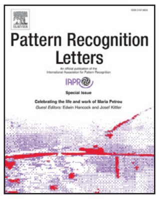
Pattern Recognition Letters 133 (2020) 48–54

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec



Learning deep face representation with long-tail data: An aggregate-and-disperse approach

Yuhao Ma[a,b], Meina Kan[a,b,*], Shiguang Shan[a,b,c], Xilin Chen[a,b]

[a] Key Laboratory of Intelligent Information Processing of Chinese Academy of Sciences, Institute of Computing Technology, CAS, Beijing 100190, China
[b] University of Chinese Academy of Sciences, Beijing 100049, China
[c] CAS Center for Excellence in Brain Science and Intelligence Technology, Shanghai 200031, China

ARTICLE INFO

Article history:
Received 5 July 2019
Revised 29 November 2019
Accepted 6 February 2020
Available online 7 February 2020

Keywords:
Face recognition
Deep representation learning
Long-tail distribution
Aggregate-and-disperse

ABSTRACT

In this work, we study the problem of deep representation learning on a large face dataset with long-tail distribution. Training convolutional neural networks on such dataset with conventional strategy suffers from imbalance problem which results in biased classification boundary, and the few-shot classes lying in tail parts further make the model prone to overfitting. Aiming to learn more discriminative CNN model from long-tail data, we propose a novel aggregate-and-disperse training schema. Firstly, our proposed method aggregates similar classes in tail part to avoid imbalance problem. Based on the aggregated super classes and those original head classes, a model is pre-trained to capture accurate discrimination in head classes as well as coarse discrimination in tail classes. Secondly, we selectively disperses those aggregated super classes to learn precise inter-class variations and refine the representation for better generalization. We perform extensive experiments on MS-Celeb-1M, BLUFR and MegaFace. Compared with baselines and existing methods, our method achieves better performance of face recognition, demonstrating its effectiveness of handling long-tail distribution.

1. Introduction

In recent years, the Convolutional Neural Networks (CNNs) have achieved great success in various computer vision tasks such as image classification [19], segmentation [23] and object detection [8], etc. In the specific field of face recognition, CNNs have also made a breakthrough with a few works exhibiting excellent performance [22,24,29,31,34]. These remarkable advancements benefit a lot from the large-scale datasets and CNN's ability to learn from massive training data. However, CNN does not perform well when encountered unusual data distribution, e.g., long-tail distribution. Particularly, in this work we study the problem of training CNN on a face dataset exhibiting long-tail characteristics, in which a considerable portion of classes have only very few face images (referred to as tail classes) while other classes have relatively abundant samples (head classes). Such long-tail distribution often arises when collecting large-scale face datasets with a large number of classes since the number of samples in each class easily varies. It is meaningful to investigate the impact of tail classes and develop an approach to handle these data for enhancement.

Fig. 1 illustrates long-tail distribution on a cleansed version of a large-scale face dataset MS-Celeb-1M [11]. In this dataset, only a small portion of identities, about 10%, have a large number of face images, while a considerable portion have only very few samples. Empirically, deep CNN models can be improved by increasing the training data, but it might not be the case when introducing tail classes. For head classes, they contain many samples with rich intra-class variations, and thus dominate the learning process without obvious imbalance problem. The tail classes, although include substantial samples in total, tend to induce imbalance problem and push the decision boundary towards them, resulting in a biased classifier. Moreover, the scarcity of few-shot samples could not guarantee an accurate estimation of class boundary, making the trained model prone to overfitting. These factors might cripple the model when tail classes are involved. A straighforward solution is to directly ignore these tail classes to avoid these negative side effects, but much valuable information is also removed. So are there any better alternatives? Although few works explore this specific problem, many existing works are highly relevant.

For face recognition tasks, [33] indicates that some metric-based loss such as [29,34] are superior to conventional softmax loss when dealing with few-shot samples. Furthermore, an improved version called range loss [37] is especially proposed to deal with the long-tail problem. There are also some other methods

* Corresponding author.
E-mail addresses: yuhao.ma@vipl.ict.ac.cn (Y. Ma), kanmeina@ict.ac.cn (M. Kan), sgshan@ict.ac.cn (S. Shan), xlchen@ict.ac.cn (X. Chen).

https://doi.org/10.1016/j.patrec.2020.02.007
0167-8655/© 2020 Published by Elsevier B.V.

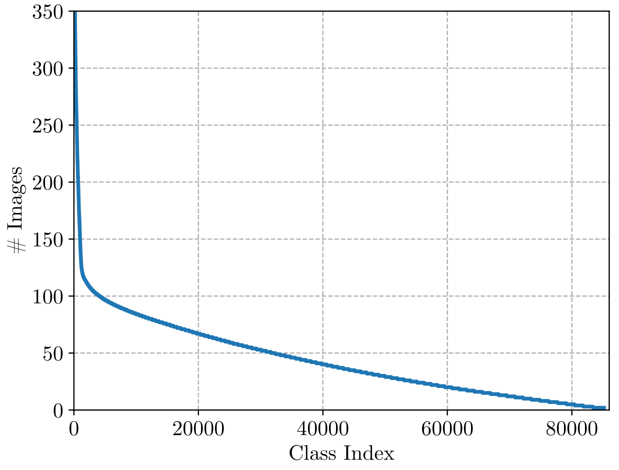


Fig. 1. Long-tail distribution of a large-scale face dataset MS-Celeb-1M [11]. Class index is sort according to the number of samples per class.

relevant to imbalance or few-shot problem, which are inherent problems of long-tail distribution. Typical strategies against imbalance problem include resampling [4,13] and cost-sensitive learning [7,32]. These principles have been applied to some recent works under deep learning frameworks [18,21]. However, these solutions have some inherent limitations such as abandoning samples due to under-sampling or introducing undesirable noises (e.g., assigning larger penalty on outlier samples). Few-shot learning aims at recognizing classes with only one or a few samples for training. Some works try to synthesize data based on generation rules as augmentation [12,35]. Some other works regularize the weight norm of classifier in adaptation of few-shot classification [10]. Most of these works rely on a base set with adequate samples and focus more on few-shot classification rather than learning a general representation under data with long-tail distribution. In [25], we proposed a method to learn deep face representation on a purely few-shot face dataset, and in this work we further extend [25] to handle long-tail face dataset.

Since the scarcity of samples makes it difficult to directly acquire accurate discrimination, we propose to utilize the inter-class similarity between those tail classes. For a group of similar classes whose samples are close in feature space, they share similar distinction from other classes, and thus could make up a super class. Based on this key insight, we design a two-step training schema, which includes an aggregation step followed by a dispersion step, in a coarse-to-fine manner. Firstly, based on visual similarity, we aggregate tail classes to super classes, which are similar to head classes in terms of data capacity and intra-class variations. With these super classes and those fine head classes, a coarsely accurate model (referred to as prototype model) is obtained by learning accurate discrimination in head classes and coarse discrimination in tail classes. Organizing tail classes with aggregation helps us to obtain a prototype model with better generalization, which also regularizes the optimization of the succeeding training to achieve better performance as well. In the second step, the aggregated classes in active super classes (whose samples appear in the current mini-batch) are dispersed to learn accurate discrimination between these tail classes. Meanwhile, those classes from inactive super classes still remain aggregated to alleviate the data imbalance problem.

The contributions of this work are summarized as follows:

- An aggregate-and-disperse training schema is proposed to simultaneously handle the data imbalance problem and few-shot problem in long-tail distribution.
- Extensive experiments are performed on two benchmarks (BLUFR [20] and MegaFace [27]) by using a long-tailed dataset sampled from MS-Celeb-1M for training, demonstrating that the proposed method could effectively ameliorate the negative influences of long-tail distribution and make better use of the information in tail classes.

2. Related work

The problem of learning with long-tail data involves both data imbalance and few-shot problems. So methods that handle these two issues are also related to this work. All related works we review in this section generally fall into four categories, including data re-sampling, data generation, regularization-based and metric-based methods.

Data re-sampling methods. Re-sampling techniques [4,13,26] aim to balance the number of samples by changing the sampling frequency to handle imbalance problem. Oversampling methods like SMOTE [4] improves replication-based method by interpolating new samples around scarce samples. However, this process still suffers from overfitting and introduces undesirable noises [7]. Another alternative of undersampling frequent classes might discard a large portion of data and remove valuable information. Recently, some of these methods designed for classic data imbalance problem have been extended in the context of deep learning [1,2]. Except for some specific problem (e.g. binary classification in detection), the improvement is quite limited due to the above mentioned drawbacks.

Data generation methods. Data generation is to augment a large number of samples with diversity, which makes it a straightforward solution to few-shot problem. As for generation in feature space, SMOTE [4] mentioned above could be regarded as a naive approach for data generation. Hariharan and Girshick [12] proposes to hallucinate the features of few-shot classes and get additional samples by learning pair-wise variations from regular classes. Dixit et al. [6] creates an attribute-guided feature descriptor for data augmentation. Yin et al. [35] assumes a Gaussian prior across all regular classes and transfers their variations to tail classes to simultaneously learn a less biased classifier and representation. As for generation in image space, Generative Adversarial Networks (GANs) [9] have achieved great progress in the last few years. GAN-based methods are capable of generating HD face images [17] or editing attributes (e.g. pose, age, etc.) of face images [30]. However, it still remains challenging to augment samples with high diversity while maintaining the identity.

Regularization-based methods. These methods modify the conventional settings (e.g. weights of samples, evaluation of loss, training schema, etc.) to regularize the learning process by design. Classic cost-sensitive learning approaches assign adaptive weights for different samples [7,32,36] to address the imbalance issue. Recent work [21] also re-weights samples for online hard example mining (OHEM) in object detection. Another method [10] adapts the classifier to recognize few-shot samples, which is achieved by aligning the norms of classification weight vectors of these few-shot classes to those of regular classes. Differently, Ma et al. [25] aims to learn face representation from a few-shot dataset. A two-step approach is proposed, which is to firstly cluster similar few-shot classes into super classes and then disperse them, forming a coarse-to-fine manner as regularization against overfitting.

Metric-based methods. For classes which only contain a scarce amount of samples, training with conventional classification easily lead to overfitting. In this scenario, metric-based methods could bring some extra supervision by adding desirable constraints on the distance between samples in the feature space. Huang et al. [15] proposes to sample quintuplets and apply hinge loss to enlarge both inter-cluster and inter-class margins for solving imbalance problem in face attribute classification. Triplet loss [29] and center loss [34] have proved their effectiveness for learning face representation, and the research of [33] indicates that such metric-based methods are superior when dealing with few-shot samples. For the long-tail problem, an improved version called range loss [37] is proposed. The objective is to minimize the harmonic mean of k largest distance in one class and maximize the shortest inter-class distance within the mini-batch. A prominent disadvantage of this method is that a suitable sampling strategy is indispensable to construct effective data batches for training.

Most of the above mentioned related works are proposed to handle imbalance or few-shot problem, which are the sub-problems of learning from long-tail data. These methods can partially solve or be inspirational to address the long-tail problem although not specifically designed for it. As an extension of [25], our method is a kind of regularization-based method. It proposes a novel idea of selectively dispersing the aggregated super classes to address both the few-shot and imbalance issue in long-tail problem.

3. Method

In this section, we firstly formulate the problem of learning deep face representation on a dataset with long-tail distribution. Then, we discuss the problems caused by the long-tail distribution and present the details of our two-step training schema.

3.1. Problem formulation

The objective of face recognition is to obtain a face descriptor which could extract discriminative features from face images. To achieve this in the context of deep learning, we adopt a deep CNN model denoted as $\mathcal{F}(\bullet, \theta)$ and apply classification task to supervise the learning process. This CNN model consists of a series of operations such as convolution, pooling and ReLU, which are parameterized as θ. This CNN model maps the input face image $\mathbf{X}$ to feature vector $\mathbf{f}$:

$$\mathbf{f} = \mathcal{F}(\mathbf{X}, \theta). \quad (1)$$

Then with the feature $\mathbf{f}$ and the weights $\mathbf{W}$ in the fully-connected layer (FC), the softmax classifier $\sigma(\mathbf{f}, \mathbf{W})$ calculates the class-wise probabilities $\mathbf{p}$:

$$\mathbf{p} = \sigma(\mathbf{f}, \mathbf{W}). \quad (2)$$

Cross entropy loss L is computed for back propagation:

$$L = -\frac{1}{n}\sum_{j=1}^{n}\sum_{i=1}^{N}\mathbb{I}(y^j = i) log \mathbf{p}_i^j. \quad (3)$$

n is the number of the samples in the input batch, and N denotes the number of classes involved in classification. $\mathbb{I}(\bullet)$ is the indicator function, y^j is the label of the j^{th} sample, and $\mathbf{p}_i^j$ denotes its predicted probability on class i.

The above-mentioned training schema is widely used in conventional settings. All samples are treated equally in the learning process no matter which class they belong to. However, for a face dataset exhibiting long-tail distribution (e.g. dataset illustrated in Fig. 1), such schema might be inappropriate because the importance of distinct classes are quite different, which is mainly caused by the difference in number of samples. So we divide the long-tail dataset into two parts based on the number of samples in a class. Thus, we get N_t tail classes which have fewer samples and the other N_h head classes. We find out that using all classes leads to worse performance than only using N_h head classes, as shown in Table 2. The tail classes seem to impose negative impact on the learning process. In this work, we argue that with proper modification to the training schema, the tail classes could help boost the performance of deep face representation learning.

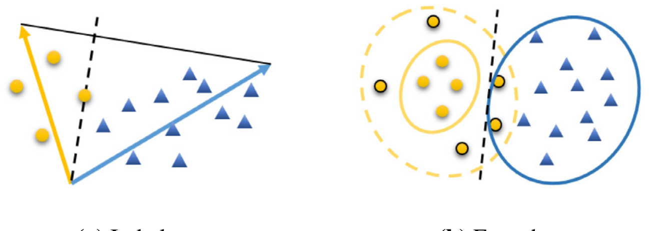


Fig. 2. Negative influences of tail classes. Yellow dots are tail class samples, and blue triangles are head class samples. Arrows are weight vectors, and the black dashed lines are the decision boundaries. (a) The imbalance problem results in a biased boundary due to different magnitude of weight norm. [10] (b) Due to few-shot property, yellow dots with black border are unseen in training. Such mistaken estimation of neighbourhood leads to an inaccurate boundary. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

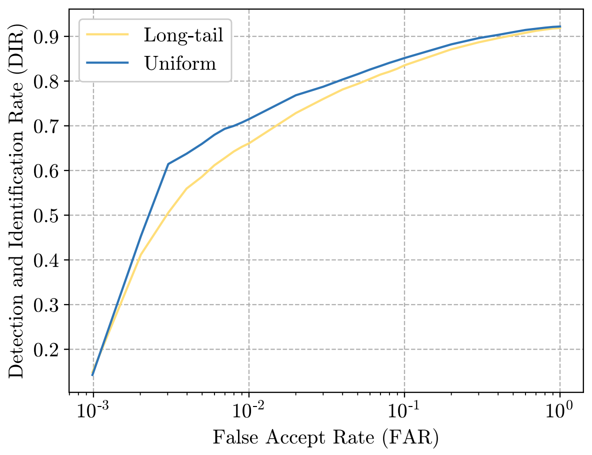


Fig. 3. Open-set identification ROC on BLUFR. The same CNN models are trained on dataset with uniform and long-tail distribution respectively.

3.2. Negative influences of long-tail distribution

To address this issue, it is crucial to understand how the long-tail distribution influences representation learning. Several previous works study this problem and provide their explanations from different perspectives. Zhang et al. [37] performs empirical analysis on feature vectors and calculates kurtosis which indicates the existence of infrequent extreme samples. Experiment shows that introducing tail classes leads to higher kurtosis, implying more outlier samples in the feature space. Guo and Zhang [10] discovers that the norm of classifier weights for one-shot classes are smaller than those of regular classes (as displayed in Fig. 2(a)), which is also the case in a long-tail settings [35]. This results in a biased classification boundary and harms the representation learning. Moreover, the few-shot property makes it difficult to learn a genuine neighbourhood of tail samples, which makes the model prone to overfitting [15]. In Fig. 2, we display the above-mentioned situations in the feature space to help better understanding of the influences of tail classes.

Furthermore, we perform an empirical study to demonstrate such negative influences. For fair comparison, we construct two datasets with 400,000 faces samples from 8000 identities based on a large-scale face dataset [3]. One have a relatively uniform distribution across different identities and another exhibits long-tail distribution. The same CNN models are trained on these two datasets with different distributions. The performance is examined on the benchmark BLUFR [20] (see Section 4.1 for details). The results in terms of ROC on open-set identification are displayed in Fig. 3. Not

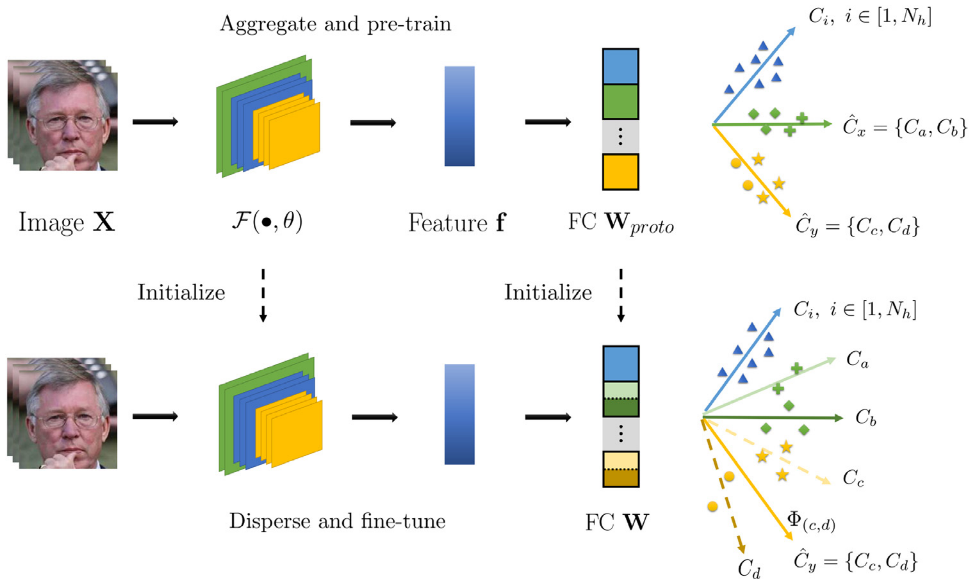


Fig. 4. An overview of the proposed aggregate-and-disperse training schema. The shape of sample indicates its intrinsic label, similar color indicates high similarity, and the arrows are weight vectors. Aggregation is to cluster similar tail classes into super classes to coarsely supervise the training of the prototype model. Dispersion is to fine-tune this prototype model by selectively dispersing the tail classes within super classes that appear in the current batch. Best viewed in color. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

surprisingly, the model trained with long-tail dataset is worse than the one with uniform distribution, illustrating the negative impact of long-tail distribution on deep face representation learning.

3.3. Proposed aggregate-and-disperse training schema

An overview of the proposed two-step training schema is illustrated in Fig. 4. The first step aggregates similar tail classes into super classes for training a robust prototype model. The second step disperses these similar tail classes in active super classes to precisely learn inter-class discrimination by fine-tuning the prototype model. Next we respectively introduce these two steps in details.

3.3.1. Step I: aggregation

As mentioned above, direct learning of inter-class discrimination from these tail classes could be harmful. Fortunately, with a large number of identities, it is not hard to find many visually similar samples which share similar distinction from the rest ones. In this step, we propose to exploit such inter-class similarity to aggregate similar tail classes into super classes which can avoid the imbalance problem and thus results in a robust training process. Then we train a prototype model based on the fine head classes and these aggregated super classes.

Clustering. Firstly we use a base CNN face descriptor to extract features of tail class samples and compute the mean feature $\mathbf{m}$ of each class as its representation:

$$\mathbf{m}_i=\frac{1}{n_i}\sum_{j=1}^{n_i}\mathbf{f}_i^j. \tag{4}$$

$\mathbf{f}_i^j$ denotes the feature of the j^{th} sample from tail class i, which has n_i samples in total. Based on $\mathbf{m}_i$, we simply adopt k-means clustering to aggregate N_t tail classes into N_k super classes $\{\hat{C}_1,\hat{C}_2,...,\hat{C}_{N_k}\}$. Each of them contains a group of similar tail classes (e.g. $\hat{C}_i=\{C_a,C_b,C_c\}$), thus the number of samples in a super class is closer to those of head classes. Suppose C_i is a head classes with a fine label if $i\in[1,N_h]$, then we have a collection of head classes and super classes $C=\{C_1,C_2,...,C_{N_h},\hat{C}_1,\hat{C}_2,...,\hat{C}_{N_k}\}$.

Prototype training. Now that we have a dataset with (N_h+N_k) classes, the prototype model is trained under the joint supervision of the coarse labels of super classes and the fine labels of head classes according to Eqs. (2) and (3), i.e. the loss for a single input sample X^j is defined as follow:

$$L^j=-\sum_{i=1}^{N_c}\mathbb{I}(y^j=i)\log\mathbf{p}_i^j. \tag{5}$$

N_c is the number of classes that equals (N_h+N_k), and correspondingly we have $\mathbf{W}_{proto}=\{\mathbf{w}_1,\mathbf{w}_2,...,\mathbf{w}_{N_h},\widehat{\mathbf{w}}_1,\widehat{\mathbf{w}}_2,...,\widehat{\mathbf{w}}_{N_k}\}$, which contains weight vectors of softmax classifier for head classes and super classes. As illustrated in Fig. 4, samples of similar tail classes (e.g. C_a, C_b) share the same color. As the tail classes in each super class share the same supervision in this step, their features lie around the corresponding super weight vector after optimization (see scatters and arrows in the same color in Fig. 4). For head classes $(C_1,C_2,...,C_{N_h})$, which have abundant samples and rich variations, accurate inter-class discrimination can be learned with ease. For super classes (e.g. $\hat{C}_x,\hat{C}_y$), they are similar to head classes in terms of data capacity and variation because they include a bunch of similar intrinsic tail classes. Classifying these super classes enables the prototype model to learn coarse (i.e. inter-cluster) discrimination from tail classes as complement. At the end of this step, we obtain a prototype model $\mathcal{F}(\bullet,\theta_{proto})$ and the corresponding classifier $\mathbf{W}_{proto}$, which have learned accurate discrimination in head classes and meanwhile captured the coarse variations in tail classes. In general, this aggregation step leads to a prototype model with better generalization and regularizes the succeeding training.

3.3.2. Step II: dispersion

The intra-cluster variations are not considered in previous training but they are relevant to distinguishing similar identities. So aiming to precisely learn detailed variations, the second dispersion step is to continue training based on the prototype model obtained in the first aggregation step. A straightforward solution is to disperse all super classes and fine-tune the prototype model with intrinsic fine labels, which treats tail classes and head classes equally. However, this would somehow bring the model back to the long-tail distribution curse again. In order to model the detailed variations and avoid the long-tail problem as well, we propose to selectively disperse them. Specifically, for a training sample from a super class appearing in the current batch, it should be distinguished from other similar classes in the same super class, so this super class should be dispersed. If a super class is inactive, which means no training sample of this super class appears in current batch, it should be regarded as a whole and thus remains aggregated to avoid imbalance problem. This is achieved with a modified softmax classifier.

Firstly we introduce how the parameters are initialized. For the CNN parameter θ, we start with θ_{proto}, which is from the prototype model in the first step. As for classifier weights, the previous $\mathbf{W}_{proto}$ for N_c classes should be adapted to a new classifier $\mathbf{W}$ for all the $(N_h + N_t)$ classes in this step. This initialization process is displayed in Fig. 4. For N_h head classes, the weight vectors of their own class from $\mathbf{W}_{proto}$ are used as the initialization, and each pair is marked in the same color. For the last N_t tail classes in $\mathbf{W}$, the weight vectors of the corresponding super classes are used as the initialization. Tail classes within the same super class are marked in similar colors.

Next we introduce the modification to the softmax classifier in details. Suppose $\mathbf{W} = \mathbf{W}_h \cup \mathbf{W}_t = \{\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_{N_h}\} \cup \{\mathbf{w}_{N_h+1}, ..., \mathbf{w}_N\}$, $N = N_h + N_t$. $\mathbf{W}_h$ and $\mathbf{W}_t$ denote the sets of weight vectors for head classes and tail classes. For a sample with label k, its corresponding weight vector is $\mathbf{w}_k$ and its feature is denoted as $\mathbf{f}_k$ (here we omit the superscript for simplicity). If C_k is a head class, we have the following form of softmax:

$$\sigma(\mathbf{f}_k, \mathbf{W}) = \frac{exp(\mathbf{w}_k^T \mathbf{f}_k)}{\sum_{\mathbf{w}_i \in \mathbf{W}_h} exp(\mathbf{w}_i^T \mathbf{f}_k) \ + \sum_{\mathbf{w}_i \in \mathbf{W}_t} \frac{1}{n_{g_i}} exp(\boldsymbol{\Phi}_i^T \mathbf{f}_k)}. \quad (6)$$

g_i denotes the index of the super class which includes the tail class i (e.g. $C_i \in \hat{C}_{g_i}$), n_{g_i} is the number of tail classes it includes. $\boldsymbol{\Phi}_i$ is defined as the mean weight vectors of the tail classes that belong to this super class:

$$\Phi_i = \frac{1}{n_{g_i}} \sum_{k \ |C_k \in \hat{C}_{g_i}} \mathbf{w}_k. \quad (7)$$

In this setting, weight vectors of those similar tail classes within a super class are updated as a whole, maintaining uniform distribution for better optimization. See the solid arrow in yellow for $\hat{C}_y$ in Fig. 4, it is the mean of the dashed arrows for tail classes C_c, C_d.

As for another situation that C_k is a tail class, we have another form of softmax:

$$\sigma(\mathbf{f}_k, \mathbf{W}) = \frac{exp(\mathbf{w}_k^T \mathbf{f}_k)}{\sum_{\mathbf{w}_i \in \mathbf{W}_h} exp(\mathbf{w}_i^T \mathbf{f}_k) + \sum_{\mathbf{w}_i \in \mathbf{W}_k} exp(\mathbf{w}_i^T \mathbf{f}_k) + \sum_{\mathbf{w}_i \in \mathbf{W}_t \setminus \mathbf{W}_k} \frac{1}{n_{g_i}} exp(\boldsymbol{\Phi}_i^T \mathbf{f}_k)}. \quad (8)$$

$\mathbf{W}_k = \{\mathbf{w}_i \ |C_i \in \hat{C}_{g_k}\}$ denotes the set of weight vectors that belong to the same super class as tail class C_k. The super class which includes the current class C_k is expanded as the second term in the denominator according to their intrinsic labels. Their weight vectors are updated separately for accurate discrimination between the classes within this super class. Fig. 4 displays the situation that C_a and C_b are dispersed within the super class $\hat{C}_x$. In contrast, for tail classes in the rest super classes, the weight vectors remain aggregated for updating.

With the modification according to Eqs. (6) and (8), we selectively determine how the weight vectors are updated in the softmax classification depending on whether they are compared with the classes within the same super class. Overall, in this dispersion step, similar tail classes are separated when distinguished from the ones within the same super class, while they are gathered and updated as a whole when compared with the ones from other super classes or the head classes. This modified softmax can achieve accurate discrimination between fine classes and also effectively ameliorate the negative influences of long-tail distribution. The information in tail classes are better utilized to enhance the representation learning.

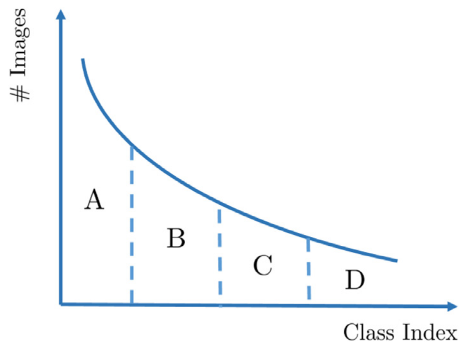


Fig. 5. The long-tail face dataset for training is divided into four groups A,B,C,D according to the number of face samples in each identity.

4. Experiments

In this section, we first introduce the two benchmarks for evaluation, i.e. BLUFR and MegaFace. Then we build a long-tail face dataset for training based on MS-Celeb-1M [11], which is the most challenging dataset collected from wild scenarios for face recognition. We further investigate the long-tail effect on this dataset and conduct extensive experiments to demonstrate the effectiveness of the proposed method in comparison with center loss [34], range loss [37], the state-of-the-art loss function ArcFace[5], and other baselines.

4.1. Experimental settings

The two benchmarks to evaluate the performance of deep face representation learning are BLUFR [20] and MegaFace [27]. **BLUFR** is a benchmark protocol based on the dataset LFW [16] to exploit all the 13,233 face images for large-scale unconstrained face recognition evaluation. It contains both verification (VR) and open-set identification (DIR) scenarios, with a focus at low false accept rate (FAR). With ten random trials of experiments, VR @FAR=0.1% and DIR @FAR=1% are adopted for performance measurement. **MegaFace** is a benchmark to evaluate performance of face recognition at the million scale of distractors. It contains gallery set with more than 1 million face images from 690K identities. The standard protocol uses Facescrub [28] as the probe set. Rank-1 identification accuracy is reported with 1 million distractors.

To thoroughly investigate the long-tail effect, we construct a long-tail face training set from a cleansed version of MS-Celeb-1M [11]. It consist of 85,164 identities and about 1.1 million face images. As illustrated in Fig. 5, we divide this dataset into 4 groups (A,B,C and D) according to the number of samples in a class. The

Table 1
The details about the division of the long-tail face dataset displayed in Fig. 5. # denotes the amount, % denotes the proportion. Image per class denotes the range of number of samples per identity in this group.

Groups	Identities		Images		Image Per Class
	#	%	#	%	
A	17,032	20.0%	695.8K	62.3%	[21,217]
B	15,000	18.6%	207.0K	18.6%	[10,20]
C	27,000	31.7%	163.5K	14.6%	[4,9]
D	26,132	30.7%	49.9K	4.5%	[1,3]
Total	85,164	100%	1116.2K	100%	[1,217]

Table 2
Performance on BLUFR and MegaFace w.r.t different compositions of training data. Trained with conventional softmax loss on ResNet34.

Groups	BLUFR		MegaFace Rank-1 Acc.
	VR @ FAR 0.1%	DIR @ FAR 1%	
A	96.08%	79.57%	61.25%
AB	**96.62%**	**83.78%**	**63.48%**
ABC	96.48%	82.45%	61.37%
ABCD	96.50%	82.11%	60.33%

exact division is displayed in Table 1. Simply, we examine the performance of conventional training schema with different compositions of training data. We use a 34-layer ResNet [14] as the base model for training. The results are shown in Table 2. We can see that training with all the samples (i.e. ABCD) does not leads to the best performance, and removing some tail classes such as C and D, i.e. training only with AB, achieves better performance. Therefore, the classes from CD are regarded as tail classes, while AB are used as head classes. In detail, we have $N_h = 32,032$ head classes and $N_t = 53,132$ tail classes in this dataset. The head classes from AB make up about 80% of all the samples but only consists of 40% of the identities. This dataset is used for training with all methods in the following experiments.

Our proposed method is implemented as follows. In the aggregation step, the base CNN model we adopt for feature extraction is trained only with N_h head classes (i.e. AB). Since tail classes from group C and group D are different in sample scarcity, we separately cluster the group C and group D. Specifically, group C and group D are respectively gathered into 1800 and 5400 super classes (i.e. N_k = 7200) to make them share similar number of samples in each category as that of head classes. The prototype model is trained with SGD optimizer with momentum for 40 epochs. The base learning rate is set as 0.1 and divided by 10 when the training error plateaus. In the dispersion step, fine-tuning starts with a smaller learning rate of 0.02 and lasts for 60 epochs for convergence. The aggregate-and-disperse training schema contains 100 epochs in total, and random sampling is adopted during the whole training process. For fair comparison, the baselines and other methods are also trained with the same number of epochs or iterations.

4.2. Performance and comparison

We evaluate the performance of the proposed method and compare it with baselines and state-of-the-art methods. The baselines include training with all the classes (i.e. ABCD), simply removing tail classes (i.e. AB), pre-training on head classes and then fine-tune with all the classes (i.e. AB-ABCD) and increasing the frequency of tail class samples (i.e. Resampling). Metric-based methods for comparison includes center loss [34], which is a classic metric-based loss for deep face representation learning, and range loss [37], which is proposed to handle such long-tail dataset. Moreover, the state-of-the-art ArcFace[5] is also included. The results are compared in Table 3.

As can be seen in Table 3, simply removing the tail classes can boost the DIR in BLUFR from 82.11% to 83.78% and the rank-1 accuracy in MegaFace from 60.33% to 63.48%, illustrating the negative influences of long-tail distribution. Pre-training on AB and then finetune on ABCD, referred to as AB-ABCD, only reports marginally better performance than only using AB, which illustrates that the tail classes are not fully utilized. Resampling the tail class samples, as another baseline, gives similar results. Center loss outperforms the inferior baseline of ABCD but still legs behind other baselines, as conventional softmax loss in center loss still makes it affected by the long-tail distribution. As for range loss, since it requires a

Table 3
Performance of the proposed method compared with baselines and other state-of-the-art methods on BLUFR and MegaFace.

Groups	BLUFR		MegaFace Rank-1 Acc.
	VR @ FAR 0.1%	DIR @ FAR 1%	
ABCD	96.50%	82.11%	60.33%
AB	96.62%	83.78%	63.48%
AB-ABCD	96.69%	84.27%	63.55%
Resampling	96.73%	84.36%	63.16%
Center Loss [34]	97.24%	83.74%	62.88%
Range Loss [37]	96.90%	84.13%	63.30%
Ours	97.33%	86.73%	64.90%
ArcFace [5]	**97.92%**	90.26%	68.63%
ArcFace+Ours	97.90%	**90.72%**	**69.27%**

Table 4
Ablation study on hyper-parameter N_k.

N_k	BLUFR		MegaFace Rank-1 Acc.
	VR @ FAR 0.1%	DIR @ FAR 1%	
Baseline	96.50%	82.11%	60.33%
1800	96.88%	84.16%	62.90%
3600	97.29%	**86.82%**	64.69%
7200	**97.33%**	86.73%	**64.90%**
14,400	97.05%	85.89%	64.02%

specific strategy to construct mini-batch, it does not show the expected performance (i.e. outperforms AB with an obvious margin) even if with carefully tuning. In contrast, our method boosts the DIR in BLUFR from 82.11% to 86.73% and the rank-1 accuracy in MegaFace from 60.33% to 64.90%, which demonstrates that it could effectively make better use of the information in tail classes by introducing super classes for the two-step training schema. Moreover, by introducing a most advanced loss function called ArcFace, the performance could be further boosted with a considerable margin. Fortunately, as an orthogonal method, our modified softmax loss could be simply added alongside ArcFace during training. Our additional term could still help improve the final performance, which demonstrate its complementarity to other methods and effectiveness when stronger baselines are involved.

4.3. Hyper-parameter sensitivity investigation

The main hyper-parameter N_k, which determines how many super classes are obtained after the first aggregation step, would have an impact on the final performance. In this section, we perform several experiments to investigate whether the performance is sensitive to N_k. We keep the principle to separately cluster group C and group D, and the ratio of corresponding number of clusters maintain 1:3 while N_k varies. Specifically, the value of N_k is set to 1800, 3600, 7200 and 14,400 respectively, and the corresponding average number of samples in each super class is about 118, 59, 30 and 15. The results are displayed in Table 4. As can be seen, it reports similar results when N_k is set to 3600 or 7200, which are much better than the baseline. While when N_k is too small or too large (e.g. 1800 or 14,400), the results become inferior. Overall speaking, our method is not so sensitive to N_k within a wide range. So empirically, we determined N_k based on the goal to have similar number of samples in the super class compared to the head class. Since for head class this number is about 28 on average, we correspondingly set N_k to 7200, which turns out to be a proper choice for this task.

5. Conclusion

In this work, we explore the problem of deep representation learning on large face dataset with long-tail distribution. Empirical study shows that with conventional training schema, tail classes with scarce samples might cripple the performance in the training process. To address this issue, we propose a two-step training schema with aggregation and dispersion in a coarse-to-fine manner. The first step aggregates those similar classes together to avoid the imbalance and few shot problem, and the second step selectively disperse the aggregated super class with a modified softmax to ensure accurate discrimination. Our proposed method achieves satisfactory results on two benchmarks (BLUFR and MegaFace) compared with baselines and other state-of-the-art methods, which demonstrates its effectiveness when dealing with long-tail data.

Declaration of Competing Interest

None.

Acknowledgements

This work was supported in part by the National Key R&D Program of China under Contract 2017YFA0700800, and in part by the Natural Science Foundation of China under Contract 61772496, and Contract 61732004, and Contract 61532018.

References

[1] M. Buda, A. Maki, M.A. Mazurowski, A systematic study of the class imbalance problem in convolutional neural networks, Neural Netw. (2018) 249–259.
[2] J. Byrd, Z. Lipton, What is the effect of importance weighting in deep learning? in: International Conference on Machine Learning, 2019, pp. 872–881.
[3] Q. Cao, L. Shen, W. Xie, O.M. Parkhi, A. Zisserman, Vggface2: a dataset for recognising faces across pose and age, in: IEEE International Conference on Automatic Face & Gesture Recognition, 2018, pp. 67–74.
[4] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, Smote: synthetic minority over-sampling technique, J. Artif. Intell. Res. (2002) 321–357.
[5] J. Deng, J. Guo, N. Xue, S. Zafeiriou, Arcface: additive angular margin loss for deep face recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 4690–4699.
[6] M. Dixit, R. Kwitt, M. Niethammer, N. Vasconcelos, Aga: attribute-guided augmentation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 7455–7463.
[7] C. Drummond, R.C. Holte, et al., C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling, in: Workshop on Learning from Imbalanced Datasets II, 2003, pp. 1–8.
[8] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 580–587.
[9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Advances inNeural Information Processing Systems, 2014, pp. 2672–2680.
[10] Y. Guo, L. Zhang, One-shot face recognition by promoting underrepresented classes, arXiv:1707.05574 (2017).
[11] Y. Guo, L. Zhang, Y. Hu, X. He, J. Gao, Ms-celeb-1m: a dataset and benchmark for large-scale face recognition, in: European Conference on Computer Vision, 2016, pp. 87–102.
[12] B. Hariharan, R. Girshick, Low-shot visual recognition by shrinking and hallucinating features, in: IEEE International Conference on Computer Vision, 2017, pp. 3018–3027.
[13] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. (2008) 1263–1284.
[14] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[15] C. Huang, Y. Li, C. Change Loy, X. Tang, Learning deep representation for imbalanced classification, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 5375–5384.
[16] G.B. Huang, M. Ramesh, T. Berg, E. Learned-Miller, Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments, Technical Report, 2007.
[17] T. Karras, S. Laine, T. Aila, A style-based generator architecture for generative adversarial networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 4401–4410.
[18] S.H. Khan, M. Hayat, M. Bennamoun, F.A. Sohel, R. Togneri, Cost-sensitive learning of deep feature representations from imbalanced data, IEEE Trans. Neural Netw. Learn. Syst. (2018) 3573–3587.
[19] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
[20] S. Liao, Z. Lei, D. Yi, S.Z. Li, A benchmark study of large-scale unconstrained face recognition, in: IEEE International Joint Conference on Biometrics, 2014, pp. 1–8.
[21] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: IEEE International Conference on Computer Vision, 2017, pp. 2980–2988.
[22] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, L. Song, Sphereface: deep hypersphere embedding for face recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 212–220.
[23] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.
[24] C. Lu, X. Tang, Surpassing human-level face verification performance on lfw with gaussian face, AAAI Conference on Artificial Intelligence, 2015.
[25] Y. Ma, M. Kan, S. Shan, X. Chen, Hierarchical training for large scale face recognition with few samples per subject, in: IEEE International Conference on Image Processing, 2018, pp. 2401–2405.
[26] T. Maciejewski, J. Stefanowski, Local neighbourhood extension of smote for mining imbalanced data, in: IEEE Symposium on Computational Intelligence and Data Mining, 2011, pp. 104–111.
[27] A. Nech, I. Kemelmacher-Shlizerman, Level playing field for million scale face recognition, IEEE Conference on Computer Vision and Pattern Recognition, 2017.
[28] H.-W. Ng, S. Winkler, A data-driven approach to cleaning large face datasets, in: IEEE International Conference on Image Processing, 2014, pp. 343–347.
[29] F. Schroff, D. Kalenichenko, J. Philbin, Facenet: a unified embedding for face recognition and clustering, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 815–823.
[30] Y. Shen, J. Gu, X. Tang, B. Zhou, Interpreting the latent space of gans for semantic face editing, arXiv:1907.10786 (2019).
[31] Y. Sun, D. Liang, X. Wang, X. Tang, Deepid3: face recognition with very deep neural networks, arXiv:1502.00873 (2015).
[32] K.M. Ting, A comparative study of cost-sensitive boosting algorithms, International Conference on Machine Learning, 2000.
[33] C. Wang, X. Zhang, X. Lan, How to train triplet networks with 100k identities? in: IEEE International Conference on Computer Vision, 2017, pp. 1907–1915.
[34] Y. Wen, K. Zhang, Z. Li, Y. Qiao, A discriminative feature learning approach for deep face recognition, in: European Conference on Computer Vision, 2016, pp. 499–515.
[35] X. Yin, X. Yu, K. Sohn, X. Liu, M. Chandraker, Feature transfer learning for deep face recognition with long-tail data, arXiv:1803.09014 (2018).
[36] B. Zadrozny, J. Langford, N. Abe, Cost-sensitive learning by cost-proportionate example weighting., in: International Conference on Data Mining, 2003, p. 435.
[37] X. Zhang, Z. Fang, Y. Wen, Z. Li, Y. Qiao, Range loss for deep face recognition with long-tailed training data, in: IEEE International Conference on Computer Vision, 2017, pp. 5409–5418.