Multimedia Tools and Applications (2019) 78:21145–21166
https://doi.org/10.1007/s11042-019-7446-2

Learning deep embedding with mini-cluster loss for person re-identification

Caihong Yuan[1,2] · Jingjuan Guo[1,3] · Ping Feng[1] · Zhiqiang Zhao[3] · Yihao Luo[1] · Chunyan Xu[4] · Tianjiang Wang[1] · Kui Duan[5]

Received: 20 July 2018 / Revised: 2 January 2019 / Accepted: 6 March 2019 /
Published online: 14 March 2019

Abstract
Recently, the triplet loss is commonly used in many deep person re-identification (ReID) frameworks to learn an embedding space in which similar data points are close and dissimilar data points are far away. However, the triplet loss simply focuses on the relative orders of points. This may lead to a relatively large intra-class variance and then a weak generalization capacity on the test set. In this paper, we propose a mini-cluster loss, which regards images belonging to the same identity as a mini-cluster and treats them as a whole during the training instead of considering them separately. For each mini-cluster in a batch, we define the largest distance between points in a mini-cluster as its inner divergence and the shortest distance with outer points as its outer divergence. By constraining the outer divergence larger than the inner divergence, our framework with the mini-cluster loss achieves the more compact mini-clusters while keeping the diversity distributions of the classes. As a result, a better generalization ability and a higher performance can be obtained. In the extensive experiments, our proposed framework achieves a state-of-the-art performance on two large-scale person ReID datasets (Market1501, DukeMTMC-reID) which clearly demonstrates its effectiveness. Specifically, 72.44% mAP and 87.05% rank-1 score are achieved on the Market1501 dataset with single query setting, 78.17% mAP and 91.05% rank-1 score with multiply query setting, and on the DukeMTMC-reID dataset, 60.19% mAP and 77.20% rank-1 score are obtained.

Keywords Person re-identification · Mini-cluster loss · The triplet loss · Deep feature embedding

1 Introduction

Person re-identification (ReID) focuses on the problem of searching for person images owning the same identity as the given anchor pedestrian image, which are captured from the

✉ Caihong Yuan
yuanch@hust.edu.cn

Extended author information available on the last page of the article.

surveillance videos across multiple cameras or across time [7, 27]. Due to its importance in the intelligent surveillance application, it has been attracting more and more attentions of the computer vision and pattern recognition committee. Despite intense studies [1, 4, 9, 10, 36, 46, 47] have been done in the recent years, ReID is still a very challenging task due to its particularity. Compared with many other tasks, such as face recognition [20, 41], there are more posture variances of the same individual and more complex backgrounds. Besides these, it also usually undergoes illumination variances and large occlusions.

Recently, deep learning gains a tremendous success in many areas, which can implicitly capture intricate distributions of large scale data by learning and representing data with multiple levels of abstraction and understand multi-modal information with multiple processing layers [28]. Motivated by this, more and more deep frameworks [2, 5, 9, 10, 22, 29, 34, 36, 46, 47] for person ReID have been proposed by researchers. The main goals of them are to learn a feature embedding where points from the same identity are close and those from different are far away. Recently, the triplet loss becomes a major means and is commonly used in many deep person ReID frameworks to keep the correct order for the anchor image with its positive and negative. In a batch there are several images for every identity which could be included in the triplets. During the training, the optimization is done on them separately. And after thousands of iterations, small intra-class variances and large inter-class variances are expected. However, as shown in Fig. 1, during an iteration, some points of the same identity are pushed to completely different directions. Thus, it will result in an unstable convergence process of the models. And since just the relative distances of a single point with its negatives and positives are constrained, it easily leads to a relatively large intra-class variance. And it is found in [4, 31] that large intra-class variances may lead to a weak generalization of the learned models on the test set. And they argue that reducing the intra-class variances and enlarging the inter-class variances may improve the performance. For this, ImpTrpLoss [5] introduces an additional item to the triplet loss by keeping the distances of positive pairs smaller than a pre-defined value. The quadruplet loss [4] adds the relative

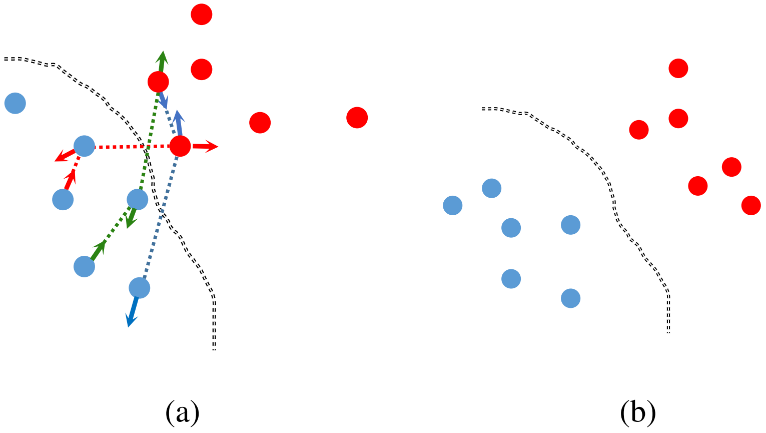


Fig. 1 Illustration of the optimization phase of the triplet loss with a toy example. **a** The movement of points in the triplets in an iteration. There are 2 mini-clusters shown here which are marked in blue and red. Three triplets are chose here which are connected with red, blue and green dotted line respectively. And the directions of the movement are marked by the solid lines with arrow. **b** The final formed clusters after optimizing. For each point here, the relative distances between its negatives and positives are well kept. However, there still exists large intra-class variances, which may lead to a weak generalization ability

constraint on the positive pairs and negative pairs without the same anchor to the triplet loss. However, both methods may destroy the diversity distributions of different classes.

Motivated by this, we propose a novel objective called mini-cluster loss defined on a blob. Images belonging to the same identity are defined as a mini-cluster and treated as a whole during the training. The proposed framework and the scheme of the mini-cluster loss are shown in Figs. 2 and 3 respectively. The input blob of the network contains P mini-clusters and each mini-cluster contains K images. For each mini-cluster in a batch, we define the largest distance between points in a mini-cluster as its inner divergence and the shortest distance with outer points as its outer divergence. By making the outer divergence larger than the inner divergence, the mini-cluster loss could keep the correct relative relationships of points and helps to form a feature embedding with smaller intra-class variances and larger inter-class variances. It can be found in our learned feature embedding which is shown in Fig. 4.

It should be mentioned that the mini-cluster loss shares similar spirits with hard samples mining methods, but the goals of them are different. For the triplet loss, hard triplets sampling is regarded as an effective measure for improving the performance. The goal of them is to mine the informative samples to enhance the gradients variance and speed up the convergence of the models while keeping the correct relative distances between the positive and negative pair with the same anchor. However for the mini-cluster loss, our searching for the inner divergence and the outer divergence is to represent the general characteristics of a cluster. And the aim is not to simply keep the relative relationship among a triplet but push the negative points far away from the anchor mini-cluster .

For explaining why the proposed method is effective, we conduct the experimental analysis in Section 4. Based on the triplet loss with batch hard [9], we explore the affects of the pairs of different hard levels in a mini-cluster for the performance. Through the analysis, we argue that the defined inner divergence and outer divergence are the appropriate representations for the general mini-cluster.

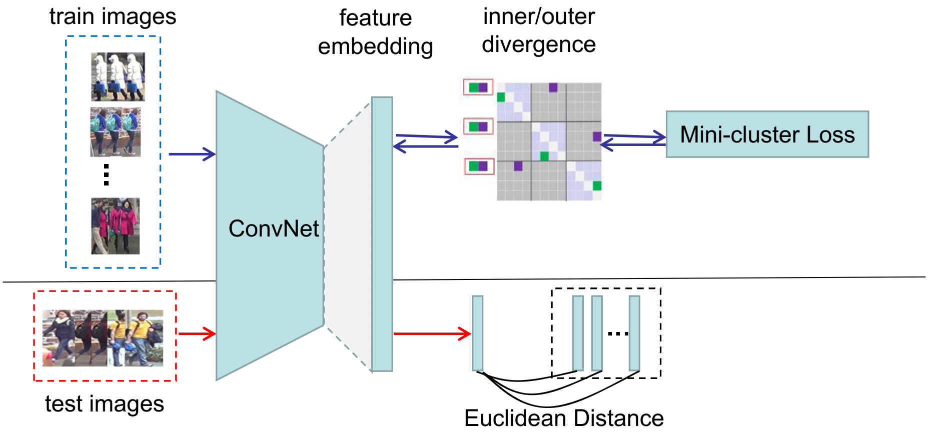


Fig. 2 Illustration of the proposed framework. The upper part is the course of training. The input blob is consisted of P identities and K images for each individual. After extracting the feature vectors, the inner divergence and outer divergence for each mini-cluster are first computed and then sent to the layer of the mini-cluster loss. And the bottom is the course of testing. Test images are first sent to the learned model and feature vectors of them are extracted. Then Euclidean distances between them are computed for performance evaluation

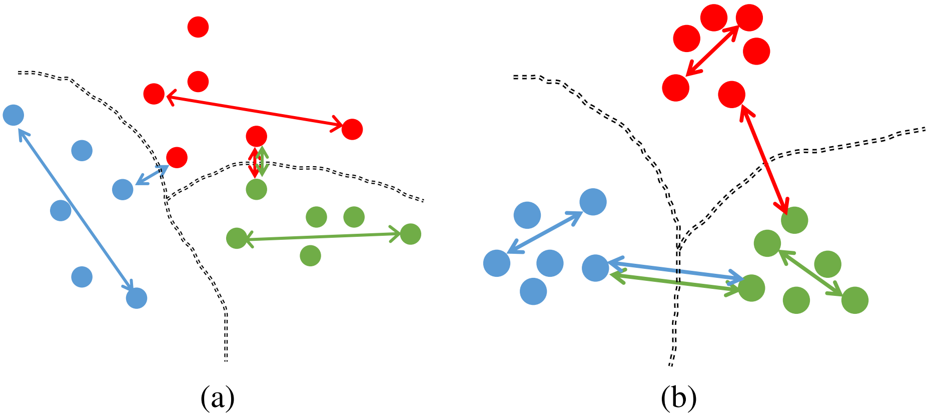


Fig. 3 The scheme of the proposed mini-cluster loss. A blob including three toy clusters colored in red, green and blue are shown here. We use the largest inner-class distance to represent the inner divergence of a mini-cluster, and the shortest inter-class distance with outer points to represent its outer divergence. **a** Before optimization. **b** After optimization

Generally, the main contributions of this paper can be listed as follows:

(1) We propose two concepts which are the inner divergence and outer divergence to represent the general characteristic of a mini-cluster.

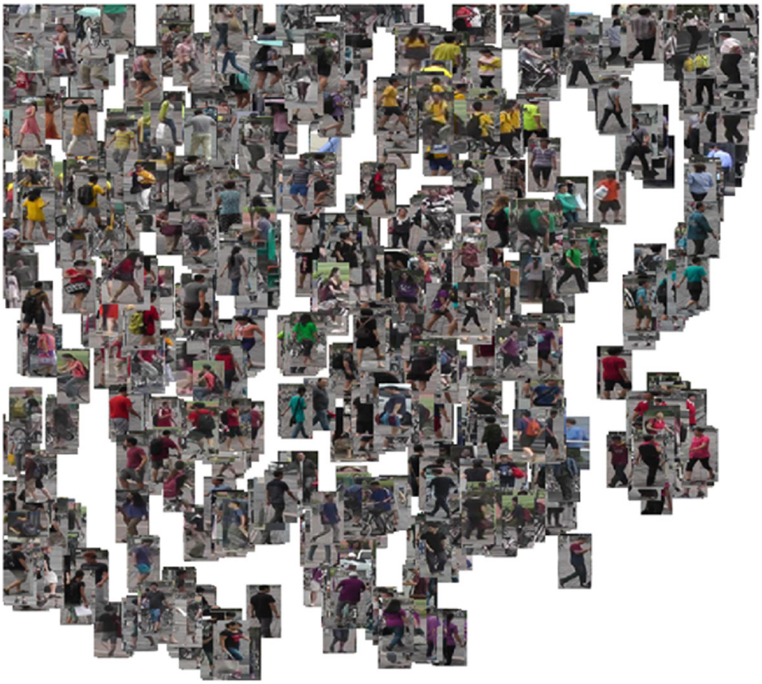


Fig. 4 A small part of the Barnes-Hut t-SNE [26] of the learned embedding for the Market1501 test set including the query images and the gallery images. Best view it when zooming in

(2) Based on them, we further propose a novel loss named mini-cluster loss, which regards images of the same identity in a batch as a whole. Just resorting to the especially selected positive (inner divergence) and negative pair (outer divergence), it could achieve more compact clusters while well keeping the diversities of different classes.

(3) Experiments on two large-scale person ReID datasets show its significant performance.

This paper is organized as follows: in Section 2 we present the related works of person ReID. Then in Section 3, the triplet loss and the quadruplet loss are first reviewed. And then we elaborate the proposed mini-cluster loss. Experimental valuations on 2 large scale datasets are demonstrated in Section 4. Moreover, the impacts of the pairs with different hard levels in a mini-cluster to the performance are experimental analyzed. And Section 5 is the conclusions.

2 Related work

There are two camps in person ReID which are traditional methods and deep methods. For those traditional methods, good feature representation [18, 24, 38, 42] and appropriate distance metric [11, 16, 21, 37] are two key subtasks [12, 13, 44]. However the solution of two separated steps is hard to achieve an optimal results [39]. With the revolutionary success of the deep learning method on many computer vision tasks, more and more deep frameworks [2, 5, 9, 10, 22, 29, 34, 36, 47] for person ReID have been proposed by the researchers. These frameworks can learn a feature embedding or a similarity metric in an end-to-end way.

There are 3 usual solutions for deep ReID: classification, verification and ranking. For these classification based person ReID frameworks [32, 34], they treat it as an identity recognition task by minimizing the softmax loss in the network. In the training, the model is learned by assigning an image to an identity. And in the testing, the top probability vector is abandoned and the previous outputted vector is extracted as the feature. Then a basic distance function (*i.e.* Euclidean distance and Cosine distance) is applied to measure the distances or similarities between images. However, since the goal of these methods is to learn boundaries between classes, it may lead to insufficient intra-class compaction [3]. And the objective of these verification based approaches [5, 39, 43] is to verify whether a pair of images are of the same person. They usually borrow the siamese network with the contrastive loss or binary classifier to reduce the intra-class variances and enlarge the inter-class variances. These methods simply focus on the absolute distance but neglect relative relationships. However, due to the complex scenarios of ReID, distributions of different classes are discrepant. Thus verification based methods may be not the best option for person ReID [33]. And for these ranking based approaches [6, 9, 33], the triplet loss is widely used to learn an embedding space in which the distance of the negative pair should be larger than the that of the positive pair in a triple. These methods could well keep the relative relationships between positive and negative pairs. However, frameworks based on triplet loss easily suffer from slow convergence and sub-optimization.

To solve this problem, many researches explore some hard triplets sampling methods or novel loss functions based on the triplet loss. For triplets sampling, there are two trends which are making full use of all the relationships as much as possible and just sampling the hard triplets in a blob. For the former, Batch All sampling [6] considers all the triplets among a blob. However, there are many easy triplets which will generate zero loss and are not effective for the gradient generation. Thus, this sampling method will reduce the gradients and easily lead to a suboptimal solution. Then the Lifted Structure loss [19] lifts the distances between vector pairs to a pairwise distance matrix and defines a structured loss based on all positive and negative samples. And N-pair loss [23] samples N classes and randomly selects a pair of images for every class. Then they propose a $N+1$ tuple loss by comparing a positive pair with all its negative pairs. The second trend is hard triplets sampling in a blob which is proved to be very efficient for improving the performance. Among them, Batch Hard [9] is an effective triplets mining method by selecting one hardest positive image and one hardest negative image for every image in a batch. Resorting to a soft margin triplet loss, it illustrates a state-of-the-art performance in the experiments. Besides these, Wu et al. [33] proposed a distance weighted sampling method and proved that sampling matters in the deep embedding learning.

At the same time, some researchers argue that the triplet loss just considers the relationships of positive pair and negative pair with the same anchor image may result in a large intra-class variance. They design some new loss functions by applying different constraints. Such as, ImpTrpLoss [5] introduces an additional item to the triplet loss by keeping the distances of positive pairs smaller than a pre-defined value. The quadruplet loss [4] is defined on a quadruplet including 4 images from 3 classes. Besides considering the relative distances between positive-negative pairs with the same anchor as the triplet loss does, an extra constraint for those without the same anchor is also added. The angular loss [30] is designed to introduce the scale invariance and a third-order geometric constraint by constraining the angle at the negative point of triplet triangles. The center loss [31] pushes the feature vectors to the center vector of the same class, and combined with the softmax loss, it effectively reduces the intra-class distance of the deep features.

Different with the triplet loss and its variants, our proposed mini-cluster loss considers images of the same identity in a batch as a whole and use the inner divergence and outer divergence to describe a mini-cluster. By limiting the outer divergence larger than the inner divergence, the relative relationships between the intra-class and inter-class are kept while more compact clusters are obtained.

3 The proposed method

Our proposed mini-cluster loss is closely related with the triplet loss and quadruplet loss. So, in this section, we introduce them firstly and then represent our mini-cluster loss in detail.

3.1 Preliminary

We define (x_i, s_i) as an input data, where $x_i \in \mathcal{X}$ is the image and s_i is the class label of x_i. The feature embedding kernel $f(\cdot;\theta) : \mathcal{X} \rightarrow \mathbb{R}^k$ maps an input image x_i to a feature vector of k dimensions. To simplify, we use $f(x_i)$ to represent $f(x_i;\theta)$.

The triplet loss The triplet loss [20] takes triplets as input, each including three images (x_i, x_j, x_k), where x_i and x_j are from the same person and x_k is from the different one. The goal of the triplet loss is to keep x_i closer to x_j than x_k. The triplet loss on a single triplet is defined as following:

$$\begin{aligned}\mathcal{L}_{triplet} = \tfrac{1}{2} \times \big[m + \big\| f(x_i) - f(x_j)\big\|_2^2 \\ - \| f(x_i) - f(x_k)\|_2^2\big]_+\end{aligned} \tag{1}$$

where $[\cdot] = max(\cdot, 0)$, and m is a predefined margin for the relative distance. If $\mathcal{L}_{triplet}(\cdot) = 0$, the partial differentials with respect to the input $f(x_i)$, $f(x_j)$ and $f(x_k)$ are all zeros. If not, they are as follows:

$$\begin{aligned} \frac{\partial(\mathcal{L}_{triplet}(\cdot))}{\partial(f(x_i))} &= f(x_k) - f(x_j) \\ \frac{\partial(\mathcal{L}_{triplet}(\cdot))}{\partial(f(x_j))} &= f(x_j) - f(x_i) \\ \frac{\partial(\mathcal{L}_{triplet}(\cdot))}{\partial(f(x_k))} &= f(x_i) - f(x_k) \end{aligned} \tag{2}$$

The quadruplet loss The quadruplet loss [4] is a variant of the triplet loss. It is based on a quadruplet (x_i, x_j, x_k, x_l), where x_i and x_j are images from the same person, but x_k and x_l are from other persons. Besides considering the relative distance between the positive and negative pair with the same probe image, the quadruplet loss also imports another constraint of the relationship between the positive and another negative pair with different probe person. The quadruplet loss is defined in (3), in which $g(\cdot, \cdot)$ represents the distance between two images, the first item is a main constraint and the second is a relatively weaker auxiliary constraint.

$$\begin{aligned} \mathcal{L}_{quad} &= \sum_{i,j,k}^{N} \left[g\big(f(x_i), f(x_j)\big)^2 - g\big(f(x_i), f(x_k)\big)^2 + \alpha_1\right]_+ \\ &+ \sum_{i,j,k,l}^{N} \left[g\big(f(x_i), f(x_j)\big)^2 - g\big(f(x_l), f(x_k)\big)^2 + \alpha_2\right]_+ \\ & s_i = s_j, s_l \neq s_k, s_i \neq s_l, s_i \neq s_k \end{aligned} \tag{3}$$

3.2 The mini-cluster loss

The scheme of the proposed mini-cluster loss is shown in Fig. 3. Suppose the input batch of our framework is consisted of P persons and K images for each person, which is called PK-batch in our paper. We consider K images of an individual as a mini-cluster. Before introducing the mini-cluster loss, we first define two concepts which are inner divergence and outer divergence to characterize a cluster. First, we define its inner divergence $\mathcal{I}dvg_a$ as the furthest distance between points of a mini-cluster :

$$\mathcal{I}dvg_a = \max_{x_i, x_j \in C_a} \left\| f(x_i) - f(x_j) \right\|_2 \tag{4}$$

in which, C_a is the ath mini-cluster, i.e., the set of the images with the same identity label l_a in a batch. x_i and x_j are different images from C_a. Moreover, for better representing the outer divergence, the distance between two mini-clusters is defined as the closest distance of points across two mini-clusters:

$$D_{a,b} = \min_{x_i \in C_a, x_j \in C_b} \left\| f(x_i) - f(x_j) \right\|_2 \tag{5}$$

Based on it, the outer divergence $\mathcal{O}dvg_a$ is defined as:

$$\mathcal{O}dvg_a = \min_{b \neq a, b \in 1 \cdots P} \{D_{a,b}\} \tag{6}$$

For each mini-cluster, there is an inner divergence representing its compact degree, and an outer divergence representing its separation degree with other mini-clusters. Intuitively, for a mini-cluster C_a, it should be satisfied with

$$\mathcal{I}dvg_a < \mathcal{O}dvg_a \tag{7}$$

So the proposed mini-cluster loss on a batch is defined as:

$$
\begin{aligned}
\mathcal{L}_{mini} &= \sum_{a=1}^{P} log\big(1+exp(\mathcal{I}dvg_a - \mathcal{O}dvg_a)\big) \\
&= \sum_{a=1}^{P} log\big(1+exp(\max_{x_i,x_j\in C_a} \big\|f(x_i)-f(x_j)\big\|_2 \\
&\quad - \min_{\substack{b=1,2\cdots P \\ b\neq a}} \{ \min_{\substack{x_k\in C_a \\ x_l\in C_b}} \|f(x_k)-f(x_l)\|_2\})\big)
\end{aligned} \tag{8}
$$

For each mini-cluster, a tuple (x_i, x_j, x_k, x_l) is carefully selected, where x_i, x_j and x_k are from the anchor mini-cluster, and x_l is from another. Thus, for a batch, P tuples are used to generate punishments for the general loss.

3.3 Runtime complexity

The computation of the proposed mini-cluster loss mainly includes three parts: distance matrix, the inner divergence and outer divergence, and the loss value. Assuming that there are M images in the training set and the batch size is $N = P \times K$, then an epoch should include $\frac{M}{N}$ batches. In each batch, the complexity of computing the distance matrix, divergences and the loss value could be expressed as $O((PK)^2)$, $O(P \times [K^2 + K \times (PK - K)])$ and $O(P)$ respectively. Thus, the complexity of an epoch should be $O(2MN + \frac{M}{K})$, which can be simply represented as $O(MN)$. As $N \rightarrow M$, we can get a worst cast complexity $O(M^2)$. Compared with the naive hard mining algorithm of the triplet loss, which has a worst case complexity of $O(M^3)$ on an epoch, our method is computationally more efficient. And, compared with the recent methods which are the batch hard sampling of the triplet loss [9] whose complexity is $O(2MN + M)$, and the quadruplet loss [4] whose complexity is $O(MN^3)$, the proposed method still has the advantage on the computational complexity.

3.4 Relationship with other metric objectives

Our mini-cluster loss is designed based on the triplet loss (1). And from the form of tuple, it is similar with the quadruplet loss (3). In this subsection, we discuss the relationship of our mini-cluster loss with them.

First, both the triplet loss and the quadruplet loss are not built on the whole mini-clusters but the individual points . They consider images of the same identity individually and keep the relative relationships among points. Thus, they apply some tuple mining method to generate tuples and several images of the same class may be contained. These selected tuples just represent themselves, not the whole mini-cluster. However, our proposed mini-cluster loss uses the inner divergence and the outer divergence to represent a mini-cluster, and ignores all other points here. Due to the particularity of the two concepts, just by limiting the outer divergence larger than the inner divergence, it would keep the correct relative relationship between intra-class and inter-class while pulling the clusters more compact. Of course, the mini-cluster loss can be regarded as a special hard tuple mining methods. In a PK-batch, for each mini-cluster a closest positive pair (x_i, x_j) and a furthest negative pair (x_k, x_l) are selected to form a tuple $(x_i, x_j, x_k, x_l)_{x_i,x_j,x_k\in C_a, x_l\in C_b}$. The mini-cluster loss is defined on such P tuples. Hard samples selecting has been proved important for the improvement of performance in many researches. The sampling method of mini-cluster loss discards most of tuples and just keeps one tuple for a mini-cluster. In the following experiments, we illustrate that our models trained on the P selected tuples can show a better performance than other hard sampling methods.

Second, from the form, the tuple $(x_i, x_j, x_k, x_l)_{x_i, x_j, x_k \in C_a, x_l \in C_b}$ our mini-cluster loss defined on is a quadruplet. But it is different with that in the quadruplet loss (3). Images in the tuple of our mini-cluster are just from two clusters. But the quadruplet $(x_i, x_j, x_k, x_l)_{x_i, x_j \in C_a, x_k \in C_b, x_l \in C_c}$ in the quadruplet loss is from three clusters. The quadruplet loss not only considers the relative distance between the positive pair and the negative pair with the same anchor as the triplet loss does, but also the relative distance between the positive pair and negative pair from other clusters. But the extra constraint item may destroy the diverse of the distributions of the clusters. However, the mini-cluster loss keeps the relative relationship of the inner divergence and the outer divergence of a mini-cluster. Due to the outer divergence is a strict constrain, it helps to pull the mini-cluster more compact. And the diverse of the different clusters can also be kept.

4 Experiments

4.1 Setup

Datasets: We evaluate our method on two large-scale ReID benchmark datasets which are Market1501 [45] and DukeMTMC-reID [47]. The Market1501 dataset is captured in front of a supermarket from 6 different camera views including 5 high-resolution cameras and one low-resolution camera. It is detected by Deformable Part Model (DPM). There are 12,936 images of 751 IDs in the training set, 3368 query images and 19,732 gallery images of the remaining 750 IDs in the testing set. The DukeMTMC-reID dataset is cropped from 85-minute high-resolution videos from 8 different cameras by handcraft. It contains 1404 IDs in total where the randomly selected 702 IDs are as the training set and the remaining 702 IDs as the testing set. As a result, there are 16,522 images in the training set, 2228 query images and 17,661 gallery images (702 IDs + 408 distractor IDs) in the testing set.

Data augmentation: In various of deep learning tasks, data augmentation has been proved to be a very effective means for alleviating the over-fitting problem. In the following implementation, all the images are first resized into 288 × 144 pixels. Then during the training, we crop a 256 × 128 pixels image region with a random perturbation and mirror it or not decided in the manner of flipping a coin. Then the cropped images subtract the mean values (123.68, 116.78, 103.94) for Market1501 and (127.50, 127.50, 127.50) for DukeMTMC-reID in the corresponding dimensions. During the test, five crops in the upper left, upper right, lower left, lower right and center and their mirrors are used to extract their feature vectors. Then the average feature vectors are applied to compute the distance matrix and evaluate the performance.

Optimization setting: We use ResNet50 [8] as the CNN network. First we discard the last fully connected layer and the softmax loss layer and add a fully connected layer with 128 dimensions and the mini-cluster loss layer. Then we fine-tune the model on the pre-trained model of ResNet50. Our experiments are conducted in the Tensorflow deep learning framework. We adopt adaptive moment estimation optimizer (Adam) to optimize our deep framework. The exponentially rate decaying schedule is applied in the training. 25,000 iterations are conducted on both datasets. The initial learning rate is set to 3e-4 for the Market1501 dataset and 1e-4 for the DukeMTMC-reID dataset, and then gradually decays after 10000 iterations till 0.01 folds of the initial learning rate.

4.2 Construction of the input blob

Since the mini-cluster loss considers a cluster as a whole, the appropriate form of the input blob in the optimizing is PK-blob which randomly samples P identities and K images for each identity. Although it is expected to set a large value for P and K, it is limited by the capacity of GPU. So the scale of the blob, i.e. P and K should be carefully selected. In our experiments, we fix the size of a blob ($P \times K$) to be 108. By set K to be 18, 12, 9, 6, 3 and 2 respectively, we investigate the influence to the results of K and explore the balance between K and P. We conduct them on the Market1501 dataset, and the results are illustrated in Table 1. From it, it can be seen that the best results are achieved when K is set to be 6 and P is 18. We argue the best balance is achieved at this setting when the batch size is set to be 108.

Moreover, it is noticed that a large performance decreasing can be found when P is larger or smaller, especially when $K = 18$ and $K = 2$. When K is set to be a larger value, P will be smaller. Then it may cause large distractions introduced by the large mini-cluster and the insufficient negative images. For example, when setting K to be 18, almost all the images of the same identity could be involved in a blob. For those classes without outliers, the inner divergence can well represent the mini-cluster. However, if the outliers are existed, it may draw the whole cluster to fit the outlier and result in an indistinct feature embedding. On the other hand, the small negative classes would lead to a weak outer divergence and then a zero loss in the later period of the optimization. It could generate an under-fitting model. On the contrary, the smaller K easily leads to a local inner divergence and then a suboptimal performance. So, in the following experiments, we set K to be 6 and P to be 18.

4.3 Progressive training strategy

Based on the keep observation on many training processes, it is found that it is easily collapsed in the early period of the optimization for models with the mini-cluster loss no matter how small of the learning rate. We analyze it is because that in the early period of the optimization, both the inner divergence and the outer divergence make no sense. Only when the feature embedding is elementary ordered, the mini-cluster loss would tap its potential in the training. So we apply a strategy of progressive training which is 2-phase training. In the first phase, we train the model by the triplet loss with the batch hard sampling defined in [9]. And then in the second phase, fine-tune the pre-trained model in the first phase. In

Table 1 Balance between K and P in a blob

Batch construction	mAP	rank-1	rank-2
$K = 18$ / $P = 6$	65.23	81.77	87.23
$K = 12$ / $P = 9$	71.17	85.75	90.29
$K = 9$ / $P = 12$	72.06	85.72	90.65
$K = 6$ / $P = 18$	**72.44**	**87.05**	**90.97**
$K = 3$ / $P = 36$	70.32	85.78	89.96
$K = 2$ / $P = 54$	69.60	85.33	90.14

These experiments are conducted on the Market1501 dataset. The size of blob is fixed as 108. By setting K to be a defined value, P is got by computing 108 divided by K. The best results are shown in bold our experiments, the general iteration number is set to 25,000 where the first 1000 iterations are trained by the method of [9] and the remaining 24,000 iterations are trained by the proposed mini-cluster loss.

4.4 Comparison with state-of-the-art methods

In Table 2, we compare our proposed method with several recent state-of-the-art works on both Market1501 and DukeMTMT-reID dataset. There are traditional methods and deep learning methods compared. The first two methods are traditional methods. Among them, **DNS** [40] designs a closed-form solution to make the images of the same person collapse into a single point while maximizing the relative between-class separation for solving the small sample size problem. And **SSM** [1] is a manifold-based affinity learning method for ReID. Methods of the next part of the Table 2 are deep ReID frameworks. For instance, **Resnet+OIM** [35] conducts the pedestrian detection and the identification task jointly by optimizing the online instance matching loss. **Latent Parts** [14] learns the localize deformable pedestrian parts first and then learns the context-aware features by integrating the learning of the full body and local parts in a united framework. Moreover, **AACN** [36] introduces pose and attention information to the deep framework. Besides these, another 2 methods aggregate the training samples using generative adversarial network (GAN), which are **Verif-Identif.+LSRO** [47] and **dMpRL**. Especially, **Triplet+Batch hard** [9] uses batch hard sampling method and soft margin triplet loss to learn the embedding which is most related with ours.

Table 2 Comparisons of our method with several state-of-the-art methods on both Market1501 and DukeMTMC-reID

Method	Market1501 SQ		Market1501 MQ		DukeMTMC-reID SQ	
	mAP	rank-1	mAP	rank-1	mAP	rank-1
DNS [40]	35.68	61.02	46.03	71.56	–	–
SSM [1]	68.80	82.21	76.18	88.18	–	–
Resnet+OIM [35]	–	82.10	–	–	–	68.10
Latent Parts [14]	57.53	80.31	66.70	86.79	–	–
P2S [48]	44.27	70.72	55.73	85.78	–	–
Multi-scale [17]	–	45.10	–	55.40	–	–
Embedding [46]	59.87	79.51	70.33	85.84	–	–
JLML [15]	65.50	85.10	74.50	89.70	–	–
SVDNet [25]	62.10	82.30	–	–	56.80	76.70
Verif-Identif.+LSRO [47]	66.07	83.97	76.10	88.42	47.13	67.68
AACN [36]	66.87	85.90	75.10	89.78	59.25	76.84
dMpRL [10]	67.53	85.75	77.85	89.88	58.56	76.81
Triplet+Batch hard [9]	69.14	84.92	76.42	90.53	–	–
Mini-cluster(ours)	**72.44**	**87.05**	**78.17**	**91.15**	**60.19**	**77.20**

For experiments on Market1501, we conduct testing of single query(SQ) and multiple query(MQ). And on DukeMTMC-reID, just testing of single query is conducted. The mAP and the match accuracy at rank-1 are listed. We emphasize the best results of comparisons with the underline in every part and show our results in bold

As it can be seen that our proposed method shows better performances compared with all the above methods on both large-scale datasets. Especially, our method improves the mAP score from 69.14% [9] to 72.44% and improves the rank-1 accuracy from 85.90% [36] to 87.05% on the Market1501 dataset at the single query mode. And at the multiple query model, there is a slight improvement compared with the best results. On the DukeMTMC-reID dataset, there is nearly a 1% performance improvement compared with 59.25% [36] at mAP. It is noted that compared with [9] which is closely with our method, there is about average 2.50% improvement of the mAP score on the Market1501.

It is mentioned above that our method is motivated by or close with some deep learning methods. However, no available published results achieved from them on the 2 large-scale datasets can be found, or it was conducted under different experimental settings. So, for better and fair comparisons, we conduct 5 groups of the related experiments under the same setting as ours, which are listed in Table 3. Concretely, we re-implement the triplet loss with random sampling, batch all sampling and batch hard sampling, the quadruplet loss and the lifted structure loss. All the experiments are trained on the pre-trained ResNet50 model, and the forms of the input blobs are PK-blob where P is set to be 18 and K is be 6. For the triplet loss, we use the form of soft-margin as is done with [9] and ours. And for the quadruplet loss [4] (3), we set the two margins to be 1.0 and 0.5 respectively. Among these 5 methods, the Triplet+Batch hard(18 × 6) achieves the best results. And compared with the results published in [9] which are listed in Table 2, there is more than 2% improvement of the mAP score under the single query mode. And the single difference is applying a different blob construction with [9], which again illustrates that the balance between P and K is important for the performance improving.

Moreover, compared with these 5 methods, our method shows a higher performance. Especially on the DukeMTMC-reID dataset, it improves 55.75% mAP achieved by the batch hard method to 60.19% and 73.61% rank-1 accuracy to 77.20%. The better performance implies that the mini-cluster loss is an effective method for person ReID. However, it should be clarified that the delightful performances of the mini-cluster loss are achieved just on the large-scale person ReID datasets, we are not sure whether it is effective for the other applications which should be tested on.

Table 3 Comparisons with some methods closely related with our method on the Market1501 and DukeMTMC-reID dataset

Method	Market1501 SQ		Market1501 MQ		DukeMTMC-reID SQ	
	mAP	rank-1	mAP	rank-1	mAP	rank-1
Triplet+Random sampling	52.75	70.46	58.45	77.75	50.55	68.90
Triplet+Batch all [6]	68.26	83.82	74.61	89.10	53.58	73.11
Triplet+Batch hard(18 × 6) [9]	71.62	86.22	77.59	91.00	55.75	73.61
Quadruplet [4]	68.64	82.96	74.90	88.33	55.35	73.11
Lifted structure [19]	70.31	85.63	76.23	90.38	54.80	72.89
Mini-cluster(ours)	**72.44**	**87.05**	**78.17**	**91.15**	**60.19**	**77.20**

For experiments on Market1501, we conduct testing of single query(SQ) and multiple query(MQ). And on DukeMTMC-reID, just testing of single query is conducted. The mAP and the match accuracy at rank-1 are listed. We emphasize the best results of comparisons with the underline and show our results in bold. Note that we re-implement all these methods and the results are achieved under the same setting

For further qualitatively investigating the performance of our method, we illustrate a small part of the Barnes-Hut t-SNE [26] of the learned embedding for the Market1501 test set including the query images and the gallery images in Fig. 4, and show some ranked gallery image lists generated by our method in Fig. 5. From Fig. 4, it can be seen that the meaningful feature embedding is learned by our method. Especially, from Fig. 5, we can clearly observe that these top ranked images are all similar with the query image in the appearance. For those mismatched images, we also find they are dressed in the similar clothing or take the similar backpack. It should be noticed that the color and the global feature can be well abstracted by our method, but the local features about the detail information of an image are not well captured.

4.5 Discussion

In Tables 2 and 3, it can be seen that the batch hard triplets mining method [9] shows a better performance compared with others. So, based on the batch hard method proposed by [9], we conduct experiments on the Market1501 dataset for exploring the informative ability of the positive pairs and negative pairs with different hard levels and explain why the inner divergence and outer divergence are an appropriate representation for a mini-cluster.

For better understanding the setting of the experiments, we show a toy example in Fig. 6. With the batch hard method shown in Fig. 6b, a hardest positive and a hardest negative are selected for each image. And there will generate K triplets for each mini-cluster and PK triplets for the whole blob. For better conducting the analysis, we define two types of hard levels for the triplets which are hard levels on the positive pairs and those on the negative

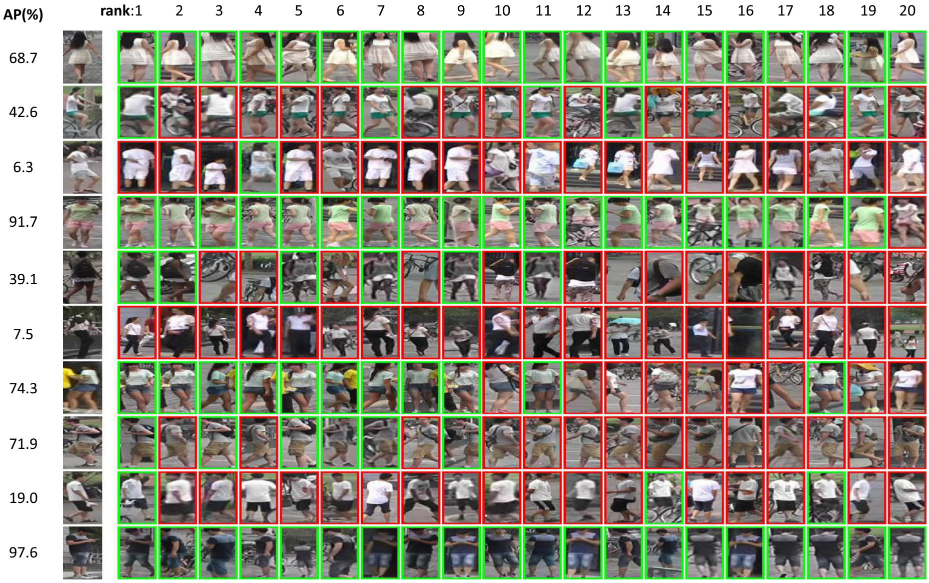


Fig. 5 Ranked gallery image lists generated by our method for some query images chosen one every 50 from the Market1501 query subset of the test set. There are 10 gallery image lists shown. And in every row, the beginning is the AP(Average Precision) score for the query which is the first image and the following right part is the ranked gallery image list. Due to space constraints, only the first 20 ranked gallery images are chosen here. The matched images are in the green border and the mismatched ones are in the red border

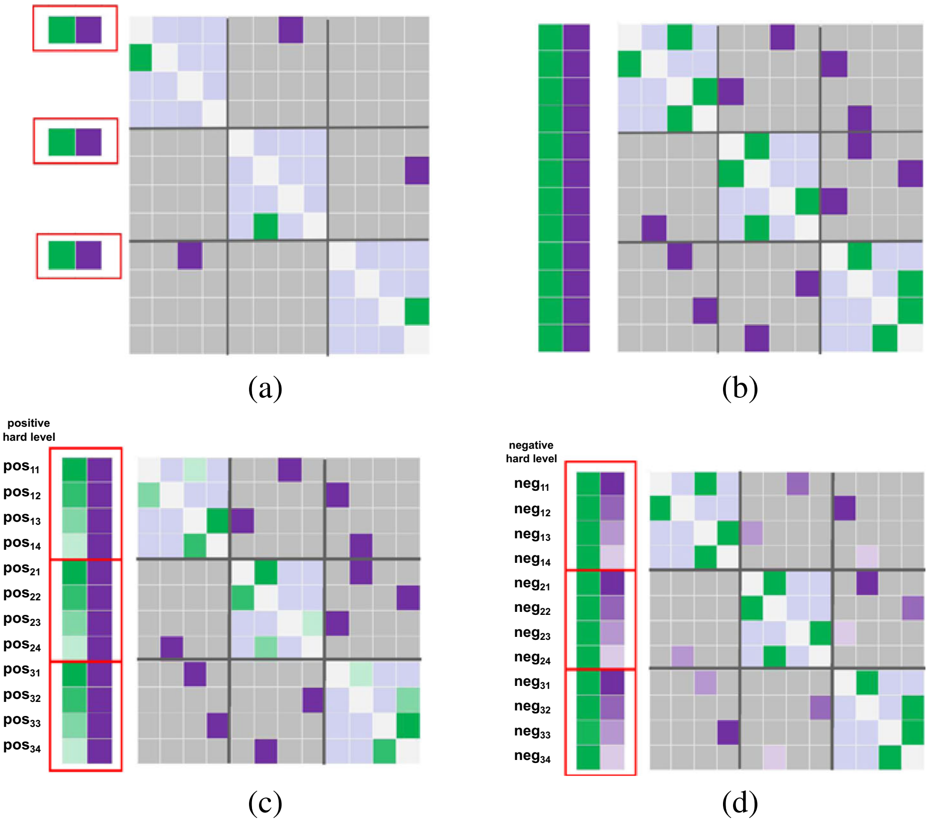


Fig. 6 Toy examples for understanding the impact of pairs with different hard levels to the ReID performance. **a** Our proposed mini-cluster loss. **b** Batch hard triplets sampling. **c** The sorted triplets of (**b**) according to the hard level of the positive pair in a triplet among each mini-cluster. **d** The sorted triplets of (**b**) according to the hard level of the negative pair in a triplet among each mini-cluster. For each sub-figure, there are two parts are included. The right part is a distance matrix, and the left is the selected positive and negative pairs in a batch. For (**b**)–(**d**), the hardest positive and hardest negative are selected for every anchor image in a batch. That is, the selected positive pair and the negative pair own the same anchor image in (**b**)–(**d**). However for (**a**), only 1 hardest positive pair and 1 hardest negative pair are chose in a mini-cluster and they may have the same anchor image or not. The selected positives are shown in green, and the selected negatives are shown in purple. And in (**c**) and (**d**), we mark the selected pairs by different shades of green or purple. The darker the color, the harder the pair is

pairs in Fig. 6c–d. Based on this, we conduct 4 groups of experiments on 4 triplet sets of different hard level, which are *pos-hard-m*, *neg-hard-m*, *pos-easy-m*, *neg-easy-m*. By setting an increasing m, we analyze the impact of the added triplets to the performance.

Specifically, for the hard levels on the positive pairs (Fig. 6c), we first sort the K triplets in each mini-cluster in the descending order according to the distance of the positive pairs. Then we sample the first $m(m = 1, 2, \cdots, K)$ triplets from every mini-cluster and the selected $m \times P$ triplets are sent to the triplet loss layer. We represent this group of experiments as *pos-hard-m*$= \{pos_{ij} | i = 1, \cdots, m, j = 1, \cdots, P\}$. And if we sample the last $m(m = 1, \cdots, K)$ triplets from every mini-cluster, this group of experiment is represented as *pos-easy-m*$= \{pos_{ij} | i = K, \cdots, K - m + 1, j = 1, \cdots, P\}$. Similar with those, we first sort K triplets in each mini-cluster in the ascending order according to the distances of

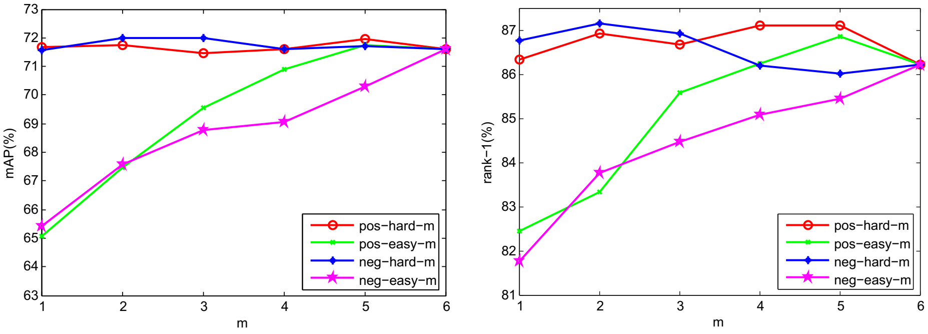
Table 4 Performance of the selected triplets from different hard levels on the Market1501 dataset

m	*pos-hard-m*			*pos-easy-m*			*neg-hard-m*			*neg-easy-m*		
	mAP	rank-1	rank-2	mAP	rank-1	rank-2	mAP	rank-1	rank-2	mAP	rank-1	rank-2
1	71.69	86.34	90.91	65.05	82.45	87.98	71.58	86.76	90.88	65.42	81.77	87.68
2	71.75	86.93	90.97	67.47	83.34	88.33	71.98	87.14	90.88	67.56	83.76	88.90
3	71.47	86.67	90.47	69.56	85.57	90.08	71.99	86.91	90.41	68.77	84.47	89.52
4	71.59	87.11	91.21	70.88	86.25	90.88	71.61	86.19	90.97	69.07	85.07	90.38
5	71.94	87.11	91.12	71.74	86.85	91.03	71.07	86.02	90.38	70.28	85.45	90.32
6	71.62	86.22	90.91	71.62	86.22	90.91	71.62	86.22	90.91	71.62	86.22	90.91

For each group of experiments, the mAP and the match scores at rank-1 and rank-2 under the single query are listed

the negative pairs (Fig. 6d), and then sample the first or last m triplets. These two groups of experiments are represented as *neg-hard-m*= $\{neg_{ij} | i = 1 \cdots m, j = 1, \cdots, K\}$ and *neg-easy-m*= $\{neg_{ij} | i = K, \cdots, K - m + 1, j = 1, \cdots, P\}$. Since K is 6 and P is 18 in our experiments, we also apply this setting in these 4 groups of experiments.

The results of the 4 groups of experiments are listed in Table 4 and shown in Fig. 7. As we have known, for *pos-hard-m*, we gradually add a triplet with positive pair of next hard level to the training set for a mini-cluster with m increasing 1. From Table 4, it can be seen that even if m is 1, that is, just 1 triplet is selected in a mini-cluster, we also achieve very competitive results. And with m increasing, there is no evident performance improvement. Similar character can be found in the columns of *neg-hard-m*. It illustrates that the hardest positive and negative pair in a mini-cluster play very important roles for the optimization and those sub-hard positives and negatives may not bring a large performance improvement. However, the performances in the columns of *pos-easy-m* and *neg-easy-m* are almost progressively improved with m increasing. This means it could bring significant performance improvements by gradually introducing the harder positives or harder negatives of a mini-cluster to the training samples. Based on this, we speculate that both the hardest positive pair and the hardest negative pair in a mini-cluster are very important for the model training. This finding lays the foundation of our mini-cluster as shown in Fig. 6a. As we have



(a) Change of mAP

(b) Change of rank-1

Fig. 7 The performance change with m increasing

introduced above, we define the distance of the hardest positive pair and the hardest negative pair as the inner divergence and outer divergence respectively. And the inner divergence and outer divergence can be used to characterize a cluster. If a cluster has a small inner divergence and a large outer divergence, that means a small intra-class variance and a large inter-class variance. And keeping the outer divergence larger than the inner divergence in the training, it would help to generate an embedding with more compact clusters and more separation between clusters.

5 Conclusion

In this work, we proposed a novel loss objective called mini-cluster loss for the person ReID task. Instead of considering samples in a blob separately, the proposed loss regards images of the same person in a blob as a mini-cluster and treats them as a whole in the training. For each mini-cluster, we define 2 concepts which are the inner divergence and the outer divergence to characterize a mini-cluster. By keeping the outer divergence larger than the inner divergence, the mini-cluster loss could learn a feature embedding with small intra-class variances and large inter-class variances. Moreover, we apply a progressive strategy to train our models. Extensive experimental results on both large-scale datasets (Market1501 and DukeMTMC-reID) clearly demonstrate the effectiveness of the proposed mini-cluster loss for person ReID. However, although our method can well capture the color and the global feature, the local features about the detail information of an image are not well obtained. We think that our method combined with the local feature extracted by using attribute information can largely improve the ReID performance. Meanwhile, we also notice that if there are just a small count of images for an identity in a ReID dataset, the proposed mini-cluster loss will not well reach its potential performance. Future work should be aimed at breaking through this limit by generating more images of an identity by using GANs(Generative Adversarial Networks).

Acknowledgment This work was supported by the National Natural Science Foundation of China (61572214, 61602244, U1536203 and U1504611), and partially sponsored by CCF-Tencent Open Research Fund.

References

1. Bai S, Bai X, Qi T (2017) Scalable person re-identification on supervised smoothed manifold. In: CVPR, pp 2530–2539
2. Barbosa IB, Cristani M, Caputo B, Rognhaugen A, Theoharis T (2017) Looking beyond appearances: synthetic training data for deep cnns in re-identification. Comput Vis Image Underst, 1–14
3. Chen Y, Chen Y, Wang X, Tang X (2014) Deep learning face representation by joint identification-verification. In: NIPS, pp 1988–1996
4. Chen W, Chen X, Zhang J, Huang K (2017) Beyond triplet loss: a deep quadruplet network for person re-identification. In: CVPR, pp 403–412
5. De C, Gong Y, Zhou S, Wang J, Zheng N (2016) Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In: CVPR, pp 1335–1344
6. Ding S, Lin L, Wang G, Chao H (2015) Deep feature learning with relative distance comparison for person re-identification. Pattern Recogn 48(10):2993–3003
7. Gong S, Cristani M, Yan S, Chen CL (2014) Person re-identification. Springer
8. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: CVPR, pp 770–778

9. Hermans A, Beyer L, Leibe B (2017) In defense of the triplet loss for person re-identification. arXiv:1703.07737, 1–10
10. Huang Y, Xu J, Wu Q, Zheng Z, Zhang Z, Zhang J (2018) Multi-pseudo regularized label for generated samples in person re-identification. arXiv:1801.06742, 1–12
11. Li W, Wang X (2013) Locally aligned feature transforms across views. In: CVPR, pp 3594–3601
12. Li W, Zhao R, Wang X (2012) Human reidentification with transferred metric learning. In: ACCV, pp 31–44
13. Li Z, Chang S, Liang F, Huang TS, Cao L, Smith JR (2013) Learning locally-adaptive decision functions for person verification. In: CVPR, pp 3610–3617
14. Li D, Chen X, Zhang Z, Huang K (2017) Learning deep context-aware features over body and latent parts for person re-identification. In: CVPR, pp 384–393
15. Li W, Zhu X, Gong S (2017) Person re-identification by deep joint learning of multi-loss classification. IJCAI, 2194–2200
16. Liao S, Li SZ (2015) Efficient psd constrained asymmetric metric learning for person re-identification. In: ICCV, pp 3685–3693
17. Liu J, Zha ZJ, Qi T, Liu D, Yao T, Ling Q, Mei T (2016) Multi-scale triplet cnn for person re-identification. In: ACM on multimedia conference, pp 192–196
18. Matsukawa T, Okabe T, Suzuki E, Sato Y (2016) Hierarchical gaussian descriptor for person re-identification. In: CVPR, pp 1363–1372
19. Song HO, Yu X, Jegelka S, Savarese S (2016) Deep metric learning via lifted structured feature embedding. In: CVPR, pp 4004–4012
20. Schroff F, Kalenichenko D, Philbin J (2015) Facenet: a unified embedding for face recognition and clustering. In: CVPR, pp 815–823
21. Shen Y, Lin W, Yan J, Xu M, Wu J, Wang J (2015) Person re-identification with correspondence structure learning. In: ICCV, pp 3200–3208
22. Shi H, Yang Y, Zhu X, Liao S, Lei Z, Zheng W, Li SZ (2016) Embedding deep metric for person re-identification: a study against large variations. In: ECCV, pp 732–748
23. Sohn K (2016) Improved deep metric learning with multi-class n-pair loss objective. In: NIPS, pp 1857–1865
24. Su C, Yang F, Zhang S, Qi T, Davis LS, Gao W (2015) Multi-task learning with low rank attribute embedding for person re-identification. In: CVPR, pp 3739–3747
25. Sun Y, Zheng L, Deng W, Wang S (2017) Svdnet for pedestrian retrieval. ICCV, 3800–3808
26. Van Der Maaten L (2014) Accelerating t-sne using tree-based algorithms, vol 15
27. Vezzani R, Baltieri D, Cucchiara R (2013) People reidentification in surveillance and forensics:a survey. Acm Comput Surv 46(2):1–37
28. Voulodimos A, Doulamis N, Doulamis A, Protopapadakis E (2018) Deep learning for computer vision: a brief review. In: Comput Intell Neurosci, pp 1–13
29. Wang F, Zuo W, Lin L, Zhang D, Zhang L (2016) Joint learning of single-image and cross-image representations for person re-identification. In: CVPR, pp 1288–1296
30. Wang J, Zhou F, Wen S, Liu X, Lin Y (2017) Deep metric learning with angular loss. In: ICCV, pp 2593–2601
31. Wen Y, Zhang K, Li Z, Yu Q (2016) A discriminative feature learning approach for deep face recognition. In: ECCV, pp 499–515
32. Wu S, Chen YC, Li X, Wu AC, You JJ, Zheng WS (2016) An enhanced deep feature representation for person re-identification. In: WACV, pp 1–8
33. Wu C-Y, Manmatha R, Smola AJ, Krahenbuhl P (2017) Sampling matters in deep embedding learning. In: CVPR, pp 2840–2848
34. Xiao T, Li H, Ouyang W, Wang X (2016) Learning deep feature representations with domain guided dropout for person re-identification. In: CVPR, pp 1249–1258
35. Xiao T, Li S, Wang B, Lin L, Wang X (2017) Joint detection and identification feature learning for person search. In: CVPR, pp 3376–3385
36. Xu J, Zhao R, Zhu F, Wang H, Ouyang W (2018) Attention-aware compositional network for person re-identification. CVPR, 2119–2128
37. Yang Y, Lei Z, Zhang S, Shi H, Li SZ (2016) Metric embedded discriminative vocabulary learning for high-level person representation. In: AAAI, pp 3648–3654
38. Yang Y, Wen L, Lyu S, Li SZ (2017) Unsupervised learning of multi-level descriptors for person re-identification. In: AAAI, vol 1, pp 4306–4312
39. Yi D, Lei Z, Li SZ (2014) Deep metric learning for practical person re-identification. Comput Sci, 34–39
40. Zhang L, Xiang T, Gong S (2016) Learning a discriminative null space for person re-identification. In: CVPR, pp 1239–1248

41. Zhang X, Fang Z, Wen Y, Li Z, Yu Q (2017) Range loss for deep face recognition with long-tail. ICCV, 1–10
42. Zhao R, Ouyang W, Wang X (2014) Learning mid-level filters for person re-identification. In: CVPR, pp 144–151
43. Zheng WS, Gong S, Xiang T (2011) Person re-identification by probabilistic relative distance comparison. In: CVPR, pp 649–656
44. Zheng W, Gong S, Xiang T (2013) Reidentification by relative distance comparison. IEEE Trans Pattern Anal Mach Intell 35(3):653–668
45. Zheng L, Shen L, Lu T, Wang S, Wang J, Qi T (2015) Scalable person re-identification: a benchmark. In: ICCV, pp 1116–1124
46. Zheng Z, Zheng L, Yang Y (2017) A discriminatively learned cnn embedding for person re-identification. arXiv:1611.05666, 1–10
47. Zheng Z, Zheng L, Yi Y (2017) Unlabeled samples generated by gan improve the person re-identification baseline in vitro. ICCV, 3754–3762
48. Zhou S, Wang J, Wang J, Gong Y, Zheng N (2017) Point to set similarity based deep feature learning for person reidentification. In: CVPR, pp 3741–3750

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Caihong Yuan received the B.Sc. degree in computer science and technology from Henan University, Kaifeng, China, in 2003. She is currently working toward the Ph.D. degree in the School of Computer Science and Technology, Huazhong University of Science and Technology(HUST), Wuhan, China. She is currently a teacher in Henan University. Her research interests include visual tracking and computer vision.

Jingjuan Guo received the M.Sc. degree in computer science and technology in 2006 from Jiangxi Normal University, Nanchang, China. He is currently working toward the Ph.D. degree in the School of Computer Science and Technology, Huazhong University of Science and Technology(HUST), Wuhan, China. Her research interests include visual tracking, computer vision, and image processing.

Ping Feng received the B.Sc. degree in 2010 from Qufu Normal university, Rizhao, China. He is pursuing the PhD degree in the School of Computer Science and Technology, Huazhong University of Science and Technology(HUST), Wuhan, China. His research interests include visual tracking, dictionary learning, neural network and saliency detection.

Zhiqiang Zhao received the B.Sc. degree in computer science and technology in 1998 from Hunan University, Changsha, China. He is currently working toward the Ph.D. degree in the School of Computer Science and Technology, Huazhong University of Science and Technology(HUST), Wuhan, China. His research interests include visual tracking, computer vision, and image processing.

Yihao Luo received the B.Sc. degree in 2017 from Huazhong University of Science and Technology(HUST), Wuhan, China. He is pursuing the PhD degree in the School of Computer Science and Technology, Huazhong University of Science and Technology(HUST), Wuhan, China. His research interests include visual tracking, dictionary learning, neural network and saliency detection.

Chunyan Xu received the B.Sc. degree from Shandong Normal University in 2007 and the M.Sc. degree from Huazhong Normal University in 2010 and the PhD degree in the School of Computer Science and Technology, Huazhong University of Science and Technology(HUST) in 2015. She is a visiting scholar at National University of Singapore from 2013 to 2015. She is currently a teacher in Nanjing University of Science and Technology. Her research interests include deep neural network, computer vision, manifold learning and kernel methods.

Tianjiang Wang received the B.Sc. degree in computational mathematics in 1982 and the PhD degree in computer science in 1999 from Huazhong University of Science and Technology(HUST), Wuhan, China. He is currently a Professor with the School of Computer Science, Huazhong University of Science and Technology, Wuhan, China. He has finished some related projects and is the author of more than 20 related papers. His research interests include machine learning, computer vision, and data mining.

Kui Duan is an associate chief physician in Huazhong University of Science and Technology(HUST) school hospital, Wuhan, China. Her research interests include medical image processing and m edical image analysis.

Affiliations

Caihong Yuan[1,2] · **Jingjuan Guo**[1,3] · **Ping Feng**[1] · **Zhiqiang Zhao**[3] · **Yihao Luo**[1] · **Chunyan Xu**[4] · **Tianjiang Wang**[1] · **Kui Duan**[5]

✉ Kui Duan
kuiduan@hust.edu.cn

Jingjuan Guo
jj_guo@hust.edu.cn

Ping Feng
fengping@hust.edu.cn

Zhiqiang Zhao
zq_zhao@hust.edu.cn

Yihao Luo
luoyihao@hust.edu.cn

Chunyan Xu
cyx@njust.edu.cn

Tianjiang Wang
tjwang@hust.edu.cn

[1] School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China

[2] School of Computer and Information Engineering, Henan University, Kaifeng 475004, China

[3] School of Information Science and Technology, Jiujiang University, Jiujiang 332005, China

[4] School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China

[5] Huazhong University of Science and Technology, Wuhan 430074, China