Deep Representation Learning on Long-tailed Data: A Learnable Embedding Augmentation Perspective

Jialun Liu[1]*, Yifan Sun[2]*, Chuchu Han[3], Zhaopeng Dou[4], Wenhui Li[1]†
[1]Jilin University [2]Megvii Inc. [3]Huazhong University of Science and Technology [4]Tsinghua University
jialun18@mails.jlu.edu.cn peter@megvii.com liwh@jlu.edu.cn

Abstract

This paper considers learning deep features from long-tailed data. We observe that in the deep feature space, the head classes and the tail classes present different distribution patterns. The head classes have a relatively large spatial span, while the tail classes have a significantly small spatial span, due to the lack of intra-class diversity. This uneven distribution between head and tail classes distorts the overall feature space, which compromises the discriminative ability of the learned features. In response, we seek to expand the distribution of the tail classes during training, so as to alleviate the distortion of the feature space. To this end, we propose to augment each instance of the tail classes with certain disturbances in the deep feature space. With the augmentation, a specified feature vector becomes a set of probable features scattered around itself, which is analogical to an atomic nucleus surrounded by the electron cloud. Intuitively, we name it as "feature cloud". The intra-class distribution of the feature cloud is learned from the head classes, and thus provides higher intra-class variation to the tail classes. Consequentially, it alleviates the distortion of the learned feature space, and improves deep representation learning on long tailed data. Extensive experimental evaluations on person re-identification and face recognition tasks confirm the effectiveness of our method.

1. Introduction

Large-scale datasets play a crucial role in deep representation learning, as well as in many other deep learning based visual tasks. In the real-world, large-scale datasets often exhibit extreme long-tailed distribution [8, 10]. Concretely, some identities have sufficient samples, while for other massive identities, only very few samples are available. They are defined as the head classes and tail classes, respectively. Long-tailed distribution poses great challenge to deep representation learning [1].

We investigate the impact of long-tailed distribution with focus on the deeply learned feature space. In a specified deep representation task, *i.e.*, person re-identification (re-ID), We visualize several neighboring head classes in Fig. 1 and find that the sample number is an important factor for intra-class diversity. Firstly we observe the original distribution of the head classes in Fig. 1 (a). The head classes can be well distinguished with a distinct margin. With rich intra-class diversity, each head class occupies a wide span in the feature space. Further, we reduce the samples of some head classes so they change to tail classes. As is shown in Fig. 1 (b), we discover that samples from tail class distribute narrowly in the learned feature space, due to the lack of intra-class diversity. This uneven distribution distorts the overall feature space and consequentially compromises the discriminative ability of the learned features.

To be more concrete, we further quantitatively investigate the intra-class diversity *w.r.t.* the long-tailed distribution. Given a specified class, we calculate the geometric angles between the features and the corresponding class center in the deep feature space. We transform a re-ID dataset (*i.e.*, DukeMTMC-reID) into a long-tailed one by setting some classes to have only 4 samples. Under a popular baseline for deep representation learning [35], the variations of head classes are distributed within 0.463 ± 0.030 (95% Confidence Interval (CI)). In contrast, the variations of tail classes are significantly small, with 0.288 ± 0.023 as the 95% CI. Such observations further confirms that 1) tail classes have smaller variance and 2) the sample number per class is the dominating factor on the variance.

With this insight, we propose to transfer the intra-class distribution of head classes to tail classes in the feature space. Our target is to encourage the tail classes to achieve similar intra-class angular variability with the head classes in training. Specifically, we first calculate the distribution of angles between the features of head class and their corresponding class center. By averaging the angular variances of all the head classes, we obtain the overall variance of head classes. Next, we consider transferring the variance

*Equal contribution.
†Corresponding author.

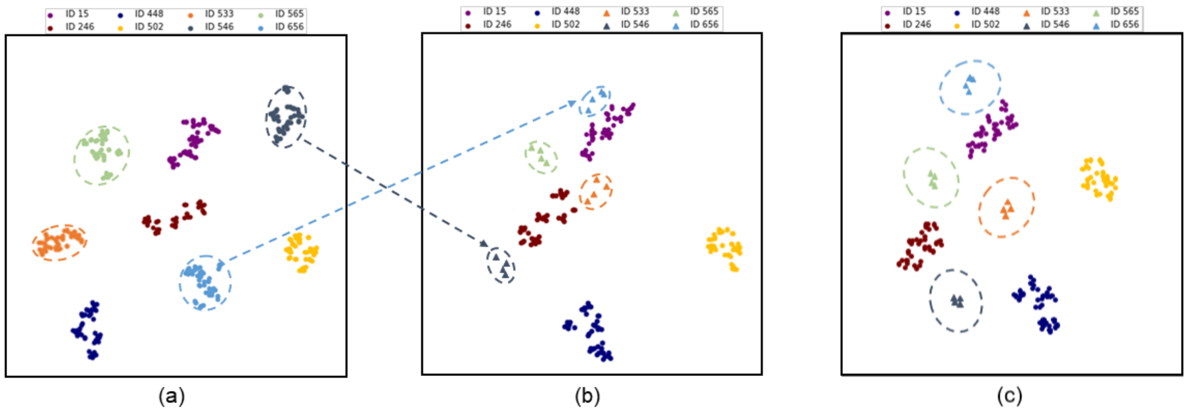


Figure 1. We select several classes from DukeMTMC-reID dataset [44, 24] then visualize features in the embedding layer with t-SNE [32]. (a) The visualization of features from head classes (dot). With the wide region in the feature space, each class can be well distinguished. (b) We reduce the samples of some head classes so that they become tail classes (triangle). With these tail classes, the spanned feature space is narrowed, which leads to the distortion of the original feature space. So it is hard for the tail classes to be separated from other classes. (c) In training, the space is expanded for the tail class so that it is pushed away from others.

of head class to each tail class. To this end, we propose to augment each tail class instance with certain disturbances in the deep feature space. With the augmentation, a specified feature vector becomes a set of probable features scattered around itself, which is termed “feature cloud”. Each instance with the corresponding feature cloud will have a relatively large distribution range, making the tail classes have a similar angular distribution with head class. Our method enforces stricter supervision on the tail classes, and thus leads to higher within-class compactness. As Fig. 1 (c) shows, with the compensation of intra-class diversity during training, the tail classes are separated from other classes by a clear margin. Under the setting of re-ID mentioned before, the intra-class angular variance of tail classes turns out over even lower (than the tail classes in baseline), which is cenntered at 0.201.

Moreover, to improve the flexibility of the method, we abandon the explicit definition of head class and tail class. Compared with some methods that divide the two classes, our approach makes the calculation entirely related to the distribution of dataset, and there is no human interference.

We summarize the contributions of our work as follows:

- We propose a learnable embedding augmentation perspective to alleviate the problem of discriminative feature learning on long-tailed data, which transfers the intra-class angular distribution learned from head classes to tail classes.
- Extensive ablation experiments on re-ID and face recognition demonstrate the effectiveness of the proposed method.

2. Related Work

Feature learning on imbalanced datasets. Recent works for feature learning on imbalanced data are mainly divided into three manners: re-sampling [1], re-weighting [21], and data augmentation[3]. The re-sampling technique includes two types: over-sampling the tail classes and under-sampling the head classes. Over-sampling manner samples the tail data repeatedly, which enables the classifier to learn tail classes better. But it may lead to over-fitting of tail classes. To reduce the risk of over-fitting, SMOTE [2] is proposed to generate synthetic data of the tail class. It randomly places the newly created instances between each tail class data point and its nearest neighbor. The under-sampling manner [6] reduces the amount of data from head classes while keeping the tail classes. But it may lose valuable information on head classes when data imbalance is extreme. The re-weighting approach assigns different weights for different classes or different samples. The traditional method re-weights classes proportionally to the inverse of their frequency of samples. Cui *et al.* [4] improve the re-weighting by the inverse effective number of samples. Li *et al.* [18] propose a method which down-weights examples with either very small gradients or large gradients because examples with small gradients are well-classified and those with large gradients tend to be outliers. Recently, data augmentation methods based on Generative Adversarial Network (GAN) [3] are popular. [41] and [9] transfer the semantic knowledge learned from the head classes to compensate tail classes, which encourage the tail classes to have similar data distribution to the head classes. All the methods divide the classes into the head or tail class, while our method abandons the constraint.

Loss function. Loss function plays an important role in deep feature learning, and the most popular one is the Softmax loss [28]. However, it mainly considers whether the samples can be correctly classified and lacks the constraint of inter-class distance and intra-class distance. In order to improve the feature discrimination, many loss func-

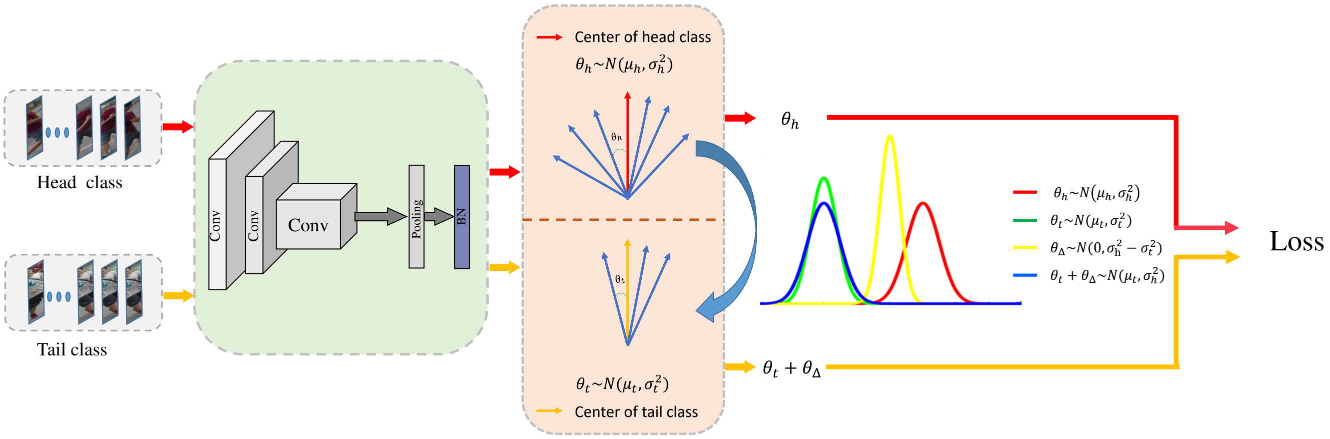

Head class
Tail class
Conv
Conv
Conv
Pooling
BN
Center of head class
$\theta_h \sim N(\mu_h, \sigma_h^2)$
θ_h
θ_t
θ_h
$\theta_h \sim N(\mu_h, \sigma_h^2)$
$\theta_t \sim N(\mu_t, \sigma_t^2)$
$\theta_\Delta \sim N(0, \sigma_h^2 - \sigma_t^2)$
$\theta_t + \theta_\Delta \sim N(\mu_t, \sigma_h^2)$
Loss
$\theta_t + \theta_\Delta$
$\theta_t \sim N(\mu_t, \sigma_t^2)$
Center of tail class

Figure 2. Overview of our proposed framework. The head data and tail data are fed into the deep network to obtain the features. We calculate the distribution of angles between the features and the class center for head class and tail class, respectively. Subsequently, we transfer the angular variance of head class (red curve) to tail class (green curve). In other words, based on the original distribution of tail class, we add an additional distribution (yellow curve). Then we get a new distribution of tail class (blue curve). Finally, we use the head data and the new tail data to calculate the loss.

tions are proposed to enhance the cosine and angular margins between different classes. Wen *et al.* [39] design a center loss to reduce the distance between the sample and the corresponding class center. The L2-Softmax [23] and NormFace [34] add normalization to produce represented features and achieve better performance. Besides normalization, adding a margin can enhance the discrimination of features by inserting distance among samples of different classes. A-Softmax Loss [20] normalizes the weights and adds multiplicative angular margins to learn more divisible angular characteristics. CosFace [35] adds an additive cosine margin to compress the features of the same class in a compact space, while enlarging the gap of features of different classes. ArcFace [5] puts an additive margin into angular space so that the loss relies on both sine and cosine dynamically to learn more angular characteristics. CosFace [35] and ArcFace [5] are chosen as baseline. Althoughwe model the intra-class angle, which is similar to them, our goal is to solve the problem of discriminative feature learning on long-tailed data.

3. The Proposed Approach

In this section, A brief description of our method is given in Section 3.1. We review the baseline in Section 3.2. We describe the updating process of the class center and the calculation of angular distribution in Section 3.3. The construction of the feature cloud is detailed in Section 3.4.

3.1. Overview of Framework

The framework of our method is shown in Fig. 2. First, the head data and tail data are fed into the deep model to extract features. And we consider to model the distribution of intra-class features by the distribution of angles between features and their corresponding class center. Then the center of each class is calculated, as to be detailed in Section 3.3. We build an angle memory for each class, which is used to store the angles between the features and their class center. Assuming the angles obey the Gaussian distribution, the angular distributions of head class and tail class can be denoted as $\theta_h \sim \mathrm{N}(\mu_h, \sigma_h^2)$ and $\theta_t \sim \mathrm{N}(\mu_t, \sigma_t^2)$, respectively. Next, we transfer the angular variance learned from the head class to every tail class. Consequently, the intra-class angular diversity of tail class is similar to the head class. Specifically, we build a feature cloud around each tail instance. An instance sampled from the feature cloud has the same identity with the tail instance. The angle between them is θ_Δ and $\theta_\Delta \sim \mathrm{N}(0, \sigma_h^2 - \sigma_t^2)$. We assume the two distribution: $\theta_t \sim \mathrm{N}(\mu_t, \sigma_t^2)$ and $\theta_\Delta \sim \mathrm{N}(0, \sigma_h^2 - \sigma_t^2)$ are independent of each other. By transformation, the new intra-class angular distribution of tail class is built as $\theta_t + \theta_\Delta \sim \mathrm{N}(\mu_t, \sigma_h^2)$ in training process. Finally, we use the original features of head classes and the reconstructed features of tail classes to calculate the loss.

3.2. Baseline Methods

The traditional softmax loss optimizes the decision boundary between two categories, but it lacks the constraint of inter-class distance and intra-class distance. CosFace [35] effectively minimizes intra-class distance and maximums inter-class distance by the introducing a cosine margin to maximize the decision margin in the angular space. The loss function can be formulated as:

$$L_1 = -\frac{1}{N}\sum_{n=1}^{N} \log \frac{e^{s(\cos(\theta_{y_n})-m_c)}}{e^{s(\cos(\theta_{y_n})-m_c)} + \sum_{j\neq y_n}^{C} e^{s\cos(\theta_j)}}, \tag{1}$$

where N and C are the mini-batch size and the number of total classes, respectively. y_n is the label of n-th image.

We define the feature vector of n-th image and the weight vector of class y_n as f_n and W_y, respectively. f_n and W_y are normalized by l_2 normalisation and the norm of feature vector is rescaled to s. θ_{y_n} is the angle between the weight W_y and the feature f_n. m_c is a hyper-parameter controlling the magnitude of the cosine margin.

Different from CosFace [35], ArcFace [5] employs an additive angular margin loss, which is formulated as:

$$L_2 = -\frac{1}{N}\sum_{n=1}^{N} \log \frac{e^{s(\cos(\theta_{y_n}+m_a))}}{e^{s(\cos(\theta_{y_n}+m_a))} + \sum_{j\neq y_n}^{C} e^{s\cos(\theta_j)}}, \tag{2}$$

where m_a is an additive angular margin penalty between feature vector f_n and its corresponding W_y. It aims to enhance the intra-class compactness and inter-class distance simultaneously.

In this paper, we choose CosFace [35] and ArcFace [5] as baseline. The reasons are as follows:

- They have achieved the state-of-the-art performance in the face recognition task, which can be seen as strong baselines in the community of deep feature learning.
- They optimize the intra-class similarity by achieving much lower intra-class angular variability. Since our method employs intra-class angles to model the intra-class feature distribution, the two loss functions can be naturally combined with our method.

3.3. Learning the intra-class angular distribution

The intra-class angular diversity can intuitively show the diversity of intra-class features. In this section, we study the distribution of angles between the features and their corresponding class center. c_i denotes the i-th class center of features. f_i^k is the k-th instance feature of class i. c_i has the same dimension as f_i^k. So, we can calculate the angle between f_i^k and c_i as follow:

$$\beta_{i,k} = arccos(\frac{f_i^k c_i}{||f_i^k||||c_i||}), \tag{3}$$

where the c_i should be updated in the training processing. Ideally, we need to take the entire training samples into account and average the features of every class in each epoch. Obviously, this approach is impractical and inefficient. Inspired by [39], we also perform the update based on a mini-batch. In each mini-batch, the class center is computed by averaging the feature vectors of the corresponding class. To avoid the misleading by some mislabelled samples, we set a center learning rate γ to update the class center. The updating method of c_i is formulated as:

$$c_i^l = (1-\gamma)c_i^l + \gamma c_i^{l-1}, \tag{4}$$

where c_i^l is the center of class i in l-th mini-batch. Each class center is updated by the center of current and previous mini-batch.

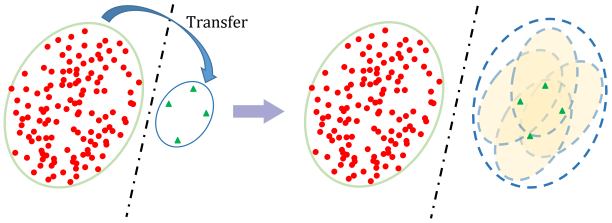


Figure 3. We transfer the intra-class angular distribution learned from the head class to the tail class.

For the class i, we maintain an angle memory β_i to store the angles between the features and their corresponding class center c_i. The size of angle memory is formulated as:

$$S_i = K_i \times P. \tag{5}$$

K_i is the sample number of the i-th class. P is a hyper-parameter determining the angle memory per class. Then we calculate the mean μ_i and variance σ_i^2 of β_i. The angular distribution of the class i is formulated as $\mathrm{N}(\mu_i, \sigma_i^2)$.

3.4. Constructing the feature cloud for tail data

In this section, we elaborate the process of constructing the feature cloud for a tail instance. First, like the previous works [41, 46], we assign a label to mark the head and tail class, yielding the vanilla version. On the other hand, we introduce a full version which abandons the explicit division of head and tail class. This manner is more flexible since it is only related to the distribution of the dataset.

Vanilla version. We strictly divide the head class and the tail class through a threshold T. If the number of samples belonging to class i is larger than T, the i-th class is defined as a head class. Otherwise, it is defined as a tail class.

In the Section 3.3, we have calculated the angular distribution of each class, which is assumed to lie in Gaussian distribution. By averaging the variance of all head classes, we obtain the overall variance of the head class. The mean is computed in the similar way. So the overall angular distribution of the head class is as follow:

$$\mu_h = \frac{\sum_{z=1}^{C_h} \mu_z}{C_h}, \qquad \sigma_h^2 = \frac{\sum_{z=1}^{C_h} \sigma_z^2}{C_h}, \tag{6}$$

where C_h is the number of head classes. μ_z and σ_z^2 is the angular mean and variance of the z-th head class, respectively. μ_h and σ_h^2 describe the overall angular distribution of the head class. We can also obtain the class center for every tail class. The angular distribution of the x-th tail class is denoted as $\mathrm{N}(\mu_t^x, {\sigma_t^x}^2)$.

For the head classes, they include sufficient samples which show the intra-class angular diversity. In general, σ_h is greater than σ_t, so our target is to transfer σ_h^2 to each tail class. As is shown in Fig. 3, we construct a feature cloud around each feature of x-th tail class. By this way, the space spanned by tail class is enlarged, in training, and the real tail instances are pushed away from other classes. The angle between the feature belonging to the x-th tail class and a feature sampled from its corresponding feature cloud is α_x, where $\alpha_x \sim \mathrm{N}(0, \sigma_h^2 - {\sigma_t^x}^2)$ and $\alpha_x \in \mathbb{R}^{1\times C}$. In training, the feature sampled from the feature cloud shares the same identity with the real tail feature. We have assumed the two distributions: $\mathrm{N}(\mu_t^x, {\sigma_t^x}^2)$ and $\mathrm{N}(0, \sigma_h^2 - {\sigma_t^x}^2)$ are independent of each other in Section 3.1. So the original angular distribution of the x-th tail class is transferred from $\mathrm{N}(\mu_t^x, {\sigma_t^x}^2)$ to $\mathrm{N}(\mu_t^x, {\sigma_h}^2)$.

The new loss functions based on CosFace [35] and ArcFace [5] are defined as:

$$L_3 = -\frac{1}{N}\sum_{n=1}^{N}\log\frac{e^{s(\cos(\theta_{y_n}+\alpha_y)-m_c)}}{e^{s(\cos(\theta_{y_n}+\alpha_y)-m_c)}+\sum_{j\neq y_n}^{C}e^{s\cos(\theta_j+\alpha_y)}}, \tag{7}$$

$$L_4 = -\frac{1}{N}\sum_{n=1}^{N}\log\frac{e^{s(\cos(\theta_{y_n}+\alpha_y+m_a))}}{e^{s(\cos(\theta_{y_n}+\alpha_y+m_a))}+\sum_{j\neq y_n}^{C}e^{s\cos(\theta_j+\alpha_y)}}, \tag{8}$$

in Eq.7 and 8, $\theta + \alpha$ and $\theta + \alpha + m_a$ are all clipped in the range $[0, \pi]$. N and C are the mini-batch size and class number, respectively. θ_{y_n} is the angle between the feature f_n and the weight W_y . s is the scale, and m_c, m_a are the cosine margin and the angular margin in CosFace [35] and ArcFace [5], respectively. If y_n is a head class, α_y = 0. As the training progresses, the tail class has the rich angular diversity as head class.

Actually, we approximate the angle (θ') between the feature sampled from feature cloud and the weight. If $\alpha > 0$, we approximate θ' by the upper bound of it, and the lower bound when $\alpha \leq 0$. The proof is given below.

Proposition. We denote a feature in the tail class as f, and W is the corresponding weight vector in the full connection layer. f' is a feature randomly sampled from the feature cloud around f.

$$\langle f, W\rangle = \theta, \quad \langle f, f'\rangle = \alpha_+, \quad \langle W, f'\rangle = \theta',$$

$$\|f\| = \|w\| = \|f'\| = 1, \quad 0 \leq \theta + \alpha_+ \leq \pi,$$

where $\langle a, b\rangle$ represents the angle between vector a and b, and $\|a\|$ represent the norm of vector a. We want to prove: $|\theta - \alpha_+| \leq \theta' \leq \theta + \alpha_+$.

Proof. Simply, we suppose that $f = [1, 0, \cdots, 0]$, then $W = [\cos\theta, w_2, \cdots, w_n]$. We use the Householder transformation [13] to transform W to V, where $V = [\cos\theta, \sin\theta, 0, \cdots, 0]$. Let $P = I - 2U \cdot U^T$, where $U = W - V/\|W - V\|$, then $f = Pf, V = PW, \hat{f}' = Pf'$. P is an orthogonal transformation which preserves the inner product and norm. Therefore, we have

$$\langle f, V\rangle = \theta, \quad \langle f, \hat{f}'\rangle = \alpha_+, \quad \langle V, \hat{f}'\rangle = \theta'.$$

Denote $\hat{f}' = [\hat{f}'_1, \hat{f}'_2, \cdots, \hat{f}'_n]$, then

$$\cos\alpha_+ = f \cdot \hat{f}' = \hat{f}'_1, \quad {\hat{f}'_2}{}^2 + \cdots + {\hat{f}'_n}{}^2 = \sin^2\alpha_+.$$

We get $\hat{f}'_2 \sin\theta \in [-sin\alpha_+ sin\theta, sin\alpha_+ sin\theta]$, where $\theta \in [0, \pi]$. Further, we have

$$\cos\theta' = \hat{f}' \cdot V = \cos\alpha_+ \cos\theta + \hat{f}'_2 \sin\theta,$$

$$\cos\theta' \in [cos(\theta + \alpha_+), cos(\theta - \alpha_+)].$$

We get the conclusion: $|\theta - \alpha_+| \leq \theta' \leq \theta + \alpha_+$.

Although $\alpha \sim N(0, \sigma^2)$, we only need to focus on $\alpha \in [-\pi, \pi]$, since $\theta + \alpha$ is clipped in the range $[0, \pi]$.

- when $0 \leq \alpha \leq \pi$, substituting α for α_+, we have $|\theta - \alpha| \leq \theta' \leq \theta + \alpha$, in which $\theta + \alpha$ is the upper bound.

- when $-\pi \leq \alpha \leq 0$, substituting $-\alpha$ for α_+, we have $|\theta - (-\alpha)| \leq \theta' \leq \theta + \alpha$, which is equivalent to $\theta + \alpha \leq \theta' \leq \theta - \alpha$, so $\theta + \alpha$ is the lower bound.

Full version. The distorted feature space is well repaired by constructing a feature cloud around a tail instance. But the process in the vanilla version is inflexible. We set a threshold T to divide the head and tail classes, artificially. The overall angular distribution in Eq.6 only depends on the head classes. In the full version, the explicit definition is discarded. We have observed that the intra-class diversity is positively correlated with the sample number, in general. Therefore, we calculate the overall variance by weighting the angular variance of each class. The weight is the sample number per class. The final variance is formulated as:

$$\sigma^2 = \sum_{i=1}^{C}\frac{(K_i - 1)\sigma_i^2}{\sum(\mathrm{K_i} - 1)}, \tag{9}$$

where C is the number of classes, and K_i is the number of samples belong to class i. σ_i^2 is the angular variance of the i-th class. A smaller K_i means that the variance of the i-th class almost has no contribution to the final variance, so the final variance mainly depends on the classes with sufficient samples. For i-th class, if $\sigma_i^2 < \sigma^2$, it means the class i has poor intra-class diversity. Therefore α_y is available in Eq.7 and 8, and we construct the feature cloud for each instance sampled from class i.

The advantage of the full version is that the calculation of feature cloud entirely depends on the distribution of the dataset. There is no human interference in the process.

4. Experiments

In this section, we conduct extensive experiments to confirm the effectiveness of our method. First we describe the experimental settings. Then we show the performance on person re-identification and face recognition with different long-tailed settings.

4.1. Settings

Person re-identification. Evaluations are conducted on three datasets: Market-1501 [42], DukeMTMC-reID [24, 44] and MSMT17 [37]. To study the impact of the ratio between head classes and tail classes on training a re-ID system, we construct several long-tailed datasets based on the original dataset. We rank the classes by their number of samples. The top $150, 100, 50$ and 20 identities are marked as the head class, respectively. The rest is treated as the tail classes, and the number of samples is reduced to 5 each class. In this way, we form the training sets of $\langle H150, S5\rangle$, $\langle H100, S5\rangle$, $\langle H50, S5\rangle$, and $\langle H20, S5\rangle$. For training, we choose the widely used ResNet-50 [11] as the backbone. The last layer of the network is followed by a Batch Normalization layer (BN). The optimizer is Adam. The scale s and m_c of CosFace [35] are set to be 24 and 0.2, respectively. The scale s and m_a of ArcFace [35] are set to be 16 and 0.2, respectively. The learning rate of class center γ is set to be 0.1. For testing, the 2048-d global features after BN are used for evaluation. The cosine distance of features is computed as the similarity score. We use two evaluation metrics: Cumulative Matching Characteristic (CMC) and mean average precision(mAP) to evaluate our method.

Face recognition. We adopt the widely used dataset MS-Celeb-1M for training. The original MS-Celeb-1M data is known to be very noisy, so we clean the dirty face images and exclude the $79K$ identities and $1M$ images. We rank the classes through the number of samples they have. The top $5K$ and $3K$ are selected as head classes. Among the rest classes, we select the first $10K$ and $20K$ as tail classes and randomly pick 5 images per class. In this way, we form the training set of $\langle H5K, T20K\rangle$, $\langle H5K, T10K\rangle$, $\langle H3K, T20K\rangle$ and $\langle H3K, T10K\rangle$. The face images are resized to 112×112. For training, we choose the ResNet-18 [11] as our backbone. We train the model for 30 epoch by adopting the triangular learning rate policy[26], and construct feature cloud at the start of the third cycle. The scale s and m_c of CosFace [35] are set to be 64 and 0.35. The scale s and m_a of ArcFace [35] are set to be 64 and 0.5. We extract 512-D features for model inference. For testing, we evaluate our method on LFW [14], MegaFace challenge1 (MF1) [17] and IJB-C [22]. We report our results on the Rank-1 accuracy of LFW and MF1, and different TPR@FPR of IJB-C TPR@FPR.

4.2. Experiments on person re-identification

Performance of baseline. Table 1 reports the results of the baseline. We compare our baseline with the advanced methods. Our baseline achieves very competitive performance, which is reliable.

Methods	Market-1501		DukeMTMC		MSMT17	
	mAP	Rank-1	mAP	Rank-1	mAP	Rank-1
HA-CNN [19]	75.7	91.2	63.8	80.5	-	-
PCB [30]	77.4	92.3	66.1	81.8	40.4	68.2
Mancs [33]	82.3	93.1	71.8	84.9	-	-
CosFace	79.5	92.4	73.0	85.6	49.2	75.3
ArcFace	81.1	92.5	73.2	85.8	50.5	75.5

Table 1. Comparison with the advanced methods on the Market-1501, DukeMTMC-reID and MSMT17 datasets

Methods		Market-1501		DukeMTMC	
		mAP	Rank-1	mAP	Rank-1
GF	SVDNet [29]	62.1	82.3	56.8	76.7
	BraidNet [36]	69.5	83.7	69.5	76.4
	CamStyle [47]	71.6	89.5	57.6	78.3
	Advesarial [15]	70.4	86.4	62.1	79.1
	Dual [7]	76.6	91.4	64.6	81.8
	Mancs [33]	82.3	93.1	84.9	71.8
	IANet [12]	83.1	94.4	73.4	87.1
	DG-Net [43]	**86.0**	**94.8**	**74.8**	86.6
PF	AACN [40]	66.9	85.9	59.2	76.8
	PSE [25]	69.0	87.7	62.0	79.8
	PCB [30]	77.4	92.3	66.1	81.8
	SPReID [16]	81.3	92.5	70.9	84.4
Ours	LEAP-CF	84.2	94.4	74.2	**87.8**
	LEAF-AF	83.2	93.5	74.2	86.9

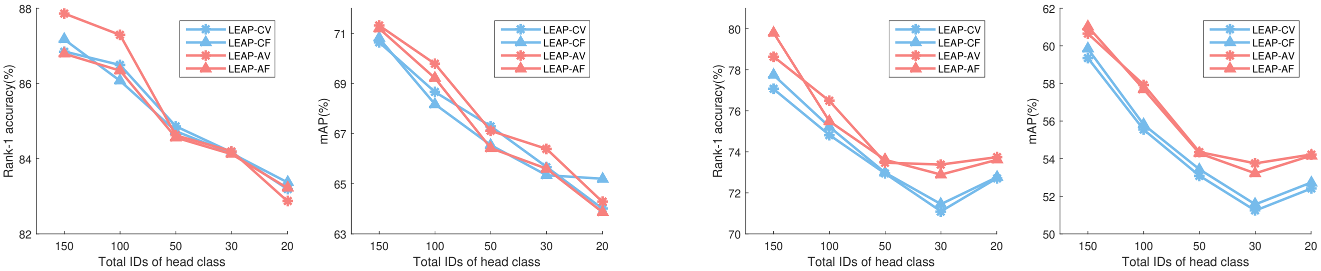
Table 2. Comparison with state-of-the-art methods on Market-1501 and DukeMTMC-reID. Three groups: global features(GF), part features(PF) and ours. LEAP-CF and LEAP-AF are our full version combined with CosFace and ArcFace, respectively.

Comparison with state-of-the-art approaches. We compare our full version with the state-of-the-art methods on Market-1501 and DukeMTMC-reID. The comparisons are reported in Table 2. It shows that our baseline has surpassed many advanced methods. And our method further improve the performance compared with baseline. Specifically, LEPA-CF achieves 94.4% on rank-1 for Market-1501, and 87.8% on rank-1 for DukeMTMC-reID. We also evaluate our method on a recently released dataset MSMT17 [37]. The result is shown in Table 3. Compared with DG-Net [43], our performance is very close to it. However, our method is a simple but efficient method, which does not use GAN to generate many image-level samples.

Methods	mAP	Rank-1	Rank-5	Rank-10
GoogleNet [31]	23.0	47.6	65.0	71.8
Pose-driven [27]	29.7	58.0	73.6	79.4
Verif-Identif [45]	31.6	60.5	76.2	81.6
GLAD [38]	34.0	61.4	76.8	81.6
PCB [30]	40.4	68.2	81.2	85.5
IANet [12]	46.8	75.5	85.5	88.7
DG-Net [43]	**52.3**	**77.2**	**87.4**	**90.5**
LEAP-CF	50.8	76.7	86.9	90.0
LEAP-AF	51.3	76.3	86.5	89.8

Table 3. Comparison with advanced methods on the MSMT17.

Evaluation with the vanilla version. We evaluate the



(a) Market-1501

(b) DukeMTMC

Figure 4. Comparison of vanilla version and full version on Market-1501 and DukeMTMC-reID. LEAP-CV and LEAP-AV are our vanilla version combined with CosFace and ArcFace, respectively. LEAP-CF and LEAP-AF are our full version combined with CosFace and ArcFace, respectively.

Dataset →		Market-1501		DukeMTMC	
Train ↓	Method ↓	mAP	Rank-1	mAP	Rank-1
$\langle H150, S5\rangle$	CosFace	67.3	86.3	57.3	75.6
	LEAP-CV	**70.6**	**86.9**	**59.4**	**77.1**
	ArcFace	70.6	87.3	60.2	77.6
	LEAP-AV	**71.3**	**87.9**	**60.6**	**78.7**
$\langle H100, S5\rangle$	CosFace	62.8	83.3	52.6	70.3
	LEAP-CV	**68.7**	**86.5**	**55.6**	**74.8**
	ArcFace	68.0	86.6	56.7	74.8
	LEAP-AV	**69.8**	**87.3**	**57.9**	**76.5**
$\langle H50, S5\rangle$	CosFace	60.5	80.7	48.0	67.7
	LEAP-CV	**67.3**	**84.9**	**53.1**	**73.0**
	ArcFace	64.2	83.8	51.1	71.1
	LEAP-AV	**67.1**	**84.6**	**54.4**	**73.5**
$\langle H20, S5\rangle$	CosFace	55.6	78.6	47.0	66.0
	LEAP-CV	**64.1**	**83.2**	**52.4**	**72.7**
	ArcFace	60.1	81.1	50.5	69.3
	LEAP-AV	**64.3**	**82.2**	**54.2**	**73.7**

Table 4. Controlled experiments by varying the ratio between head and tail data. H is the number of head class. S denotes that the sample number per tail class. CosFace and ArcFace are baselines. LEAP-CV and LEAP-AV are vanilla versions combined with CosFace and ArcFace.

effectiveness of the vanilla version. For comparison, we train the baseline model on the long-tailed re-ID datasets under the supervision of CosFace [35] and ArcFace [5]. We compare our method with baseline methods. The results are shown in Table 4. We have the following observations. First, compared with CosFace, ArcFace has higher Rank-1 and mAP accuracy on the same long-tailed setting. For example, on Market-1501 with $\langle H20, S5\rangle$, ArcFace achieves the Rank-1 accuracy of 81.1%, while the Rank-1 accuracy of CosFace is 78.6%. This indicates that Arcface has a stronger robustness for the long-tailed re-ID. Second, in different long-tailed settings, the proposed LEAP method combined with CosFace and ArcFace achieves consistently better results than the baseline with significant margins. This indicates that the LEAP is a robust method for long-tailed data distribution. Third, as the long-tailed distribution is more serious, the improvement of our method becomes even more obvious. For example, in the $\langle H20, S5\rangle$ setting on DukeMTMC-reID, the improvement of LEAP-CV reaches $+6.7\%$ (from 66.0% to 72.7%) in the Rank-1 accuracy.

Comparison between vanilla version and full version. We show the results comparison of vanilla version and full version under different long-tailed settings in Fig. 4. We observe that the full version obtains the results very close to vanilla version, and even better results in some settings. By this experiment, we justify that compared with those methods which need a label to distinguish between head class and tail class, the full version is more flexible.

Dataset →		Market-1501		DukeMTMC	
Train ↓	Method ↓	mAP	Rank-1	mAP	Rank-1
$\langle H20, S5\rangle$	CosFace	55.6	78.6	47.0	66.0
	LEAP-CF	**65.2**	**83.4**	**52.7**	**72.8**
	ArcFace	60.1	81.1	50.5	69.3
	LEAP-AF	**63.9**	**83.2**	**54.2**	**73.6**
$\langle H20, S4\rangle$	CosFace	43.1	67.7	36.0	53.7
	LEAP-CF	**54.7**	**76.8**	**42.6**	**63.0**
	ArcFace	49.4	73.8	39.7	58.8
	LEAP-AF	**56.5**	**77.9**	**44.2**	**64.4**
$\langle H20, S3\rangle$	CosFace	31.9	55.5	25.6	40.8
	LEAP-CF	**43.5**	**67.2**	**33.2**	**51.1**
	ArcFace	36.2	60.1	28.9	46.7
	LEAP-AF	**44.1**	**66.1**	**34.3**	**53.3**

Table 5. Impact analysis of different tail data for feature learning.

The impact of tail data. When the head class is reduced gradually and the tail data is increasing, the results are shown in Table 5, we observe the effect of tail data on performance. We gradually reduce the samples of each tail class, which results in insufficient training data, and the performance of the model drops dramatically. However, our method still makes a large margin improvement over the baseline. For example, in the $\langle H20, S3\rangle$ setting on Market-1501, even the number of samples for each tail class is only 3, the improvement of LEAP-CF reaches $+11.7\%$ (from

Test →		LFW	MegaFace	IJB-C(TPR@FPR)		
Train ↓	Method ↓	Rank-1	Rank-1	1e-3	1e-4	1e-5
$\langle H5K, T10K\rangle$	CosFace	98.73	81.41	83.35	73.32	63.42
	LEAP-CV	**98.88**	**81.78**	**83.83**	**73.96**	**64.64**
	ArcFace	98.60	81.08	82.30	72.45	62.46
	LEAP-AV	**98.67**	**81.69**	**83.16**	**72.97**	**63.22**
$\langle H5K, T20K\rangle$	CosFace	98.87	82.72	84.77	76.71	68.19
	LEAP-CV	**98.98**	**83.16**	**84.82**	**77.21**	**68.88**
	ArcFace	98.73	82.76	84.45	76.22	66.93
	LEAP-AV	**99.10**	**83.36**	**85.70**	**77.77**	**68.05**
$\langle H3K, T10K\rangle$	CosFace	97.65	72.27	79.08	68.06	56.52
	LEAP-CV	**97.97**	**73.19**	**79.60**	**69.18**	**58.89**
	ArcFace	97.82	72.45	78.24	66.99	55.31
	LEAP-AV	**98.07**	**73.43**	**78.84**	**67.82**	**55.75**
$\langle H3K, T20K\rangle$	CosFace	98.02	74.06	81.21	71.68	61.03
	LEAP-CV	**98.23**	**75.18**	**81.87**	**72.16**	**62.62**
	ArcFace	98.28	75.24	81.09	71.36	61.60
	LEAP-AV	**98.73**	**76.28**	**82.61**	**73.21**	**62.72**

Table 6. Face recognition results on LFW, MF1 and IJB-C are reported by varying the ratio between head and tail classes in training sets. H and T is the number of head class and tail class, respectively.

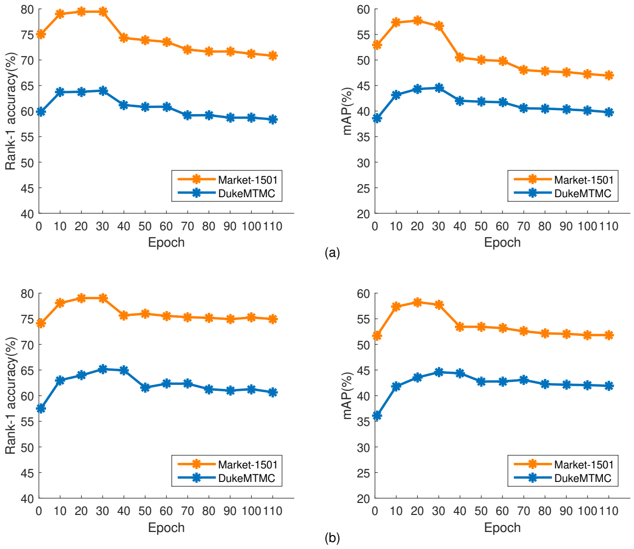


Figure 5. Different timings of constructing the feature cloud for tail data. (a) Combined our method with CosFace [35]. (b) Combined our method with ArcFace [5]

55.5% to 67.2%) in the Rank-1 accuracy.

Timing of feature cloud for tail data. We investigate the effect of timing of constructing a feature cloud for tail data on Market-1501 and DukeMTMC-reID dataset. We take a long-tailed version: $\langle H20, S4\rangle$ as an example. The varying curve of the results is shown in Fig. 5. (a) Combined with CosFace [35]. When epoch is in the range of 10 to 30, our results are just marginally impacted and the best results are achieved. (b) Combined with ArcFace [5]. Our results are impacted just marginally and the best results are achieved from 20-th to 30-th epoch.

4.3. Experiments on face recognition

To further verify the observations in the re-ID task, we perform a similar set of experiments on the face recognition task. Different from re-ID, the dataset of face recognition has a relatively large scale. In order to improve the training efficiency, we update the class center every 5 iteration. The result is shown in Table 6. On LFW, our performance is improved slightly since LFW has been well solved. MF1 and IJB-C are the most challenging testing benchmark for face recognition. We report the Rank-1 accuracy of MF1 and TPR@FPR of IJB-C. Compared with the baseline, our method obtains consistency improvement. For example, in the $\langle H3K, T10K\rangle$ setting, we evaluate our method on IJB-C, the LEAP-CV improves TPR@FPR(1e-5) from 56.52% to 58.89%. in the $\langle H3K, T20K\rangle$ setting, we evaluate our method on MF1, the LEAP-CV improves Rank-1 accuracy from 74.06% to 75.18%.

5. Conclusions

This paper proposes a novel approach for deep representation learning on long-tailed data. We observe that in the deeply-learned feature space, the tail classes are prone to lack of intra-class diversity, which consequentially distorts the overall distribution of feature space. In response, we enhance the diversity of tail class with augmentation embedded in deep feature space. The pattern of the augmentation is learned from the head classes (with abundant intra-class diversity) and transferred to tail classes in the manner of feature cloud. Experiments on person re-identification and face recognition demonstrate the effectiveness of our method on deep feature learning with long-tailed distribution.

References

[1] Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018. 1, 2

[2] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority oversampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002. 2

[3] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8789–8797, 2018. 2

[4] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9268–9277, 2019. 2

[5] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019. 3, 4, 5, 7, 8

[6] Chris Drummond, Robert 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*, volume 11, pages 1–8. Citeseer, 2003. 2

[7] Yang Du, Chunfeng Yuan, Bing Li, Lili Zhao, Yangxi Li, and Weiming Hu. Interaction-aware spatio-temporal pyramid attention networks for action classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 373–389, 2018. 6

[8] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 1

[9] Hang Gao, Zheng Shou, Alireza Zareian, Hanwang Zhang, and Shih-Fu Chang. Low-shot learning via covariance-preserving adversarial augmentation networks. In *Advances in Neural Information Processing Systems*, pages 975–985, 2018. 2

[10] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *European Conference on Computer Vision*, pages 87–102. Springer, 2016. 1

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[12] Ruibing Hou, Bingpeng Ma, Hong Chang, Xinqian Gu, Shiguang Shan, and Xilin Chen. Interaction-and-aggregation network for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9317–9326, 2019. 6

[13] Alston S Householder. Unitary triangularization of a nonsymmetric matrix. *Journal of the ACM (JACM)*, 5(4):339–342, 1958. 5

[14] Gary B Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. 2008. 6

[15] Houjing Huang, Dangwei Li, Zhang Zhang, Xiaotang Chen, and Kaiqi Huang. Adversarially occluded samples for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5098–5107, 2018. 6

[16] Mahdi M Kalayeh, Emrah Basaran, Muhittin Gökmen, Mustafa E Kamasak, and Mubarak Shah. Human semantic parsing for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1062–1071, 2018. 6

[17] Ira Kemelmacher-Shlizerman, Steven M Seitz, Daniel Miller, and Evan Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4873–4882, 2016. 6

[18] Buyu Li, Yu Liu, and Xiaogang Wang. Gradient harmonized single-stage detector. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8577–8584, 2019. 2

[19] Wei Li, Xiatian Zhu, and Shaogang Gong. Harmonious attention network for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2285–2294, 2018. 6

[20] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 212–220, 2017. 3

[21] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 181–196, 2018. 2

[22] Brianna Maze, Jocelyn Adams, James A Duncan, Nathan Kalka, Tim Miller, Charles Otto, Anil K Jain, W Tyler Niggel, Janet Anderson, Jordan Cheney, et al. Iarpa janus benchmark-c: Face dataset and protocol. In *2018 International Conference on Biometrics (ICB)*, pages 158–165. IEEE, 2018. 6

[23] Rajeev Ranjan, Carlos D Castillo, and Rama Chellappa. L2-constrained softmax loss for discriminative face verification. *arXiv preprint arXiv:1703.09507*, 2017. 3

[24] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European Conference on Computer Vision*, pages 17–35. Springer, 2016. 2, 6

[25] M Saquib Sarfraz, Arne Schumann, Andreas Eberle, and Rainer Stiefelhagen. A pose-sensitive embedding for person re-identification with expanded cross neighborhood re-

ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 420–429, 2018. 6

[26] Leslie N Smith. Cyclical learning rates for training neural networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 464–472. IEEE, 2017. 6

[27] Chi Su, Jianing Li, Shiliang Zhang, Junliang Xing, Wen Gao, and Qi Tian. Pose-driven deep convolutional model for person re-identification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3960–3969, 2017. 6

[28] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation from predicting 10,000 classes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1891–1898, 2014. 2

[29] Yifan Sun, Liang Zheng, Weijian Deng, and Shengjin Wang. Svdnet for pedestrian retrieval. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3800–3808, 2017. 6

[30] Yifan Sun, Liang Zheng, Yi Yang, Qi Tian, and Shengjin Wang. Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline). In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 480–496, 2018. 6

[31] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 6

[32] Laurens Van Der Maaten. Accelerating t-sne using tree-based algorithms. *The Journal of Machine Learning Research*, 15(1):3221–3245, 2014. 2

[33] Cheng Wang, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Mancs: A multi-task attentional network with curriculum sampling for person re-identification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 365–381, 2018. 6

[34] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: l 2 hypersphere embedding for face verification. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1041–1049. ACM, 2017. 3

[35] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5265–5274, 2018. 1, 3, 4, 5, 6, 7, 8

[36] Yicheng Wang, Zhenzhong Chen, Feng Wu, and Gang Wang. Person re-identification with cascaded pairwise convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1470–1478, 2018. 6

[37] Longhui Wei, Shiliang Zhang, Wen Gao, and Qi Tian. Person transfer gan to bridge domain gap for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 79–88, 2018. 6

[38] Longhui Wei, Shiliang Zhang, Hantao Yao, Wen Gao, and Qi Tian. Glad: Global-local-alignment descriptor for pedestrian retrieval. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 420–428. ACM, 2017. 6

[39] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*, pages 499–515. Springer, 2016. 3, 4

[40] Jing Xu, Rui Zhao, Feng Zhu, Huaming Wang, and Wanli Ouyang. Attention-aware compositional network for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2119–2128, 2018. 6

[41] Xi Yin, Xiang Yu, Kihyuk Sohn, Xiaoming Liu, and Manmohan Chandraker. Feature transfer learning for face recognition with under-represented data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5704–5713, 2019. 2, 4

[42] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *Proceedings of the IEEE international conference on computer vision*, pages 1116–1124, 2015. 6

[43] Zhedong Zheng, Xiaodong Yang, Zhiding Yu, Liang Zheng, Yi Yang, and Jan Kautz. Joint discriminative and generative learning for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2138–2147, 2019. 6

[44] Zhedong Zheng, Liang Zheng, and Yi Yang. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3754–3762, 2017. 2, 6

[45] Zhedong Zheng, Liang Zheng, and Yi Yang. A discriminatively learned cnn embedding for person reidentification. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 14(1):13, 2018. 6

[46] Yaoyao Zhong, Weihong Deng, Mei Wang, Jiani Hu, Jianteng Peng, Xunqiang Tao, and Yaohai Huang. Unequal-training for deep face recognition with long-tailed noisy data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7812–7821, 2019. 4

[47] Zhun Zhong, Liang Zheng, Zhedong Zheng, Shaozi Li, and Yi Yang. Camera style adaptation for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5157–5166, 2018. 6