million-scale face benchmark WebFace42M [55]. We randomly select 4M samples from the dataset as labeled data for training and take 4M, 8M and 12M samples respectively as unlabeled data for testing. There is no identities overlap between the training set and the testing set. We reproduce a

#unlabeled	4M				8M				12M				
Method / Metrics	Pre	Recall	F_P	F_B	Pre	Recall	F_P	F_B	Pre	Recall	F_P	F_B	Time
K-Means [30]	95.99	50.05	65.80	78.29	92.20	49.91	64.34	76.47	88.75	49.69	63.71	75.04	2h
HAC [37]	98.25	59.76	74.31	85.46	96.55	58.98	73.23	84.57	OOM	OOM	OOM	OOM	OOM
DBSCAN [10]	94.77	44.12	60.21	77.87	89.97	43.55	58.69	77.02	85.57	43.76	57.91	76.38	3h
ARO [32]	**99.34**	62.83	76.98	88.83	**98.44**	62.01	76.09	88.66	**97.49**	62.34	76.05	88.60	4h
GCN-D [49]	98.05	52.54	68.42	71.47	96.47	51.82	67.42	71.24	95.08	53.70	68.63	72.39	8h
STAR-FC	96.77	**94.00**	**95.36**	**94.93**	93.95	**93.99**	**93.97**	**94.77**	90.86	**94.06**	**92.43**	**94.63**	**1.7h**

Table 5: Comparison on face clustering when training with 4M face images and testing with different numbers of unlabeled data from the WebFace42M. Inference time on 12M testing data is shown in the right-most column. GCN-V+E [47] fails to perform training on 4M data due to out-of-memory, so we don't show it in this table. HAC [37] is able to train on large-scale data, however it fails to perform large-scale inference with 12M testing data. The proposed STAR-FC achieves superior results on different testing settings and can complete inference on 12M data within 1.7h.

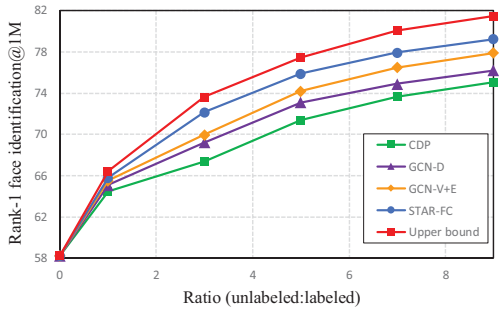


Figure 6: Rank-1 face identification accuracy on MegaFace with 1M distractors. The X-axis indicates the ratio of unlabeled to labeled data. The point where ratio is 0 indicates that only a split of labeled data is used for training. The upper bound is trained using data with ground-truth labels.

series of clustering baselines on the WebFace42M dataset. Table 5 shows their clustering performance and inference time on 12M testing data with acceleration by faiss [18]. Given the large-scale graph with 4M nodes for training, GCN-V will be directly out of memory. HAC fails to handle large-scale inference when the testing data size increases up to 12M. Under such settings of large-scale training and large-scale testing, our method once again achieves superior face clustering performance, and can complete the inference efficiently on 12M testing data within 1.7h.

Furthermore, we make an exploration about the training upper bound of the proposed STAR-FC. We gradually increase the size of the training set and observe the performance changes. Specifically, we randomly select different scales of data (0.4M, 4M, 12M, 20M) from WebFace42M as labeled data for training, and test their face clustering performance on a testing set with 12M data. Experimental results in Table 6 and Figure 1 show that global-based methods such as GCN-V fail to handle large-scale training while the local-based method GCN-D has poor performance. Nevertheless, with the increase of training data, our method has been consistently improved and finally achieves 95.1 *pairwise F-score*. These experiments prove the performance superiority of the STAR-FC and its ability to handle large-scale training. What's more, we are the first to conduct face clustering training on a very large-scale graph with 10^7 nodes, thus provide a strong baseline for large-scale face clustering. Our method is promising to perform excellently when a larger training set appears.

Training set	Pre	Recall	F_P	Pre	Recall	F_B	NMI
0.4M	96.7	84.6	90.2	99.6	75.9	86.1	97.8
4M	90.9	94.1	92.4	99.0	90.7	94.6	99.1
12M	95.4	92.9	94.2	99.4	88.3	93.5	99.0
20M	97.8	92.5	95.1	99.4	88.1	93.4	99.0

Table 6: Face clustering performance of the STAR-FC with different scales of training data from the WebFace42M and testing on 12M unlabeled data from the WebFace42M.

5. Conclusion

In this paper, we have proposed a structure-aware face clustering method STAR-FC which addresses the dilemma of large-scale training and efficient inference. A structure-preserved subgraph sampling method is introduced to explore the power of larger-scale training data, and it can achieve satisfactory performance with less GPU memory usage. Moreover, a two-step graph refinement strategy with full-graph operation is developed to perform efficient inference. For the first time, a face clustering model is trained on a very large-scale graph with 10^7 nodes. Extensive experiments on MS1M and WebFace42M demonstrate the superior face clustering performance of the proposed STAR-FC.

Acknowledgement This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFA0700802, in part by the National Natural Science Foundation of China under Grant U1813218, Grant 61822603, Grant U1713214, in part by Beijing Academy of Artificial Intelligence (BAAI), and in part by a grant from the Institute for Guo Qiang, Tsinghua University.

References

[1] Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *IR*, 2009. 6

[2] Rianne van den Berg, Thomas N Kipf, and Max Welling. Graph convolutional matrix completion. *arXiv:1706.02263*, 2017. 2

[3] Jie Chen, Tengfei Ma, and Cao Xiao. FastGCN: fast learning with graph convolutional networks via importance sampling. In *ICLR*, 2018. 3, 4

[4] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-GCN: An efficient algorithm for training deep and large graph convolutional networks. In *KDDM*, 2019. 2

[5] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *CVPR*, 2017. 2

[6] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *TIT*, 1967. 2, 3, 6

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 2, 4

[8] Jiankang Deng, Jia Guo, Tongliang Liu, Mingming Gong, and Stefanos Zafeiriou. Sub-center ArcFace: Boosting face recognition by large-scale noisy web faces. In *ECCV*, 2020. 2, 4

[9] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. ArcFace: Additive angular margin loss for deep face recognition. In *CVPR*, 2019. 1, 5

[10] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *SIGKDD*, 1996. 1, 2, 7, 8

[11] Ross Girshick. Fast R-CNN. In *ICCV*, 2015. 2, 4

[12] Senhui Guo, Jing Xu, Dapeng Chen, Chao Zhang, Xiaogang Wang, and Rui Zhao. Density-aware feature embedding for face clustering. In *CVPR*, 2020. 1, 2, 7

[13] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. MS-Celeb-1M: A dataset and benchmark for large-scale face recognition. In *ECCV*, 2016. 2, 4, 5, 6

[14] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017. 2, 4

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2, 4

[16] Jeffrey Ho, Ming-Husang Yang, Jongwoo Lim, Kuang-Chih Lee, and David Kriegman. Clustering appearances of objects under varying illumination conditions. In *CVPR*, 2003. 1, 2

[17] Anil K Jain, Jianchang Mao, and K Moidin Mohiuddin. Artificial neural networks: A tutorial. *Computer*, 1996. 4

[18] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *TBD*, 2019. 8

[19] Ira Kemelmacher-Shlizerman, Steven M Seitz, Daniel Miller, and Evan Brossard. The MegaFace Benchmark: 1 million faces for recognition at scale. In *CVPR*, 2016. 6, 7

[20] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017. 2

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. In *NeruIPS*, 2012. 2

[22] Peizhao Li, Han Zhao, and Hongfu Liu. Deep fair clustering for visual learning. In *CVPR*, 2020. 1

[23] Wanhua Li, Yueqi Duan, Jiwen Lu, Jianjiang Feng, and Jie Zhou. Graph-based social relation reasoning. In *ECCV*, 2020. 2

[24] Wanhua Li, Yingqiang Zhang, Kangchen Lv, Jiwen Lu, Jianjiang Feng, and Jie Zhou. Graph-based kinship reasoning network. In *ICME*, 2020. 2

[25] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 2, 4

[26] Wei-An Lin, Jun-Cheng Chen, Carlos D Castillo, and Rama Chellappa. Deep density clustering of unconstrained faces. In *CVPR*, 2018. 2

[27] Wei-An Lin, Jun-Cheng Chen, and Rama Chellappa. A proximity-aware hierarchical clustering of faces. In *FG*, 2017. 2

[28] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, 2017. 1

[29] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *ICML*, 2016. 1

[30] Stuart Lloyd. Least squares quantization in pcm. *TIP*, 1982. 1, 2, 7, 8

[31] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010. 4

[32] Charles Otto, Dayong Wang, and Anil K Jain. Clustering millions of faces by identity. *TPAMI*, 2017. 2, 7, 8

[33] Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. *BMVC*, 2015. 2

[34] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 2, 4

[35] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *ESWC*, 2018. 2

[36] Yichun Shi, Charles Otto, and Anil K Jain. Face clustering: representation and pairwise constraints. *TIFS*, 2018. 2

[37] Robin Sibson. Slink: an optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 1973. 7, 8

[38] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *NeurIPS*, 2014. 1

[39] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*, 2014. 1

[40] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018. 2

[41] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, 2018. 1

[42] Zhongdao Wang, Liang Zheng, Yali Li, and Shengjin Wang.

Linkage based face clustering via graph convolution network. In *CVPR*, 2019. 1, 2, 4, 7

[43] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *TNNLS*, 2020. 2

[44] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. *arXiv:1505.00853*, 2015. 4

[45] Sijie Yan, Zhizhong Li, Yuanjun Xiong, Huahan Yan, and Dahua Lin. Convolutional sequence generation for skeleton-based action synthesis. In *ICCV*, 2019. 2

[46] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI*, 2018. 2

[47] Lei Yang, Dapeng Chen, Xiaohang Zhan, Rui Zhao, Chen Change Loy, and Dahua Lin. Learning to cluster faces via confidence and connectivity estimation. In *CVPR*, 2020. 1, 2, 4, 7, 8

[48] Lei Yang, Qingqiu Huang, Huaiyi Huang, Linning Xu, and Dahua Lin. Learn to propagate reliably on noisy affinity graphs. In *ECCV*, 2020. 1, 2

[49] Lei Yang, Xiaohang Zhan, Dapeng Chen, Junjie Yan, Chen Change Loy, and Dahua Lin. Learning to cluster faces on an affinity graph. In *CVPR*, 2019. 1, 2, 4, 5, 7, 8

[50] Xiaohang Zhan, Ziwei Liu, Junjie Yan, Dahua Lin, and Chen Change Loy. Consensus-driven propagation in massive unlabeled data for face recognition. In *ECCV*, 2018. 1, 2, 6, 7

[51] Xiaohang Zhan, Jiahao Xie, Ziwei Liu, Yew-Soon Ong, and Chen Change Loy. Online deep clustering for unsupervised representation learning. In *CVPR*, 2020. 1

[52] Yaobin Zhang, Weihong Deng, Mei Wang, Jiani Hu, Xian Li, Dongyue Zhao, and Dongchao Wen. Global-Local GCN: Large-scale label noise cleansing for face recognition. In *CVPR*, 2020. 1, 2

[53] Ming Zhao, Yong Wei Teo, Siliang Liu, Tat-Seng Chua, and Ramesh Jain. Automatic person annotation of family photo album. In *ICIVR*, 2006. 1, 2

[54] Chunhui Zhu, Fang Wen, and Jian Sun. A rank-order distance based clustering algorithm for face tagging. In *CVPR*, 2011. 2

[55] Zheng Zhu, Guan Huang, Jiankang Deng, Yun Ye, Junjie Huang, Xinze Chen, Jiagang Zhu, Tian Yang, Jiwen Lu, Dalong Du, and Zhou Jie. Webface260M: A benchmark unveiling the power of million-scale deep face recognition. In *CVPR*, 2021. 1, 2, 5, 6, 7