International Journal of Machine Learning and Cybernetics (2021) 12:2899–2916
https://doi.org/10.1007/s13042-021-01376-8

ORIGINAL ARTICLE

Unsupervised person re-identification via *K*-reciprocal encoding and style transfer

Kun Xie[1] · You Wu[2] · Jing Xiao[1] · Jingjing Li[1] · Guohui Xiao[3] · Yang Cao[1]

Received: 14 October 2020 / Accepted: 5 July 2021 / Published online: 16 July 2021

Abstract
In this paper, we study the unsupervised person re-identification (re-ID) problem, which does not require any annotation information. Our approach considers three aspects in unsupervised re-ID task, i.e., variance across various cameras, label allocation to unlabeled images and hard negative mining. First, an unsupervised style transfer model is adopted to generate style-transferred images with different camera styles, which contributes to reduce the variance across various cameras. Then we apply *k*-reciprocal encoding method to obtain *k*-reciprocal nearest neighbors. According to the feature similarity of the probe person with its neighbors, soft pseudo labels are allocated to the probe person iteratively. Due to lack of annotation information to pairwise images, we propose the *k*-reciprocal nearest neighbors loss (KNNL) to learn discriminative features. Furthermore, a hard negative mining strategy is adopted to improve the accuracy and robustness of our framework. We conduct experiments on three large-scale datasets: Market-1501, DukeMTMC-reID and MSMT17. Results show that our method not only outperforms the state-of-the-art unsupervised re-ID approaches, but also is superior to unsupervised domain adaptation methods (UDA) and semi-supervised learning methods.

Keywords Unsupervised learning · Person re-ID · *K*-reciprocal encoding · Style transfer

1 Introduction

Person re-identification (re-ID) aims at matching a query person image in a set of gallery pedestrian images. In recent year, re-ID has drawn increasing attention from academic research and has achieved impressive progress. However, overwhelming majority of them are supervised approaches [12, 26, 27, 39, 42], which require intensive manual labeling and not applicable to real scenarios. Therefore, in this paper, we focus on the unsupervised setting of re-ID problem. In the last few years, some unsupervised domain adaptation (UDA) methods [6, 7, 32, 44, 45] which aim to transfer source dataset knowledge to other unseen target re-ID dataset are proposed. These methods have achieved great success, however, UDA methods require a large labeled source dataset and the inter-class variance between source dataset and target dataset is hard to compensate.

In our work, we attempt to study the fully unsupervised person re-ID problem, which means no annotation information is required. Our approach is mainly based on the following three aspects: (1) overcoming the variance of image style across different cameras, such as illumination, occluder, viewpoint. (2) allocating soft pseudo labels to each unlabeled identity. (3) learning discriminative and robust features from hard negative samples.

Overcoming the variance of image style across different cameras As shown in Fig. 1, pedestrians captured by different cameras are suffering great appearance changes. To deal with it, we consider each camera as a style domain and adopt StarGAN [4] model to learn the camera style transfer model. specifically, with StarGAN, we can generate several new images of other camera styles and use these images for similarity mining to alleviate the effects of camera

Jingjing Li and Jing Xiao have contributed equally to this study.

✉ Jing Xiao
xiaojing@scnu.edu.cn

✉ Jingjing Li
lijingjing@scnu.edu.cn

[1] School of Computer Science, South China Normal University, Guangzhou 510631, China

[2] School of Mathematics and Statistics, Hunan Normal University, Changsha 410081, China

[3] Faculty of Computer Science, Free University of Bozen-Bolzano, 39100 Bolzano, Italy

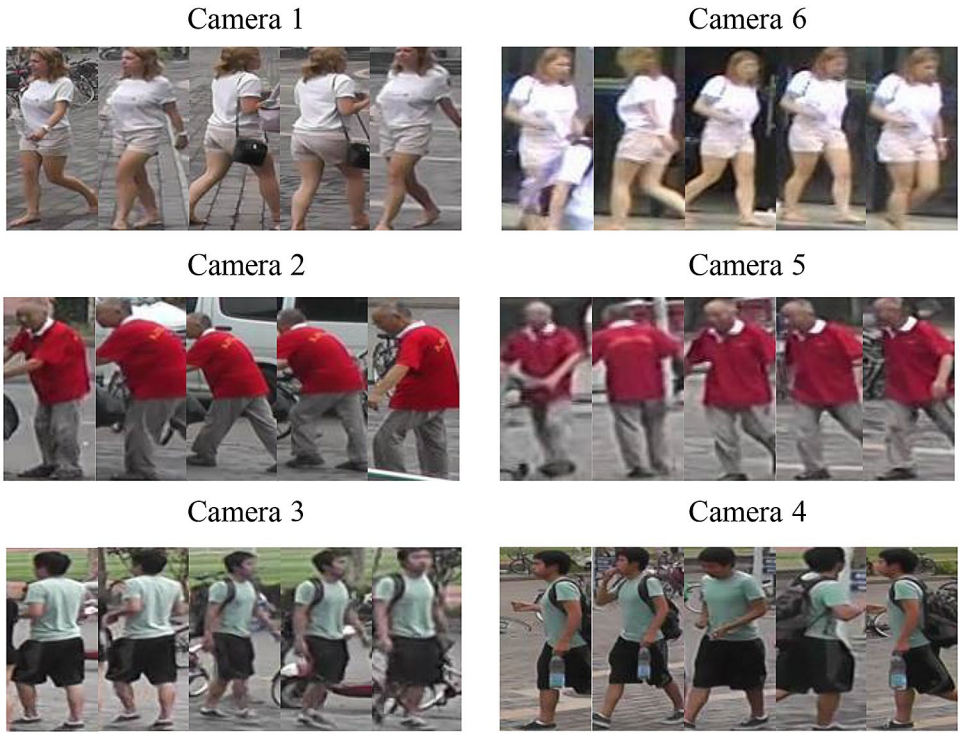

Fig. 1 Examples of images from different cameras. The images in the same row represent the same pedestrian. It is very evident that, different cameras have different lightness, occluder and viewpoint, *etc*

differences. Note that our training set is randomly selected from the combination of the original training images and the style-transferred images. More details are given in Sect. 3.1.

Allocating soft pseudo labels In order to allocate pseudo labels to each unlabeled identity, our idea starts by considering each training image as a different class, then we propose to iteratively apply k-reciprocal encoding method [15, 24, 43] to find out k-reciprocal nearest neighbors. We first select a probe person, and then the initial ranking list is obtained according to the similarity between the probe and the others. As shown in Fig. 2, from left to right, the similarity of the image features extracted by the model decreases sequentially. P1–P3 have the same identity with the probe. We call they are *true matches* to the probe. However, all of them are not included in the top-3 ranks. N1–N3 are *false matches*, but they also get high ranks. In order to alleviate the pollution of these false matches, we adopt k-reciprocal encoding method to obtain the k-reciprocal nearest neighbors of the probe that excludes many false matches. Apparently, excluding these false matches will help to reduce the noise during the re-ID training process. After that, the soft pseudo labels of the probe person are allocated according to the feature similarity of the k-reciprocal nearest neighbors.

Hard negative mining In the process of obtaining k-reciprocal nearest neighbors, we found that k-reciprocal encoding method will bring some hard negative samples which are considerably informative for training. Therefore we adopt a hard negative mining strategy to learn robust and discriminative features. Experimental results demonstrate that the strategy is the key to achieve high performance for our model.

Finally, during the iterative training procedure, our framework exploits the cross-camera similarity of the identities; allocates soft pseudo labels to each unlabeled identity and learns discriminative features from hard negative samples. We evaluate our proposed method on three large scale re-ID datasets. The experimental results reveal that our method outperforms state-of-the-art unsupervised methods on all the three datasets. It is worth mentioning that our method also exceeds the UDA re-ID methods which utilize a lot of annotation information. In summary, the contribution of this work is three-fold:

- We apply camera style transfer to generate images with different styles for decreasing the variance caused by different cameras. Experiments show that camera style transfer are indispensable for improving the performance of re-ID models.
- We propose an unsupervised re-ID framework via k-reciprocal encoding. By iteratively assigning soft pseudo labels to images according to the feature similarity of k-reciprocal nearest neighbors, our framework can learn robust and discriminative features.
- The experimental results demonstrate that our method outperforms the state-of-the-art unsupervised and UDA approaches by a large margin on three large-scale datasets, *i.e.*, Market-1501, DukeMTMC-reID and MSMT17.

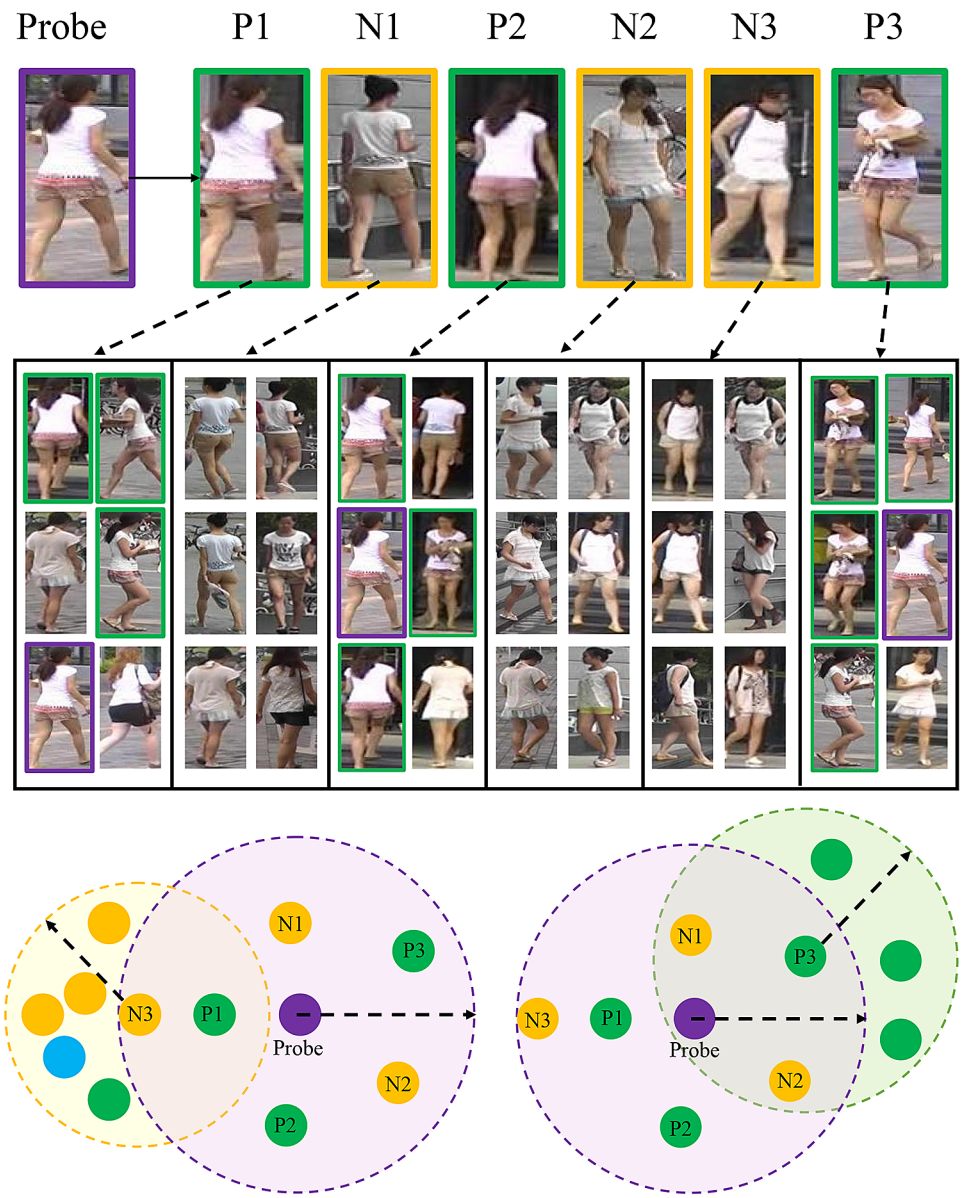

Fig. 2 Illustration of the *k*-reciprocal nearest neighborhoods in person re-ID. Top: The probe images and its 6-nearest neighborhoods, where P1–P3 in green are positives, N1–N3 in orange are negatives. Middle: Each two columns shows 6-nearest neighbors of the candidate image. For example, the first two columns are the 6-nearest neighbors of the candidate image P1. Purple and green box correspond to the probe and positives, respectively. Bottom: Purple, green and orange circle correspond to the probe, positives and negatives, respectively. On the right, P3 and the probe are 6-nearest neighbors reciprocally, while N3 and the probe on the left are not

2 Related work

In recent years, the deep convolutional neural networks (CNN) [11, 14, 18, 28] have made remarkable progress. Meanwhile, supervised deep learning on person re-ID [12, 26, 27, 39, 40] also achieve great success. However, supervised method means extensive annotations are needed, which leads to the deployment in real-world applications becoming very difficult. Therefore, in this paper, we focus on the unsupervised setting.

2.1 Unsupervised person re-identification

The fully unsupervised re-ID works can be summarized into three categories. The first category designs hand-craft features [8, 19, 23]. However, it is difficult to design discriminative features for images captured by different cameras, especially under different viewpoint and illumination condition. The second category [7, 20] utilizes clustering analysis in machine learning to predict pseudo labels for deep training model. However, these methods are not only hard to determine the number of clusters, but also difficult to quantify loss in clustering. The third category [22, 30] iteratively filter similar samples from training sets for each pedestrian image and assigns pseudo labels during training. The key of this method is to determine similar sample filters and pseudo-label assignments. In this paper, we follow the third category and adopt *k*-reciprocal encoding method to filter *k*-reciprocal nearest neighbors which is similar to the probe person image. We propose to assign soft pseudo labels

instead of hard pseudo labels to the probe person image according to the feature similarity of the neighbors.

2.2 Generative adversarial networks

In recent years, Generative adversarial networks (GANs) [10] have achieved tremendous success such as style transfer [2, 9, 16] and cross domain image generation [1, 29]. CycleGAN [47] introduces a cycle consistency loss to translate an image from a source domain X to a target domain Y in the absence of paired examples. However, when dealing with more than two domains, CycleGAN is time-consuming and limited in robustness. To address this limitation, Choi et al. [4] proposed StarGAN which can perform image-to-image translations for multiple domains using only a single model. In our work, we adopt the model of StarGAN [4] to generate images in different camera styles.

GANs is widely utilized in re-ID domain. DG-Net [40] applies GANs to generate high-quality pedestrian images and integrates them with re-ID models, while improving the quality of generated image and the accuracy of re-ID. SPGAN [6] and PTGAN [32] first generate different styles of pedestrian images, and then use the style-transferred image for training. CamStyle [46] transfers the labeled training images to styles of the each cameras, which increasing data diversity against overfitting. However, these supervised re-ID methods need to assign labels to the transferred images, and incorrect label assignment methods may reduce the accuracy and robustness of the model. Different from supervised re-ID works, our method utilizes GANs for unsupervised re-ID tasks which do not require label assignment to the transferred images.

2.3 Unsupervised domain adaptation

The main idea of unsupervised domain adaptation (UDA) [3, 7, 13, 31, 33, 37, 45] is to learn a re-ID model from a labeled source domain and an unlabeled target domain. UDA can effectively solve the scalability problem of supervised re-ID, and has achieved excellent success. MAR [37] learns a soft multi-label for each unlabeled person by comparing the unlabeled person with a set of known reference persons from an auxiliary domain. Zhong et al. [45] comprehensively investigate into the intra-domain variations of the target domain and propose a new framework which consists of two elaborate components, classification module and exemplar memory module. Classification module calculates the cross-entropy loss for labeled source data. Exemplar memory module saves the up-to-date features for target data and computes the invariance learning loss for unlabeled target data. Our approach and ECN [45] both contain a similar example memory module. However, our approach is completely unsupervised and does not require any annotated source domains for training.

3 Methodology

Preliminary Given an unlabeled training set $\mathrm{X} = \{x_i\}_{i=1}^{N}$ containing N cropped person images, our purpose is to learn a feature embedding function $\phi(\theta;x_i)$ from X without any manual annotations, where θ is the collection of parameters to the function ϕ. After training, this feature embedding function can be employed to the gallery set $\mathrm{X}^g = \{x_i^g\}_{i=1}^{N_g}$ of N_g images, and the query set $\mathrm{X}^q = \{x_i^q\}_{i=1}^{N_q}$ of N_q images. In the stage of testing, we extract the feature of query image $\phi(\theta;x_i^q)$ to search the similar image features from the gallery set. We use Euclidean distance to define the distance between each pair of images, *i.e.*, $d(x_i^q, x_i^g) = \left\|\phi(\theta;x_i^q) - \phi(\theta;x_i^g)\right\|$. The more similar of two images, the feature embedding distance between them are supposed to be smaller.

3.1 Camera style transfer

As shown in Fig. 1, pedestrians captured by different cameras are suffering great appearance changes, such as lightness, occluder, viewpoint, *etc.* In order to solve the problem of camera invariance, we propose to generate style-transferred images which can not only maintain the identity of a person, but also reflect the style of another camera. As mentioned above, we apply StarGAN [4] to learn camera style transfer model in the unlabelled training set. Different from CycleGAN [47] which can only capable of learning the relations between two different domains at a time, StarGAN takes in training data of multiple domains, and learns the relations between all available domains using only a single generator. For example, in Fig. 3, given three domains,

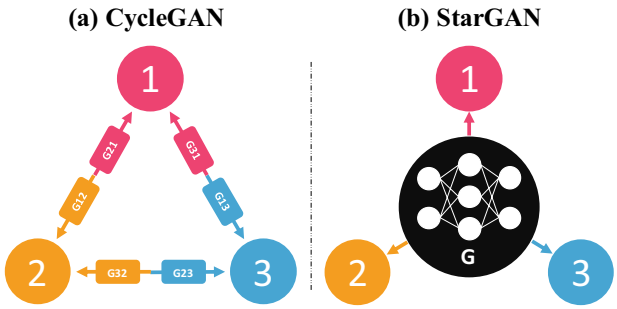

Fig. 3 The difference between CycleGAN and StarGAN. **a** To handle multiple domains, the CycleGAN model learns a generator for each pair of images. **b** StarGAN learns mappings between multiple domains using only one generator

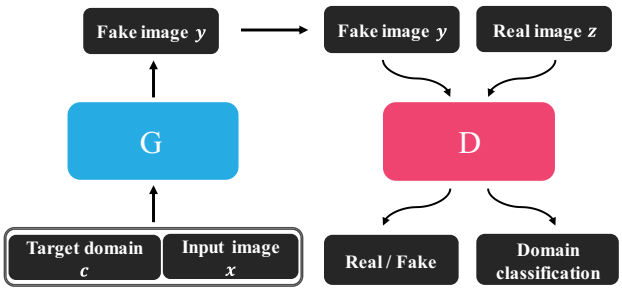

Fig. 4 Brief overview of StarGAN which consist of two modules, a generator G and a discriminator D

the CycleGAN model must learn six generators for three pairs of images, however StarGAN can learn the mapping between three domains with only one generator. Note that the pedestrian images generated by StartGAN are comparable or even better than CycleGAN. As shown in Fig. 5, we present several examples of images generated by both methods. Apparently, the background of the images generated by CycleGAN for the camera6 domain is darkened and the pedestrians become somewhat blurred. When learning to synthesize pedestrian images, it is important for the generator to detect body features internally, such as body shape or clothing color. Instead of learning a fixed translation like CycleGAN, which may be prone to overfitting, StarGAN

Fig. 5 Top: Examples of images generated by CycleGAN and StartGAN. The images in the blue box are generated by CycleGAN, and the images in the orange box are generated by StartGAN. Bottom: Examples of style-transferred images. The images framed in green are randomly selected

learns to translate the input image flexibly to the corresponding target domain. This is probably the reason why StarGAN performs better than CycleGAN. In this paper, we assume that the camera-ID of each image is known, since the camera-ID can be easily obtained when collecting pedestrian images from video sequences.

As shown in Fig. 4, StarGAN consists of two modules, a generator G and a discriminator D. The goal of StarGAN is to train a single generator G that learns mappings among multiple domains. Specifically, the StarGAN trains generator G to translate an input image x into a fake image y conditioned on the target domain label c, $G(x, c) \rightarrow y$. Meanwhile, the discriminator D is adopted to distinguish between real image z and fake image y generated by G and classify the real image z to its corresponding domain, $D : z \rightarrow \{D_{src}(z), D_{cls}(z)\}$. In consequence, after several epochs of training, G can generate images indistinguishable from real images and classifiable as target domain by D.

In our work, the images captured by different cameras are considered as different domains. Given a person re-ID dataset containing images captured by C different cameras, we aim to learn style transfer models for each camera pair with StarGAN, *i.e.*, each real target image collected from camera c is augmented with $C-1$ fake images in the styles of other cameras while remaining the original identity. For a training data x_i, the combination of the original training images and the style-transferred fake images can be denoted as, $\mathrm{X}_i^{style} = \{x_i^1, x_i^2, \ldots, x_i^C\}$. In the paper [21], camera style transfer is also adopted to overcome camera invariance. For an original images x_i, Lin et al. [21] use all the style-transferred images in X_i^{style} for training, which will be very time-consuming and introduce excessive noise. Our method adopts a different strategy by randomly selecting only one image from X_i^{style} for each x_i. For example, in the first row of Fig. 5, we generate 5 images of other camera style, and randomly select only one image framed in green as a training image.

3.2 Overview of framework

Following [20, 22, 45], we adopt ResNet-50 [11] pre-trained on ImageNet [5] as the backbone. Intuitively, the ResNet-50 backbone network takes an image as input and output a 2048-dim feature vector. There are two basic blocks in ResNet-50, one is Identity Block (ID Block) and the other is the Conv Block. In ID Block, the input and output dimensions are the same, so it can be connected multiple ones in series. The input and output dimensions in Conv Block are different and therefore cannot be connected in series. In ResNet-50, Conv Block is used to change the dimension of the feature map. As shown in Fig. 6, the ID Block takes a feature map I as input; after two stacked *(Convolution)-(Batch Norm)-(Relu Unit)* blocks and one *(Convolution)-(Batch Norm)* block, the output is F(I); then a shortcut connection is adopted to perform identity mapping, and its output are added to the output F(I) of the stacked layers. The original mapping is then recasted into F(I)+I; finally, ID Block is ended by a Batch Norm layer.

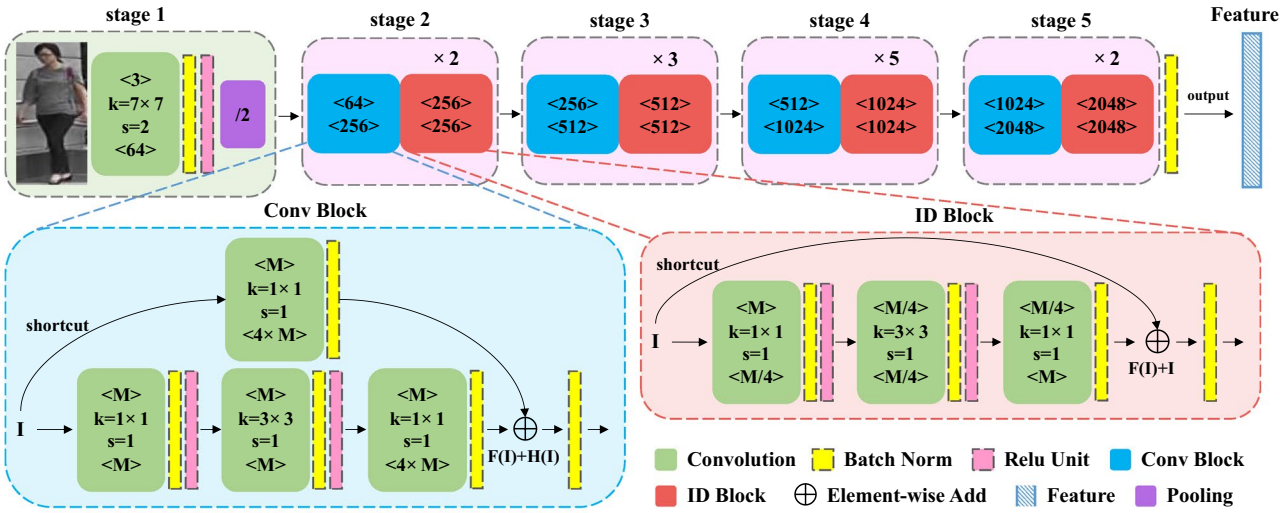

Fig. 6 Architecture of ResNet-50 network. Input images are resized into a fixed size of 256×128. There are two basic blocks in ResNet-50, one is Conv Block and the other is Identity Block (ID Block). The whole ResNet-50 network can be divided into 5 stages. In stage 1, Convolution layer takes images as input, and then a batch normalization (Batch Norm) is adopted right after Convolution and before activation (Relu Unit), finally Pooling layer is adopted to do downsampling. In stages 2–5, each of them is composed of a Conv Block and several ID Blocks. After 5 stages, we add a Batch Norm layer, which produces a 2048-dim feature. In this figure, k represents the convolution kernel size; s stands for stride; symbol < .. > represents the number of input and output channel respectively

Conv Block is similar to ID Block, but there are two differences. One is that it inserts a *(Convolution)-(Batch Norm)* block in the shortcut connection, and the other is that it changes the dimension of the feature map. In Conv Block, the shortcut connection takes I as input and output H(I), and the stacked layers output F(I) whose dimension is the same as H(I). The final output of Conv Block is the batch normalization of F(I)+H(I). The whole ResNet-50 network can be divided into 5 stages as shown in Fig. 6. After 5 stages, we perform batch normalization on the output and get the final 2048-dim feature.

The framework of our method is shown in Fig. 7. We divide our approach into two stages: Initialization and Re-train. Since the dataset does not have ground truth identity label for each image x_i, we start by regarding each image as an individual class and assign x_i with a hard label y_i as,

$$y_i[j] = \begin{cases} 1 & j = i \\ 0 & j \neq i \end{cases}, \tag{1}$$

with this hard label, each training sample is learned to push other training images away. However, for the images of the same identity, they are supposed to be close in the feature space. This hard label converts X into a labeled dataset, so that re-ID network training can be performed to obtain basic discriminative capabilities.

Then we feed the input x_i into backbone CNN to extract the feature v_i. In order to estimate the similarity between x_i and other images effectively, we construct a memory bank [36, 45] $V \in \mathbb{R}^{D \times N}$ for storing the up-to-date features of all training images, where D is the dimension of the feature, and v_i is a D-dimensional L2-normalized feature vector extracted by the person re-ID model. This computes the similarity between x_i and an another image x_j as,

$$S_i[j] = v_i^{\mathsf{T}} \cdot v_j, \tag{2}$$

where S_i denotes the similarity between v_i and other images. During the back-propagation, we update the features in the memory bank V for the training sample x_i through,

$$V^{\mathsf{T}}[i] = \alpha V^{\mathsf{T}}[i] + (1 - \alpha) v_i, \tag{3}$$

where V^{T} is the transposition of the memory bank matrix V, and $V^{\mathsf{T}}[i]$ is the memory of feature v_i. The hyper-parameter α controls the updating rate. $V^{\mathsf{T}}[i]$ is then L2-normalized by $V^{\mathsf{T}}[i] \leftarrow \left\| V^{\mathsf{T}}[i] \right\|_2$.

At Re-train stage, we regard feature v_i extracted by CNN from x_i as a probe and apply k-reciprocal encoding (KR-encoding) method to obtain k-reciprocal nearest neighbors $R(x_i, k)$ of x_i, then according to the feature similarity of the probe with neighbors, the soft pseudo label vector in entry j is computed as,

$$L_i[j] = \begin{cases} e^{S_i[j]-1} & x_j \in R(x_i, k) \\ 0 & x_j \notin R(x_i, k) \end{cases} \tag{4}$$

where $L_i[j] \in [0, 1]$ and L_i is the soft pseudo label for image x_i.

After obtaining the similarity vector S_i and soft label L_i of x_i, we propose the K-reciprocal Nearest Neighbors Loss (KNNL), which takes hard label y_i and similarity S_i as input for re-ID model training at the Initialization stage. At the

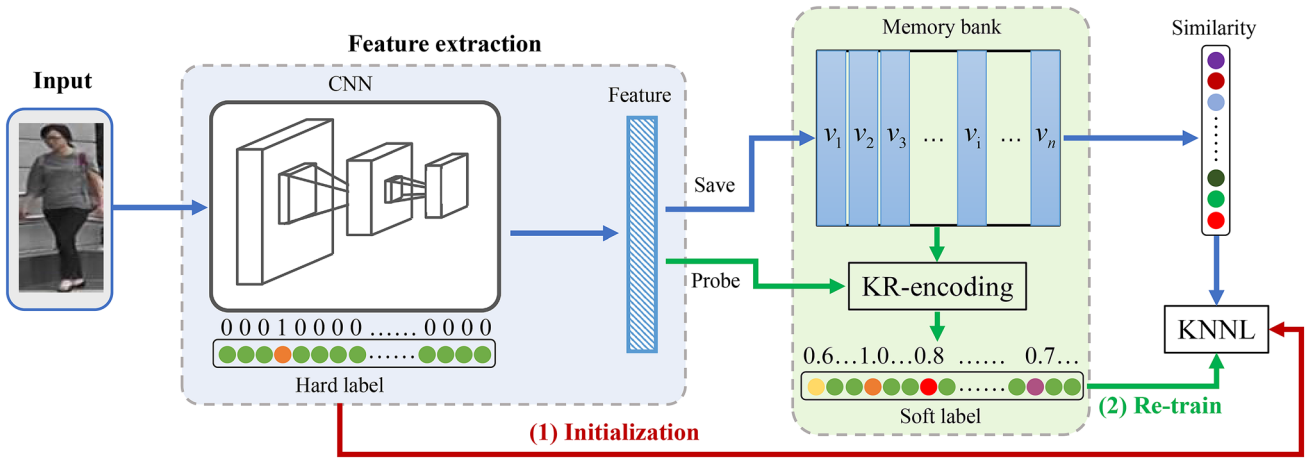

Fig. 7 The pipeline of the proposed approach. At first, we assign a hard single-class label to input image, then extract features and save to memory bank. We divide our approach into two stages: (1) Initialization, which is shown in the red arrows. In this stage, we just use hard single-class label as an initialization. (2) Re-train, shown in the green arrows. In this stage, we regard feature as a probe and apply k-reciprocal Encoding (KR-encoding) to generate soft pseudo label. The loss function takes soft pseudo label and similarity of the probe and other images as inputs to narrow the distance of k-reciprocal nearest neighbors

Re-train stage, KNNL takes soft label L_i and similarity S_i as input for training. Intuitively, For a probe x_i in the stage of Re-train, KNNL calculates the Euclidean distance between the similarity S_i and soft label L_i, pulling the similarity and soft pseudo labels closer and increasing the similarity between x_i and its k-reciprocal nearest neighbors $R(x_i, k)$. The KNNL can be represented as,

$$L_{KNNL} = \sum_{i=1}^{N} \frac{1}{\left|R(x_i,k)\right|} \sum_{x_j \in R(x_i,k)} \left\|S_i[j] - L_i[j]\right\|_2, \tag{5}$$

where $\left|R(x_i,k)\right|$ is the number of k-reciprocal nearest neighbors of x_i, and $\|\cdot\|_2$ represent Euclidean distance. At Initialization stage, L_i is equal to y_i. The overall training procedure is summarized in Algorithm 1.

Algorithm 1 Unsupervised Person Re-identification Approach Via k-reciprocal Encoding and Style Transfer

Input: The unlabeled dataset $\mathrm{X} = \{x_i\}_{i=1}^{N}$; CNN model $\phi(\theta^0; x)$.

Output: Best CNN model $\phi(\theta^*; x)$.

Style Transfer Step:
Applying StarGAN to learn camera style transfer model in the unlabeled training set.
Using generator G of the trained StarGAN to augment each training data x_i to $\mathrm{X_i^{style}} = \{\mathrm{x_i^1}, \mathrm{x_i^2}, ..., \mathrm{x_i^C}\}$.
Randomly selecting one image from each $\mathrm{X_i^{style}}$ and constructing a new training set $\mathrm{X^{style}}$.
Training Step:
Assigning each x_i in $\mathrm{X^{style}}$ with a hard label y_i and initializing memory bank V.
epoch = 0.
repeat
    For each x_i, using memory bank V to compute the similarity S_i between x_i and other images.
    if epoch $<$ $\mathrm{T_{ini}}$ **then**
        Initialization Stage:
        Training CNN model $\phi(\theta; x)$ with the KNNL loss on $\{(x_i, y_i)\}_{i=1}^{N}$ and $\{S_i\}_{i=1}^{N}$.
    else
        Re-train Stage:
        Applying KR-encoding to obtain k-reciprocal nearest neighbors $R(x_i, k)$ of the probe x_i and assign soft pseudo labels
        L_i according to the feature similarity of the probe x_i with its neighbors $R(x_i, k)$.
        Training CNN model $\phi(\theta; x)$ with the KNNL loss on $\{(x_i, L_i)\}_{i=1}^{N}$ and $\{S_i\}_{i=1}^{N}$.
    end if
    Extracting the features $\{v_i\}_{i=1}^{N}$, then updating the memory bank V with features $\{v_i\}_{i=1}^{N}$ before normalizing it.
    Evaluating on the test set $\rightarrow$ performance P.
    if $P > P^*$ **then**
        $P^* = P$
        Best model = $\phi(\theta; x)$
    end if
    epoch = epoch + 1;
until epoch = T.

3.3 *K*-reciprocal encoding

In [43], *k*-reciprocal encoding (KR-encoding) method is proposed to re-rank the supervised re-ID results and improve its accuracy. But in our work, *k*-reciprocal encoding method is applied to obtain soft pseudo labels for unsupervised re-ID. As illustrated in Fig. 7, KR-encoding takes the probe and feature memory bank of all training images as input and outputs soft pseudo labels. We can know from Eq. (4) that, the key for obtaining soft pseudo labels is to find the *k*-reciprocal nearest neighbors of the probe. Next we will elaborate on how to get the *k*-reciprocal nearest neighbors [15, 24, 43] of the probe x_i.

We define $E(x_i, k)$ as the *k*-nearest neighbors of a probe x_i,

$$E(x_i, k) = \{p_1, p_2, \ldots, p_k\}, \left|E(x_i, k)\right| = k, \tag{6}$$

where $|\cdot|$ denotes the number of candidates in the set and the *k*-nearest neighbors refers to the *k* neighbors with the closest similarity to the probe. The *k*-reciprocal nearest neighbors $R(x_i, k)$ can be defined as,

$$R(x_i, k) = \{p_j | (p_j \in E(x_i, k)) \wedge (x_i \in E(p_j, k))\}. \tag{7}$$

Compared with *k*-nearest neighbors, the *k*-reciprocal nearest neighbors and the probe x_i are more correlated and contain less noise. As shown in Fig. 2, the *k*-reciprocal nearest neighbor can effectively filter out the pollution of false matches to the top-k images. However, due to variations in occlusion, illumination, pose, and view, true matches may be excluded from the k-nearest neighbors and subsequently not included in the k-reciprocal nearest neighbors. Therefore, following [43], we incrementally add the ϵk-reciprocal nearest neighbors of each candidate in $R(x_i, k)$ into a more robust set $R^*(x_i, k)$ as,

$$\begin{aligned} R^*(x_i, k) &= R(x_i, k) \cup R(p_j, \epsilon k) \\ s.t. &\left|R(x_i, k) \cap R(p_j, \epsilon k)\right| \geqslant \frac{2}{3}\left|R(p_j, \epsilon k)\right|, \end{aligned} \tag{8}$$

where $p_j \in R(x_i, k)$, $\epsilon \in (0, 1)$. ϵ is the expansion rate, which controls the number of k-reciprocal nearest neighbors expansion for each candidate in $R(x_i, k)$. For example, when $\epsilon = 0.5$ and $k = 20$, we will add 10-reciprocal nearest neighbors of each candidate into $R^*(x_i, k)$. The parameter selection of ϵ is discussed in Sect. 4.4. After expansion process, we can add into $R^*(x_i, k)$ more positive samples which are more similar to the candidates in $R(x_i, k)$ than to the probe x_i. In our experiments, adding ϵk-reciprocal nearest neighbors to $R(x_i, k)$ set improves the accuracy of the model and obtains more robust features, which is discussed in Sect. 4.5.

Obviously, after expansion process, the Eq. (4) should be updated slightly as,

$$L_i[j] = \begin{cases} e^{S_i[j]-1} & x_j \in R^*(x_i, k) \\ 0 & x_j \notin R^*(x_i, k), \end{cases} \tag{9}$$

and Eq. (5) should also be updated as,

$$L_{KNNL} = \sum_{i=1}^{N} \frac{1}{\left|R^*(x_i, k)\right|} \sum_{x_j \in R^*(x_i, k)} \left\|S_i[j] - L_i[j]\right\|_2. \tag{10}$$

We can know from the Eq. (9) that the derivative of the function $L_i[j]$ to similarity $S_i[j]$ is greater than zero when $x_j \in R^*(x_i, k)$. This means the more similar x_j and x_i are, the larger the *j*-th entry of the soft pseudo label L_i vector is. The larger value of *j*-th entry in L_i allows the model to learn more similar features from x_j and x_i. In Sect. 4.5, we demonstrate that this soft pseudo label strategy really contributes to our model performance.

3.4 *K*-reciprocal nearest neighbors loss

As illustrated in Eq. (10), KNNL uses Euclidean distance to pull the similarity and soft pseudo label closer in the training of the re-ID model progresses. At initialization stage, KNNL takes the hard label y_i and similarity S_i as input, which will push all the training images away. However, for the images of the same identity, they are supposed to be close in the feature space. As consequence, the initialization process enables the re-ID model to have basic discrimination ability. At Re-train stage, KNNL takes the soft label L_i and similarity S_i as input. We can know from the Eq. (9) that the value of *k*-reciprocal nearest neighbors corresponding to the entry in the soft label L_i is not zero, and the values of other entries are all zero, thus KNNL will pull the probe x_i and all the elements in *k*-reciprocal nearest neighbors set $R^*(x_i, k)$ closer.

However, as shown in Fig. 2, we found that *k*-reciprocal encoding method will bring some hard negative samples. For example, N1–N3 are not the same identity as the probe, they are the probe's *k*-nearest neighbors but not the *k*-reciprocal nearest neighbors. Based on the above facts, we take the top *d*% images which do not belong to the *k*-reciprocal nearest neighbors as hard negative samples. The collection of hard negative samples for x_i can be denoted as $N(x_i)$. Intuitively, collection $N(x_i)$ contains images that look similar to the probe x_i but not the same identity as the probe. The length of collection $N(x_i)$ can be computed as,

$$\left|N(x_i)\right| = d\% \cdot \left(N - \left|R^*(x_i, k)\right|\right). \tag{11}$$

The hard negative loss can be illustrated as,

$$L_{HN} = \sum_{i=1}^{N} \frac{1}{\left|N(x_i)\right|} \sum_{x_j \in N(x_i)} \left\|S_i[j]\right\|_2, \tag{12}$$

when $x_j \in N(x_i)$, the soft label of x_i in entry j equals zero, *i.e.*, $L_i[j] = 0$, therefore we can ignore it in the Eq. (12). The hard negative loss L_{HN} makes the similarity S_i between the probe x_i and its hard negative samples close to zero, *i.e.*, L_{HN} will push away the probe and its hard negative samples, which will improve accuracy and robustness of the re-ID model.

All in all, we adopt a hard negative mining strategy to learn robust discriminated features. The final KNNL can be computed as,

$$L^*_{KNNL} = \lambda L_{KNNL} + L_{HN}, \tag{13}$$

where λ is a coefficient that balances the importance of L_{KNNL} and hard negative loss L_{HN}, which will be tested in experiments.

4 Experiments

4.1 Dataset

We evaluate our unsupervised method on three large-scale person re-identification (re-ID) benchmarks: Market-1501 [38], DukeMTMC-reID [25, 41] and MSMT17 [32]. Market-1501 contains 32,668 labeled person images of 1501 identities collected from 6 non-overlapping camera views. DukeMTMC-reID has 8 cameras and contains 36,411 labeled images of 1404 identities. MSMT17 is composed of 126,411 person images from 4101 identities collected by 15 cameras.

4.2 Performance evaluation

Our model performance is evaluated by the Cumulative Matching Characteristic (CMC) curves and mean Average Precision (mAP).

CMC curves are the most popular evaluation metrics for person re-identification methods. Considering a simple *single-gallery-shot* setting, where each gallery identity has only one instance. For each query image x, an algorithm will rank all the gallery samples according to their distances to the query from small to large. After that, the top-k samples are collected to form a set S, and the CMC top-k accuracy is

$$Acc_k = \begin{cases} 1 & x \in S \\ 0 & x \notin S, \end{cases} \tag{14}$$

which is a shifted step function. The final CMC curve is computed by averaging the shifted step functions over all the queries. For example, if the value of top-k is small, it means that after sorting, the similarity of the classification results is very low, indicating that the performance of the classifier is poor. For *multi-gallery-shot* setting, where each gallery identity could have multiple instances, the query will always match the "easiest" positive sample in the gallery while does not care other harder positive samples when computing CMC. Note that in the three datasets of this paper, the query and gallery sets could have same camera view, for each individual query identity, his/her gallery samples from the same camera are excluded.

mAP for a set of queries is defined as,

$$mAP = \frac{\sum_{x=1}^{Q} AveP(x)}{Q}. \tag{15}$$

where Q is the number of queries in the set and $AveP(x)$ is the average precision (AP) for a given query x. The formula is essentially telling us is that, for a given query x, we calculate its corresponding AP. Then the mean of the all these AP scores would give us a single number, called the mAP, which quantifies how good our model is at performing the query. Obviously, the key to calculating mAP is to get the AP of all queries. Next we introduce a related concept i.e. *precision* to calculate AP. The *precision* (also called positive predictive value) is the fraction of true positive (TP) instances and the total number of retrieved positive instances. The formula is given as,

$$precision = \frac{TP}{TP + FP}, \tag{16}$$

where FP is false positive instances. By default, precision takes all the retrieved instances into account, however, it can also be evaluated at a given number of retrieved instances, commonly known as cut-off rank, where the model is only assessed by considering only its top-most queries. The measure is called precision at k or $P@k$. In our retrieval task, given a query x, its corresponding AP is computed as,

$$AveP(x) = \frac{1}{GTP} \sum_{q=1}^{Q} P@k \times rel@k, \tag{17}$$

where GTP refers to the total number of ground truth positives, Q refers to the total number of images you are interested in, $rel@k$ is a relevance function. The relevance function is an indicator function which equals 1 if the instance at rank k is TP and equals to 0 otherwise. So far, For each query x, we can calculate a corresponding AP. The mAP is simply the mean of all the queries.

4.3 Experiment setting

For the camera style transfer model, we following the training strategy in [44]. In training, we employ random flipping and random cropping with a probability of 0.5 as data augmentation. Adam [17] with $\beta_1 = 0.5$ and $\beta_2 = 0.999$ are

applied as optimizer. At the first 100 epochs, the learning rate is 0.0001 for both generator and discriminator, and in the remaining 100 epochs the learning rate linearly decays to zero. For each image in the target set, we generate $C-1$ style-transferred fake images, where C is the number of cameras. We randomly select only one image from the combination of the original training images and the style-transferred images as our training set. We train StarGAN models for Market-1501, DukeMTMC-reID and MSMT17, respectively.

For the re-ID model, we adopt ResNet-50 [11] as the CNN backbone and initialize the model with the parameters pre-trained on ImageNet [5]. We apply random crop, random rotation, color jitter, and random erasing as data augmentation. The input image is resized to 256×128. We use SGD as optimizer for the re-ID model, the learning rate for ResNet-50 base layers are 0.02, and others are 0.2. We train the model for 60 epochs and the mini-batch size for model training is 128. In initialization stage, the memory bank starts saving all the features extracted by backbone, and the size of memory bank is $2048 \times N$. The memory updating rate α starts from 0 and grows linearly with the number of epochs to 0.5. After 5 epochs, our model enters the stage of Re-train and starts applying KR-encoding to calculate the soft label of each image. In KNNL, the coefficient λ is set to 3.0. In hard negative mining, we take top 0.4% images which do not belong to the k-reciprocal nearest neighbors as hard negative samples. The number of the k-reciprocal nearest neighbors k needs to be selected according to different datasets. The above parameter values are obtained from the parameter analysis in Sect. 4.4.

4.4 Parameter analysis

In this section, we investigate the sensitivities of our approach to five important hyper-parameters, i.e., the number of the k-reciprocal nearest neighbors k, the expansion rate ϵ, the coefficient of the loss λ, the hard negative mining rate d%, and finally the learning rate lr. For all parameter selection experiments, we performed 10-fold cross-validation. We divide the training set into 10 parts, and take one of them as the validation set for each training process. After that, 10 experiments are conducted and the results of the 10 experiments are averaged as the final test results.

Number of k-reciprocal nearest neighbors k In Fig. 8, we report the effect of the k-reciprocal nearest neighbors k. We vary k from 1 to 30 and set $\epsilon = 0.6, \lambda = 3, d = 0.4, lr = 0.2$ to test the model performance. When k is equal to 1, the expansion process mentioned in Sect. 3.3 is not considered. From the figure, we can clearly conclude that when k is between 16 and 24, the Rank-1 accuracy and mAP of our model fluctuate very little and the best results are obtained. This indicates that the performance of our re-ID model will reach the best when k is within a reasonable range. Small k will weaken the ability of KR-encoding to select more true matches, at the same time, large k will introduce many negative labels. The best k is around 22 for both Market-1501 and 20 for DukeMTMC-reID. The DukeMTMC-reID dataset has on average more images per pedestrian than the Market-1501 dataset. Intuitively, it should be the case that the k of the

Table 1 Performance evaluation with different value of ϵ in Eq. (8)

ϵ	Market-1501		DukeMTMC-reID	
	Rank-1	mAP	Rank-1	mAP
0.1	78.8	48.6	65.7	40.7
0.2	81.2	52.7	66.0	41.7
0.3	81.4	51.4	67.1	42.3
0.4	81.3	52.6	68.0	43.7
0.5	81.7	52.5	67.5	43.2
0.6	**82.6**	**54.1**	**68.5**	45.1
0.7	82.5	53.8	67.9	**45.3**
0.8	81.9	53.2	67.2	44.3
0.9	81.2	52.3	66.4	42.5

Bold values indicate the values of Rank accuracy or mAP of the best method in this column

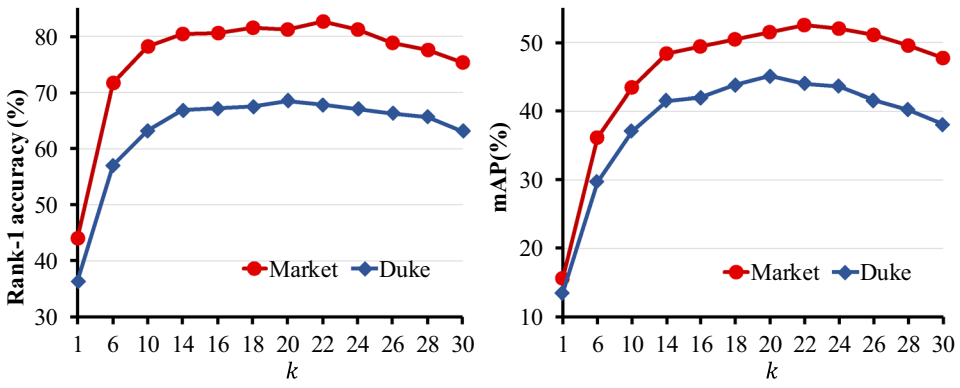

Fig. 8 Evaluation with different value of k in Eq. (6)

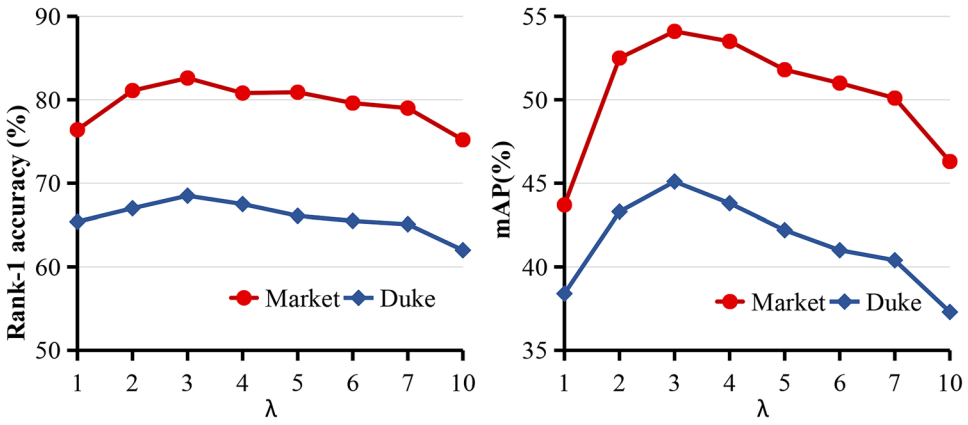

Fig. 9 Evaluation with different values of λ in Eq. (13)

DukeMTMC-reID is larger than that of the Market-1501, but the result is exactly the opposite. Empirically, we believe this may be due to the fact that the duke dataset has more noise than the market dataset, so the larger k introduces more negative labels.

Expansion rate ϵ In Table 1, we explore the effect of different expansion rates ϵ on model performance. We vary ϵ from 0.1 to 0.9 to test the performance of our model. In dataset Market-1501, we set $k = 22, \lambda = 3, d = 0.4, lr = 0.2$, and the results indicate that the best result is obtained when expansion rate ϵ equal to 0.6. In dataset DukeMTMC-reID, we set $k = 20, \lambda = 3, d = 0.4, lr = 0.2$. The best Rank-1 accuracy is also obtained when expansion rate is equal to 0.6, however, the best mAP is obtained when expansion rate is equal to 0.7. After our consideration, we finally set $\epsilon = 0.6$ as the default setting for the later parameter selection experiments.

Coefficient of the loss λ Figure 9 investigates the effect of coefficient λ in L^*_{KNNL}. Coefficient λ balances the importance of L_{KNNL} and hard negative loss L_{HN}. We can know from Sect. 3 that the L_{KNNL} increases the similarity between a probe image and its k-reciprocal nearest neighbors. Meanwhile, L_{HN} pushes away the probe image and its hard negative samples. When λ is too small, the loss function will be more inclined to push away the probe image and its hard negative samples than to pull closer with its k-reciprocal nearest neighbors. This may lead to large intra-class distances and small inter-class distances, thus impairing the performance of the model. When λ is too large, the loss function will prefer to increase the similarity between the

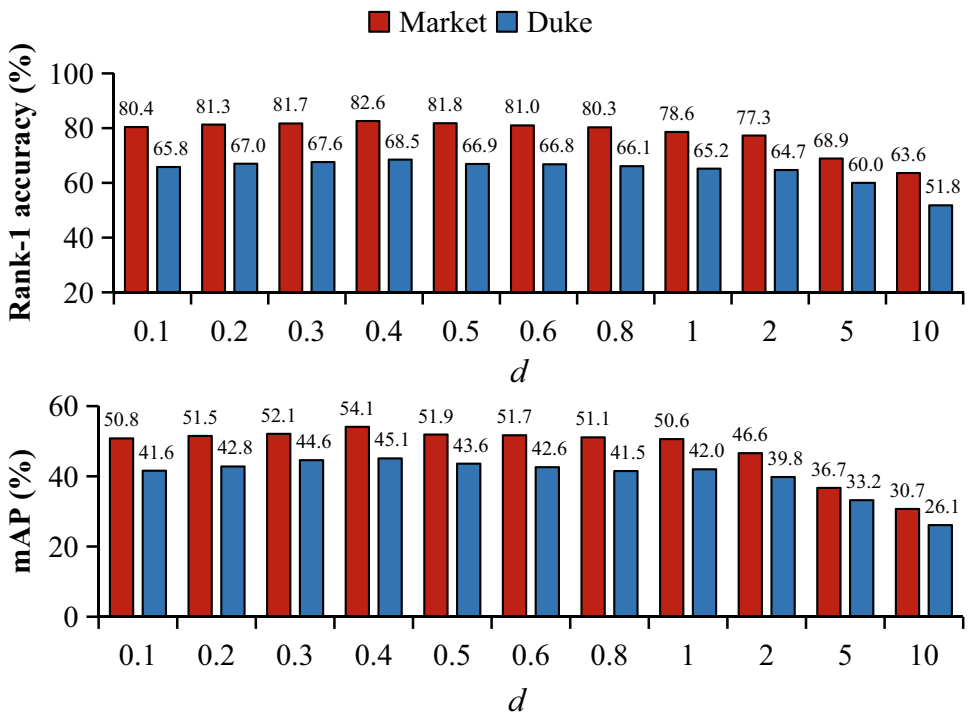

Fig. 10 Evaluation with different values of d in Eq. (11)

probe image and its k-reciprocal nearest neighbors, ignoring its hard negative samples, which is obviously not a good strategy. Therefore, in order to get the best model performance, it is necessary to find a reasonable λ. We evaluate different values for the coefficient λ. When $\lambda = 1$, the importance of L_{KNNL} and L_{HN} are the same. When λ is between 2 and 5, our result is impacted just marginally and the best results are obtained. This shows that our method is insensitive to λ in an appropriate range. When λ is greater than 5, the performance of mAP begins to decline a lot. Finally, We take the best result and set $\lambda = 3$.

Hard negative mining rate d Figure 10 illustrates the effect of hard negative mining rate d. We vary d from 0.1 to 10 and set $\epsilon = 0.6, \lambda = 3, d = 0.4, lr = 0.2, k = 22$ for dataset Market-1501 and $k = 20$ for dataset DukeMTMC-reID. When $r = 10$, we use 10% negative classes for loss computation. Obviously, $d = 10$ greatly reduces the performance of the model. When d is between 0.1 and 0.8, our result is also impacted marginally and insensitive to parameter d. When d is greater than 1, performance begins to drop dramatically. As same as the coefficient of the loss λ, we take the best result and set $d = 0.4$.

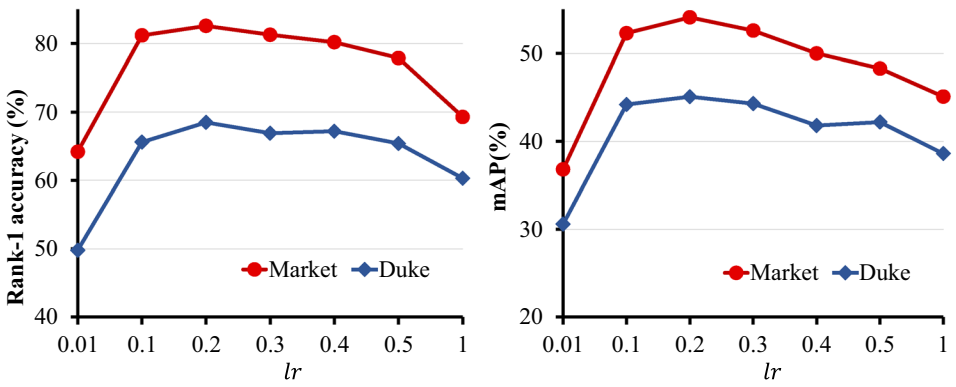

Fig. 11 Evaluation with different values of learning rate *lr*

Table 2 Comparison with the state-of-the-art methods on Market-1501 and DukeMTMC-reID

Methods	Reference	Setting	Market-1501				DukeMTMC-reID			
			Rank-1	Rank-5	Rank-10	mAP	Rank-1	Rank-5	Rank-10	mAP
LOMO[19]	CVPR15	**Unsupervised**	27.2	41.6	49.1	8.0	12.3	21.3	26.6	4.8
BOW[38]	ICCV15	**Unsupervised**	35.8	52.4	60.3	14.8	17.1	28.8	34.9	8.3
OIM[36]	CVPR17	**Unsupervised**	38.0	58.0	66.3	14.0	24.5	38.8	46.0	11.3
PTGAN[32]	CVPR18	UDA	38.6	–	66.1	–	27.4	–	50.7	–
PUL[7]	TOMM18	UDA	45.5	60.7	66.7	20.5	30.0	43.4	48.5	16.4
EUG[35]	CVPR18	OneEx	49.8	66.4	72.7	22.5	45.2	59.2	63.4	24.5
Progressive[34]	TIP19	OneEx	55.8	72.3	78.4	26.2	48.8	63.4	68.4	28.5
SPGAN[6]	CVPR18	UDA	58.1	76.0	82.7	26.7	46.9	62.6	68.5	26.4
HHL[44]	ECCV18	UDA	62.2	78.8	84.0	31.4	46.9	61.0	66.7	27.2
BUC[20]	AAAI19	**Unsupervised**	66.2	79.6	84.5	38.3	47.4	62.6	68.4	27.5
CSE [21]	TIP20	**Unsupervised**	73.7	84.0	87.9	38.0	56.1	66.7	71.5	30.6
ECN[45]	CVPR19	UDA	75.1	87.6	91.6	43.0	63.3	75.8	*80.4*	*40.4*
MMCL+MPLP[30]	CVPR20	**Unsupervised**	*80.3*	*89.4*	*92.3*	*45.5*	*65.2*	*75.9*	80.0	40.2
Ours (w/o hardNeg)	–	**Unsupervised**	27.1	45.6	54.8	10.0	26.5	41.0	48.1	9.6
Ours (w/o styleTrans)	–	**Unsupervised**	64.2	78.3	84.2	38.3	54.5	67.6	72.0	34.4
Ours (w/o expansion)	–	**Unsupervised**	79.1	89.9	92.8	49.4	66.5	76.9	81.2	40.2
Ours	-	**Unsupervised**	**82.6**	**91.3**	**94.5**	**54.1**	**68.5**	**79.0**	**82.8**	**45.1**

Bold values indicate the values of Rank accuracy or mAP of the best method in this column

Italics values represent the values of Rank accuracy or mAP of the second best method in this column

In the column "Setting", "Unsupervised" denotes the unsupervised methods. "UDA" denotes the unsupervised domain adaptation methods. "OneEx" denotes the methods use the one-example annotation, in which each person in the dataset is annotated with one labeled example

Learning rate lr It is well known that a small learning rate will cause the model to converge slowly and never reach the best performance, while a large learning rate will cause the model curve to oscillate around the convergence point. Therefore, it is crucial to select a proper learning rate. In our experiments, we vary lr from 0.01 to 0.5 and set $\epsilon = 0.6, \lambda = 3, d = 0.4, k = 22$ for the Market-1501 dataset and $k = 20$ for the DukeMTMC-reID dataset. Figure 11 demonstrates the impact of different learning rates. From the results, we can see that our model achieves best performance in both two datasets when the learning rate equals to 0.2. Therefore, we set $lr = 0.2$.

4.5 Ablation study

The Impact of hard negative mining To investigate the impact of the proposed camera style transfer for training dataset, we conducted an ablation study. The results with and without the hard negative (hardNeg) mining loss are shown in Ours (w/o hardNeg) of Tables 2 and 4. Specifically, on Market-1501, the hard negative mining loss improves the Rank-1 accuracy and mAP by 55.5 points and 44.1 points, respectively. On DukeMTMC-reID, the Rank-1 accuracy and mAP are improved 42.0 points and 35.5 points, respectively. On MSMT17, Rank-1 accuracy and mAP improved by 27.4 points and 10.3 points, respectively. The huge boost shows that without hard negative mining loss, the positive samples and the hard negative sample can not be separated effectively, thus impairing the ability of the re-ID model to discriminate false positive samples.

The Impact of camera style transfer The result of without style transfer are demonstrated in Ours (w/o styleTrans) of Tables 2 and 4. We observe that the Rank-1 accuracy of the datasets Market-1501 and DukeMTMC-reID decreased from 82.6% and 68.5% to 64.2% and 54.5%, respectively. On MSMT17, if style transfer was not performed, the Rank-1 accuracy and mAP decreased by 10.2% and 5.2%, respectively. The impressive result indicates that the style-transferred images effectively eliminate the variance of different cameras by training with the original images.

The Impact of k-reciprocal nearest neighbors expansion As shown in Table 2 Ours (w/o expansion), the Rank-1 accuracy is 79.1% and 66.5% for Market-1501 and DukeMTMC-reID, respectively, which is 3.5 and 2.0 points lower than our complete method. In Table 4 Ours (w/o expansion), we observe an improvements of 2.1 points in Rank-1 accuracy with expansion process. The above evidence proves that adding ϵ k-reciprocal nearest neighbors of each candidate truly introduce more positive samples into $R^*(x_i, k)$, which improves the accuracy of the model and obtains more robust features.

Table 3 Performance comparison of the soft and hard pseudo labels

Dataset	Market-1501		DukeMTMC-reID	
Metric(%)	Rank-1	mAP	Rank-1	mAP
Ours(w/ hard labels)	76.9	43.8	64.1	39.0
Ours	**82.6**	**54.1**	**68.5**	**45.1**

Bold values indicate the values of Rank accuracy or mAP of the best method in this column

The Impact of soft pseudo labels Different from Wang et al. [30] and Lin et al. [20], where they assign hard pseudo labels to each unlabeled identity when training models iteratively, we assign soft pseudo labels to probe person image according to the feature similarity of the k-reciprocal nearest neighbors as Eq. (4). As shown in Table 3, we test the impact of different label assignment strategy on experimental performance. The hard pseudo labels assignment can be represented as,

$$L_i[j] = \begin{cases} 1 & x_j \in R^*(x_i, k) \\ 0 & x_j \notin R^*(x_i, k) \end{cases}, \tag{18}$$

where $R^*(x_i, k)$ is the expansion of k-reciprocal nearest neighbors sets $R(x_i, k)$. In Table 3 Ours (w/ hard labels), the Rank-1 accuracy is 76.9% and 64.1% for Market-1501 and DukeMTMC-reID, respectively, which are 5.7 and 4.4 points lower than the soft pseudo labels setting. We think the reason of the performance degradation is that the model fails to learn the similarity and consistency between images of the same person when learning with hard pseudo labels. In contrast, our soft pseudo labels assignment can capacitate the model to learn the similarity among identities in a more smooth way.

The Impact of randomly selecting strategy In our approach, we adopt camera style transfer to overcome camera invariance. However, different from [21] which uses all the style-transferred images for training, our training set is randomly selected from the combination of the original training images and the style-transferred images. Because using all the style-transferred images for training is very time-consuming and introduce excessive noise. We train our re-ID model in the dataset Market-1501, using all image strategy and randomly selecting strategy, respectively. For using all image strategy, We first set k in Eq. (6) equals 22 as we analyzed in Section. 4.4. Result turns out that the Rank-1 accuracy and mAP is 66.3% and 29.9%, respectively, which are 16.3 and 24.2 points lower than the randomly selecting strategy. What's more, for training on GTX 1080TI GPU, the time required to use the randomly selecting strategy is about 6.8 h, while the training time required to use all image strategy is about 60.7 h, which is 9 times the former. However, during the experiment, we realize that when using all the style-transferred images, the size of the training set is several times that of the original data set; therefore, the

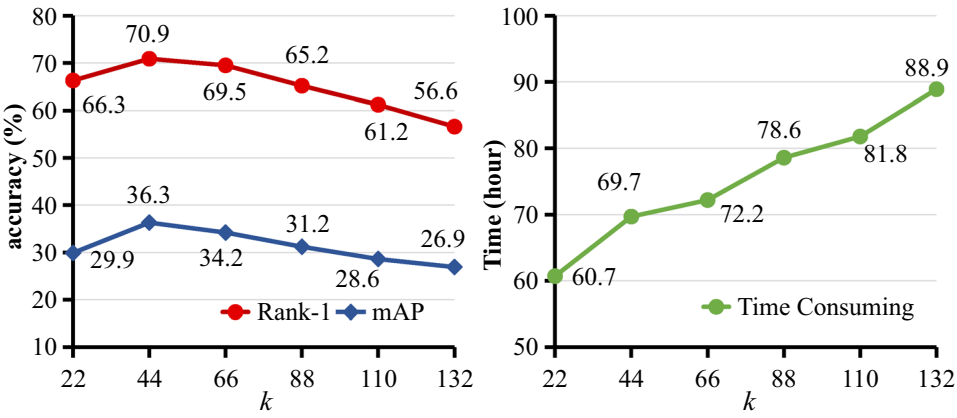

Fig. 12 Left: Performance of the using all image strategy. Right: Time Consuming

Table 4 Performance comparison with state-of-the-art methods on MSMT17

Methods	MSMT17			
	Rank-1	Rank-5	Rank-10	mAP
OIM[36]	7.3	14.4	18.5	1.7
PTGAN[32]	11.8	–	27.4	3.3
ECN[45]	30.2	41.5	46.8	10.2
CSE[21]	31.4	41.4	45.7	9.9
Ours(w/o hardNeg)	10.1	18.5	24.1	3.2
Ours(w/o styleTrans)	27.3	41.1	44.5	8.3
Ours(w/o expansion)	35.4	46.7	50.2	11.8
Ours	**37.5**	**48.5**	**52.0**	**13.4**

Bold values indicate the values of Rank accuracy or mAP of the best method in this column

number of k-reciprocal nearest neighbors of a probe image also becomes larger. Based on the above analysis, we start to set k equal to 22 and double k each time until $k = C \times k$, where C is the number of cameras. As shown in Fig. 12, when $k = 44$, the model reaches to best performance. The Rank-1 accuracy and mAP is 70.3% and 35.1%, respectively. However, this is still much lower than our performance using randomly selecting strategy. Not only that, when $k = 44$, the training time increases to 69.7 h, which is more than 10 times the randomly selecting method. The training time with using all image strategy is long without improving accuracy, which is why we recommend using randomly selecting strategy.

4.6 Comparison with state-of-the-art methods

We compare our approach against state-of-the-art unsupervised learning methods, unsupervised domain adaptation methods (UDA) and semi-supervised learning methods on three large-scale datasets, *i.e.*, Market-1501, DukeMTMC-reID and MSMT17.

The performance on Market-1501 and DukeMTMC-reID are summarized in Table 2. Note that the performance in [35, 36] are reproduced by [21] and we borrow the numbers to our table. In our paper, we compare with five unsupervised methods: LOMO [19], BOW [38], OIM [36], BUC [20], MMCL+MPLP [30], five unsupervised domain adaptation (UDA) methods: PTGAN [32], PUL [7], SPGAN [6], HHL [44], ECN [45], and two one-example annotation methods: EUG [35], Progressive [34]. In the one-example annotation methods, each person in the dataset is annotated with one labeled example.

On Market-1501, we obtain the best performance among the compared methods with **Rank-1 = 82.6%, mAP = 54.1%**. Compared to the state-of-the-art unsupervised method MMCL+MPLP [30] in the fully unsupervised setting, we achieve 2.3 points and 8.6 points improvement in Rank-1 accuracy and mAP, respectively. The 8.6 points improvement of mAP is mainly attributed to the k-reciprocal encoding, which greatly alleviate the pollution of false matches. Our method is also superior to UDA methods and semi-supervised learning methods. For example, our approach outperforms the ECN [45] and Progressive [34] by a large margin of 7.5% and 26.8% in Rank-1, respectively.

On DukeMTMC-reID, we achieve the best accuracy with **Rank-1 = 68.5%, mAP = 45.1%**. Compared to fully unsupervised methods MMCL+MPLP [30], our method improves the accuracy of Rank-1 and mAP by 3.3% and 4.9%, respectively. Compared to UDA methods ECN [45] and semi-supervised learning methods Progressive [34], our method achieve 5.2 points and 19.7 points improvement in Rank-1 accuracy, respectively.

We also evaluate our approach on a larger and more challenging dataset, i.e., MSMT17. The performance are demonstrated in Table 4. On MSMT17, our method produces the best performance with **Rank-1 = 37.5% and mAP =13.4%**. Compared to UDA methods ECN [45], we achieve 7.3 points and 3.2 points improvement in Rank-1 accuracy and mAP, respectively. The main reason is that the UDA method

Table 5 Performance of training on one dataset and testing on another

Training Dataset	Testing Dataset					
	Market-1501		DukeMTMC-reID		MSMT17	
	Rank-1	mAP	Rank-1	mAP	Rank-1	mAP
Market-1501	**82.6**	**54.1**	49.8	27.8	26.6	8.2
DukeMTMC-reID	59.3	28.1	**68.5**	**45.1**	27.9	8.6
MSMT17	43.8	17.7	36.9	16.0	**37.5**	**13.4**

Bold values indicate the values of Rank accuracy or mAP of the best method in this column

When the training dataset and the test dataset are the same, it means both training and test are performed on the same dataset

cannot directly learn discriminative features from the target dataset. Meanwhile, the inter-class variance between source dataset and target dataset is hard to compensate. Our performance are also 6.1 points and 3.5 points higher than fully unsupervised methods CSE [21] in Rank-1 accuracy and mAP, respectively.

4.7 Cross-dataset testing study

To discover the weaknesses and potential of our proposed framework, in this section, we design two cross-dataset testing experiments. In the first experiment, we train our framework on dataset *A* and then test it directly on the test set of another dataset *B*. In the second experiment, similarly, we first train on a dataset *A*, the difference is that we fine-tune the trained framework with images from the training set of dataset *B* before testing.

In the first cross-dataset experiment, we train our framework on datasets Market-1501 [38], DukeMTMC-reID [25, 41] and MSMT17 [32], respectively, and then test it on the other datasets. The results are demonstrated in Table 5. If we train the framework on DukeMTMC-reID and test it on Market-1501, the Rank-1 accuracy and mAP are 59.3% and 28.1%, which are 23.3% and 26% less than training and testing directly on Market-1501, respectively. Obviously, the results of the test are not promising. The reason is simple, i.e., each dataset varies very much. If you train on one image domain but tests on another different image domain, the results are predictable. This weakness of not being able to adapt well to different domains is not only present in our framework, but also in almost all other frameworks.

In the second experiment, to discover the potential of our framework, we fine-tuned the already trained model using 20%, 60%, and 100% of the images of the testing dataset, respectively. The results are shown in Table 6. Apparently, The framework performs increasingly well as the proportion of the fine-tuned training set increases. Note that our fine-tuning experiments are different from unsupervised domain adaptation (UDA) [6, 7, 32, 44, 45], we do not need any labels, whereas the UDA approach requires a lot of labeling information. Observing the experimental results, we found two interesting phenomena. First, after training on the DukeMTMC-reID dataset and then fine-tuning with 100% of the Market-1501 dataset, Rank-1 achieves an accuracy of 84.2%, which is 1.6% higher than the 82.6% accuracy when trained and tested directly on Market-1501. Second, after

Table 6 Performance of the fine-tuned model

Training Dataset	Testing Dataset						
	Percentage(%)	Market-1501		DukeMTMC-reID		MSMT17	
		Rank-1	mAP	Rank-1	mAP	Rank-1	mAP
Market-1501	20	–	–	57.0	32.3	25.3	7.7
	60	–	–	65.3	41.3	28.9	9.2
	100	–	–	**68.0**	**43.3**	33.6	10.7
DukeMTMC-reID	20	71.6	36.5	–	–	27.0	8.1
	60	80.5	48.9	–	-	31.0	9.8
	100	**84.2**	**54.3**	–	–	**35.3**	**11.7**
MSMT17	20	61.3	30.3	54.5	30.0	–	–
	60	75.1	44.1	62.1	37.6	–	–
	100	80.7	48.4	65.4	40.0	–	–

Bold values indicate the values of Rank accuracy or mAP of the best method in this column

We fine-tuned the trained model with images from 20%, 60%, and 100% of the test dataset, respectively

training on datasets Market-1501 and DukeMTMC-reID, respectively, we tested our framework directly on dataset MSMT17 and obtained Rank-1 accuracies of 26.6% and 27.9%, however, when we performed fine-tuning experiments using 20% of the MSMT17 dataset, the framework's Rank-1 accuracy unexpectedly dropped by 1.3% and 0.9%, respectively. Before illustrating these two phenomena, we discuss the complexity of the three datasets. Suppose the size of the training set of Market-1501 is m. Then, the training sets of DukeMTMC-reID and MSMT17 are about $1.28m$ and $2.52m$, respectively. this indicates that DukeMTMC-reID is a bit more complex than Market-1501, while MSMT17 is much more complex than Market- 1501 and DukeMTMC-reID. The first phenomenon suggests that the framework parameters obtained from the DukeMTMC-reID dataset are beneficial to the performance of the Market-1501 dataset when DukeMTMC-reID is a bit more complex than Market-1501. This illustrates the generality of the features extracted by our framework and reflects the potential of our framework to extract generic features. The second phenomenon illustrates that when the complexity of DukeMTMC-reID and Market-1501 is much smaller than that of MSMT17, the parameters obtained from these two datasets are harmful to the performance of the MSMT17 dataset. We believe this may be due to the high complexity of MSMT17, and the features of DukeMTMC-reID and Market-1501 are very different from those of MSMT17, thus, conducting fine-tuning experiments after DukeMTMC-reID or Market-1501 training is equivalent to introducing noise.

5 Conclusion

This paper proposes a method based on k-reciprocal encoding and style transfer to address the fully unsupervised person ReID which does not require any annotation information. With the k-reciprocal encoding, we can directly obtain k-reciprocal nearest neighbors and filter out the pollution of false matches. According to the feature similarity of the probe person and its neighbors, soft pseudo labels are allocated to the probe person. KNNL with hard negative loss is introduced to learn robust discriminated features and improve the re-ID model accuracy. Experiments on three large-scale datasets validate the effectiveness of the our approach in unsupervised person re-ID.

Acknowledgements This work is supported by the National Natural Science Foundation of China No. 61872153, the National Science Foundation of Guangdong Province No. 2018A030313318 and the Key-Area Research and Development Program of Guangdong Province No. 2019B111101001.

Declarations

Conflicts of interest The authors declare that there is no conflict of interests regarding the publication of this article.

References

1. Bousmalis K, Silberman N, Dohan D, Erhan D, Krishnan D (2017) Unsupervised pixel-level domain adaptation with generative adversarial networks. In: 2017 IEEE Conference on computer vision and pattern recognition (CVPR), vol 1, pp 95–104
2. Chen D, Yuan L, Liao J, Yu N, Hua G (2017) Stylebank: an explicit representation for neural image style transfer. In: 2017 IEEE Conference on computer vision and pattern recognition (CVPR), vol 1, pp 2770–2779
3. Chen Y, Zhu X, Gong S (2019) Instance-guided context rendering for cross-domain person re-identification. In: 2019 IEEE/CVF International Conference on computer vision (ICCV), vol 1, pp 232–242
4. Choi Y, Choi MJ, Kim M, Ha JW, Kim S, Choo J (2018) Stargan: unified generative adversarial networks for multi-domain image-to-image translation. In: 2018 IEEE/CVF Conference on computer vision and pattern recognition, vol 1, pp 8789–8797
5. Deng J, Dong W, Socher R, Li L, Li K, Fei-Fei L (2009) Imagenet: a large-scale hierarchical image database. In: 2009 IEEE Conference on computer vision and pattern recognition, vol 1, pp 248–255
6. Deng W, Zheng L, Kang G, Yang Y, Ye Q, Jiao J (2018) Image-image domain adaptation with preserved self-similarity and domain-dissimilarity for person re-identification. In: 2018 IEEE/CVF Conference on computer vision and pattern recognition, vol 1, pp 994–1003
7. Fan H, Zheng L, Yang Y (2018) Unsupervised person re-identification. ACM Trans Multimed Comput Commun Appl (TOMM) 14:1–18
8. Farenzena M, Bazzani L, Perina A, Murino V, Cristani M (2010) Person re-identification by symmetry-driven accumulation of local features. In: 2010 IEEE Computer Society Conference on computer vision and pattern recognition, vol 1, pp 2360–2367
9. Gatys LA, Ecker AS, Bethge M (2016) Image style transfer using convolutional neural networks. In: 2016 IEEE Conference on computer vision and pattern recognition (CVPR), vol 1, pp 2414–2423
10. Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville AC, Bengio Y (2014) Generative adversarial networks. In: Proceedings of the International Conference on neural information processing systems, vol 27, pp 2672–2680
11. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer vision and pattern recognition (CVPR), vol 1, pp 770–778
12. Hermans A, Beyer L, Leibe B (2017) In defense of the triplet loss for person re-identification. arXiv preprint arXiv:1703.07737
13. Hoffman J, Tzeng E, Darrell T, Saenko K (2015) Simultaneous deep transfer across domains and tasks. 2015 IEEE International Conference on computer vision (ICCV), vol 1, pp 4068–4076
14. Huang G, Liu Z, Weinberger KQ (2017) Densely connected convolutional networks. In: 2017 IEEE Conference on computer vision and pattern recognition (CVPR) 1:2261–2269
15. Jégou H, Harzallah H, Schmid C (2007) A contextual dissimilarity measure for accurate and efficient image search. In: 2007 IEEE Conference on computer vision and pattern recognition, vol 1, pp 1–8

16. Johnson J, Alahi A, Fei-Fei L (2016) Perceptual losses for real-time style transfer and super-resolution. In: European Conference on computer vision, pp 694–711
17. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980
18. Krizhevsky A, Sutskever I, Hinton G (2012) Imagenet classification with deep convolutional neural networks. Neural Inf Proces Syst. https://doi.org/10.1145/3065386
19. Liao S, Hu Y, Xiangyu Zhu, Li SZ (2015) Person re-identification by local maximal occurrence representation and metric learning. In: 2015 IEEE Conference on computer vision and pattern recognition (CVPR), vol 1, pp 2197–2206
20. Lin Y, Dong X, Zheng L, Yan Y, Yang Y (2019) A bottom-up clustering approach to unsupervised person re-identification. Proc AAAI Conf Artif Intell 33:8738–8745. https://doi.org/10.1609/aaai.v33i01.33018738
21. Lin Y, Wu Y, Yan C, Xu M, Yang Y (2020) Unsupervised person re-identification via cross-camera similarity exploration. IEEE Trans Image Process 29:5481–5490
22. Lin Y, Xie L, Wu Y, Yan C, Tian Q (2020) Unsupervised person re-identification via softened similarity learning. In: 2020 IEEE/CVF Conference on computer vision and pattern recognition (CVPR), vol 1, pp 3387–3396
23. Lisanti G, Masi I, Bagdanov AD, Bimbo AD (2015) Person re-identification by iterative re-weighted sparse ranking. IEEE Trans Pattern Anal Mach Intell 37(8):1629–1642
24. Qin D, Gammeter S, Bossard L, Quack T, Gool L (2011) Hello neighbor: accurate object retrieval with k-reciprocal nearest neighbors. CVPR 2011:777–784
25. Ristani E, Solera F, Zou RS, Cucchiara R, Tomasi C (2016) Performance measures and a data set for multi-target, multi-camera tracking. In: ECCV Workshops, pp 17–35
26. Sun Y, Zheng L, Deng W, Wang S (2017) Svdnet for pedestrian retrieval. In: 2017 IEEE International Conference on computer vision (ICCV), vol 1, pp 3820–3828
27. Sun Y, Zheng L, Yang Y, Tian Q, Wang S (2018) Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline). In: Proceedings of the European Conference on computer vision (ECCV), pp 501–518
28. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: 2015 IEEE Conference on computer vision and pattern recognition (CVPR), vol 1, pp 1–9
29. Taigman Y, Polyak A, Wolf L (2017) Unsupervised Cross-Domain Image Generation. arXiv preprint arXiv:1611.02200
30. Wang, D., Zhang, S.: Unsupervised person re-identification via multi-label classification. In: 2020 IEEE/CVF Conference on computer vision and pattern recognition (CVPR), vol 1, pp 10978–10987
31. Wang J, Zhu X, Gong S, Li W (2018) Transferable joint attribute-identity deep learning for unsupervised person re-identification. In: 2018 IEEE/CVF Conference on computer vision and pattern recognition, vol 1, pp 2275–2284
32. Wei L, Zhang S, Gao W, Tian Q (2018) Person transfer gan to bridge domain gap for person re-identification. In: 2018 IEEE/CVF Conference on computer vision and pattern recognition, vol 1, pp 79–88
33. Wu A, Zheng W, Lai J (2019) Unsupervised person re-identification by camera-aware similarity consistency learning. In: 2019 IEEE/CVF International Conference on computer vision (ICCV), vol 1, pp 6921–6930
34. Wu Y, Lin Y, Dong X, Yan Y, Bian W, Yang Y (2019) Progressive learning for person re-identification with one example. IEEE Trans Image Process 28:2872–2881
35. Wu Y, Lin Y, Dong X, Yan Y, Ouyang W, Yang Y (2018) Exploit the unknown gradually: one-shot video-based person re-identification by stepwise learning. In: 2018 IEEE/CVF Conference on computer vision and pattern recognition, vol 1, pp 5177–5186
36. Xiao T, Li S, Wang B, Lin L, Wang X (2017) Joint detection and identification feature learning for person search. In: 2017 IEEE Conference on computer vision and pattern recognition (CVPR), vol 1, pp 3376–3385
37. Yu HX, Zheng W, Wu A, Guo X, Gong S, Lai J (2019) Unsupervised person re-identification by soft multilabel learning. In: 2019 IEEE/CVF Conference on computer vision and pattern recognition (CVPR), vol 1, pp 2143–2152
38. Zheng L, Shen L, Tian L, Wang S, Wang J, Tian Q (2015) Scalable person re-identification: a benchmark. 2015 IEEE International Conference on computer vision (ICCV), vol 1, pp 1116–1124
39. Zheng L, Yang Y, Hauptmann AG (2016) Person re-identification: past, present and future. arXiv preprint arXiv:1610.02984
40. Zheng Z, Yang X, Yu Z, Zheng L, Yang Y, Kautz J (2019) Joint discriminative and generative learning for person re-identification. In: 2019 IEEE/CVF Conference on computer vision and pattern recognition (CVPR), vol 1, pp 2133–2142
41. Zheng Z, Zheng L, Yang Y (2017) Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In: 2017 IEEE International Conference on computer vision (ICCV), vol 1, pp 3774–3782
42. Zheng Z, Zheng L, Yang Y (2018) A discriminatively learned cnn embedding for person reidentification. ACM Trans Multimed Comput Commun Appl (TOMM) 14:1–20
43. Zhong Z, Zheng L, Cao D, Li S (2017) Re-ranking person re-identification with k-reciprocal encoding. In: 2017 IEEE Conference on computer vision and pattern recognition (CVPR), vol 1, pp 3652–3661
44. Zhong Z, Zheng L, Li S, Yang Y (2018) Generalizing a person retrieval model hetero- and homogeneously. In: Proceedings of the European Conference on computer vision (ECCV), pp 172–188
45. Zhong Z, Zheng L, Luo Z, Li S, Yang Y (2019) Invariance matters: exemplar memory for domain adaptive–person re-identification. In: 2019 IEEE/CVF Conference on computer vision and pattern recognition (CVPR), vol 1, pp 598–607
46. Zhong Z, Zheng L, Zheng Z, Li S, Yang Y (2019) Camstyle: a novel data augmentation method for person re-identification. IEEE Trans Image Process 28:1176–1190
47. Zhu JY, Park T, Isola P, Efros AA (2017) Unpaired image-to-image translation using cycle-consistent adversarial networks. In: 2017 IEEE International Conference on computer vision (ICCV), vol 1, pp 2242–2251