Deep Feature Learning with Relative Distance Comparison for Person Re-identification

Shengyong Ding , Liang Lin , Guangrun Wang , Hongyang Chao

Sun Yat-sen University, Guangzhou 510006, China

Abstract

Identifying the same individual across different scenes is an important yet difficult task in intelligent video surveillance. Its main difficulty lies in how to preserve similarity of the same person against large appearance and structure variation while discriminating different individuals. In this paper, we present a scalable distance driven feature learning framework based on the deep neural network for person re-identification, and demonstrate its effectiveness to handle the existing challenges. Specifically, given the training images with the class labels (person IDs), we first produce a large number of triplet units, each of which contains three images, i.e. one person with a matched reference and a mismatched reference. Treating the units as the input, we build the convolutional neural network to generate the layered representations, and follow with the $L2$ distance metric. By means of parameter optimization, our framework tends to maximize the relative distance between the matched pair and the mismatched pair for each triplet unit. Moreover, a nontrivial issue arising with the framework is that the triplet organization cubically enlarges the number of training triplets, as one image can be involved into several triplet units. To overcome this problem, we develop an effective triplet generation scheme and an optimized gradient descent algorithm, making the computational load mainly depends on the number of original images instead of the number of triplets. On several challenging databases, our approach achieves very promising results and outperforms other state-of-the-art approaches.

Keywords: Person Re-identification, Deep Learning, Distance Comparison

1. Introduction

Person re-identification, the aim of which is to match the same individual across multiple cameras, has attracted widespread attention in recent years due to its wide applications in video surveillance. It is the foundation of threat detection, behavioral understanding and other applications. Despite the considerable efforts of computer vision researchers, however, it is still an unsolved problem due to the dramatic variations caused by light, viewpoint and pose changes [1]. Figure 1 shows some typical examples from two cameras.

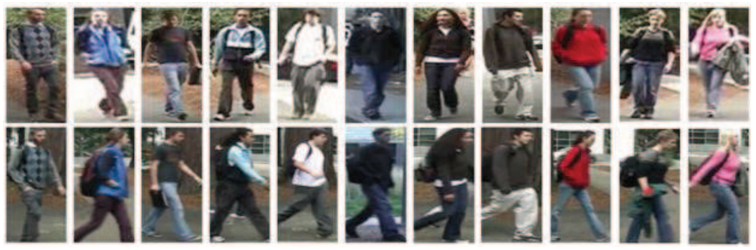

Figure 1: Typical examples of pedestrians shot by different cameras. Each column corresponds to one person. Huge variations exist due to the light, pose and view point changes.

There are two crucial components, i.e. feature representations and distance metric in person re-identification systems. In these two components, feature representation is more fundamental because it is the foundation of distance learning. The features used in person re-identification range from the color histogram [2], spatial cooccurrence representation model [3], attributes model [4] to combination of multiple features [2, 5]. These handcrafted features can hardly be optimal in practice because of the different viewing conditions that prevail [6]. Given a particular feature representation, a distance function is learned to construct a similarity measure [7, 8] with good similarity constraints . Although the effectiveness of the distance function has been demonstrated, it heavily relies on the quality of the features selected, and such selection requires deep domain knowledge and expertise [1].

In this paper, we present a scalable distance driven feature leaning framework via the convolutional network to learn representations for the person re-identification problem. Unlike the traditional deep feature learning methods aimed at minimizing the classification error, in our framework, features are learned to maximize the relative distance. More specifically, we train the network through a set of triplets. Each triplet contains three images, i.e. a query image, one matched reference (an image of the same person as that in the query image) and one mismatched reference. The network produces features with which the L_2 distance between the matched pair and the mismatched pair should be as large as possible for each triplet. This encourages the distances between matched pairs to take smaller values than those between the mismatched pairs. Figure 2 illustrates the overall principles. As discussed in [9], the triplet-based model is a natural model for the person re-identification problem for two main reasons. First, the intra-class and inter-class variation can vary significantly for different classes, and it may thus be inappropriate to require the distance between a matched pair or mismatched pair to fall within an absolute range. Second, person re-identification training images are relatively scarce, and the triplet-based training model can generate more constraints for distance learning, thereby helping to alleviate the over-fitting problem.

Similar to traditional neural networks, our triplet-based model also uses gradient descent algorithms in solving the parameters. Owing to limitations in memory size, it is impossible to load all the triplets for a given labeled image set into the memory to calculate the gradient. A practical means is to train the network iteratively in mini-batches, that is, in each iteration, a subset of the triplets are generated and the network parameters are then updated with the gradient derived from that batch. However, as we will see in the later sections, randomly generating the triplets at each iteration is inefficient as only a small number of distance constraints are imposed on the images within the triplets. Therefore we propose a more efficient triplet generation scheme. In each iteration, we randomly select a small number of classes (persons) from the dataset and generate the triplets using only those images, which guarantees that

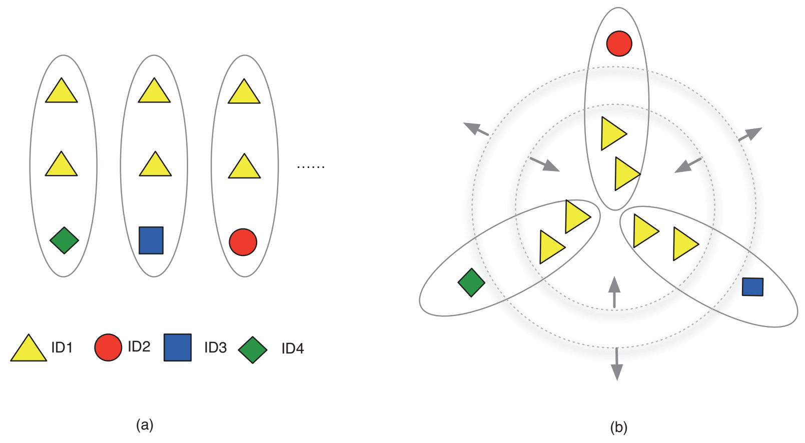

Figure 2: Illustration of deep feature learning via relative distance maximization. The network is trained by a set of triplets to produce effective feature representations with which the true matched images are closer than the mismatched images.

only a small number of images are selected in each iteration and rich distance constraints are imposed. In our proposed triplet generation scheme, one image can occur in several triplets in each iteration with a high degree of probability, and we thus design an extended network propagation algorithm to avoid recalculating the gradients of the same images. Our triplet generation scheme and the extended network propagation algorithm render the overall computational load of our model dependent mainly on the number of the training images, not on the number of triplets. Our approach also enables us to use the existing deep learning implementations to solve our model parameters with only slight modifications. In summary, we make two contributions to the literature:

1) A scalable deep feature learning method for person re-identification via maximum relative distance.

2) An effective learning algorithm for which the training cost mainly depends on the number of images rather than the number of triplets.

The remainder of this paper is organized as follows. In section two, we review the related work on person re-identification problems. In section three, we present our formulation and network architecture. In section four, we derive the algorithms for solving the model parameters using gradient descent methods for a small triplet set. In section five, we show how to train the network in batch mode with an efficient triplet generation scheme, and in section six, we present our experimental results. Section seven concludes our work.

2. Related Work

Feature representation and distance metric are the two main components of person re-identification systems. The existing approaches to person re-identification tasks primarily make use of handcrafted features such as color and texture histograms [2, 3, 10]. To increase their representative capability, features have been designed to carry spatial information [2, 5]. For example, Farezena et al. utilized the symmetry property of pedestrian images to propose a method called Symmetry Driven Accumulation of Local Features (SDALF) which is robust to background clutter [5]. The body configuration-based pictorial structure features have been also well studied to cope with individual variations [11, 12].

In addition to handcrafted feature designs, some studies addressed learning features for person re-identification tasks. For example, Gray and Tao [2] proposed the use of Adaboost to learn effective representations from an ensemble of local features. Zhao et al. [13] proposed the learning of mid-level features from hierarchical clusters of patches.

Another important research direction in person re-identification is distance learning. Zheng et al. [9] formulated the distance learning as a Probabilistic Relative Distance Comparison model (PRDC) to maximize the likelihood that correctly matched pairs will have a smaller distance between them than incorrectly matched pairs. In addition, Mignon and Jurie proposed Pairwise Constrained Component Analysis (PCCA) to project the original data into a lower dimensional space [14], in which the distance between pairs has the desired properties. Li et al. introduced a locally adaptive thresholding rule to metric learning models (LADF), and reported that it achieved good performance on person re-identification tasks [8]. RankSVM has also been proposed for learning a subspace in which the matched images have a higher rank than the mismatched images for a given query. There are also a number of general distance learning methods that have been rarely exploited in the context of person re-identification problems [15, 16, 17, 18].

Inspired by the success of deep learning, there are also some literatures applying neural network models to address the person re-identification problems. Dong Yi et al. [19] applied a deep neural network to learn pair-wise similarity and achieved state-of-the-art performance. Hao Liu et al. [20] presented a Set-Label Model, which applies DBN (Deep Belief Network) and NCA (Neighborhood Component Analysis) on the proposed concatenated features of the query image and the gallery image to improve the person re-identification performance. Xu et al. [12] adopted a cluster sampling algorithm [21] for re-identifying persons with templates. Li et al. [22] proposed a deep learning framework for learning filter pairs that tries to automatically encode the photometric transforms across cameras. Our work differs from these methods in its loss function and learning algorithm.

The model most similar to that proposed herein was introduced by Wang et al. [23] and involved learning features for fine-grained image retrieval. They borrowed the network architecture designed by Krizhevsky et al. [24], and pre-trained the network using soft-max loss function. It is unclear whether the triplet-based deep model can be effectively trained from triplets without other pre-training techniques. Here, we extend the triplet-based model to the person re-identification problem with an efficient learning algorithm and triplet generation scheme. We demonstrate its effectiveness without pre-training techniques using a relatively simple network .

3. Model

3.1. Formulation

Our objective is to use a deep convolutional network to learn effective feature representations that can satisfy the relative distance relationship under the L_2 distance. In other words, we apply a deep convolutional network to produce the feature for each image. And with these generated features, we require the distances between matched pairs should be smaller than those between mismatched pairs as depicted in Figure 3.

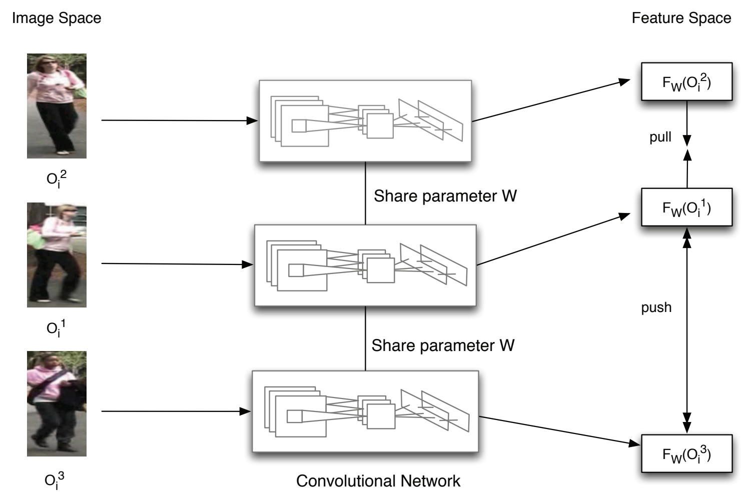

Figure 3: Illustration of maximizing the distance for person re-identification. The L_2 distance in the feature space between the matched pair should be smaller than the mismatched pair in each triplet.

In our model, the relative distance relationship is reflected by a set of triplet units $\{O_i\}$ where $O_i =< O_i^1, O_i^2, O_i^3 >$, in which O_i^1 and O_i^2 are a matched pair and O_i^1 and O_i^3 are a mismatched pair. Let $W = \{W_j\}$ denote the network parameters and $F_W(I)$ denote the network output of image I, i.e. feature representation for image I. For a training triplet O_i, the desired feature should satisfy the following condition under the L_2 norm.

$$||F_W(O_i^1) - F_W(O_i^2)|| < ||F_W(O_i^1) - F_W(O_i^3)|| \tag{1}$$

or equally:

$$||F_W(O_i^1) - F_W(O_i^2)||^2 < ||F_W(O_i^1) - F_W(O_i^3)||^2 \tag{2}$$

Here, we use the squared form to facilitate the partial derivative calculation. For a given training set $O=\{O_i\}$, the relative distance constraints are converted to the minimization problem of the following objective, i.e. maximizing the distance between matched pairs and mismatched pairs, where n is the number of the training triplets.

$$f(W, O) = \Sigma_{i=1}^{n} \max\{||F_W(O_i^1) - F_W(O_i^2)||^2 - ||F_W(O_i^1) - F_W(O_i^3)||^2, C\} \tag{3}$$

The role of the max operation with the constant C is to prevent the overall value of the objective function from being dominated by easily identifiable triplets, which is similar to the technique widely used in hinge-loss functions. We set C=-1 throughout the paper.

Note the network in our model still takes one image as input both for training and testing as the conventional convolutional network does. The triplet-based loss function is introduced for parameter optimization in the training stage. During the testing, we feed each testing image to the trained network to get its feature and use these features for performance evaluation under the normal L_2 distance.

3.2. Network Architecture

All existing person re-identification datasets are relatively small, and we thus designed a simplified network architecture for our model. Figure 4 shows the overall network architecture, which comprises five layers. The first and third layers are convolutional layers and the second and fourth layers are pooling layers. The first convolutional layer includes 32 kernels of size 5×5×3 with a stride of 2 pixels. The second convolutional layer takes the pooled output of the first convolutional layer as input and filters it with 32 kernels of size 5×5×32 with a stride of 1 pixel. The final 400 dimensional layer is fully connected to the pooled output of the second convolutional layer with the following normalization:

Let $\{x_i\}$ denote the output before normalization, with the normalized output then calculated by:

$$y_i = \frac{x_i}{\sqrt{\Sigma x_i^2}} \tag{4}$$

Note that this normalization differs from the normalization scheme applied by Krizhevsky et al. [24] over different channels. Our normalization ensures that the distance derived from each triplet cannot easily exceeds the margin C so that more triplet constraints can take effect for the whole objective function. Accordingly, the back propagation process accounts for the normalization operation using the chain rule during calculation of the partial derivative.

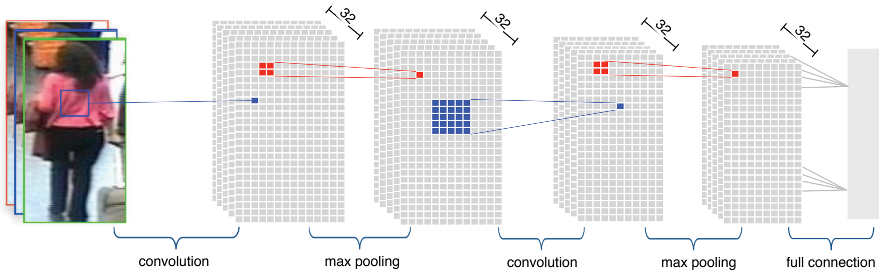

Figure 4: An illustration of the network architecture. The first and third layers are convolutional layers and the second and fourth layers are max pooling layers. The final layer is a full connection layer.

We use overlapped max pooling for the pooling operation. More precisely, the pooling operation can be thought of as comprising a grid of pooling units spaced s pixels apart, with each summarizing a neighborhood of size $z \times z$ centered at the location of the pooling unit. We set s=1 and z=2 in our network. For the neuron activation functions, we use Rectified Linear Units to transform the neuron inputs, thereby speeding up the learning process and achieving good performance, as discussed in [24].

4. Learning Algorithm

In this section, we show how to solve the network given a fixed set of training triplets. We assume the memory is sufficiently large to load all of the triplets. The procedures for generating triplets from labeled images and training the network using the batch mode is relegated to the next section.

4.1. Triplet-based gradient descent algorithm

We first present a direct learning algorithm derived from the definition of the objective function. For ease of exposition, we introduce $d(W, O_i)$, which denotes the difference in distance between the matched pair and the mismatched pair in the triplet O_i.

$$d(W, O_i) = ||F_W(O_i^1) - F_W(O_i^2)||^2 - ||F_W(O_i^1) - F_W(O_i^3)||^2 \tag{5}$$

and the objective function can be rewritten as,

$$f(W, O) = \Sigma_{O_i} \max\{d(W, O_i), C\} \tag{6}$$

Then the partial derivative of the objective becomes

$$\frac{\partial f}{\partial W_j} = \Sigma_{O_i} h(O_i) \tag{7}$$

$$h(O_i) = \begin{cases} \frac{\partial d(W,O_i)}{\partial W_j}, & \text{if } d(W, O_i) > C; \\ 0, & \text{if } d(W, O_i) <= C; \end{cases} \tag{8}$$

By the definition of $d(W, O_i)$, we can obtain the gradient of $d(W, O_i)$ as follows:

$$\begin{aligned} \frac{\partial d(W, O_i)}{\partial W_j} &= 2(F_W(O_i^1) - F_W(O_i^2))' \cdot \frac{\partial F_W(O_i^1) - \partial F_W(O_i^2)}{\partial W_j} \\ &\quad -2(F_W(O_i^1) - F_W(O_i^3))' \cdot \frac{\partial F_W(O_i^1) - \partial F_W(O_i^3)}{\partial W_j} \end{aligned} \tag{9}$$

We can now see that the gradient on each triplet can be easily calculated given the values of $F_W(O_i^1), F_W(O_i^2), F_W(O_i^3)$ and $\frac{\partial F_W(O_i^1)}{\partial W_j}, \frac{\partial F_W(O_i^2)}{\partial W_j}, \frac{\partial F_W(O_i^3)}{\partial W_j}$, which can be obtained by separately running the standard forward and backward propagation for each image in the triplet. As the algorithm needs to go through all of the triplets to accumulate the gradients for each iteration, we call it the triplet-based gradient descent algorithm. Algorithm 1 shows the overall process.

Algorithm 1: Triplet-based gradient descent algorithm

```
Input:
    Training samples $\{O_i\}$;
Output:
    The network parameters $\{W_j\}$
1: while $t < T$ do
2:   $t \leftarrow t + 1$;
3:   $\frac{\partial f}{\partial W_j} = 0$
4:   for all training triplet $O_i$ do
5:     Calculate $F_W(O_i^1)$,$F_W(O_i^2)$,$F_W(O_i^3)$ by forward propagation;
6:     Calculate $\frac{\partial F_W(O_i^1)}{\partial W_j} \frac{\partial F_W(O_i^2)}{\partial W_j} \frac{\partial F_W(O_i^3)}{\partial W_j}$ by back propagation;
7:     Calculate $\frac{\partial d(W,O_i)}{\partial W_j}$ according to equation 9;
8:     Increment the gradient $\frac{\partial f}{\partial W_j}$ according to equation $7, 8$;
9:   end for
10:  $W_j^t = W_j^{t-1} - \lambda_t \frac{\partial f}{\partial W_j}$;
11: end while
```

4.2. Image-based gradient descent algorithm

In the triplet-based gradient descent algorithm, the number of network propagations depends on the number of training triplets in each iteration, with each triplet involving three rounds of forward and backward propagation during the calculation of the gradient. However, if the same image occurs in different triplets, the forward and backward propagation of that image can be reused. Recognition of this potential shortcut inspired us to look for an optimized algorithm in which the network propagation executions depend only on the number of distinct images in the triplets. Before considering that algorithm, we first review the way in which the standard propagation algorithm is deduced in the conventional CNN learning algorithm, where the objective function often takes the following form. Here n is the number of training images.

$$f(I_1, I_2, ..., I_n) = \frac{1}{n}\Sigma_{i=1}^{n} loss(F_W(I_i)) \tag{10}$$

As the objective function is defined as the sum of the loss function on each image I_i, we have:

$$\frac{\partial f}{\partial W} = \frac{1}{n} \Sigma_{i=1}^{n} \frac{\partial loss(F_W(I_i))}{\partial W} \tag{11}$$

This shows that we can calculate the gradient of the loss function for each image separately and then sum these image-based gradients to obtain the overall gradient of the objective function. In the case of a single image, the gradient can be calculated recursively by the chain rule, which is given as follows.

$$\frac{\partial loss(F_W(I_i))}{\partial W^l} = \frac{\partial loss(F_W(I_i))}{\partial X_i^l} \frac{\partial X_i^l}{\partial W^l} \tag{12}$$

$$\frac{\partial loss(F_W(I_i))}{\partial X_i^l} = \frac{\partial loss(F_W(I_i))}{\partial X_i^{l+1}} \frac{\partial X_i^{l+1}}{\partial X_i^l} \tag{13}$$

In the above equations, W^l represents the network parameters at the l^{th} layer and X_i^l represents the feature maps of the image I_i at the same layer. The Equation 12 holds because X_i^l depends on the parameter W^l and the Equation 13 holds because the feature maps at the $(l+1)^{th}$ layer depend on those at the l^{th} layer. As the partial derivative of the loss function with respect to the output feature can be simply calculated according to the loss function definition, the gradient on each image can be calculated recursively. Simple summation of the image gradients produces the overall gradient of the objective function.

We now turn to the triplet-based objective function and show that the overall gradient can also be obtained from the image-based gradients, which can be calculated separately. The difficulty lies in the impossibility of writing the objective function directly as the sum of the loss functions on the images, as in Equation 10, because it takes the following form, where n is the number of triplets:

$$f = \Sigma_{i=1}^{n} loss(F_W(O_i^1), F_W(O_i^2), F_W(O_i^3)) \tag{14}$$

However, because the loss function for each triplet is still defined on the outputs of the images in each triplet, this objective function can also be seen as follows, where $\{I'_k\}$ represents the set of all the distinct images in the triplets, i.e. $\{I'_k\} =$

$\{O_i^1\}\bigcup\{O_i^2\}\bigcup\{O_i^3\}$ and m is the number of the images in the triplets.

$$f = f(F_W(I_1'), F_W(I_2'), ..., F_W(I_m')) \tag{15}$$

As $F_W(I_k')$ is some function of the feature map X_k^l at the l^{th} layer, the objective function can also be seen as follows:

$$f = f(X_1^l, X_2^l, ..., X_m^l) \tag{16}$$

Then the derivative rule gives us the following equations with X_k^l depending on W^l and X_k^{l+1} depending on X_k^l.

$$\frac{\partial f}{\partial W^l} = \Sigma_{k=1}^m \frac{\partial f}{\partial X_k^l}\frac{\partial X_k^l}{\partial W^l} \tag{17}$$

$$\frac{\partial f}{\partial X_k^l} = \frac{\partial f}{\partial X_k^{l+1}}\frac{\partial X_k^{l+1}}{\partial X_k^l} \tag{18}$$

The first equation shows the gradient of the loss function with respect to the network parameters takes image-based form (summation over images) and tells us how to get this gradient given $\frac{\partial f}{\partial X_k^l}$ for all k. Actually, $\frac{\partial f}{\partial W^l}$ can be obtained by $\Sigma\alpha_k\frac{\partial f}{\partial X_k^l}$ with $\alpha_k = \frac{\partial X_k^l}{\partial W^l}$ whose computation only relies on image I_k'. If we get $\frac{\partial f}{\partial W^l}$ for all the layers, then we get the overall gradient of the triplet-based loss function, i.e. $\Delta W = \frac{\partial f}{\partial W}$.

The second equation tells us how to get the partial derivative of the loss function with respect to the feature map of each image I_k' at the l^{th} layer, i.e. $\frac{\partial f}{\partial X_k^l}$ recursively. More precisely, if we have known the partial derivative with respect to the feature maps of the upper layer, say the $(l+1)^{th}$ layer, then the derivative with respect to this layer can be derived by simply multiplying a matrix $\frac{\partial X_k^{l+1}}{\partial X_k^l}$ which can also be calculated for each image I_k' separately.

So if we get the partial derivative of the loss function with respect to the output (feature map of the top layer) of each image, i.e. $\frac{\partial f}{\partial F_W(I_k')}$, we can get the gradient $\frac{\partial f}{\partial W}$ by applying Equation 18 and Equation 17 recursively (standard backward propagation). Luckily, the derivative with respect to the output of each image can be easily obtained as follows since it is defined analytically on

$\{F_W(I'_k)\}$.

$$\frac{\partial f}{\partial F_W(I'_k)} = \Sigma_{i=1}^{n} \frac{\partial \max\{||F_W(O_i^1) - F_W(O_i^2)||^2 - ||F_W(O_i^1) - F_W(O_i^3)||^2, C\}}{\partial F_W(I'_k)} \quad (19)$$

Algorithm 3 provides the details of calculating $\frac{\partial f}{\partial F_W(I'_k)}$. As the algorithm shows, we need to collect the derivative from each triplet. If the triplet contains the target image I'_k and the distance $d(W, O_i)$ is greater than the constant C (implementing the max operation in equation 3), then this triplet contributes its derivative with respect to $F_W(I'_k)$. The form of this derivative depends on the position where the image I'_k appears in the triplet O_i as listed in the algorithm. Otherwise, this triplet will be simply passed. With this image-based gradient calculation method, the whole training process is given in Algorithm 2. It is not hard to see that our optimized learning algorithm is very similar to the traditional neural network algorithm except that calculating the partial derivative with respect to the output of one image for the triplet-based loss function relies on the outputs of other images while the traditional loss function does not. This optimized learning algorithm has two obvious merits:

1. We can apply a recent deep learning implementation framework such as Caffe [25] simply by modifying the loss layer.

2. The number of network propagation executions can be reduced to the number of distinct images in the triplets, a crucial advantage for large scale datasets.

5. Batch Learning and Triplet Generation

Suppose that we have a labelled dataset with M classes (persons) and that each class has N images. The number of possible triplets would be $M(M-1)N^2(N-1)$. It would be impossible to load all of these triplets into the memory to train the network even for a moderate dataset. It is thus necessary to train the network using the batch mode, which allows it to be trained iteratively. In each iteration, only a small part of triplets are selected from all the possible triplets, and these triplets are used to calculate the gradient and then

Algorithm 2: Image-based gradient descent algorithm

```
Input:
    Training triplets $\{O_i\}$;
Output:
    The network parameters $W$;
1: Collect all the distinct images $\{I'_k\}$ in $\{O_i\}$;
2: while $t < T$ do
3:     $t \leftarrow t + 1$;
4:     $\frac{\partial f}{\partial W} = 0$;
5:     Calculate the outputs for each image $I'_k$ by forward propagation;
6:     for all $I'_k$ do
7:         Calculate $\frac{\partial f}{\partial F_W(I'_k)}$ for image $I'_k$ according to Algorithm 3;
8:         Calculate $\frac{\partial f}{\partial W}(I'_k)$ using back propagation;
9:         Increment the partial derivative: $\frac{\partial f}{\partial W} += \frac{\partial f}{\partial W}(I'_k)$;
10:    end for
11:    $W^t = W^{t-1} - \lambda_t \frac{\partial f}{\partial W}$;
12: end while
```

to update the network parameters. There are several ways to select triplets from the full population of triplets. The simplest method is to select them randomly. However, in random selection, the distinct image size is approximately three times of the selected triplet size because each triplet contains three images, and the likelihood of two triplets sharing the same image is very low. This triplet generation approach is very inefficient because only a few distance constraints are placed on the selected images in each iteration. Instead, according to our optimized gradient derivation, we know that the number of network propagations depends on the number of images contained in the triplets. So we should produce more triplets to train the model with the same number of images in each iteration. This leads to our following triplet generation scheme. In each iteration, we select a fixed number of classes (persons), and for each image in each

Algorithm 3: Partial derivative with respect to the output of image I'_k

```
Input:
    Training triplets $\{O_i\}$, image $I'_k$;
Output:
    The partial derivative: $\frac{\partial f}{\partial F_W(I'_k)}$;
1: $\frac{\partial f}{\partial F_W(I'_k)} = 0$;
2: for all $O_i =< O_i^1, O_i^2, O_i^3 >$ do
3:   if $d(W, O_i) > C$ then
4:     if $I'_k = O_i^1$ then
5:       $\frac{\partial f}{\partial F_W(I'_k)} + = 2(F_W(O_i^3) - F_W(O_i^2))$;
6:     else if $I'_k = O_i^2$ then
7:       $\frac{\partial f}{\partial F_W(I'_k)} - = 2(F_W(O_i^1) - F_W(O_i^2))$;
8:     else if $I'_k = O_i^3$ then
9:       $\frac{\partial f}{\partial F_W(I'_k)} + = 2(F_W(O_i^1) - F_W(O_i^3))$;
10:    end if
11:  end if
12: end for
```

class, we randomly construct a large number of triplets, in which the matched references are randomly selected from the same class and the mismatched references are randomly selected from the remaining selected classes. This policy ensures large amounts of distance constraints are posed on a small number of images, which can be loaded into the limited memory in each iteration. And with the increasing number of iterations are executed, the sampled triplets still can cover all the possible triplet pattern, ensuring the model to converge to a local minimum.

As a comparison, suppose the memory can only load 300 images (a typical case for 2G GPU memory device). Then in the random triplet generation scheme, only about 100 triplets can be applied to train the model in one iteration. However, our proposed scheme can use thousands of triplets to train the model without obvious computation load increase. Algorithm 4 gives the complete batch training process. As described in the ablation study section, our proposed triplet generation scheme shows obvious advantages both in convergence time and matching rate.

Algorithm 4: Learning deep features from relative distance comparison in the batch mode

```
Input:
    Labelled training images $\{I_i\}$;
Output:
    Network Parameters $W$;
1: while $t < T$ do
2:     $t \leftarrow t + 1$;
3:     Randomly select a subset of classes (persons) from the training set;
4:     Collect images from the selected classes: $\{I'_k\}$ ;
5:     Construct a set of triplets from the selected classes;
6:     $\Delta W = 0$;
7:     for all $I'_k$ do
8:         Run forward propagation for $I'_k$
9:         Calculate the partial derivative of the loss function with respect to
           $F_W(I'_k)$ according to Algorithm 3;
10:        Run the standard backward propagation for $I'_k$;
11:        Accumulate the gradient: $\Delta W + = \Delta W(I'_k)$;
12:    end for
13:    $W^t = W^{t-1} - \lambda_t \Delta W$;
14: end while
```

6. Experiments

6.1. Datasets and Evaluation Protocol

We used two well-known and challenging datasets, i.e., iLIDS [26] and VIPeR [2], for our experiments. Both datasets contain a set of persons, each of whom has several images captured by different cameras. All the images were resized to 250 × 100 pixels to train our network.

iLIDS dataset The iLIDS dataset [26] was constructed from video images captured in a busy airport arrival hall. It features 119 pedestrians, with 479 images normalized to 128 × 64 pixels. The images come from non-overlapping cameras, and were subject to quite large illumination changes and occlusions. On average, there are four images of each individual pedestrian.

VIPeR dataset The VIPeR dataset [2] contains two views of 632 pedestrians. The pair of images of each pedestrian was captured by different cameras under different viewpoint, pose and light conditions. It is the most challenging dataset in the person re-identification arena owing to the huge variance and discrepancy.

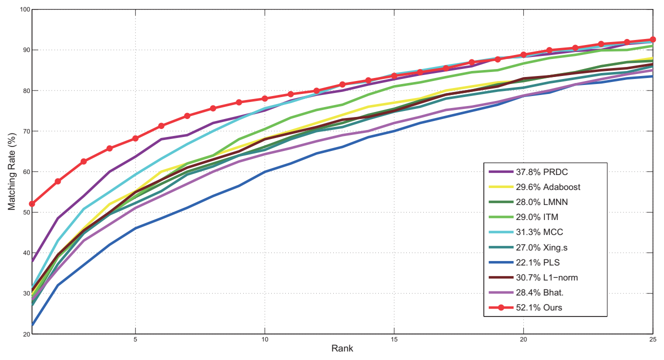

Figure 5: Performance comparison using CMC curves on i-LIDS dataset.

Evaluation Protocol We adopted the widely used cumulative match curve (CMC) approach [27] for quantitive evaluation. We randomly selected about half of the persons for training (69 for iLIDS and 316 for VIPeR), with the

Method	Top1	Top5	Top10	Top15	Top20	Top30
Ours	**52.1**	**68.2**	**78.0**	**83.6**	**88.8**	**95.0**
Adaboost	29.6	55.2	68.1	77.0	82.4	92.1
LMNN	28.0	53.8	66.1	75.5	82.3	91.0
ITML	29.0	54.0	70.5	81.0	86.7	95.0
MCC	31.3	59.3	75.6	84.0	88.3	95.0
Xing's	27.0	52.3	63.4	74.8	80.7	93.0
PLS	22.1	46.0	60.0	70.0	78.7	87.5
L1-norm	30.7	55.0	68.0	75.0	83.0	90.0
Bhat.	28.4	51.1	64.3	72.0	78.8	89.0
PRDC	37.8	63.7	75.1	82.8	88.4	95.0

Table 1: Performance of different models on i-LIDS dataset.

remainder used for testing. To obtain the CMC values, we divided the testing set into a gallery set and a probe set, with no overlap between them. The gallery set comprised one image for each person. For each image in the probe set, we returned the n nearest images in the gallery set using the $L2$ distance with the features produced by the trained network. If the returned list contained an image featuring the same person as that in the query image, this query was considered as success of rank n. We repeated the procedure 10 times, and used the average rate as the metric.

6.2. Performance Comparison

Training Setting The weights of the filters and the full connection parameters were initialized from two zero-mean Gaussian distributions with standard deviation 0.01 and 0.001 respectively. The bias terms were set with the constant 0. We generated the triplets as follows. In each iteration, we selected 40 persons and generate 80 triplets for each person. When there were less than 10 triplets whose distance constraints could not be satisfied, i.e. the distance between the matched pair is larger than the distance between the mismatched pair, the learning process was taken as converged.

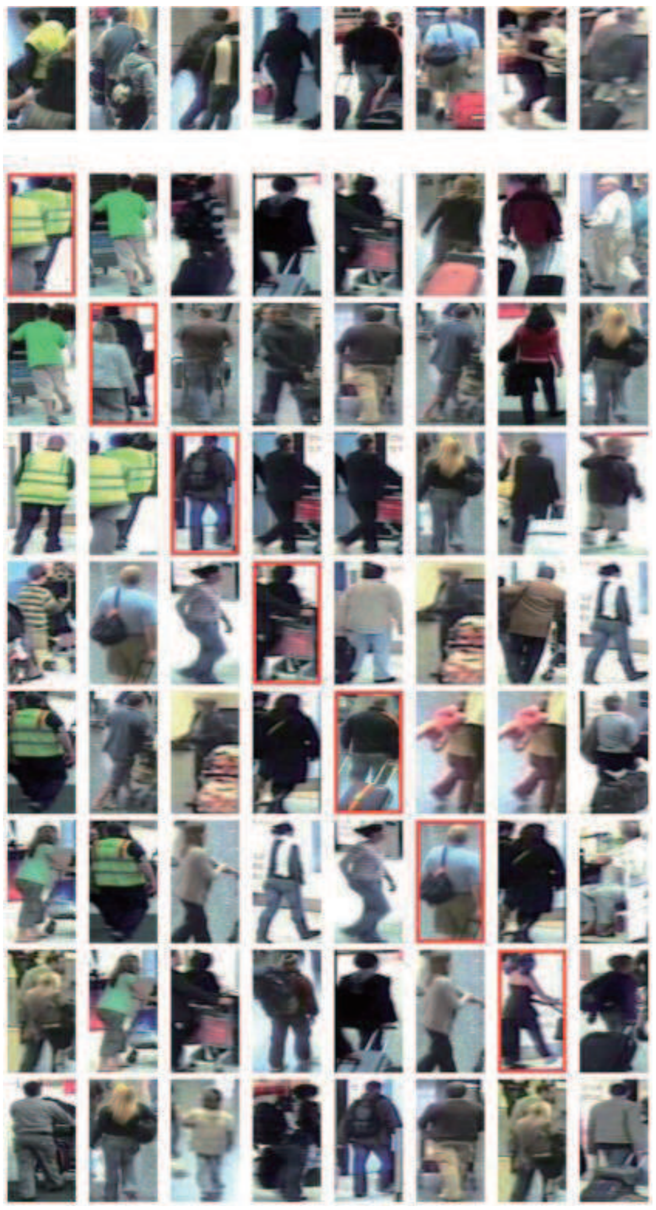

Figure 6: Search examples on iLIDS dataset. Each column represents a ranking result with the top image being the query and the rest images being the returned list. The image with the red bounding box is the matched one.

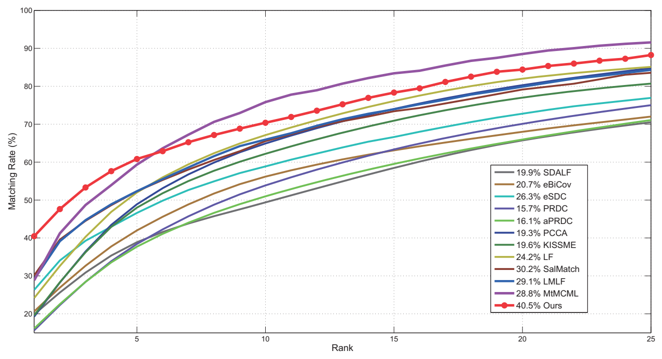

Figure 7: Performance comparison using CMC curves on VIPeR dataset.

Comparison on iLIDS dataset Using the iLIDS dataset, we compared our method with PRDC [9] and other metric learning methods (i.e. Adaboost[2], Xing's [15], LMNN [16], ITML [17], PLS [28], Bhat. [2], L1-norm [3] and MCC [29]). The features were an ensemble of color histograms and texture histograms, as described in [9]. We used 69 persons for training and the rest for testing (the same setting as used by the compared methods). Figure 5 shows the curves of the various models, and Table 1 shows the top 1 , top 5, top 10, top 15, top 20 and top 30 performance. Our method achieved rank-1 accuracy 52.1%, which clearly outperformed the other methods. Figure 6 shows several query examples for the iLIDS dataset. In this figure, each column represents a ranking result with the top image being the query image. The matched one in the returned list is marked by a red bounding box.

Comparison on VIPeR dataset Using the VIPeR dataset, we compared our method with such state-of-the-art methods as MtMCML [30], LMLF [13], SDALF [5], eBiCov [31], eSDC [32], PRDC [9], aPRDC [33], PCCA [14], KISSME [34], LF [35] and SalMatch [36]. Half of the persons were used for training, and the rest for testing (the same setting as used by the compared methods). Figure 7 presents the CMC curves of the various models, and Table 2 presents the top 1 , top 5, top 10, top 15, top 20 and top 30 ranking results. Our method achieved rank-1 accuracy 40.5% that clearly outperformed

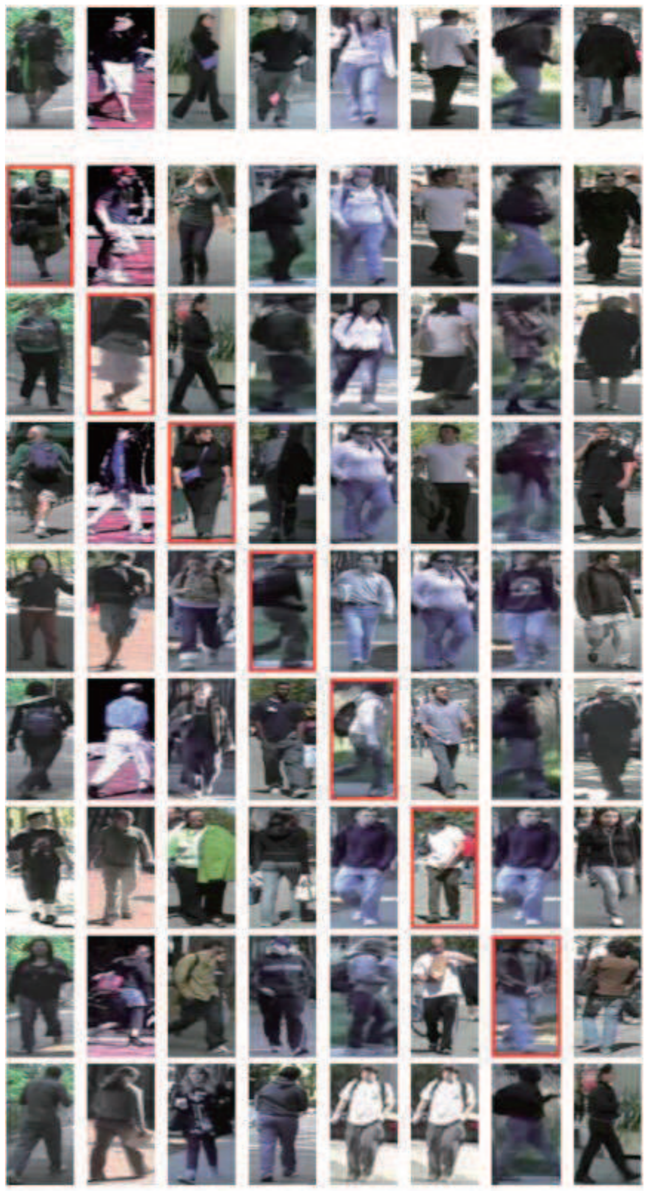

Figure 8: Search examples on VIPeR dataset. Each column represents a ranking result with the top image being the query and the rest images being the returned list. The image with the red bounding box is the matched one.

Method	Top1	Top5	Top10	Top15	Top20	Top30
Ours	**40.5**	**60.8**	70.4	78.3	84.4	90.9
MtMCML	28.8	59.3	**75.8**	**83.4**	**88.5**	**93.5**
SDALF	19.9	38.4	49.4	58.5	66.0	74.4
eBiCov	20.7	42.0	56.2	63.3	68.0	76.0
eSDC	26.3	46.4	58.6	66.6	72.8	80.5
PRDC	15.7	38.4	53.9	63.3	70.1	78.5
aPRDC	16.1	37.7	51.0	59.5	66.0	75.0
PCCA	19.3	48.9	64.9	73.9	80.3	87.2
KISSME	19.6	48.0	62.2	70.9	77.0	83.7
LF	24.2	52.3	67.1	76.1	82.2	87.9
SalMatch	30.2	52.3	66.0	73.4	79.2	86.0
LMLF	29.1	52.3	66.0	73.9	79.9	87.9

Table 2: Performance of different models on VIPeR dataset.

most available benchmarking methods. Figure 8 shows some query examples for the VIPeR dataset. Each column represents a ranking result with the top image being the query image and the rest being the result list. The matched one in the returned list is highlighted by a red bounding box. This figure shows the difficulty of this dataset. Actually, in the failed examples (rank 1 image does contain the same person as the query), the images ranked higher than the matched one often look more closer to the query image as in columns 2-7.

6.3. Ablation Studies of Learning

In this section, we explore the learning details on the VIPeR dataset, as it is more challenging and contains more images.

Data Augmentation Data augmentation is an important mechanism for alleviating the over-fitting problem. In our implementation, we crop a center region 230 $\times$ 80 in size with a small random perturbation for each image to augment the training data. Such augmentation is critical to the performance, particularly when the training dataset is small. In our experiment, the performance declined by 33 percent without it.

Normalization Normalization is a common approach in CNN networks [24], but these networks normalize the feature map over different channels. In our model, the output feature is normalized to 1 under the $L2$ norm. Without this normalization, the top 1 performance drops by 25 percent. Normalization also helps to reduce the convergence time. In our experiment, the learning process roughly converged in four 4,000 iterations with normalization and in roughly 7,000 without it.

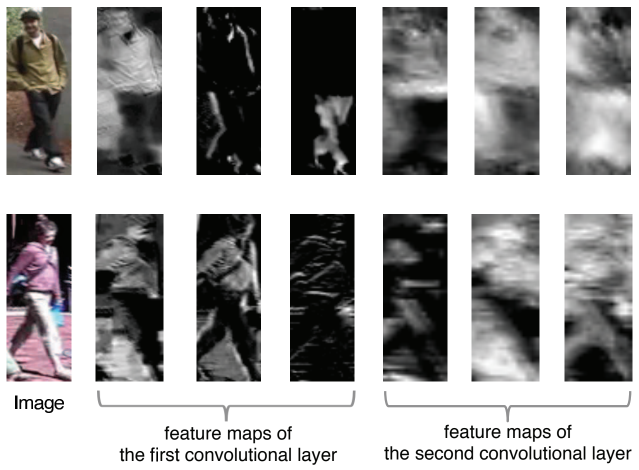
Image
feature maps of the first convolutional layer
feature maps of the second convolutional layer

Figure 9: Visualization of feature maps generated by our approach.

Triplet Generation The triplet generation scheme affects the convergence time and matching rate, as pointed out in previous sections. We compared the model's performance under two triplet generation schemes. In the first scheme, we selected 40 persons in each iteration, and constructed 80 triplets for each person using the images of those 40 persons. In the second scheme, we again selected 40 persons in each iteration, but constructed only one triplet for each person (approximating random selection). The first scheme achieved its best performance in about 4,000 iterations while the second scheme achieved its best performance (90 percent matching rate of the first scheme) in 20,000 iterations. However, the training time in each iteration for these two schemes is almost the same as we expected.

Implementation Detail We implemented our model based on the Caffe framework [24], with only the data layer and loss layer replaced. We trained the network on a GTX 780 GPU with 2G memory. When there were fewer than 10 triplets whose distance constraints had been violated, the model was taken as converged. Our model usually converged in less than one hour thanks to its simplified network architecture and effective triplet generation scheme.

Feature map visualization In addition, we visualize the intermediate features generated by our model to validate the effectiveness of representation learning. Figure 9 shows two examples, where we present some feature maps of the first and the second convolutional layers, respectively. As we expect, the lower layer feature maps tend to have strong responses at the edges, showing some characteristics of low level features.

7. Conclusion

In this paper, we present a scalable deep feature learning model for person re-identification via relative distance comparison. In this model, we construct a CNN network that is trained by a set of triplets to produce features that can satisfy the relative distance constraints organized by that triplet set. To cope with the cubically growing number of triplets, we present an effective triplet generation scheme and an extended network propagation algorithm to efficiently train the network iteratively. Our learning algorithm ensures the overall computation load mainly depends on the number of training images rather than the number of triplets. The results of extensive experiments demonstrate the superior performance of our model compared with the state-of-the-art methods. In future research, we plan to extend our model to more datasets and tasks.

Acknowledgement

This work was supported by the National Natural Science Foundation of China (No. 61173082 and No.61173081), Guangdong Science and Technology Program (No. 2012B031500006), Guangdong Natural Science Foundation (No. S2013050014548), Special Project on Integration of Industry, Education and Research of Guangdong Province (No. 2012B091000101), and Program of Guangzhou Zhujiang Star of Science and Technology (No. 2013J2200067). Corresponding authors of this work are L. Lin and H. Chao.

References

References

[1] L. Lin, T. Wu, J. Porway, Z. Xu, A stochastic graph grammar for compositional object representation and recognition, Pattern Recognition 42 (7) (2009) 1297–1307.

[2] D. Gray, H. Tao, Viewpoint invariant pedestrian recognition with an ensemble of localized features, in: ECCV, Springer, 2008, pp. 262–275.

[3] X. Wang, G. Doretto, T. Sebastian, J. Rittscher, P. Tu, Shape and appearance context modeling, in: ICCV, IEEE, 2007, pp. 1–8.

[4] R. Layne, T. M. Hospedales, S. Gong, Towards person identification and re-identification with attributes, in: ECCV, Springer, 2012, pp. 402–412.

[5] M. Farenzena, L. Bazzani, A. Perina, V. Murino, M. Cristani, Person re-identification by symmetry-driven accumulation of local features, in: CVPR, IEEE, 2010, pp. 2360–2367.

[6] L. Lin, R. Zhang, X. Duan, Adaptive scene category discovery with generative learning and compositional sampling, Circuits and Systems for Video Technology, IEEE Transactions on 25 (2) (2015) 251–260.

[7] W. Li, R. Zhao, X. Wang, Human reidentification with transferred metric learning, in: ACCV, Springer, 2013, pp. 31–44.

[8] Z. Li, S. Chang, F. Liang, T. S. Huang, L. Cao, J. R. Smith, Learning locally-adaptive decision functions for person verification, in: CVPR, IEEE, 2013, pp. 3610–3617.

[9] W.-S. Zheng, S. Gong, T. Xiang, Person re-identification by probabilistic relative distance comparison, in: CVPR, IEEE, 2011, pp. 649–656.

[10] L. Lin, P. Luo, X. Chen, K. Zeng, Representing and recognizing objects with massive local image patches, Pattern Recognition 45 (1) (2012) 231–240.

[11] L. Lin, X. Wang, W. Yang, J. Lai, Discriminatively trained and-or graph models for object shape detection, Pattern Analysis and Machine Intelligence, IEEE Transactions on 37 (5) (2015) 959–972.

[12] Y. Xu, L. Lin, W.-S. Zheng, X. Liu, Human re-identification by matching compositional template with cluster sampling, in: IEEE International Conference on Computer Vision (ICCV), IEEE, 2013, pp. 3152–3159.

[13] R. Zhao, W. Ouyang, X. Wang, Learning mid-level filters for person re-identification, in: CVPR, 2013, pp. 144–151.

[14] A. Mignon, F. Jurie, Pcca: A new approach for distance learning from sparse pairwise constraints, in: CVPR, IEEE, 2012, pp. 2666–2672.

[15] E. P. Xing, M. I. Jordan, S. Russell, A. Y. Ng, Distance metric learning with application to clustering with side-information, in: NIPS, 2002, pp. 505–512.

[16] K. Q. Weinberger, J. Blitzer, L. K. Saul, Distance metric learning for large margin nearest neighbor classification, in: NIPS, 2005, pp. 1473–1480.

[17] J. V. Davis, B. Kulis, P. Jain, S. Sra, I. S. Dhillon, Information-theoretic metric learning, in: ICML, ACM, 2007, pp. 209–216.

[18] S. Xiang, F. Nie, C. Zhang, Learning a mahalanobis distance metric for data clustering and classification, Pattern Recognition 41 (12) (2008) 3600–3612.

[19] D. Yi, Z. Lei, S. Z. Li, Deep metric learning for practical person re-identification, CoRR abs/1407.4979.

[20] H. Liu, B. Ma, L. Qin, J. Pang, C. Zhang, Q. Huang, Set-label modeling and deep metric learning on person re-identification, Neurocomputing 151 (2015) 1283–1292.

[21] L. Lin, X. Liu, S.-C. Zhu, Layered graph matching with composite cluster sampling, Pattern Analysis and Machine Intelligence, IEEE Transactions on 32 (8) (2010) 1426–1442.

[22] W. Li, R. Zhao, T. Xiao, X. Wang, Deepreid: Deep filter pairing neural network for person re-identification, in: CVPR, 2014, pp. 152–159.

[23] J. Wang, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, Y. Wu, et al., Learning fine-grained image similarity with deep ranking, in: CVPR, 2014.

[24] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: NIPS, 2012, pp. 1097–1105.

[25] Y. Jia, Caffe: An open source convolutional architecture for fast feature embedding, `http://caffe.berkeleyvision.org/` (2013).

[26] T. Wang, S. Gong, X. Zhu, S. Wang, Person re-identification by video ranking, in: European Conference on Computer Vision, IEEE, 2014.

[27] D. Gray, S. Brennan, H. Tao, Evaluating appearance models for recognition, reacquisition, and tracking, in: PETS, Citeseer, 2007.

[28] W. R. Schwartz, L. S. Davis, Learning discriminative appearance-based models using partial least squares, in: Computer Graphics and Image Processing (SIBGRAPI), 2009 XXII Brazilian Symposium on, IEEE, 2009, pp. 322–329.

[29] A. Globerson, S. T. Roweis, Metric learning by collapsing classes, in: NIPS, 2005, pp. 451–458.

[30] L. Ma, X. Yang, D. Tao, Person re-identification over camera networks using multi-task distance metric learning, TIP 23 (8) (2014) 3656–3670.

[31] B. Ma, Y. Su, F. Jurie, et al., Bicov: a novel image representation for person re-identification and face verification, in: BMVC, 2012.

[32] R. Zhao, W. Ouyang, X. Wang, Unsupervised salience learning for person re-identification, in: CVPR, IEEE, 2013, pp. 3586–3593.

[33] C. Liu, S. Gong, C. C. Loy, X. Lin, Person re-identification: what features are important?, in: ECCV, Springer, 2012, pp. 391–401.

[34] M. Kostinger, M. Hirzer, P. Wohlhart, P. M. Roth, H. Bischof, Large scale metric learning from equivalence constraints, in: CVPR, IEEE, 2012, pp. 2288–2295.

[35] S. Pedagadi, J. Orwell, S. Velastin, B. Boghossian, Local fisher discriminant analysis for pedestrian re-identification, in: CVPR, IEEE, 2013, pp. 3318–3325.

[36] R. Zhao, W. Ouyang, X. Wang, Person re-identification by salience matching, in: ICCV, IEEE, 2013, pp. 2528–2535.