Beyond Part Models: Person Retrieval with Refined Part Pooling (and A Strong Convolutional Baseline)

Yifan Sun[1], Liang Zheng[2], Yi Yang[3], Qi Tian[4], and Shengjin Wang[1]*

[1] Department of Electronic Engineering, Tsinghua University, China
[2] Research School of Computer Science, Australian National University, Australia
[3] Centre for Artificial Intelligence, University of Technology Sydney, Australia
[4] (1) Huawei Noah's Ark Lab (2) University of Texas at San Antonio
sunyf15@mails.tsinghua.edu.cn, wgsgj@tsinghua.edu.cn

Abstract. Employing part-level features offers fine-grained information for pedestrian image description.A prerequisite of part discovery is that each part should be well located. Instead of using external resources like pose estimator, we consider content consistency within each part for precise part location. Specifically, we target at learning discriminative part-informed features for person retrieval and make two contributions. (i) A network named Part-based Convolutional Baseline (PCB). Given an image input, it outputs a convolutional descriptor consisting of several part-level features. With a uniform partition strategy, PCB achieves competitive results with the state-of-the-art methods, proving itself as a strong convolutional baseline for person retrieval. (ii) A refined part pooling (RPP) method. Uniform partition inevitably incurs outliers in each part, which are in fact more similar to other parts. RPP re-assigns these outliers to the parts they are closest to, resulting in refined parts with enhanced within-part consistency. Experiment confirms that RPP allows PCB to gain another round of performance boost. For instance, on the Market-1501 dataset, we achieve (77.4+4.2)% mAP and (92.3+1.5)% rank-1 accuracy, surpassing the state of the art by a large margin. Code is available at: https://github.com/syfafterzy/PCB_RPP

Keywords: person retrieval, part-level feature, part refinement

1 Introduction

Person retrieval, also known as person re-identification (re-ID), aims at retrieving images of a specified pedestrian in a large database, given a query person-of-interest. Presently, deep learning methods dominate this community, with convincing superiority against hand-crafted competitors [44]. Deeply-learned representations provide high discriminative ability, especially when aggregated from deeply-learned part features. The latest state of the art on re-ID benchmarks are achieved with part-informed deep features [39,31,41].

* corresponding author

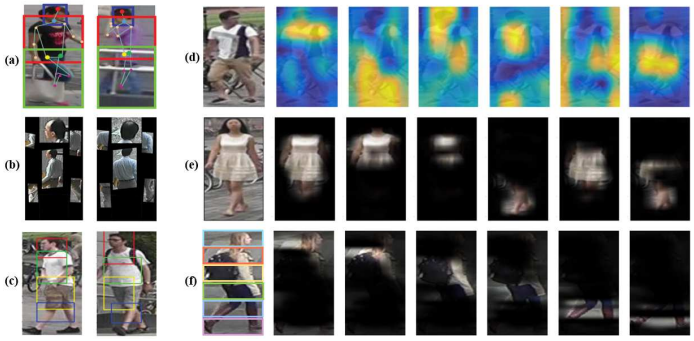


Fig. 1. Partition strategies of several deep part models in person retrieval. (a) to (e): Partitioned parts by GLAD [35], PDC [31], DPL [39], Hydra-plus [25] and PAR [41], respectively.(f): Our method employs a uniform partition and then refines each stripe. Both PAR [41] and our method conduct "soft" partition, but our method differs significantly from [41], as detailed in Section 2.

An essential prerequisite of learning discriminative part features is that parts should be accurately located. Recent state-of-the-art methods vary on their partition strategies and can be divided into two groups accordingly. The first group [42,31,35] leverage external cues, *e.g.,* assistance from human pose estimation [26,36,16,29,2]. They rely on external human pose estimation datasets and sophisticated pose estimator. The underlying datasets bias between pose estimation and person retrieval remains an obstacle against ideal semantic partition on person images. The other group [39,41,25] abandon cues from semantic parts. They require no part labeling and yet achieve competitive accuracy with the first group. Some partition strategies are compared in Fig. 1. Against this background of progress on learning part-level deep features, we rethink the problem of what makes well-aligned parts. Semantic partitions may offer stable cues to good alignment but are prone to noisy pose detections. This paper, from another perspective, lays emphasis on the consistency within each part, which we speculate is vital to the spatial alignment. Then we arrive at our motivation that given coarsely partitioned parts, we aim to refine them to reinforce within-part consistency. Specifically, we make the following two contributions:

First, we propose a network named Part-based Convolutional Baseline (PCB) which conducts uniform partition on the conv-layer for learning part-level features. It does not explicitly partition the images. PCB takes a whole image as the input and outputs a convolutional feature. Being a classification net, the architecture of PCB is concise, with slight modifications on the backbone network. The training procedure is standard and requires no bells and whistles. We show that the convolutional descriptor has much higher discriminative ability than the commonly used fully-connected (FC) descriptor. On the Market-1501 dataset, for instance, the performance increases from 85.3% rank-1 accuracy and 68.5% mAP to 92.3% (+7.0%) rank-1 accuracy and 77.4% (+8.9%) mAP, surpassing many state-of-the-art methods by a large margin.

Second, we propose an adaptive pooling method named Refined Part Pooling (RPP) to improve the uniform partition. We consider the motivation that within each part the contents should be consistent. We observe that under uniform partition, there exist outliers in each part. These outliers are, in fact, closer to contents in some other part, implying within-part inconsistency. Therefore, we refine the uniform partition by relocating those outliers to the part they are closest to, so that the within-part consistency is reinforced. An example of the refined parts is illustrated in Fig. 1(f). RPP does not require part labels for training and improves the retrieval accuracy over the high baseline achieved by PCB. For example on Market-1501, RPP further increases the performance to 93.8% (+1.5%) rank-1 accuracy and 81.6% (+4.2%) mAP.

2 Related works

Hand-crafted part features for person retrieval. Before deep learning methods dominated the re-ID research community, hand-crafted algorithms had developed approaches to learn part or local features. Gray and Tao [13] partition pedestrians into horizontal stripes to extract color and texture features. Similar partitions have then been adopted by many works [9,45,28,23]. Some other works employ more sophisticated strategy. Gheissari *et al.* [12] divide the pedestrian into several triangles for part feature extraction. Cheng *et al.* [4] employ pictorial structure to parse the pedestrian into semantic parts. Das *et al.* [6] apply HSV histograms on the head, torso and legs to capture spatial information.

Deeply-learned part features. The state of the art on most person retrieval datasets is presently maintained by deep learning methods [44]. When learning part features for re-ID, the advantages of deep learning over hand-crafted algorithms are two-fold. First, deep features generically obtain stronger discriminative ability. Second, deep learning offers better tools for parsing pedestrians, which further benefits the part features. In particular, human pose estimation and landmark detection have achieved impressive progress [26,29,2,36,16]. Several recent works in re-ID employ these tools for pedestrian partition and report encouraging improvement [42,31,35]. However, the underlying gap between datasets for pose estimation and person retrieval remains a problem when directly utilizing these pose estimation methods in an off-the-shelf manner. Others abandon the semantic cues for partition. Yao *et al.* [39] cluster the coordinates of max activations on feature maps to locate several regions of interest. Both Liu *et al.* [25] and Zhao *et al.* [41] embed the attention mechanism [38] in the network, allowing the model to decide where to focus by itself.

Deeply-learned part with attention mechanism. A major contribution of this paper is the refined part pooling. We compare it with a recent work, PAR [39] by Zhao *et al.* in details. Both works employ a part-classifier to conduct "soft" partition on pedestrian images, as shown in Fig. 1. Two works share the merit of requiring no part labeling for learning discriminative parts. However, the motivation, training methods, mechanism, and final performance of the two methods are quite different, to be detailed below.

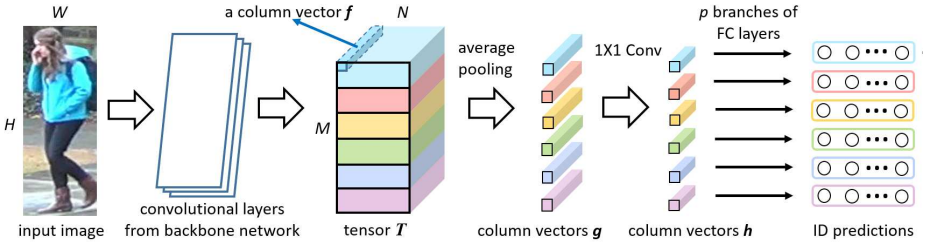


Fig. 2. Structure of PCB. The input image goes forward through the stacked convolutional layers from the backbone network to form a 3D tensor $\boldsymbol{T}$. PCB replaces the original global pooling layer with a conventional pooling layer, to spatially down-sample $\boldsymbol{T}$ into p pieces of column vectors $\boldsymbol{g}$. A following 1×1 kernel-sized convolutional layer reduces the dimension of $\boldsymbol{g}$. Finally, each dimension-reduced column vector $\boldsymbol{h}$ is input into a classifier, respectively. Each classifier is implemented with a fully-connected (FC) layer and a sequential Softmax layer. Either p pieces of $\boldsymbol{g}$ or $\boldsymbol{h}$ are concatenated to form the final descriptor of the input image.

Motivation: PAR aims at directly learning aligned parts while RPP aims to refine the pre-partitioned parts. *Working mechanism:* using attention method, PAR trains the part classifier in an unsupervised manner, while the training of RPP can be viewed as a weakly-supervised process. *Training process:* RPP firstly trains an identity classification model with uniform partition and then utilizes the learned knowledge to induce the training of part classifier. *Performance:* the slightly more complicated training procedure rewards RPP with better interpretation and significantly higher performance. For instance on Market-1501, mAP achieved by PAR, PCB cooperating attention mechanism and the proposed RPP are 63.4%, 74.6% and 81.6%, respectively. In addition, RPP has the potential to cooperate with various partition strategies.

3 Proposed Method

Sec. 3.1 first proposes a part-based convolutional baseline (PCB). PCB employs the simple strategy of uniform partition on convolutional features. Sec. 3.2 describes the phenomenon of within-part inconsistency, which reveals the problem of uniform partition. Sec. 3.3 proposes the refined part pooling (RPP) method. RPP reduces the partition errors by conducting pixel-level refinement on the convolutional feature. RPP is also featured for learning without part label information, which is detailed in Sec. 3.4.

3.1 PCB: A Part-based Convolutional Baseline

Backbone network. PCB can take any network without hidden fully-connected layers designed for image classification as the backbone, *e.g.*, Google Inception [33] and ResNet [14]. This paper mainly employs ResNet50 with consideration of its competitive performance as well as its relatively concise architecture.

From backbone to PCB. We reshape the backbone network to PCB with slight modifications, as illustrated in Fig. 2. The structure before the original global average pooling (GAP) layer is maintained exactly the same as the backbone model. The difference is that the GAP layer and what follows are removed. When an image undergoes all the layers inherited from the backbone network, it becomes a 3D tensor $\boldsymbol{T}$ of activations. In this paper, we define the vector of activations viewed along the channel axis as a **column vector**. Then, with a conventional average pooling, PCB partitions $\boldsymbol{T}$ into p horizontal stripes and averages all the column vectors in a same stripe into a single part-level column vector $\boldsymbol{g}_i$ ($i = 1, 2, \cdots, p$, the subscripts will be omitted unless necessary). Afterwards, PCB employs a convolutional layer to reduce the dimension of $\boldsymbol{g}$. According to our preliminary experiment, the dimension-reduced column vectors $\boldsymbol{h}$ are set to 256-dim. Finally, each $\boldsymbol{h}$ is input into a classifier, which is implemented with a fully-connected (FC) layer and a following Softmax function, to predict the identity (ID) of the input.

During training, PCB is optimized by minimizing the sum of Cross-Entropy losses over p pieces of ID predictions. During testing, either p pieces of $\boldsymbol{g}$ or $\boldsymbol{h}$ are concatenated to form the final descriptor $\mathcal{G}$ or $\mathcal{H}$, *i.e.*, $\mathcal{G} = [\boldsymbol{g}_1, \boldsymbol{g}_2, \cdots, \boldsymbol{g}_p]$ or $\mathcal{H} = [\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_p]$. As observed in our experiment, employing G achieves slightly higher accuracy, but at a larger computation cost, which is consistent with the observation in [32].

Important Parameters. PCB benefits from fine-grained spatial integration. Several key parameters, *i.e.,* the input image size (*i.e.*, $[\boldsymbol{H}, \boldsymbol{W}]$), the spatial size of the tensor $\boldsymbol{T}$ (*i.e.*, $[\boldsymbol{M}, \boldsymbol{N}]$), and the number of pooled column vectors (*i.e.*, $\boldsymbol{p}$) are important to the performance of PCB. Note that $[\boldsymbol{M}, \boldsymbol{N}]$ is determined by the spatial down-sampling rate of the backbone model, given the fixed-size input. Some deep object detection methods, *e.g.*, SSD [24] and R-FCN [5], show that decreasing the down-sampling rate of the backbone network efficiently enriches the granularity of feature. PCB follows their success by removing the last spatial down-sampling operation in the backbone network to increase the size of $\boldsymbol{T}$. This manipulation considerably increases retrieval accuracy with very light computation cost added. The details can be accessed in Section 4.4.

Through our experiment, the optimized parameter settings for PCB are:

- The input images are resized to 384×128, with a height to width ratio of 3:1.
- The spatial size of $\boldsymbol{T}$ is set to 24×8.
- $\boldsymbol{T}$ is equally partitioned into 6 horizontal stripes.

3.2 Within-Part Inconsistency

Uniform partition for PCB is simple, effective, and yet to be improved. It inevitably introduces partition errors to each part and consequentially compromises the discriminative ability of the learned feature. We analyze the partition errors from a new perspective: the within-part inconsistency.

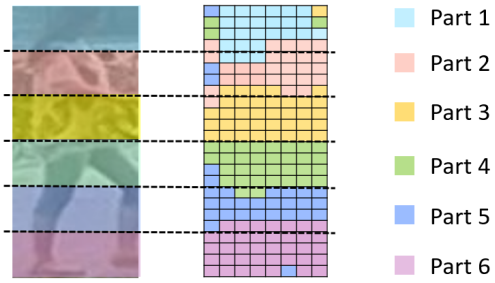


Fig. 3. Visualization of within-part inconsistency. $\boldsymbol{T}$. Left: $\boldsymbol{T}$ is equally partitioned to $p = 6$ horizontal stripes (parts) during training. Right: Every column vector in $\boldsymbol{T}$ is denoted with a small rectangle and painted in the color of its closest part.

With focus on the tensor $\boldsymbol{T}$ to be spatially partitioned, our intuition of within-part inconsistency is: column vector f in a same part of $\boldsymbol{T}$ should be similar to each other and be dissimilar to column vectors in other parts; otherwise the phenomenon of within-part inconsistency occurs, implying that the parts are partitioned inappropriately.

After training PCB to convergence, we compare the similarities between each f and $\boldsymbol{g}_i$ $(i = 1, 2, \cdots, p)$, *i.e.*, the average-pooled column vector of each part, by measuring cosine distance. By doing this, we find the closest part to each f, as exampled in Fig. 3. Each column vector is denoted by a small rectangle and painted in the color of its closest part. We observe that there exist many outliers, while designated to a specified horizontal stripe (part) during training, which are more similar to another part. The existence of these outliers suggests that they are inherently more consistent with column vectors in another part.

3.3 Refined Part Pooling

We propose the refined part pooling (RPP) to correct within-part inconsistency. Our goal is to assign all the column vectors according to their similarities to each part, so that the outliers will be relocated. More concretely, we quantitatively measure the similarity value $S(f \leftrightarrow P_i)$ between column vector f and each part Pi. Then the column vector f is sampled into part P_i according to the similarity value $S(f \leftrightarrow P_i)$, which is formulated by,

$$P_i = \{S(f \leftrightarrow P_i)f, \forall f \in F\}, \tag{1}$$

where F is the complete set of column vectors in tensor $\boldsymbol{T}$, $\{\bullet\}$ denotes the sampling operation to form an aggregate.

It is non-trivial to directly measure the similarity value between a given f and each part. Assume that we have performed a sampling operation defined in Eq. 1 to update each part, then the "already-measured" similarities don't stand anymore. We have to perform the "similarity measuring" $\rightarrow$ "sampling" procedure iteratively until convergence, which evolves a non-trivial clustering embedded in deep learning.

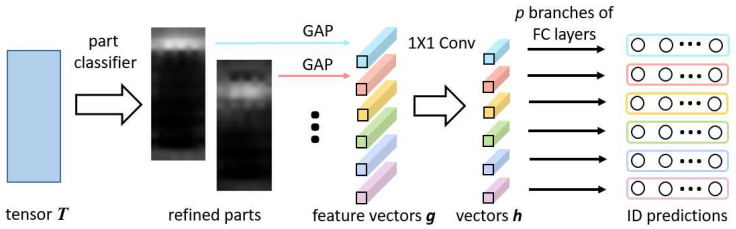


Fig. 4. PCB in combination with refined part pooling. The 3D tensor $\boldsymbol{T}$ is denoted simply by a rectangle instead of a cube as we focus on the spatial partition. Layers before $\boldsymbol{T}$ are omitted as they remain unchanged compared with Fig. 2. A part classifier predicts the probability of each column vector belonging to p parts. Then each part is sampled from all the column vectors with the corresponding probability as the sampling weight. GAP denotes global average pooling.

So instead of measuring the similarity between each f and each P_i, RPP employs a part classifier to predict the value of $S(f \leftrightarrow P_i)$ (which can also be interpreted as the probability of f belonging to P_i) as follows:

$$S(f \leftrightarrow P_i) = softmax(W_i^T f) = \frac{\exp(W_i^T f)}{\sum_{j=1}^{p} \exp(W_j^T f)}, \tag{2}$$

where p is the number of pre-defined parts (*i.e.*, $p = 6$ in PCB), and W is the trainable weight matrix of the part classifier.

The proposed refined part pooling conducts a "soft" and adaptive partition to refine the original "hard" and uniform partition, and the outliers originated from the uniform partition will be relocated. In combination with refined part pooling described above, PCB is further reshaped into Fig. 4. Refined part pooling, *i.e.*, the part classifier along with the following sampling operation, replaces the original average pooling. The structure of all the other layers remains exactly the same as in Fig. 2.

W has to be learned without part label information. To this end, we design an induced training procedure, as detailed in the following Section 3.4.

3.4 Induced Training for Part Classifier

The key idea of the proposed induced training is that: without part label information, we can use the already-learned knowledge in the pre-trained PCB to induce the training of the newly-appended part classifier. The algorithm is as follows.

- First, a standard PCB model is trained to convergence with $\boldsymbol{T}$ equally partitioned.
- Second, we remove the original average pooling layer after $\boldsymbol{T}$ and append a p-category part classifier on $\boldsymbol{T}$. New parts are sampled from $\boldsymbol{T}$ according to the prediction of the part classifier, as detailed in Section 3.3.

– Third, we set all the already learned layers in PCB fixed, leaving only the part classifier trainable. Then we retrain the model on training set. In this condition, the model still expects the tensor $\boldsymbol{T}$ to be equally partitioned, otherwise it will predict incorrect about the identities of training images. So Step 3 penalizes the part classifier until it conducts partition close to the original uniform partition, whereas the part classifier is prone to categorize inherently similar column vectors into a same part. A state of balance will be reached as a result of Step 3.
– Finally, all the layers are allowed to be updated. The whole net, *i.e.*, PCB along with the part classifier are fine-tuned for overall optimization.

In the above training procedure, PCB model trained in Step1 induces the training of the part classifier. Step3 and 4 converges very fast, requiring 10 more epochs in total.

Algorithm 1: Induced training for part classifier

Step 1. A standard PCB is trained to convergence with uniform partition.
Step 2. A p-category part classifier is appended on the tensor $\boldsymbol{T}$.
Step 3. All the pre-trained layers of PCB are fixed. Only the part classifier is trainable. The model is trained until convergence again.
Step 4. The whole net is fine-tuned to convergence for overall optimization.

4 Experiments

4.1 Datasets and Settings

Datasets. We three datasets for evaluation, *i.e.*, **Market-1501** [43], **DukeMTMC-reID** [30,47], and **CUHK03** [19]. The Market-1501 dataset contains 1,501 identities observed under 6 camera viewpoints, 19,732 gallery images and 12,936 training images detected by DPM [10]. The DukeMTMC-reID dataset contains 1,404 identities, 16,522 training images, 2,228 queries, and 17,661 gallery images. With so many images captured by 8 cameras, DukeMTMC-reID manifests itself as one of the most challenging re-ID datasets up to now. The CUHK03 dataset contains 13,164 images of 1,467 identities. Each identity is observed by 2 cameras. CUHK03 offers both hand-labeled and DPM-detected bounding boxes, and we use the latter in this paper. CUHK03 originally adopts 20 random train/test splits, which is time-consuming for deep learning. So we adopt the new training/testing protocol proposed in [48]. For Market-1501 and DukeMTMC-reID, we use the evaluation packages provided by [43] and [47], respectively. All the experiment evaluates the single-query setting. Moreover, for simplicity we do not use re-ranking algorithms which considerably improve mAP [48]. Our results are compared with reported results without re-ranking.

4.2 Implementation details

Implementation of IDE for comparison. We note that the IDE model specified in [44] is a commonly used baseline in deep re-ID systems [44,42,37,11,32,46,47,49]. In contrast to the proposed PCB, the IDE model learns a global descriptor. For comparison, we implement the IDE model on the same backbone network, *i.e.*, ResNet50, and with several optimizations over the original one in [44], as follows. 1) After the "pool5" layer in ResNet50, we append a fully-connected layer followed by Batch Normalization and ReLU. The output dimension of the appended FC layer is set to 256-dim. 2) We apply dropout on "pool5" layer. Although there are no trainable parameters in "pool5" layer, there is evidence that applying Dropout on it, which outputs a high dimensional feature vector of 2048d, effectively avoids over-fitting and gains considerable improvement [46,47]. We empirically set the dropout ratio to 0.5. On Market-1501, our implemented IDE achieves 85.3% rank-1 accuracy and 68.5% mAP, which is a bit higher than the implementation in [49].

Implementation of two Potential Alternative Structures of PCB for comparison. Given a same backbone network, there exist several potential alternative structures to learn part-level features. We enumerate two structures for comparison with PCB.

- Variant 1. Instead of making an ID prediction based on each $\boldsymbol{h}_i$ $(i = 1, 2, \cdots, p)$, it averages all $\boldsymbol{h}_i$ into a single vector $\overline{\boldsymbol{h}}$, which is then fully connected to an ID prediction vector. During testing, it also concatenates $\boldsymbol{g}$ or $\boldsymbol{h}$ to form the final descriptor. Variant 1 is featured by learning a convolutional descriptor under a single loss.
- Variant 2. It adopts exactly the same structure as PCB in Fig. 2. However, all the branches of FC classifiers in Variant 2 share a same set of parameters.

Training. The training images are augmented with horizontal flip and normalization. We set batch size to 64 and train the model for 60 epochs with base learning rate initialized at 0.1 and decayed to 0.01 after 40 epochs. The backbone model is pre-trained on ImageNet [7]. The learning rate for all the pre-trained layers are set to 0.1× of the base learning rate. When employing refined part pooling for boosting, we append another 10 epochs with learning rate set to 0.01. With two NVIDIA TITAN XP GPUs and Pytorch as the platform, training an IDE model and a standard PCB on Market-1501 (12,936 training images) consumes about 40 and 50 minutes, respectively. The increased training time of PCB is mainly caused by the cancellation of the last spatial down-sample operation in the Conv5 layer, which enlarges the tensor $\boldsymbol{T}$ by 4×.

4.3 Performance evaluation

We evaluate our method on three datasets, with results shown in Table 4.3. Both uniform partition (PCB) and refined part pooling (PCB+RPP) are tested.

Table 1. Comparison of the proposed method with IDE and 2 variants. pool5: output of Pool5 layer in ResNet50. FC: output of the appended FC layer for dimension reduction. $\mathcal{G}$ ($\mathcal{H}$): feature representation assembled with column vectors $\boldsymbol{g}$ ($\boldsymbol{h}$). Both $\boldsymbol{g}$ and $\boldsymbol{h}$ are illustrated in Fig. 2

Models	Feature	dim	Market-1501		DukeMTMC-reID		CUHK03	
			R-1	mAP	R-1	mAP	R-1	mAP
IDE	pool5	2048	85.3	68.5	73.2	52.8	43.8	38.9
IDE	FC	256	83.8	67.7	72.4	51.6	43.3	38.3
Variant 1	$\mathcal{G}$	12288	86.7	69.4	73.9	53.2	43.6	38.8
Variant 1	$\mathcal{H}$	1536	85.6	68.3	72.8	52.5	44.1	39.1
Variant 2	$\mathcal{G}$	12288	91.2	75.0	80.2	62.8	52.6	45.8
Variant 2	$\mathcal{H}$	1536	91.0	75.3	80.0	62.6	54.0	47.2
PCB	$\mathcal{G}$	12288	92.3	77.4	81.7	66.1	59.7	53.2
PCB	$\mathcal{H}$	1536	92.4	77.3	81.9	65.3	61.3	54.2
PCB+RPP	$\mathcal{G}$	12288	**93.8**	**81.6**	**83.3**	**69.2**	62.8	56.7
PCB+RPP	$\mathcal{H}$	1536	93.1	81.0	82.9	68.5	**63.7**	**57.5**

PCB is a strong baseline. Comparing PCB and IDE, the prior commonly used baseline in many works [44,42,37,11,32,46,47,49], we clearly observe the significant advantage of PCB: mAP on three datasets increases from 68.5%, 52.8% and 38.9% to 77.4% (+8.9%), 66.1% (+13.3%) and 54.2% (+15.3%), respectively. This indicates that integrating part information increases the discriminative ability of the feature. The structure of PCB is as concise as that of IDE, and training PCB requires nothing more than training a canonical classification network. We hope it will serve as a baseline for person retrieval task.

Refined part pooling (RPP) improves PCB especially in mAP. From Table 4.3, while PCB already has a high accuracy, RPP brings further improvement to it. On the three datasets, the improvement in rank-1 accuracy is +1.5%, +1.6%, and +3.1%, respectively; the improvement in mAP is +4.2%, +3.1%, and +3.5%, respectively. The improvement is larger in mAP than in rank-1 accuracy. In fact, rank-1 accuracy characterizes the ability to retrieve the easiest match in the camera network, while mAP indicates the ability to find all the matches. So the results indicate that RPP is especially beneficial in finding more challenging matches.

The benefit of using p losses. To validate the usage of p branches of losses in Fig. 2, we compare our method with Variant 1 which learns the convolutional descriptor under a single classification loss. Table 4.3 suggests that Variant 1 yields much lower accuracy than PCB, implying that employing a respective loss for each part is vital for learning discriminative part features.

The benefit of NOT sharing parameters among identity classifiers. In Fig. 2, PCB inputs each column vector $\boldsymbol{h}$ to a FC layer before the Softmax loss. We compare our proposal (not sharing FC layer parameters) with Variant 2 (sharing FC layer parameters). From Table 4.3, PCB is higher than Variant 2 by 2.4%, 3.3%, and 7.4% on the three datasets, respectively. This suggests that sharing parameters among the final FC layers is inferior.

Table 2. Comparison of the proposed method with the art on Market-1501. The compared methods are categorized into 3 groups. Group 1: hand-crafted methods. Group 2: deep learning methods employing global feature. Group 3: deep learning methods employing part features. * denotes those requiring auxiliary part labels. Our method is denoted by "PCB" and "PCB+RPP"

Methods	R-1	R-5	R-10	mAP
BoW+kissme [43]	44.4	63.9	72.2	20.8
WARCA[17]	45.2	68.1	76.0	-
KLFDA[18]	46.5	71.1	79.9	-
SOMAnet[1]	73.9	-	-	47.9
SVDNet[32]	82.3	92.3	95.2	62.1
Triplet Loss [15]	84.9	94.2	-	69.1
DML [40]	87.7	-	-	68.8
Cam-GAN [50]	88.1	-	-	68.7
MultiRegion [34]	66.4	85.0	90.2	41.2
PAR [41]	81.0	92.0	94.7	63.4
MultiLoss [20]	83.9	-	-	64.4
PDC* [31]	84.4	92.7	94.9	63.4
MultiScale [3]	88.9	-	-	73.1
GLAD* [35]	89.9	-	-	73.9
HA-CNN [21]	91.2	-	-	75.7
PCB	92.3	97.2	98.2	77.4
PCB+RPP	**93.8**	**97.5**	**98.5**	**81.6**

Comparison with state of the art. We compare PCB and PCB+RPP with state of the art. Comparisons on Market-1501 are detailed in Table 2. The compared methods are categorized into three groups, *i.e.*, hand-crafted methods, deep learning methods with global feature and deep learning methods with part features. Relying on uniform partition only, PCB surpasses all the prior methods, including [31,35] which require auxiliary part labeling to deliberately align parts. The performance lead is further enlarged by the proposed refined part pooling.

Comparisons on DukeMTMC-reID and CUHK03 (new training/testing protocol) are summarized in Table 3. In the compared methods, PCB exceeds [3] by +5.5% and 17.2% in mAP on the two datasets, respectively. PCB+RPP (refined part pooling) further surpasses it by a large margin of +8.6% mAP on DukeMTMC-reID and +20.5% mAP on CUHK03. PCB+RPP yields higher accuracy than "TriNet+Era" and "SVDNet+Era" [49] which are enhanced by extra data augmentation.

In this paper, **we report mAP = 81.6%, 69.2%, 57.5% and Rank-1 = 93.8%, 83.3% and 63.7% for Market-1501, Duke and CUHK03**, respectively, setting new state of the art on the three datasets. All the results are achieved under the single-query mode without re-ranking. Re-ranking methods will further boost the performance especially mAP. For example, when "PCB+RPP" is combined with the method in [48], mAP and Rank-1 accuracy on Market-1501 increases to **91.9%** and **95.1%**, respectively.

Table 3. Comparison with prior art on DukeMTMC-reID and CUHK03. Rank-1 accuracy (%) and mAP (%) are shown

Methods	DukeMTMC-reID		CUHK03	
	rank-1	mAP	rank-1	mAP
BoW+kissme [43]	25.1	12.2	6.4	6.4
LOMO+XQDA [23]	30.8	17.0	12.8	11.5
GAN [47]	67.7	47.1	-	-
SVDNet [32]	76.7	56.8	41.5	37.3
MultiScale [3]	79.2	60.6	40.7	37.0
SVDNet+Era [49]	79.3	62.4	48.7	43.5
Cam-GAN [50]	75.3	53.5	-	-
HA-CNN [21]	80.5	63.8	41.7	38.6
PCB (UP)	81.8	66.1	61.3	54.2
PCB (RPP)	**83.3**	**69.2**	**63.7**	**57.5**

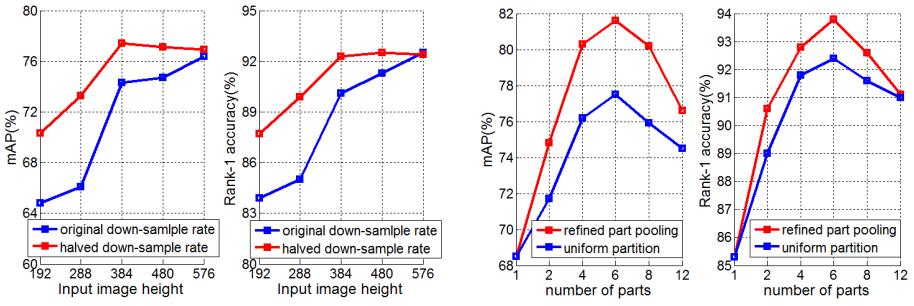
4.4 Parameters Analysis

We analyze some important parameters of PCB (and with RPP) introduced in Section 3.1 on Market-1501. Once optimized, the same parameters are used for all the three datasets.

The size of images and tensor $\boldsymbol{T}$. We vary the image size from 192×64 to 576×192, using 96×32 as interval. Two down-sampling rates are tested, *i.e.*, the original rate, and a halved rate (larger $\boldsymbol{T}$). We train all these models on PCB and report their performance in Fig. 5. Two phenomena are observed.

First, a larger image size benefits the learned part feature. Both mAP and rank-1 accuracy increase with the image size until reaching a stable performance.

Second, a smaller down-sampling rate, *i.e.*, a larger spatial size of tensor $\boldsymbol{T}$ enhances the performance, especially when using relatively small images as input. In Fig. 5 (a), PCB using 384×128 input and halved down-sampling rate achieves almost the same performance as PCB using 576×192 input and the



(a) Impact of the size (b) Impact of p

Fig. 5. Parameter analysis. (a): The impact of image size. We use the original and halved down-sampling rates. (b): The impact of number of parts p. We compare PCB with and without the refined part pooling.

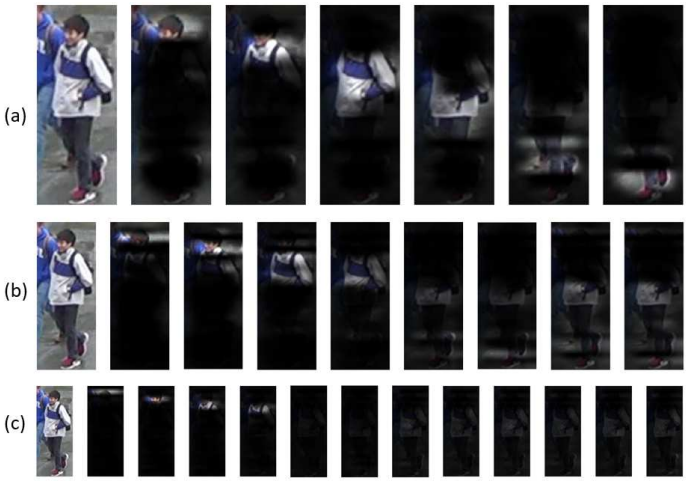


Fig. 6. Visualization of the refined parts under different p values. When $p = 8$ or 12, some parts repeat with others or become empty.

original down-sampling rate. We recommend the manipulation of halving the down-sampling rate with consideration of the computing efficiency.

The number of parts p. Intuitively, p determines the granularity of the part feature. When p=1, the learned feature is a global one. As p increases, retrieval accuracy improves at first. However, accuracy does not always increase with p, as illustrated in Fig. 5 (b). When $p = 8$ or 12, the performance drops dramatically, regardless of using refined part pooling. A visualization of the refined parts offers insights into this phenomenon, as illustrated in Fig. 6. When p increases to 8 or 12, some of the refined parts are very similar to others and some may collapse to an empty part. As a result, an over-increased p actually compromises the discriminative ability of the part features. In real-world applications, we would recommend to use $p = 6$ parts.

4.5 Induction and Attention Mechanism

In this work, when training the part classifier in Alg. 1, a PCB pre-trained with uniform partition is required. The knowledge learned under uniform partition induces the subsequent training of the part classifier. *Without PCB pre-training, the network learns to partition* $\boldsymbol{T}$ *under no induction and becomes similar to methods driven by attention mechanism.* We conduct an ablation experiment on Market-1501 and DukeMTMC-reID to compare the two approaches. Results are presented in Table 4, from which three observations can be drawn.

First, no matter which partition strategy is applied in PCB, it significantly outperforms PAR [41], which learns to partition through attention mechanism. Second, the attention mechanism also works based on the structure of PCB. Under the "RPP (w/o induction)" setting, the network learns to focus on several parts through attention mechanism, and achieves substantial improvement

Table 4. Ablation study of induction on Market-1501. PAR learns to focus on several parts to discriminate person with attention mechanisms. RPP (w/o induction) means no induction for learning the refined parts and the network learns to focus on several parts with attention mechanism. It is equivalent to PAR on the structure of PCB

Methods	Market-1501		DukeMTMC-reID	
	rank-1	mAP	rank-1	mAP
PAR [41]	81.0	63.4	-	-
IDE	85.3	68.5	73.2	52.8
RPP (w/o induction)	88.7	74.6	78.8	60.9
PCB	92.3	77.4	81.7	66.1
PCB+RPP	93.8	81.6	83.3	69.2

over IDE, which learns a global descriptor. Third, the induction procedure (PCB training) is critical. When the part classifier is trained without induction, the retrieval performance drops dramatically, compared with the performance achieved by "PCB+RPP". It implies that the refined parts learned through induction is superior to the parts learned through attention mechanism. Partitioned results with induction and attention mechanism are visualized in Fig. 1.

Moreover, for learning the part classifier without labeling information, we compare RPP with another potential method derived from Mid-level Element Mining [22,27,8]. Specifically, we follow [8] by assigning each stripe on tensor $\boldsymbol{T}$ with a pseudo part label to train the part classifier. Then we slide the trained part classifier on $\boldsymbol{T}$ to predict the similarity between every column vector on $\boldsymbol{T}$ and each part. The predicted similarity values are used for refining the uniformly-partitioned stripes of PCB, as the same in RPP. The above described approach achieves 93.0% (82.1%) rank-1 accuracy and 79.0% (66.9%) mAP on Market-1501 (DukeMTMC-reID). It also improves PCB, but is inferior to RPP. We guess the superiority of RPP originates from: given no part labels, the part classifier of RPP and the ID classifier are jointly optimized to recognize training identities, and thus gains better pedestrian discriminative ability.

5 Conclusion

This paper makes two contributions to solving the pedestrian retrieval problem. First, we propose a Part-based Convolutional Baseline (PCB) for learning part-informed features. PCB employs a simple uniform partition strategy and assembles part-informed features into a convolutional descriptor. PCB advances the state of the art to a new level, proving itself as a strong baseline for learning part-informed features. Despite the fact that PCB with uniform partition is simple and effective, it is yet to be improved. We propose the refined part pooling to reinforce the within-part consistency in each part. After refinement, similar column vectors are concluded into a same part, making each part more internally consistent. Refined part pooling requires no part labeling information and improves PCB considerably.

References

1. Barbosa, I.B., Cristani, M., Caputo, B., Rognhaugen, A., Theoharis, T.: Looking beyond appearances: Synthetic training data for deep cnns in re-identification. arXiv preprint arXiv:1701.03153 (2017)
2. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: CVPR (2017)
3. Chen, Y., Zhu, X., Gong, S.: Person re-identification by deep learning multi-scale representations. In: In International Conference on Computer Vision, Workshop on Cross-Domain Human Identification (CHI) (2017)
4. Cheng, D.S., Cristani, M., Stoppa, M., Bazzani, L., Murino, V.: Custom pictorial structures for re-identification. In: BMVC (2011)
5. Dai, J., Li, Y., He, K., Sun, J.: R-FCN: object detection via region-based fully convolutional networks. In: NIPS (2016)
6. Das, A., Chakraborty, A., Roy-Chowdhury, A.K.: Consistent Re-identification in a Camera Network. Springer International Publishing (2014)
7. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Li, F.F.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
8. Diba, A., Pazandeh, A.M., Pirsiavash, H., Gool, L.V.: Deepcamp: Deep convolutional action & attribute mid-level patterns. In: CVPR (2016)
9. Engel, C., Baumgartner, P., Holzmann, M., Nutzel, J.F.: Person re-identification by support vector ranking. In: BMVC (2010)
10. Felzenszwalb, P., McAllester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In: CVPR (2008)
11. Geng, M., Wang, Y., Xiang, T., Tian, Y.: Deep transfer learning for person re-identification. arXiv preprint arXiv:1611.05244 (2016)
12. Gheissari, N., Sebastian, T.B., Hartley, R.: Person reidentification using spatiotemporal appearance. In: CVPR (2006)
13. Gray, D., Tao, H.: Viewpoint invariant pedestrian recognition with an ensemble of localized features. In: ECCV (2008)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
15. Hermans, A., Beyer, L., Leibe, B.: In defense of the triplet loss for person re-identification. arXiv preprint arXiv: 1703.07737 (2017)
16. Insafutdinov, E., Pishchulin, L., Andres, B., Andriluka, M., Schiele, B.: Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In: ECCV (2016)
17. Jose, C., Fleuret, F.: Scalable metric learning via weighted approximate rank component analysis. In: ECCV (2016)
18. Karanam, S., Gou, M., Wu, Z., Rates-Borras, A., Camps, O., Radke, R.J.: A comprehensive evaluation and benchmark for person re-identification: Features, metrics, and datasets. arXiv preprint arXiv: 1605.09653 (2016)
19. Li, W., Zhao, R., Xiao, T., Wang, X.: Deepreid: Deep filter pairing neural network for person re-identification. In: CVPR (2014)
20. Li, W., Zhu, X., Gong, S.: Person re-identification by deep joint learning of multi-loss classification. In: IJCAI (2017)
21. Li, W., Zhu, X., Gong, S.: Harmonious attention network for person re-identification. arXiv preprint arXiv:1802.08122 (2018)
22. Li, Y., Liu, L., Shen, C., van den Hengel, A.: Mining mid-level visual patterns with deep CNN activations. International Journal of Computer Vision (2017)

23. Liao, S., Hu, Y., Zhu, X., Li, S.Z.: Person re-identification by local maximal occurrence representation and metric learning. In: CVPR (2015)
24. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S.E., Fu, C., Berg, A.C.: SSD: single shot multibox detector. In: ECCV (2016)
25. Liu, X., Zhao, H., Tian, M., Sheng, L., Shao, J., Yi, S., Yan, J., Wang, X.: Hydraplus-net: Attentive deep features for pedestrian analysis. In: ICCV (2017)
26. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR (2015)
27. M., J.O., Tuytelaars, T.: Modeling visual compatibility through hierarchical mid-level elements. In: ECCV (2016)
28. Ma, A.J., Yuen, P.C., Li, J.: Domain transfer support vector ranking for person re-identification without target camera label information. In: ICCV (2013)
29. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: ECCV (2016)
30. Ristani, E., Solera, F., Zou, R., Cucchiara, R., Tomasi, C.: Performance measures and a data set for multi-target, multi-camera tracking. In: European Conference on Computer Vision workshop on Benchmarking Multi-Target Tracking (2016)
31. Su, C., Li, J., Zhang, S., Xing, J., Gao, W., Tian, Q.: Pose-driven deep convolutional model for person re-identification. In: ICCV (2017)
32. Sun, Y., Zheng, L., Deng, W., Wang, S.: SVDNet for pedestrian retrieval. In: ICCV (2017)
33. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: AAAI (2017)
34. Ustinova, E., Ganin, Y., Lempitsky, V.: Multiregion bilinear convolutional neural networks for person re-identification. arXiv preprint arXiv: 1512.05300 (2015)
35. Wei, L., Zhang, S., Yao, H., Gao, W., Tian, Q.: GLAD: Global-local-alignment descriptor for pedestrian retrieval. ACM Multimedia (2017)
36. Wei, S.E., Ramakrishna, V., Kanade, T., Sheikh, Y.: Convolutional pose machines. In: CVPR (2016)
37. Xiao, T., Li, H., Ouyang, W., Wang, X.: Learning deep feature representations with domain guided dropout for person re-identification. In: CVPR (2016)
38. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R., Bengio, Y.: Show, attend and tell: Neural image caption generation with visual attention. In: ICML (2015)
39. Yao, H., Zhang, S., Zhang, Y., Li, J., Tian, Q.: Deep representation learning with part loss for person re-identification. arXiv preprint arXiv:1707.00798 (2017)
40. Zhang, Y., Xiang, T., Hospedales, T.M., Lu, H.: Deep mutual learning. arXiv preprint arXiv: 1705.00384 (2017)
41. Zhao, L., Li, X., Wang, J., Zhuang, Y.: Deeply-learned part-aligned representations for person re-identification. In: ICCV (2017)
42. Zheng, L., Huang, Y., Lu, H., Yang, Y.: Pose invariant embedding for deep person re-identification. arXiv preprint arXiv:1701.07732 (2017)
43. Zheng, L., Shen, L., Tian, L., Wang, S., Wang, J., Tian, Q.: Scalable person re-identification: A benchmark. In: ICCV (2015)
44. Zheng, L., Yang, Y., Hauptmann, A.G.: Person re-identification: Past, present and future. arXiv preprint arXiv:1610.02984 (2016)
45. Zheng, W., Gong, S., Xiang, T.: Reidentification by relative distance comparison. TPAMI (2013)
46. Zheng, Z., Zheng, L., Yang, Y.: Pedestrian alignment network for large-scale person re-identification. arXiv preprint arXiv: 1707.00408 (2017)

47. Zheng, Z., Zheng, L., Yang, Y.: Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In: ICCV (2017)
48. Zhong, Z., Zheng, L., Cao, D., Li, S.: Re-ranking person re-identification with k-reciprocal encoding. In: CVPR (2017)
49. Zhong, Z., Zheng, L., Kang, G., Li, S., Yang, Y.: Random erasing data augmentation. arXiv preprint arXiv: 1708.04896 (2017)
50. Zhong, Z., Zheng, L., Zheng, Z., Li, S., Yang, Y.: Camera style adaptation for person re-identification. arXiv preprint arXiv:1711.10295 (2017)