Person Re-identification with Cascaded Pairwise Convolutions

Yicheng Wang[1], Zhenzhong Chen[1]*, Feng Wu[2], Gang Wang[3]
[1]School of Remote Sensing and Information Engineering, Wuhan University, China
[2]School of Information Science and Technology, University of Science and Technology of China, China
[3]Alibaba Group, China
wyc@whu.edu.cn, zzchen@whu.edu.cn, fengwu@ustc.edu.cn, wg134231@alibaba-inc.com

Abstract

In this paper, a novel deep architecture named BraidNet is proposed for person re-identification. BraidNet has a specially designed WConv layer, and the cascaded WConv structure learns to extract the comparison features of two images, which are robust to misalignments and color differences across cameras. Furthermore, a Channel Scaling layer is designed to optimize the scaling factor of each input channel, which helps mitigate the zero gradient problem in the training phase. To solve the problem of imbalanced volume of negative and positive training samples, a Sample Rate Learning strategy is proposed to adaptively update the ratio between positive and negative samples in each batch. Experiments conducted on CUHK03-Detected, CUHK03-Labeled, CUHK01, Market-1501 and DukeMTMC-reID datasets demonstrate that our method achieves competitive performance when compared to state-of-the-art methods.

1. Introduction

Person re-identification (Person Re-ID) is the process of discriminating whether two person images taken from different camera views belong to the same person [4]. It's challenging due to the problems brought by the variations of viewpoints, poses, illumination conditions and backgrounds [30].

The misalignment problem means that two images contain different contents in the same location, due to the variations of viewpoints and poses. The color differences across cameras means that the cameras in different illumination conditions take the same color into different RGB value. To solve these problems, we propose a novel deep architecture named **BraidNet**. BraidNet has a type of specially designed convolutional layer: WConv. As illustrated in Fig. 1, WConv uses two convolution kernels for each of the two inputs and crossly sums the four intermediate outputs to form two outputs. In our design, the cascaded WConv structure can learn to extract the comparison features of two images, which are robust to misalignments and color differences across cameras.

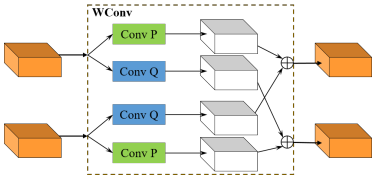

Figure 1. Illustration of the computation in a WConv layer. The 3D boxes represent feature maps. "Conv P/Q" means convolution with kernel P/Q.

The zero gradient problem exists in the DNN/CNN models with ReLU activation. The unit/channel which frequently makes negative response can rarely get non-zero gradient in the backward propagation (BP) process, thus corresponding weights to compute the response can't be sufficiently trained and become useless. To mitigate this problem, we propose a simple **Channel Scaling (CS) layer** which optimizes the scaling factor of each input channel, to guide the weights away from being negative.

The imbalanced training data problem in Person Re-ID means that the negative samples are much more than positive samples. If all the samples are used for training, the trained model tends to make negative prediction all the time; if only a limited percentage of negative samples is used, the abundant information in vast negative samples may be seriously wasted [1]. To deal with this problem without predefined percentage, an online batch generating strategy named **Sample Rate Learning (SRL)** is proposed. Like the Hard Negative Mining strategy [1, 35] which selects the negative samples with large loss on current model, the SRL strategy tries to increase the expected loss in each batch by adaptively adjusting the ratio of positive samples and negative samples.

*Corresponding author: Zhenzhong Chen

The benefits of CS layer and SRL strategy are verified by the controlled experiments. With the combination of *BraidNet-CS* + SRL, our method achieves competitive performance on five datasets. Finally, the visualization of WConv features proves the theoretical functions of the cascaded WConv structure.

2. Related Work

To deal with the misalignment problem, many works incorporate one-to-many patch matching or extract features from detected body parts. For example, DNNIM [24] performs inexact matching over a wide search space; DMLLV [25] uses specially designed distance functions to consider vertical misalignments, horizontal misalignments and leg posture variations; Spindle [36] separately captures semantic features from different body regions thus the macro- and micro-body features can be well aligned across images. The problem of color differences across cameras is rarely considered directly by existing approaches but also important. OSML [3] tries to solve this problem by using a single pair of ColorChecker images to learn a Mahalanobis metric that directly models the relationship between color features across a pair of cameras. We notice that some special combinations of different contents in the same location of two images can also provide cues for matching. The cascaded WConv structure is designed to learn these special combinations, and judge whether these combinations exist in the same location of two images.

To avoid the zero gradient problem, LReLU [18] and PReLU [8] modify the ReLU function, but bring a little extra computational cost in the model's inference. Batch normalization (BN) [9] also solves this problem: when a channel makes nearly all-zero responses among all the samples, the normalization process transfers some responses of this channel to positive. We design a simple CS layer to solve this problem by guiding the weights away from being negative. The CS layer brings no more computational cost in the model's inference.

To solve the imbalanced training data problem, one solution is to assign different treatments on positive and negative samples. LOMO+MLAPG [16] uses asymmetric sample weighting strategy to take larger weight on the loss of positive samples; DML [32] assigns asymmetric labels to positive and negative samples; SSSL [34] imposes different penalty parameters on positive and negative samples in objective function. Another solution is to define the percentage of positive and negative training samples[1, 13, 28], where the percentage of negative sample can be manually adjusted in the training phase. As the optimal percentage setting may vary with different datasets and training stages, we design the SRL strategy to adaptively optimize the percentage of positive samples and negative samples in each batch during the training phase.

3. BraidNet

3.1. Architecture

When the size of convolution window is $d \times d$, input feature map has m channels and output feature map has n channels, **conventional convolution operation** in one specific convolution window can be written as:

$$y_i = b_i + \sum_{j=1}^{m} w_{ji} x_j, \qquad i = 1, 2, \cdots, n \tag{1}$$

where x_j is a $d^2 \times 1$ vector that represents co-located $d \times d$ elements in the jth input channel, w_{ji} is a $1 \times d^2$ weight vector, b_i is the bias factor of the ith output channel, and y_i is the response value at corresponding pixel in the ith output channel.

Our proposed WConv Layer has two input feature maps with the same size and two output feature maps with the same size. **WConv operation** in one specific convolution window is defined as:

$$\begin{cases} y_i^{(p)} = b_i + \sum_{j=1}^{m} (w_{ji}^{(p)} x_j^{(p)} + w_{ji}^{(q)} x_j^{(q)}) \\ y_i^{(q)} = b_i + \sum_{j=1}^{m} (w_{ji}^{(q)} x_j^{(p)} + w_{ji}^{(p)} x_j^{(q)}) \\ \quad i = 1, 2, \cdots, n \end{cases} \tag{2}$$

where $x_j^{(p)}$ and $x_j^{(q)}$ represent co-located elements in the jth channel of two input feature maps, respectively; $w_{ji}^{(p)}$ and $w_{ji}^{(q)}$ are two different weight vectors; b_i is the bias factor of the ith channel in the two output feature maps; $y_i^{(p)}$ and $y_i^{(q)}$ are the response value at corresponding pixels in the ith channel of the two output feature maps, respectively. The computation of a WConv layer is illustrated in Fig. 1.

In a **BraidNet**, feature maps of two input images are firstly extracted by weight-sharing subnetworks, then these feature maps are compared by cascaded WConv (with ReLU, pooling, *etc*., the same below) structure. The output feature maps of the last WConv layer are element-wisely added together to obtain one feature map (*i.e*., comparison features), then these comparison features are fed to another subnetwork to obtain the final matching score.

3.2. Mechanism Analysis

In Person Re-ID, the variations of viewpoints, poses and illumination make two pedestrian images usually contain different local contents (*e.g*., objects, shapes, colors) in the same location, even if the two images belong to the same person. Therefore, directly judging whether two images contain the same co-located content leads to unsatisfactory Re-ID accuracy. Luckily, some special combinations

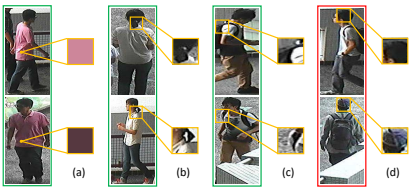

Figure 2. Examples of asymmetric matching pattern. (a) two specific RGB value provide evidence for matching, as they may come from the same color but be effected by different illumination conditions; (b) two specific plane shapes of braid provide evidence for matching, as they may be caused by different viewpoints; (c) a backpack and a strap provide evidence for matching, as almost all the backpacks have straps; (d) hair and a hat provide evidence for mismatching, as a man in a hat can't reveal his hair.

of different local contents can provide cues for the matching, which are named as "asymmetric matching pattern". Some examples of asymmetric matching pattern are shown in Fig. 2. An asymmetric matching pattern can be written as a set of two different subpatterns: $\{m^{(p)}, m^{(q)}\}$. Given one asymmetric matching pattern $\{m^{(p)}, m^{(q)}\}$, two images I_A, I_B and one specific location, when "$m^{(p)}$ and $m^{(q)}$ exist in the same location of I_A and I_B, respectively", or "$m^{(q)}$ and $m^{(p)}$ exist in the same location of I_A and I_B, respectively", we say this asymmetric matching pattern exist in this location of these two images, and it is useful cues for subsequent matching computation. Above judging process is named as asymmetric pattern matching, which can be implemented by our proposed cascaded WConv structure. Detailed explanation is presented below.

When the L2-Norm regularization is applied on the convolution kernels, some trained weights in the kernels become very small. Given a very small positive scalar ϵ, weight vectors in Eq. 2 can be divided into two categories:

$$c_{ji}^{(s)} = \begin{cases} 0, & \|w_{ji}^{(s)}\|_2^2 \leq \epsilon \\ 1, & \|w_{ji}^{(s)}\|_2^2 > \epsilon \end{cases} \tag{3}$$

where $s \in \{p, q\}$, the same below.

According to Eq. 2, when $c_{ji}^{(p)} = 0$, $w_{ji}^{(p)} x_j^{(p)} \approx 0$ and $w_{ji}^{(p)} x_j^{(q)} \approx 0$, thus the effect of $x_j^{(p)}$ to $y_i^{(p)}$ and the effect of $x_j^{(q)}$ to $y_i^{(q)}$ can be ignored; when $c_{ji}^{(q)} = 0$, $w_{ji}^{(q)} x_j^{(q)} \approx 0$ and $w_{ji}^{(q)} x_j^{(p)} \approx 0$, thus the effect of $x_j^{(q)}$ to $y_i^{(p)}$ and the effect of $x_j^{(p)}$ to $y_i^{(q)}$ can be ignored.

We divide the channels in the cascaded WConv structure into five categories:
0) meaningless channel, whose responses are always approximately zero;
1) the feature of I_A;
2) the feature of I_B;
3) the result of an asymmetric pattern matching in the combination case of $m^{(p)}$-I_A, $m^{(q)}$-I_B;
4) the result of an asymmetric pattern matching in the combination case of $m^{(q)}$-I_A, $m^{(p)}$-I_B.

For each WConv layer, we use $t_j^{(p)}$ and $t_j^{(q)}$ to represent the category of the jth channel in the input feature maps marked with p and q, respectively. We use $l_i^{(p)}$ and $l_i^{(q)}$ to represent the category of the ith channel in the output feature maps marked with p and q, respectively.

For the first WConv layer, the two input feature maps come from the same subnetwork but different images, so $x_j^{(p)}$ and $x_j^{(q)}$ represent features in the same type and $[t_j^{(p)}, t_j^{(q)}] \in \{[0,0],[1,2]\}$. The categories of output channels can be judged as:

$$[l_i^{(p)}, l_i^{(q)}] = \begin{cases} [0,0] & \forall\, j|_{t_j^{(p)}\neq 0}, c_{ji}^{(p)} = 0; \forall\, j|_{t_j^{(q)}\neq 0}, c_{ji}^{(q)} = 0 \\ [1,2] & \exists\, j|_{t_j^{(p)}=1}, c_{ji}^{(p)} = 1; \forall\, j|_{t_j^{(q)}\neq 0}, c_{ji}^{(q)} = 0 \\ [2,1] & \forall\, j|_{t_j^{(p)}\neq 0}, c_{ji}^{(p)} = 0; \exists\, j|_{t_j^{(q)}=2}, c_{ji}^{(q)} = 1 \\ [3,4] & \exists\, j|_{t_j^{(p)}=1}, c_{ji}^{(p)} = 1; \exists\, j|_{t_j^{(q)}=2}, c_{ji}^{(q)} = 1 \end{cases} \tag{4}$$

For the second WConv layer, the two input feature maps come from previous WConv layer, so $[t_j^{(p)}, t_j^{(q)}] \in \{[0,0],[1,2],[2,1],[3,4]\}$. The categories of output channels can be judged as:

$$[l_i^{(p)}, l_i^{(q)}] = \begin{cases} [0,0] & \forall\, (j,s)|_{t_j^{(s)}\neq 0}, c_{ji}^{(s)} = 0 \\ [1,2] & \exists\, (j,s)|_{t_j^{(s)}=1}, c_{ji}^{(s)} = 1; \forall\, (j,s)|_{t_j^{(s)}\neq 1}, c_{ji}^{(s)} = 0 \\ [2,1] & \exists\, (j,s)|_{t_j^{(s)}=2}, c_{ji}^{(s)} = 1; \forall\, (j,s)|_{t_j^{(s)}\neq 2}, c_{ji}^{(s)} = 0 \\ [3,4] & \exists\, (j,s)|_{t_j^{(s)}=1}, c_{ji}^{(s)} = 1; \exists\, (j,s)|_{t_j^{(s)}=2}, c_{ji}^{(s)} = 1 \\ [3,4] & \exists\, (j,s)|_{t_j^{(s)}\in\{3,4\}}, c_{ji}^{(s)} = 1 \end{cases} \tag{5}$$

As described in Eqs. 4 and 5, $l_i^{(s)} = 0$ when the effects of I_A and I_B on $y_i^{(s)}$ can be ignored; $l_i^{(s)} = 1$ when the effect of I_A on $y_i^{(s)}$ can't be ignored, but the effect of I_B on $y_i^{(s)}$ can be ignored; $l_i^{(s)} = 2$ when the effect of I_B on $y_i^{(s)}$ can't be ignored, but the effect of I_A on $y_i^{(s)}$ can be ignored; $l_i^{(s)} \in \{3,4\}$ when the effect of neither I_A nor I_B on $y_i^{(s)}$ can be ignored (the effects of I_A and I_B are different as corresponding weights are different, *i.e.*, the subpatterns corresponding to I_A and I_B are different). When $[l_i^{(p)}, l_i^{(q)}] \in \{[1,2],[2,1]\}$, $y_i^{(p)}$ and $y_i^{(q)}$ represent the same type of single-image feature, but correspond to different images; when $[l_i^{(p)}, l_i^{(q)}] = [3,4]$, $y_i^{(p)}$ and $y_i^{(q)}$ are the results of the same asymmetric pattern matching, but in different combination cases of subpatterns and images.

The discussions on subsequent WConv layers is the same as that of the second WConv layer.

To summarize, the cascaded WConv structure not only extracts multi-level single-image features, but also extracts multi-level cross-image features via asymmetric pattern matching.

We add together the two output feature maps of the cascaded WConv structure, to judge whether corresponding asymmetric matching patterns exist in the same location of two images in any combination cases of images and subpatterns (or judge whether the two images contain the same content in the same location). As different comparison features have different roles in matching, a subnetwork is used to further synthesize previous comparison features and make the final matching score.

4. Channel Scaling Layer

4.1. Formulation

In one CS layer, the operation applied on one pixel of the ith input channel is:

$$z_i = \gamma_i y_i \tag{6}$$

where γ_i is the scaling factor and is shared across all the pixels of corresponding channel. Additionally, in order to confirm the scaling factor $\gamma_i > 0$, we let:

$$\gamma_i = e^{\alpha_i} \tag{7}$$

where $\alpha_i \in \mathbb{R}$, α_i is initialized to 0 and is trained in the training phase. Eqs. 6 and 7 define the operation of CS layer.

Each CS layer is trained using mini-batch SGD method simultaneously with other layers. In the BP process, the gradients of y_i and α_i are calculated as:

$$\begin{cases} \nabla_{y_i} = \nabla_{z_i} \dfrac{\partial z_i}{\partial y_i} = \nabla_{z_i} \gamma_i \\ \nabla_{\gamma_i} = \displaystyle\sum_{z_i} \nabla_{z_i} \frac{\partial z_i}{\partial \gamma_i} = \sum_{z_i} \nabla_{z_i} y_i \\ \nabla_{\alpha_i} = \nabla_{\gamma_i} \dfrac{\partial \gamma_i}{\partial \alpha_i} = \nabla_{\gamma_i} e^{\alpha_i} = \nabla_{\gamma_i} \gamma_i \end{cases} \tag{8}$$

4.2. Mechanism Analysis

For the convenience of discussion, we study the CS layer with input feature map of 1×1 spatial size and n channels. The input feature map of this CS layer comes from a convolutional layer, and the output feature map of this CS layer is fed to a ReLU layer. The computation in above layers is:

$$o_i = max\{\gamma_i \sum_{j=1}^{r} v_{ji} x_j, 0\}, \qquad i = 1, 2, \cdots, n \tag{9}$$

where r is the number of input elements in one convolutional window (including a constant 1 for the bias operation), x_j is the jth element and v_{ji} is the corresponding weight to calculate the response value in the ith output channel, $x_j \geq 0$ as it comes from previous ReLU layer.

In the BP process,the gradient of γ_i is calculated as:

$$\nabla_{\gamma_i} = \begin{cases} \nabla_{o_i} \sum_{k=1}^{r} v_{ki} x_k & \sum_{k=1}^{r} v_{ki} x_k > 0 \\ 0 & \sum_{k=1}^{r} v_{ki} x_k \leq 0 \end{cases} \tag{10}$$

and the gradient of v_{ji} is calculated as:

$$\nabla_{v_{ji}} = \begin{cases} \nabla_{o_i} \gamma_i x_j & \sum_{k=1}^{r} v_{ki} x_k > 0 \\ 0 & \sum_{k=1}^{r} v_{ki} x_k \leq 0 \end{cases} \tag{11}$$

In the training phase, the value of a parameter will be significantly changed when the gradient of it keeps to be positive/negative in multiple continuous iterations (one iteration means the processes of forward propagation, backward propagation and gradient descent with one batch, the same below).

When $\nabla_{o_i} > 0$ and $\sum_{k=1}^{r} v_{ki} x_k > 0$ in multiple continuous iterations, $\nabla_{v_{ji}} \geq 0$ in these iterations and v_{ji} may decrease according to Eq. 11, $\nabla_{\gamma_i} > 0$ in these iterations and γ_i decreases according to Eq. 10. As γ_i is a multiplication factor to compute $\nabla_{v_{ji}}$, the decrease of v_{ji} is suppressed.

When $\nabla_{o_i} < 0$ and $\sum_{k=1}^{r} v_{ki} x_k > 0$ in multiple continuous iterations, $\nabla_{v_{ji}} \leq 0$ in these iterations and v_{ji} may increase; $\nabla_{\gamma_i} < 0$ in these iterations and γ_i increases. As γ_i is a multiplication factor to compute $\nabla_{v_{ji}}$, the increase of v_{ji} is promoted.

When $\sum_{k=1}^{r} v_{ki} x_k$ is always non-positive, which is easily to occur when $v_{\cdot i}$ ("$\cdot$" represents 1, 2, $\cdots$, r) are partial to be negative, $\nabla_{v_{\cdot i}} = 0$ all the time. Thereafter, $v_{\cdot i}$ can't get non-zero gradients to update their value, $o_i = 0$ all the time, so $v_{\cdot i}$ and corresponding channel become useless in the network. We can use CS layer to guide the weights away from being negative, so as to alleviate this zero gradient problem.

After the training phase, we can update $v_{\cdot i}$ with the value of $\gamma_i v_{\cdot i}$ and then remove the CS layer, so CS layer brings no additional computational cost in the model's inference.

5. Sample Rate Learning

To solve the problem of imbalanced training data, some methods use fixed percentage of positive samples in each batch, *i.e.*, sample rate. As different datasets may have different optimal percentage settings, and the optimal percentage may varies in different training phase, we design the Sample Rate Learning (SRL) strategy to adaptively optimize the percentage of positive and negative samples in each batch during the training phase.

Given sample rate $r \in (0, 1)$ and batch size n, the number of positive samples (n^+) and the number of negative samples (n^-) in the batch can be calculated as:

$$\begin{cases} n^+ = \text{round}(nr) \\ n^- = n - n^+ \end{cases} \tag{12}$$

In order to confirm $r \in (0, 1)$, we let:

$$r = \frac{1}{1 + e^{-\lambda}} \tag{13}$$

where $\lambda \in \mathbb{R}$, and λ is a learned parameter in the training phase.

Given prediction model $s = f(X; \theta)$ (X is a sample, θ are the parameters in the model, s is the matching score) and loss function $l(s, c)$ (c is the label of corresponding sample, $c \in \{+1, -1\}$), the expectations of loss on positive sample and negative sample respectively in condition of θ are supposed to exist and are written as $\mathrm{E}[Loss^+|\theta]$, $\mathrm{E}[Loss^-|\theta]$. They can be estimated in each batch via:

$$\begin{cases} \hat{\mathrm{E}}[Loss^+|\theta] = \dfrac{1}{n^+} \displaystyle\sum_{i=1}^{n^+} l(S_i^+, +1) \\ \hat{\mathrm{E}}[Loss^-|\theta] = \dfrac{1}{n^-} \displaystyle\sum_{i=1}^{n^-} l(S_i^-, -1) \end{cases} \tag{14}$$

where S_i^+ (S_i^-) is the matching score of ith positive (negative) sample.

The expectation of loss in one batch which is generated with sample rate r is:

$$\mathrm{E}[Loss|\theta, r] = r\mathrm{E}[Loss^+|\theta] + (1 - r)\mathrm{E}[Loss^-|\theta] \tag{15}$$

The partial derivative of $\mathrm{E}[Loss|\theta, r]$ with respect to r (the gradient of r) is:

$$\nabla_r = \mathrm{E}[Loss^+|\theta] - \mathrm{E}[Loss^-|\theta] \tag{16}$$

and can be estimated via:

$$\hat{\nabla}_r = \hat{\mathrm{E}}[Loss^+|\theta] - \hat{\mathrm{E}}[Loss^-|\theta] \tag{17}$$

Inspired by the hard negative mining strategy used in ImprovedDL [1] and DCSL [35], which chooses the negative samples with large loss on current model, we try to increase $\mathrm{E}[Loss|\theta, r]$ by adjusting r, and the objective function in the training phase can be written in a minimax form:

$$\min_{\theta}\{\max_{r} \mathrm{E}[Loss|\theta, r]\} \tag{18}$$

To satisfy Eq. 18, r should be updated to 1 when $\nabla_r > 0$, and updated to 0 when $\nabla_r < 0$. Due to the uncertainty of estimation, we can't guarantee $\nabla_r > 0$ or $\nabla_r < 0$ when $\hat{\nabla}_r > 0$ or $\hat{\nabla}_r < 0$, but the larger the $|\hat{\nabla}_r|$ is, the more confident we are. Therefore, we don't update r to 1 or 0 directly, but increase or decrease it gradually, and $|\hat{\nabla}_r|$ can be used as a dynamic changing 'step size' in this process.

Based on the above discussion, rather than using the gradient descent method which updates parameters along the opposite direction of gradients, r (λ indeed) should be updated along the direction of gradient. As it is unnecessary to let r converge to a fixed value, the learning rate of λ remains unchanged in the training phase. The pseudo code of mini-batch stochastic gradient descent method with SRL strategy is shown in Algorithm 1.

Algorithm 1 Mini-batch SGD Method with SRL for Training Binary Classification Model

Require:
Training dataset $\{X_1, C_1\}, \{X_2, C_2\}, \cdots$,
Binary classification model $s = f(x; \theta)$,
Loss function $l(s, c)$,
Batchsize n,
Learning rate of model parameters μ,
Learning rate of λ μ_λ.
Ensure: Trained parameters θ.
$\lambda \leftarrow 0$.
Initiate θ by a random generator.
for $e \leftarrow 1$ to E **do**
  for $t \leftarrow 1$ to T_e **do**
    Calculate r using Eq. 13 .
    Calculate n^+ and n^- using Eq. 12 .
    Choose n^+ positive samples and n^- negative samples to make a batch $\{x_1, c_1\}, \{x_2, c_2\}, \cdots \{x_n, c_n\}$.
    $s_i \leftarrow f(x_i; \theta), i = 1, 2, \cdots, n$.
    Calculate $\hat{\nabla}_r$ using Eqs. 14 and 17 , calculate $\hat{\nabla}_\theta$ using BP method.
    $\hat{\nabla}_\lambda \leftarrow r(1 - r)\hat{\nabla}_r$.
    $\lambda \leftarrow \lambda + \mu_\lambda \hat{\nabla}_\lambda, \theta \leftarrow \theta - \mu\hat{\nabla}_\theta$.
  end for
  Decrease μ.
end for
return θ

6. Experiments

Experiments are implemented on the DagNN wrapper in MatConvNet toolbox [29]. Firstly, we compare the performance of networks with different settings to verify the benefits brought by BraidNet architecture, CS layer and SRL strategy, respectively. Then our method and some other recently proposed methods are compared on multiple datasets. Finally, some features in the cascaded WConv structure are visualized to prove our theoretical analysis of cascaded WConv structure.

Datasets and Evaluation Protocol. Five Person Re-ID datasets are used in our experiments: CUHK03-Detected (the detected version of CUHK03 [13]), CUHK03-Labeled (the labeled version of CUHK03), CUHK01 [12], Market-1501 [37], and DukeMTMC-reID [38] which was developed based on DukeMTMC dataset [22].

The single-query setting is followed in the test phase. For each gallery image, the matching scores of the most matching probe ID (in different camera views) are increased to ensure the uniqueness of the matched one. For Market-1501 and DukeMTMC-reID datasets, we use the dataset split case and evaluation packages provided by [37] and [38], respectively. For CUHK03-Detected, CUHK03-Labeled and CUHK01 datasets, the number of individuals in testing subset is set to 100. The images from the second camera view are used as the probe set. In one evaluation process, we randomly select one image from the first camera view for each identity to make the gallery set, then we average the CMC curves of all the probe images. Above evaluation process is repeated 100 times and the mean value is the final result.

Architecture. Three BraidNet architectures are included

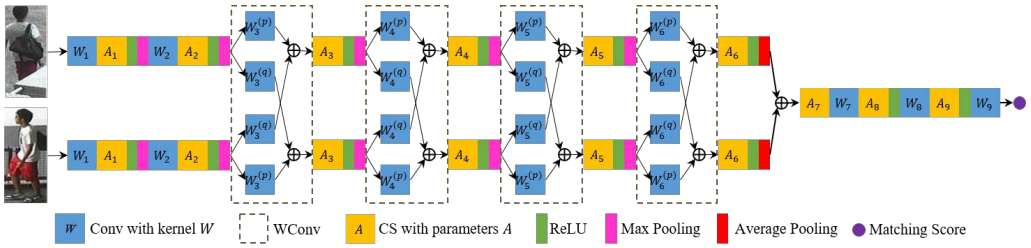

Figure 3. Illustration of *BraidNet-CS* architecture. Each input image is resized to [128,64]. Each Conv layer and WConv layer except the last three Conv layers has window size 3×3, stride [1, 1] and padding [1, 1, 1, 1]; each Max Pooling layer has pooling size 2×2 and padding [0, 0, 0, 0]; each Average Pooling layer has pooling size 4×2 and padding [0, 0, 0, 0]; the last three Conv layers apply 1×1 convolutions upon 1×1 feature maps. W_1 corresponds to M output channels and $W_2 \sim W_8$ correspond to N output channels. For CUHK03-Detected, CUHK03-Labeled and CUHK01 datasets, M=64, N=128; for Market-1501 and DukeMTMC-reID datasets, M=80, N=160.

in our experiments: *BraidNet*, *BraidNet-BN*, and *BraidNet-CS*. The architecture of *BraidNet-CS* is illustrated in Fig. 3. Feature maps of two images are firstly extracted by the weight-sharing subnetworks with two Conv layers, and fed to a four-layer cascaded WConv structure. The outputs of the cascaded WConv structure are pooled to 1×1 spacial size and then added together. Finally, a subnetwork with three Conv layers is used to synthesize previous comparison features and output the final matching score. As Market-1501 and DukeMTMC-reID datasets contain more cameras and have more possible asymmetric matching patterns, we use larger channel numbers when experiment is implemented on these datasets. We remove the CS layers from *BraidNet-CS* architecture to make the *BraidNet* architecture. We replace the CS layers with the BN layers in *BraidNet-CS* architecture to make the *BraidNet-BN* architecture (the two BN layers connecting to the same WConv layer share parameters).

Training Setting. Following the Binomial Deviance used in [32], the loss function on each sample is:

$$loss = log(1 + e^{-sc}) + \frac{\alpha}{2}\|\theta\|_2^2 \tag{19}$$

where s ($s \in \mathbb{R}$) is the matching score and c ($c \in \{+1, -1\}$) is the sample label; $\frac{\alpha}{2}\|\theta\|_2^2$ is the L2-Norm regularization term applied on all the convolution kernels, and the weight decay parameter α is set to 0.0005.

The BraidNet is trained from scratch on each dataset separately. Data augmentation is applied by adding left-right flipping version of original images. The batch size n is set to 256. The momentum factor is set to 0.9; the learning rate of λ in SRL strategy is set to 0.1. For CUHK03-Detected, CUHK03-Labeled and CUHK01 datasets, the learning rate of model parameters is set to 0.02 in the first 50k iterations, then multiplied by 0.5 every 10k iterations until the 140k-th iteration. For Market-1501 and DukeMTMC-reID datasets, the learning rate of model parameters is set to 0.02 in the first 70k iterations, then multiplied by 0.5 every 10k iterations until the 190k-th iteration.

To reduce the memory cost, we generate samples online in each iteration. To generate n^+ positive samples, we firstly randomly choose n^+ different identities, then for each chosen identity, we randomly choose two images to make a positive sample; to generate each one of n^- negative samples, we randomly choose two different identities, then randomly choose one image from each chosen identity.

6.1. The Effectiveness of Our Method

On each dataset of CUHK03-Detected CUHK03-Labeled and CUHK01, we evaluate the performance of *BraidNet*, *BraidNet-BN* and *BraidNet-CS* trained with fixed sample rate $1/4$, and the performance of *BraidNet-CS* trained with SRL strategy. The evaluation results are listed with that of some recently proposed methods in Table 1.

The Effectiveness of BraidNet According to Table 1, *BraidNet* outperforms most recently proposed methods. In contrast with DNNIM [24] which performs inexact matching over a wide search space in a single Normalized Correlation layer, our BraidNet performs asymmetric pattern matching in the same location of two feature maps in multiple WConv layers, and outperforms DNNIM by a large margin.

The Effectiveness of Channel Scaling Layer According to Table 1, the additional CS layers contribute obvious Rank-1 increase of 3.08% and 2.48% on CUHK03-Detected and CUHK03-Labeled datasets, respectively. On CUHK01 dataset, the Rank-1 score of *BraidNet-CS* is a bit inferior to that of *BraidNet*. As the CUHK01 dataset has only two cameras, it contains less variations in misalignments and color differences across cameras than the other datasets. For CUHK01 dataset, limited numbers of useful channels and parameters in the BraidNet are required, and additional CS layers make no help to the performance improvement.

The BN layer can speed up the convergence and brings performance improvement in most other works. As discussed in Section 2, BN layer also helps avoid the zero gradient problem. The additional BN layers improve

Table 1. Rank-1, Rank-10, Rank-20 scores (%) of different methods on CUHK03-Detected, CUHK03-Labeled and CUHK01 datasets.

Methods	CUHK03-Detected (p=100)			CUHK03-Labeled (p=100)			CUHK01 (p=100)		
	Rank-1	Rank-10	Rank-20	Rank-1	Rank-10	Rank-20	Rank-1	Rank-10	Rank-20
FPNN [13]	19.89	50.00	78.50	20.65	66.50	80.00	-	-	-
ImprovedDL [1]	44.96	83.47	93.15	54.74	93.88	98.10	65.00	93.12	97.20
LOMO + XQDA [15]	46.25	88.55	94.25	52.20	92.14	96.25	-	-	-
LOMO + MLAPG [16]	51.15	92.05	96.90	57.96	94.74	98.00	-	-	-
SSSL [34]	51.20	-	-	57.00	-	-	-	-	-
EDM [23]	52.09	-	-	61.32	-	-	86.59	-	-
SICIR [30]	52.17	91.00	95.00	-	-	-	71.80	94.00	98.00
Ensembles [21]	-	-	-	62.10	94.30	97.80	-	-	-
LDNS [33]	54.70	84.75	95.20	62.55	90.05	98.10	-	-	-
SLSTM [28]	57.3	88.3	-	-	-	-	-	-	-
GOG + XQDA [20]	65.5	93.7	-	67.3	96.0	-	-	-	-
MSCAN [11]	67.99	95.36	97.83	74.21	97.54	99.25	-	-	-
MTDnet [6]	-	-	-	74.68	97.47	-	78.50	97.50	-
GSCNN [27]	68.1	94.6	-	-	-	-	-	-	-
DNNIM [24]	72.04	96.00	98.26	72.43	95.51	98.40	81.23	97.39	98.60
SSM [2]	72.70	-	96.05	76.63	-	97.95	-	-	-
LSRO [38]	73.1	96.7	-	-	-	-	-	-	-
DCSL [35]	-	-	-	80.20	99.17	-	89.60	98.90	-
JLML [14]	80.6	**98.7**	99.2	83.2	**99.4**	**99.8**	87.0	98.6	99.4
SVDNet [26]	81.8	97.2	-	-	-	-	-	-	-
CRAFT-MFA + LOMO [7]	**87.5**	**98.7**	99.5	-	-	-	-	-	-
BraidNet	80.24	97.72	99.34	84.81	98.59	99.33	90.74	99.51	**100.00**
BraidNet-BN	82.69	91.51	92.26	86.02	93.07	94.05	81.98	87.27	87.91
BraidNet-CS	83.32	98.00	99.11	87.29	98.82	99.23	89.14	99.96	**100.00**
BraidNet-CS + SRL	85.85	98.46	**99.55**	**88.18**	98.66	99.48	**93.04**	**99.97**	**100.00**

Table 2. Rank-1 scores (%) and mAP scores (%) of different methods on Market-1501 and DukeMTMC-reID datasets.

Methods	Market-1501		DukeMTMC-reID	
	Rank-1	mAP	Rank-1	mAP
BoW + KISSME [37]	44.42	20.76	25.13	12.17
LOMO + XQDA [15]	-	-	30.75	17.04
BoW + WARCA [10]	45.16	-	-	-
TMA [19]	47.92	22.31	-	-
SCSP [5]	51.90	26.35	-	-
LDNS [33]	55.43	29.87	-	-
SLSTM [28]	61.6	35.3	-	-
GSCNN [27]	65.88	39.55	-	-
CRAFT-MFA + LOMO [7]	71.8	45.5	-	-
CADL [17]	73.84	47.11	-	-
Spindle [36]	76.9	-		
K-reciprocal [39]	77.11	63.63	-	-
LSRO [38]	78.06	56.23	67.68	47.13
MSCAN [11]	80.31	57.53	-	-
OIM [31]	-	-	68.1	-
SSM [2]	82.21	68.80	-	-
SVDNet [26]	82.3	62.1	**76.7**	56.8
JLML [14]	**85.1**	65.5	-	-
BraidNet-CS + SRL	83.70	**69.48**	76.44	**59.49**

Table 3. Rank-1 scores (%) of *BraidNet-CS* trained with different sample strategies on three datasets.

Sample Rate	CUHK03-Detected	CUHK03-Labeled	CUHK01
1/2	83.63	86.72	90.78
1/4	83.32	87.29	89.14
1/8	82.82	86.48	88.87
Learned	**85.85**	**88.18**	**93.04**

the Rank-1 scores on CUHK03-Detected and CUHK03-Labeled datasets, but decrease the other scores significantly. This unsatisfactory result may relate to the large fluctuations of the expectations and variances' estimations during the training phase. In our experiments, the CS layer is more stable and brings more performance improvements than the BN layer.

The Effectiveness of Sample Rate Learning In addition to training with fixed sample rate $1/4$ and SRL strategy, we evaluate the performance of *BraidNet-CS* trained with fixed sample rates $1/2$ and $1/8$ on three datasets, and compare corresponding Rank-1 scores in Table 3. In Rank-1 score, *BraidNet-CS* is not very sensitive to fixed sample rate in the field $[1/8, 1/2]$. Even so, SRL strategy makes about 2% improvements in Rank-1 score among three datasets.

The evolutions of sample rate during training *BraidNet-CS* with SRL strategy on the three datasets are shown in Fig. 5. On each dataset, the sample rate decreases to a certain degree in earlier training stage, then continuously decreases with the learning rate update. As CUHK03-Detected and CUHK03-Labeled datasets only vary in the detection method, the sample rate evolutions of them are almost the same, while the sample rate evolution of CUHK01 is obviously different from them. Above observations demonstrate that the sample rate learned by SRL strategy varies with different training stages and different datasets.

6.2. Comparison with Other Methods

In addition to CUHK03-Detected, CUHK03-Labeled and CUHK01 datasets, we also evaluate the performance of *BraidNet-CS* trained with SRL strategy on Market-1501 and DukeMTMC-reID datasets, respectively. Rank-1 scores and mAP scores of some recently proposed methods and our method are compared in Table 2.

According to Tables 1 and 2, our method achieves competitive performance on all the five datasets. It outperforms all the competitors in Rank-1 score on CUHK03-Labeled

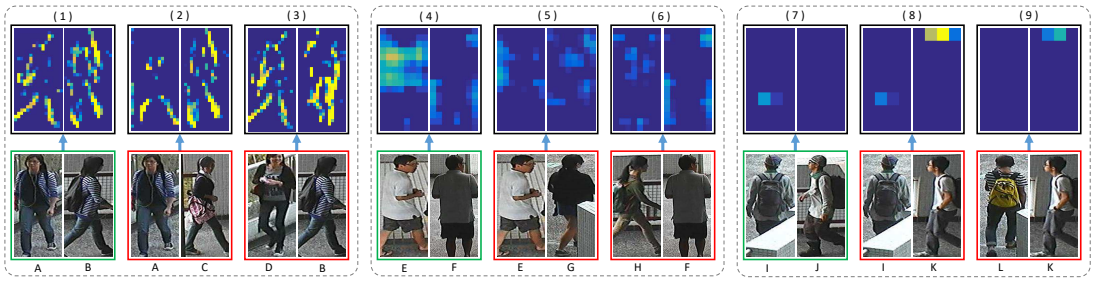

Figure 4. Visualization of some feature pairs of given image pairs in the cascaded WConv structure. Feature pairs in the three dotted boxes come from the 92nd channel of the first WConv-CS-ReLU operation's outputs, the 127th channel of the second WConv-CS-ReLU operation's outputs and the 91st channel of the third WConv-CS-ReLU operation's outputs, respectively.

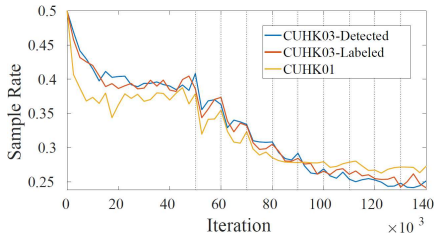

Figure 5. The evolutions of sample rate during training *BraidNet-CS* with SRL strategy on three datasets. The dotted lines mean the learning rate of model parameters is updated.

and CUHK01 datasets, and in mAP score on Market-1501 and DukeMTMC-reID datasets.

6.3. Visualization of WConv Features

To verify the function of cascaded WConv structure, we feed some samples into the *BraidNet-CS* architecture trained with SRL strategy on CUHK03-Labeled dataset, and visualize some feature pairs (*i.e.*, the responses on the ith channel of pairwise feature maps) in the cascaded WConv structure in Fig. 4.

In the first dotted box, it is obvious that the right feature in feature pair (1) and the right feature in feature pair (2) are the same feature of image A; the left feature in feature pair (1) and the left feature in feature pair (3) are the same feature of image B. The effect of the other image on one image's feature can be neglected.

In the second dotted box, the responses of feature pair (5) and feature pair (6) are low, so corresponding channels can't extract meaningful single-image feature from images of E, G, H or F. However, images E and F make high responses in the left feature of feature pair (4), indicating an asymmetric matching pattern is found in the high-response locations of the two input images. According to the high-response field and the content of images E and F, this asymmetric matching pattern may be $\{White, Gray\}$. $\{White, Gray\}$ may provide cues to support the matching decision, as they may come from the same original color but are changed to current RGB value by different illumination conditions.

In the third dotted box, the responses of feature pair (7) and feature pair (9) are low, so corresponding channels can't extract meaningful single-image feature from images I, J, L or K. However, images I and K make high responses in the right feature of feature pair (8), indicating an asymmetric matching pattern is found in the high-response locations of the two input images. According to the high-response field and the content of images I and K, this asymmetric matching pattern may be $\{Hair, Hat\}$. $\{Hair, Hat\}$ may provide cues to support the mismatching decision, as a man in a hat can't reveal his hair.

Above discussions verify the theoretical functions of cascaded WConv structure: extracting features of two images and comparing two images via asymmetric pattern matching on multiple semantic levels.

7. Conclusion

This work proposes a deep architecture named BraidNet for Person Re-ID, which can effectively handle the problems of misalignment and color differences across cameras. In addition, the CS layer and SRL strategy are proposed to solve the zero gradient problem and the imbalanced training data problem in the training phase. Experiments demonstrate the effectiveness of our method.

Acknowledgements

This work was supported in part by the National Key R&D Program of China under Grant No. 2017YFB1002202, National Natural Science Foundation of China under Grant No. 61471273 and No. 61771348, Wuhan Morning Light Plan of Youth Science and Technology under Grant No. 2017050304010302, and LIESMARS Special Research Funding.

References

[1] E. Ahmed, M. Jones, and T. K. Marks. An improved deep learning architecture for person re-identification. In *CVPR*,

pages 3908–3916, 2015.

[2] S. Bai, X. Bai, and Q. Tian. Scalable person re-identification on supervised smoothed manifold. In *CVPR*, pages 2530–2539, 2017.

[3] S. Bak and P. Carr. One-shot metric learning for person re-identification. In *CVPR*, pages 2990–2999, 2017.

[4] A. Bedagkar-Gala and S. K. Shah. A survey of approaches and trends in person re-identification. *Image and Vision Computing*, 32(4):270–286, 2014.

[5] D. Chen, Z. Yuan, B. Chen, and N. Zheng. Similarity learning with spatial constraints for person re-identification. In *CVPR*, pages 1268–1277, 2016.

[6] W. Chen, X. Chen, J. Zhang, and K. Huang. A multi-task deep network for person re-identification. In *AAAI*, pages 3988–3994, 2016.

[7] Y. Chen, X. Zhu, W. Zheng, and J. Lai. Person re-identification by camera correlation aware feature augmentation. *IEEE TPAMI*, 40(2):392–408, 2018.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, pages 1026–1034, 2015.

[9] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015.

[10] C. Jose and F. Fleuret. Scalable metric learning via weighted approximate rank component analysis. In *ECCV*, pages 875–890, 2016.

[11] D. Li, X. Chen, Z. Zhang, and K. Huang. Learning deep context-aware features over body and latent parts for person re-identification. In *CVPR*, pages 384–393, 2017.

[12] W. Li, R. Zhao, and X. Wang. Human reidentification with transferred metric learning. In *ACCV*, pages 31–44, 2012.

[13] W. Li, R. Zhao, T. Xiao, and X. Wang. Deepreid: Deep filter pairing neural network for person re-identification. In *CVPR*, pages 152–159, 2014.

[14] W. Li, X. Zhu, and S. Gong. Person re-identification by deep joint learning of multi-loss classification. In *IJCAI*, pages 2194–2200, 2017.

[15] S. Liao, Y. Hu, X. Zhu, and S. Z. Li. Person re-identification by local maximal occurrence representation and metric learning. In *CVPR*, pages 2197–2206, 2015.

[16] S. Liao and S. Z. Li. Efficient PSD constrained asymmetric metric learning for person re-identification. In *ICCV*, pages 3685–3693, 2015.

[17] J. Lin, L. Ren, J. Lu, J. Feng, and J. Zhou. Consistent-aware deep learning for person re-identification in a camera network. In *CVPR*, pages 5771–5780, 2017.

[18] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013.

[19] N. Martinel, A. Das, C. Micheloni, and A. K. Roy-Chowdhury. Temporal model adaptation for person re-identification. In *ECCV*, pages 858–877, 2016.

[20] T. Matsukawa, T. Okabe, E. Suzuki, and Y. Sato. Hierarchical gaussian descriptor for person re-identification. In *CVPR*, pages 1363–1372, 2016.

[21] S. Paisitkriangkrai, C. Shen, and A. van den Hengel. Learning to rank in person re-identification with metric ensembles. In *CVPR*, pages 1846–1855, 2015.

[22] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *ECCV workshop on Benchmarking Multi-Target Tracking*, pages 17–35, 2016.

[23] H. Shi, Y. Yang, X. Zhu, S. Liao, Z. Lei, W. Zheng, and S. Z. Li. Embedding deep metric for person re-identification: A study against large variations. In *ECCV*, pages 732–748, 2016.

[24] A. Subramaniam, M. Chatterjee, and A. Mittal. Deep neural networks with inexact matching for person re-identification. In *NIPS*, pages 2667–2675, 2016.

[25] C. Sun, D. Wang, and H. Lu. Person re-identification via distance metric learning with latent variables. *IEEE Trans. Image Processing*, 26(1):23–34, 2017.

[26] Y. Sun, L. Zheng, W. Deng, and S. Wang. SVDNet for pedestrian retrieval. In *ICCV*, pages 3800–3808, 2017.

[27] R. R. Varior, M. Haloi, and G. Wang. Gated siamese convolutional neural network architecture for human re-identification. In *ECCV*, pages 791–808, 2016.

[28] R. R. Varior, B. Shuai, J. Lu, D. Xu, and G. Wang. A siamese long short-term memory architecture for human re-identification. In *ECCV*, pages 135–153, 2016.

[29] A. Vedaldi and K. Lenc. Matconvnet: Convolutional neural networks for MATLAB. In *ACM MM*, pages 689–692, 2015.

[30] F. Wang, W. Zuo, L. Lin, D. Zhang, and L. Zhang. Joint learning of single-image and cross-image representations for person re-identification. In *CVPR*, pages 1288–1296, 2016.

[31] T. Xiao, S. Li, B. Wang, L. Lin, and X. Wang. Joint detection and identification feature learning for person search. In *CVPR*, pages 3415–3424, 2017.

[32] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Deep metric learning for person re-identification. In *ICPR*, pages 34–39, 2014.

[33] L. Zhang, T. Xiang, and S. Gong. Learning a discriminative null space for person re-identification. In *CVPR*, pages 1239–1248, 2016.

[34] Y. Zhang, B. Li, H. Lu, A. Irie, and X. Ruan. Sample-specific SVM learning for person re-identification. In *CVPR*, pages 1278–1287, 2016.

[35] Y. Zhang, X. Li, L. Zhao, and Z. Zhang. Semantics-aware deep correspondence structure learning for robust person re-identification. In *IJCAI*, pages 3545–3551, 2016.

[36] H. Zhao, M. Tian, S. Sun, J. Shao, J. Yan, S. Yi, X. Wang, and X. Tang. Spindle net: Person re-identification with human body region guided feature decomposition and fusion. In *CVPR*, pages 1077–1085, 2017.

[37] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian. Scalable person re-identification: A benchmark. In *ICCV*, pages 1116–1124, 2015.

[38] Z. Zheng, L. Zheng, and Y. Yang. Unlabeled samples generated by GAN improve the person re-identification baseline in vitro. In *ICCV*, 2017.

[39] Z. Zhong, L. Zheng, D. Cao, and S. Li. Re-ranking person re-identification with k-reciprocal encoding. In *CVPR*, pages 1318–1327, 2017.