Faster Person Re-Identification

Guan'an Wang[1,2⋆], Shaogang Gong[3], Jian Cheng[4,2], and Zengguang Hou[1,2]

[1]SKLMCCS, CASIA [2]SAI, UCAS [3]QMUL [4]NLPR, CASIA
wangguanan2015@ia.ac.cn, s.gong@qmul.ac.uk,
jcheng@nlpr.ia.ac.cn, zengguang.hou@ia.ac.cn

Abstract. Fast person re-identification (ReID) aims to search person images quickly and accurately. The main idea of recent fast ReID methods is the hashing algorithm, which learns compact binary codes and performs fast Hamming distance and counting sort. However, a very long code is needed for high accuracy (*e.g.* 2048), which compromises search speed. In this work, we introduce a new solution for fast ReID by formulating a novel Coarse-to-Fine (CtF) hashing code search strategy, which complementarily uses short and long codes, achieving both faster speed and better accuracy. It uses shorter codes to coarsely rank broad matching similarities and longer codes to refine only a few top candidates for more accurate instance ReID. Specifically, we design an All-in-One (AiO) framework together with a Distance Threshold Optimization (DTO) algorithm. In AiO, we simultaneously learn and enhance multiple codes of different lengths in a single model. It learns multiple codes in a pyramid structure, and encourage shorter codes to mimic longer codes by self-distillation. DTO solves a complex threshold search problem by a simple optimization process, and the balance between accuracy and speed is easily controlled by a single parameter. It formulates the optimization target as a F_β score that can be optimised by Gaussian cumulative distribution functions. Experimental results on 2 datasets show that our proposed method (CtF) is not only 8% more accurate but also 5× faster than contemporary hashing ReID methods. Compared with non-hashing ReID methods, CtF is 50× faster with comparable accuracy. Code is available at `https://github.com/wangguanan/light-reid`.

1 Introduction

Person re-identification (ReID) [8,50] aims to match images of a person across disjoint cameras, which is widely used in video surveillance, security and smart city. Many methods [26,43,21,51,16,22,50,11,32] have been proposed for person ReID. However, for higher accuracy, most of them utilize a large deep network to learn high-dimensional real-value features for computing similarities by Euclidean distance and returning a rank list by quick-sort [13]. Quick-sort of high-dimensional deep features can be slow, especially when the gallery set is large. Table 1 shows that the query time per ReID probe image increases massively with the increase of the ReID gallery size; and counting-sort [1] is much more

⋆ This work was done when Guan'an Wang was at QMUL supervised by *Prof.* Shaogang Gong

Gallery Size	Query Time (s) Quick-Sort	Counting-Sort
1×10^3	3.4×10^{-3}	4.7×10^{-4}
1×10^4	1.0×10^{-1}	2.7×10^{-3}
1×10^5	4.3×10^{-1}	2.7×10^{-2}
1×10^6	6.4×10^{0}	2.6×10^{-1}
1×10^7	1.1×10^{2}	2.7×10^{0}
Per Sample Complexity	- $O(nlogn)$	2.6×10^{-7} $O(n)$

Table 1. ReID search time per probe image by quick-sort (real-value) and counting-sort (binary). The latter is much faster.

Code Length	Computation Time (s) Euclidean	Hamming
32	6.8×10^{-5}	2.4×10^{-6}
64	1.3×10^{-4}	2.7×10^{-6}
128	2.6×10^{-4}	2.8×10^{-6}
256	5.0×10^{-4}	3.3×10^{-6}
512	1.0×10^{-3}	4.4×10^{-6}
1,024	2.0×10^{-3}	7.1×10^{-6}
2,048	3.9×10^{-3}	1.7×10^{-5}

Table 2. Comparing Euclidean and Hamming distances, Euclidean and longer lengths are slow to compute.

efficient than quick-sort, in which the former has a linear complexity w.r.t the gallery size ($O(n)$) whilst the latter has a logarithm complexity ($O(nlogn)$).

Several fast ReID methods [5,47,41,55,4,7,56,24] have been proposed to increase ReID speed whist retaining ReID accuracy. The common main idea is hashing, which learns a binary code instead of real-value features. To sort binary codes, the inefficient Euclidean distance and quick-sort are replaced by the Hamming-distance and counting-sort [1]. Table 2 shows that computing a Hamming distance between 2048-dimensional binary-codes is 229× faster than that of a Euclidean distance between real-value features.

Different from common image retrieval tasks, which are category-level matching in a close-set, ReID is instance-level matching in an open-set (zero-shot setting). For image retrieval in the ImageNet [28], the classes of training and test sets are the same and imagery appearances of different classes diverse a lot, such as dog, car, and airplane. In contrast, the training and test ReID images have completely different ID classes without any overlap (ZSL) whilst the appearances of different persons can be very similar to subtle changes (fine-grained) on clothing, body characteristics, gender, and carried-objects. The ZSL and fine-grained characteristics of ReID require state-of-the-art hashing-based fast ReID models [24] to employ very long binary codes, *e.g.* 2048, in order to retain competitive ReID accuracy. However, the binary code length affects significantly the cost of computing Hamming distance. Table 2 shows that computing a Hamming distance between two 2048-dimensional binary codes takes 1.7×10^{-5} seconds, which is 7× slower than computing that of 32-dimensional binary codes at 2.4×10^{-6} seconds. This motivates us to solve the following problem: How to yield higher accuracy from hashing-based ReID using shorter binary codes.

To that end, we propose a novel Coarse-to-Fine (CtF) search strategy for faster ReID whilst also retaining competitive accuracy. At test time, our model (CtF) first uses shorter codes to coarsely rank a gallery, then iteratively utilises longer codes to further rank selected top candidates where the top-ranked candidates are defined iteratively by a set of Hamming distance thresholds. Thus, the long codes are only used for a decreasingly fewer matches in ranking in or-

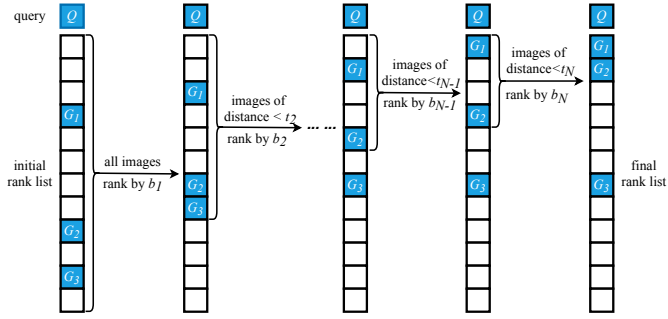

Fig. 1. A Coarse-to-Fine (CtF) hashing code search strategy to speed up ReID, where Q is a query image, $\{G_i\}_{i=1}^{3}$ are the positive images in the gallery set, $B = \{b_k\}_{k=1}^{N}$ are binary codes of lengths $L = \{l_k\}_{k=1}^{N}$, $T = \{k_i\}_{k=2}^{N}$ are Hamming distance thresholds where gallery images are selected by each t_k for further comparison by increasingly longer codes b_k.

der to reduce the overall search time whilst retaining ReID accuracy. This is an intuitively straightforward idea but not easily computable for ReID due to three difficulties: (1) Coarse-to-fine search requires multiple codes of different lengths. Asymmetrically, computing them with multiple models is both time-consuming and sub-optimal. (2) The coarse ranking must be accurate enough to minimise missing true-match candidates in fine-grained ranking whilst keeping their numbers small, thus reduce the total search time. Paradoxically, shorter codes perform much worse than longer codes in ReID task therefore hard to be sufficiently accurate. (3) The set of distance thresholds for guiding the coarse search affect both final accuracy and overall speed. How to determine *automatically* these thresholds to balance optimally accuracy and speed is both important and nontrivial.

In this work, we propose a novel All-in-One (AiO) framework together with a Distance Threshold Optimization (DTO) algorithm to simultaneously solve these three problems. The AiO framework can simultaneously learn and enhance multiple codes of different lengths in a single model. It progressively learns multiple codes in a pyramid structure, where the knowledge from the bottom long code is shared by the top short code. We promote shorter codes to mimic longer codes by both probability- and similarity- distillation. This makes shorter codes more powerful without importing extra teacher networks. The DTO algorithm solves a complex threshold search problem by a simple optimization process and the balance between search accuracy and speed is easily controlled by a single parameter. It explores a F_β score as the optimization target formulated as Gaussian cumulative distribution functions. So that we can estimate its parameters by the statistics of Gaussian probability distributions modeling the distances of positive and negative pairs. Finally, by maximizing the F_β score, we compute iteratively optimal distance thresholds.

Our contributions are: (1) We propose a novel Coarse-to-Fine (CtF) search strategy for Faster ReID, not only speeding up hashing ReID, but also improv-

ing their accuracy. To the best of our knowledge, this is the first work to introduce such search strategy into ReID. (2) A novel All-in-One (AiO) framework is proposed to learn and enhance multiple codes of different lengths in a single framework by viewing it as a multi-channel self-distillation problem. In the framework, the multiple codes are learned in a pyramid structure and shorter codes mimic longer codes via probability- and similarity- distillation loss. (3) A novel Distance Threshold Optimization (DTO) algorithm is proposed to find the optimal thresholds for coarse-to-fine search by concluding the threshold search task to a F_β distance optimization problem. The F_β score is represented with Gaussian cumulative distribution functions, whose mean and variance can be estimated by fitting a small validation set. (4) Extensive experimental results on two datasets show that, our proposed method is 50× faster than non-hashing ReID methods, 5× faster and 8% more accurate than hashing ReID methods.

2 Related Works

In this work, we wish to solve the fast ReID problem under the framework of hashing by proposing an All-in-One (AiO) hashing learning module and a Distance Threshold Optimization (DTO) algorithm. Thus, we mainly discuss the related works including non-fast person re-identification (ReID) task, fast ReID task and hashing algorithm.

Person Re-Identification. Person re-identification addresses the problem of matching pedestrian images across disjoint cameras [8]. The key challenges lie in the large intra-class and small inter-class variation caused by different views, poses, illuminations, and occlusions. Existing methods can be grouped into hand-crafted descriptors [26,43,21], metric learning methods [51,16,22] and deep learning algorithms [50,11,32,37,36,38,35]. The goal of hand-crafted descriptors is to design robust features. Metric learning aims to make a pair of true matches have a relatively smaller distance than that of a wrong match pair in a discriminant manner. Deep learning algorithms adopt deep neural networks to straightly learn robust and discriminative features in an end-to-end manner and achieve the best performance. However, all the ReID methods above learn real-value features for high accuracy, which is slow.

Hashing Algorithm. Hashing algorithm mainly divided into unsupervised and (semi-)supervised ones. Unsupervised hashing methods (LSH [6], SH [40], ITQ [19]) employ unlabeled data even no data. (Semi-)Supervised (SSH [39], BRE [17], KSH [23], SDH [30], SSGAH[34]) utilize labeled information to improve binary codes. Recently, inspired by powerful deep networks, some deep hashing methods (CNNH [42], NINH [18], DPSH [20]) have been proposed and achieve much better performance. They usually utilize a CNN to extract meaningful features, formulate the hashing function as a fully-connected layer with $tanh/sigmoid$ activation function, and quantize features by *signature* function. The framework can be optimized with a related layer or some iteration strategies. However, all the hashing methods are designed for close-set category-level

retrieval tasks, which cannot be directly used for person ReID, an open-set fine-grained search problem.

Fast Person Re-Identification. Fast ReID methods aims to search in a fast speed meanwhile obtaining accuracy as high as possible. The main idea of those methods is hashing algorithm, which learns binary code instead of real-value features. Based on the binary codes, the inefficient Euclidean distance and quick-sorting can be replaced by efficient Hamming distance and counting sort. Zheng *et al.* [47] learn cross-view binary codes using two hash functions for two different views. Wu *et al.* [41] simultaneously learn both CNN feature and hash functions to get robust yet discriminative features and similarity-preserving binary codes. CSBT [4] solves the cross-camera variations problem by employing a subspace projection to maximize intra-person similarity and inter-person discrepancies. In [55] integrate spatial information for discriminative features by representing horizontal parts to binary codes. ABC [24] improves binary codes by implicitly fits the feature distribution to a pre-defined binary one with Wasserstein distance. However, all the fast ReID methods take very long binary codes (*e.g.* 2048) for high accuracy. Different from them, we propose a coarse-to-fine search strategy which complementarily uses codes of different lengths, obtaining not only faster speed but also higher accuracy.

3 Proposed Method

In this work, we propose a coarse-to-fine (CtF) search strategy for fast and accurate ReID. For effectively implementing the strategy, we design an All-in-One (AiO) framework together with a Distance Threshold Optimization (DTO) algorithm. The former learns and enhances multiple codes of different lengths in a single framework. The latter finds the optimal distance thresholds to balance time and accuracy.

3.1 Coarse-to-Fine Search

As we illustrated in the introduction section, although the long binary codes can get high accuracy, it takes much longer time than short codes. This motivates us to think about that can we reduce the usage of long codes to further speed hashing ReID methods up. Thus, a simple but efficient solution is complementarily using both short and long codes. Here, shorter codes fast return a rough rank list of gallery, and longer codes carefully refine a small number of top candidates. Figure 1 shows its procedures.

Although the idea is straightforward, there are still three difficulties preventing it being applied to ReID. (1) Coarse-to-fine search requires multiple codes of different lengths. Asymmetrically, computing them with multiple models is both time-consuming and sub-optimal. (2) The coarse ranking must be accurate enough to minimise missing true-match candidates in fine-grained ranking whilst keeping their numbers small, thus reduce the total search time. Paradoxically, shorter codes perform much worse than longer codes in ReID task. (3) The set

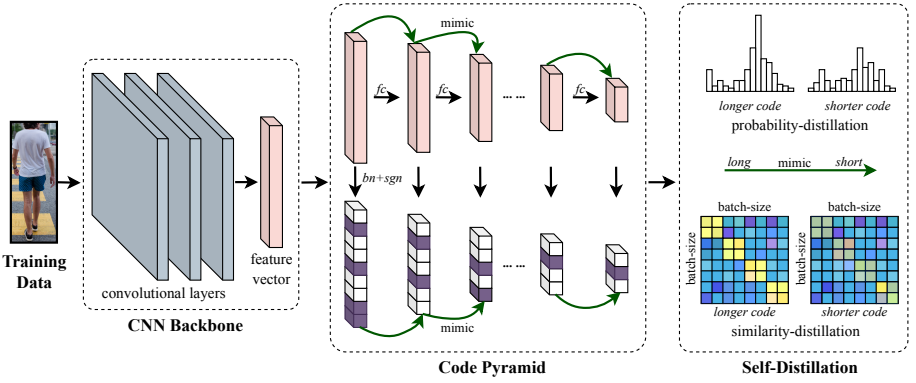

Fig. 2. All-in-One framework. It learns and enhances multiple codes of different lengths in a single framework with a code pyramid structure and self-distillation learning.

of distance thresholds for guiding the coarse search affect both final accuracy and overall speed. How to determine *automatically* these thresholds to balance optimally accuracy and speed is both important and nontrivial. To solve the problems, we propose an All-in-One (AiO) framework and a Distance Threshold Optimization (DTO) algorithm. Please see the next two parts for more details.

3.2 All-in-One Framework

The All-in-One (AiO) framework aims to simultaneously learn and enhance multiple codes of different lengths in a single model, whose architecture can be seen in Figure 2. Specifically, it first utilizes a convolutional network to extract the real-value feature vectors, then learns multiple codes of different lengths in a pyramid structure, finally enhances the codes by encouraging shorter codes mimic longer codes via self-distillation.

Learn Multiple Codes in a Pyramid Structure. The code pyramid learns multiple codes of different lengths, where the shorter codes are based on the longer codes. With such a structure, we can not only learn many codes in one shot, but also share the knowledge of longer codes with shorter codes. The equations are as below:

$$v_0 = F(x), \quad v_k = FC_k(v_{k-1}), \quad k \in 1, 2, ..., N, \tag{1}$$

where x is input image, F is the CNN backbone, N is the code number, $V = \{v_k\}_{k=1}^{N}$ are the real-value feature vectors with different lengths $L = \{l_k\}_{k=1}^{N}$, FC_k is the fully-connected layers with l_{k-1} input- and l_k output-sizes. After getting real-value features of different lengths, we can obtain their binary codes $B = \{b_k\}_{k=1}^{N}$ in the following equation.

$$b_k = sgn(bn(v_k)), \tag{2}$$

where bn is the batch normalization layer, sgn is the symbolic function. We use the batch normalization layer because it normalizes the real-value features to be symmetric to 0 and reduces the quantization loss.

Enhance Codes with Self-Distillation Learning. As we discussed in the introduction section, the coarse ranking must be accurate enough to minimise missing true-match candidates in fine-grained ranking. Inspired by [12,33], we introduce self-distillation learning to enhance the multiple codes in a single framework without importing extra teacher network. Different from conventional distillation models, which imports an extra large teacher network to supervise a small student network, we perform distillation learning in a single network and achieve better performance, which is important for fast ReID.

Specifically, our self-distillation learning is composed of a probability- and a similarity- distillation. The probability-distillation transfers the instance-level knowledge in a from of softened class scores. Its formulation is given by

$$\mathcal{L}_{pro} = \frac{1}{N-1}\sum_{k=1}^{N-1}\mathcal{L}_{ce}(\sigma(\frac{z_{k+1}}{T}), \sigma(\frac{\hat{z}_k}{T})), \tag{3}$$

where $\mathcal{L}_{ce}(\cdot,\cdot)$ denotes the cross-entropy loss, σ is the softmax function, $\hat{z}_k/z_{k+1}$ means the output logits of the binary code b_k/b_{k+1}, $\hat{z}_k$ means it act as a teacher and fixed during training, T is a temperature hyperparameter, which is set 1.0 empirically. The similarity-distillation transfers the knowledge of relationship from longer codes to shorter one, whose formulation is in Eq.(4). This is motivated by that as an image search task, ReID features should also focus on the relationship among samples, *i.e.* to what extent the sample A is similar/dissimilar to sample B.

$$\mathcal{L}_{sim} = \frac{1}{N-1}\sum_{k=1}^{N-1}\sum_{i,j}||\frac{1}{l_{k+1}}G_{k+1}^{i,j} - \frac{1}{l_k}\hat{G}_k^{i,j}||^2, \tag{4}$$

where $G_k^{i,j}/G_{k+1}^{i,j}$ is the Hamming distance between b_k^i/b_{k+1}^i and b_k^j/b_{k+1}^j, $b_{k/k+1}^{i/j}$ is the binary code of image x_i/x_j with length l_k/l_{k+1}, the $\hat{G}$ means that G acts as a label and is fixed during the optimization process, thus contributes nothing to the gradients.

Overall Objective Function and Training. Recent progresses on ReID have shown the effectiveness of the classification [50] and triplet [11] losses. Thus, our final objective function includes our proposed probability- and similarity-distillation losses together with the classification and triplet losses as the final objective function. The formulation can be found in Eq.(5),

$$\mathcal{L} = \mathcal{L}_{ce} + \mathcal{L}_{tri} + \lambda_1\mathcal{L}_{prob} + \lambda_2\mathcal{L}_{sim} \tag{5}$$

Considering that the mapping function sgn in Eq.(2) is discrete and Hamming distance in Eq.(2) is not differentiable, a natural relaxation [20] is utilised in Eq.(5) by replacing sgn with $tanh$ and changing the Hamming distance to the inner-product distance. Finally, our All-in-One framework can be optimized in an end-to-end way by minimizing the loss in Eq.(5).

Algorithm 1. Distance Threshold Optimization

Input: Trained Model in Eq.(2), Validation Data (X_v, Y_v)
Output: Thresholds $\{T_i\}_{i=2}^{N}$

for $k = \{1, 2, ..., n-1\}$ **do**
  B_k: Extract binary codes of validation set with length l_k via Eq.(2)
  D^r: Hamming distances of relevant pairs (b_k^i, b_k^j), where $y^i = y^j$
  D^n: Hamming distances of non-relevant pairs (b_{k-1}^i, b_{k-1}^j), where $y^i \neq y^j$
  PDF^r, PDF^n: Probability distribution function of D^r and D^n of in Eq.(7)
  CDF^r, CDF^n: Cumulative Distribution Function of D^r and D^n in Eq.(7)
  t_{n+1}: Maximize F_β score in Eq.(8) and return t_{n+1}
return $T = \{t_i\}_{i=2}^{N}$

3.3 Distance Threshold Optimization

After getting the multiple codes of different lengths $B = \{b_i\}_{i=1}^{N}$, we can perform the Coarse-to-Fine (CtF) search. There are two tips in CtF search, *i.e.* high accuracy and fast speed. For fast speed, the candidate number returned by coarse search should be small. For high accuracy, the candidates returned by coarse search should include relevant images as more as possible. But the two requirements are naturally conflicting. Thus, it is important to find the proper thresholds to optimally balance the two targets, *i.e.* both high accuracy and fast speed. One simple solution is brute search via cross-validation. However, the search space is too large. For example, if we have multiple binary codes of lengths $L = \{32, 128, 512, 2048\}$, the complexity of the brute search will be $\prod_L > 4 \times 10^9$ times.

In this part, we propose a novel Distance Threshold Optimization (DTO) algorithm which solves the time-consuming brute parameter search task with a simple optimization process. Specifically, inspired by [9], we first explicitly formulate the two sub-targets as two scores in Eq.(6), *i.e.* precision (P) and recall (R) scores. Then we balance the two sub-targets by mixing the two scores with a single parameter β and get F_β score in Eq.(6).

$$P = \frac{TP}{TP + FP}, \quad R = \frac{TP}{TP + FN}, \quad F_\beta = (\beta^2 + 1)\frac{PR}{\beta^2 P + R} \tag{6}$$

Here, TP is the number of relevant images in the candidates, FP is the number of non-relevant images in the candidates and FN is not retrieved relevant samples. As we can see, the precision score P means the rate of relevant images in the candidates. Usually a high P means a small candidate number, which is good for fast speed. The recall score R represents the rate of returned relevant samples in the total relevant samples. A high R score means more returned relevant samples, which is important for high accuracy. The F_β mixed the precision and recall scores with a parameter β, which considers both speed and accuracy.

$$PDF(t) = \frac{1}{\sigma\sqrt{2\pi}} exp(-\frac{(t-u)^2}{\sigma\sqrt{2}}), \quad CDF(t) = \frac{1}{2}(1 + erf\frac{t-u}{\sigma\sqrt{2}}) \tag{7}$$

$$F_\beta = \frac{CDF^r(\beta^2+1)}{CDF^n + CDF^r + \beta^2(1-CDF^n+CDF^r)} \tag{8}$$

Considering that TP/FP/FN are statistics which cannot be optimized, we replace them with two Gaussian cumulative distribution functions in form of Eq.(7) (right), whose parameters u and σ are estimated by fitting a validation set using the Gaussian probability distribution function in Eq.(7) (left). Finally, by maximizing the F_β in Eq.(8), we can get the optimal distance thresholds $T = \{t_k\}_{k=2}^N$ balanced by β.

4 Experiments

4.1 Dataset and Evaluation Protocols

Datasets. We extensively evaluate our proposed method on two common datasets (Market-1501 [49] and DukeMTMC-reID [52]) and one large-scale dataset (Market-1501+500k[49]). The Market-1501 dataset contains 1,501 identities observed under 6 cameras, which are splited into 12,936 training, 3,368 query and 15,913 gallery images. The Market-1501+500k enlarges the gallery of Market-1501 with extra 500,000 distractors, making it more challenging for both accuracy and speed. DukeMTMC-reID contains 1,404 identities with 16,5522 training, 2,228 query and 17,661 gallery images.

Evaluation Protocols. For accuracy, we use standard metrics including Cumulative Matching Characteristic (CMC) curves and mean average precision (mAP). All the results are from a single query setting. To evaluate speed, we use average query time per image, including distance computation and sorting time. For fair evaluation, we do not use any parallel algorithm for distance computation and sorting.

4.2 Implementation Details

We implemented our method with Pytorch on a PC with 2.6Ghz Intel Core i5 CPUs, 10GB memory, and a NVIDIA RTX 2080Ti GPU. For a fair comparison and following [25,24], we use ResNet50 [10] as the CNN backbone. In training stage, each image is resized to 256×128 and augmented by horizontal flip and random erasing [53]. A batch data includes 64 images from 16 different persons, where every person includes 4 images. The lengths $L = \{l_k\}_{k=1}^N$ of multiple codes are empirically set $\{32, 128, 512, 2048\}$. The margin in the triplet loss in Eq.(5) is 0.3. The framework is optimized by Adam [15] with total epochs 120. Its initial learning rate is 0.00035, which is warmed up for 10 epochs and decayed to its $0.1\times$ and $0.01\times$ at 40 and 70 epochs. We randomly split the training data into a training and a validation set according to 6 : 4, then decide the parameters via cross-validation, After that, we train our method with all training data. λ_1 and λ_2 in Eq.(5) are set as 1.0 and 1,000, and β in Eq.(8) is set 2.0. The three paramters are decided via cross validation. Code is available at github[1].

[1] `https://github.com/wangguanan/light-reid`

4.3 Comparisons with Non-Hashing ReID Methods

Methods	Code		Market-1501			DukeMTMC-reID		
	Type	Length	R1(%)	mAP(%)	***Q.Time***(s)	R1(%)	mAP(%)	***Q.Time***(s)
PSE [29]	**R**	1,536	78.7	56.0	-	-	-	-
PN-GAN [27]	**R**	1,024	89.4	72.6	-	73.6	53.2	-
IDE [50]	**R**	2,048	88.1	72.8	-	69.4	55.4	-
Camstyle [54]	**R**	2,048	88.1	68.7	-	75.3	53.5	-
PIE [48]	**R**	2,062	87.7	69.0	-	79.8	62.0	-
BoT [25]	**R**	2,048	94.1	85.7	2.2×10^0	86.4	76.4	2.0×10^0
SPReID [14]	**R**	10,240	92.5	81.3	-	84.4	71.0	-
PCB [32]	**R**	12,288	93.8	81.6	6.9×10^0	83.3	69.2	6.3×10^0
VPM [31]	**R**	14,336	93.0	80.8	-	83.6	72.6	-
CtF (ours)	**B**	2,048	**93.7**	**84.9**	$\mathbf{4.6 \times 10^{-2}}$	**87.6**	**74.8**	$\mathbf{3.7 \times 10^{-2}}$

Table 3. Comparisons with non-hashing ReID methods using real-value features of different lengths on Market-1501 and DukeTMTC-reID. **B**: binary code, **R**: real-value feature. Longer real-value features have higher accuracy but slower query speed. Our model CtF (including AiO) has very fast query speed (two orders of magnitude faster) and comparable accuracy with non-hashing ReID methods.

Non-hashing ReID use longer real-value features, such as 2048-dimensional $float64$ features, for a better accuracy. This significantly affects their speed, *i.e.* query time. Table 3 shows that our proposed CtF (including AiO) method is significantly faster than non-hashing ReID methods (two orders of magnitude). CtF also achieves very competitive accuracy with close Rank-1 (93.7% vs. 94.1%) and mAP (87.6% vs. 86.4%) scores of the best non-hashing ReID mehtod BoT [25] on Market-1501 and DukeMTMC-reID, and better than all the other non-hashing methods using different feature length, of which 5 methods have features shorter than 2,062 (PSE [29], IDE [50], PN-GAN [27], CamStyle [54], PIE [48]) and 3 methods have features longer than 10,240 (SPReID [14], PCB [32], VPM [31]). Overall, longer feature usually contributes to higher accuracy but with slower speed. For example, SPReID, PCB and VPM take features longer than 10,240 and achieves 92%-93% and 83%-84% Rank-1 scores on Market-1501 and DukeMTMC-reID datasets, respectively. The others utilize features no longer than 2,048 achieving Rank-1 score less than 92% and 80%. On the other hand, the query speed of those methods with long features is much slower. For example, PCB takes $6.9s$ and $6.3s$ for query each image on the two datasets respectively. This is 3-4× slower than IDE with $2s$ on either dataset. Specifically, CtF performs much faster than non-hashing methods and significantly, it achieves much better accuracy than comparable length real-value feature model. For example, CtF achieves 93.7%/87.6% Rank-1 scores on Market-1501/DukeMTMC-reID, as compared to BoT having 94.1%/86.4% respectively. This is because CtF (including AiO) utilizes all-in-one framework together with coarse-to-fine search

strategy, which not only learns powerful binary code, but also complementarily uses short and long codes for both high accuracy and fast speed.

4.4 Comparisons with Hashing ReID Methods

Methods	Code Length	Market-1501			DukeMTMC-reID		
		R1(%)	mAP(%)	**Q.Time**(s)	R1(%)	mAP(%)	**Q.Time**(s)
DRSCH [44]	512	17.1	11.5	-	19.3	13.6	-
DSRH [45]	512	27.1	17.7	-	25.6	18.6	-
HashNet [3]	512	29.2	19.1	-	40.8	28.6	-
DCH [2]	512	40.7	20.2	-	57.4	37.3	-
CSBT [4]	512	42.9	20.3	-	47.2	33.1	-
PDH [55]	512	44.6	24.3	-	-	-	-
DeepSSH [46]	512	46.5	24.1	-	-	-	-
ABC [24]	512	69.4	48.5	9.8×10^{-2}	69.9	52.6	7.5×10^{-2}
ABC [24]	2,048	81.4	64.7	2.8×10^{-1}	82.5	61.2	2.0×10^{-1}
CtF (ours)	AiO+32	60.0	37.7	$\underline{3.4 \times 10^{-2}}$	49.5	28.7	$\underline{2.3 \times 10^{-2}}$
	AiO+128	88.9	71.0	4.2×10^{-2}	78.6	59.4	3.2×10^{-2}
	AiO+512	92.8	82.2	9.8×10^{-2}	85.4	71.6	7.5×10^{-2}
	AiO+2,048	<u>93.7</u>	<u>85.4</u>	2.8×10^{-1}	<u>87.7</u>	<u>75.7</u>	2.0×10^{-1}
	AiO+CtF	**93.7**	**84.0**	$\mathbf{4.6 \times 10^{-2}}$	**87.6**	**74.8**	$\mathbf{3.7 \times 10^{-2}}$

Table 4. Comparisons with state-of-the-art hashing ReID methods on Market-1501 and DukeTMTC-reID. AiO+k means learning multiple codes with all-in-one framework, but querying with only the code of length l_k. Aio+CtF not only learns multiple codes with all-in-one framework, but also query with coarse-to-fine search strategy. Our AiO+CtF achieve a good balance between accuracy and speed.

Hashing ReID methods learn binary codes using a hashing algorithm. Binary codes are good for speed but sacrifice model accuracy. To mitigate this problem, the state-of-the-art hashing ReID methods usually employ long codes such as 2048. In binary coding, 2048 is relatively very long as compared to the more commonly used 512 length, unlike in real-value feature length compared above. Table 4 shows that CtF (with AiO) not only achieves the best accuracy (even compared to much shorter code length used by other hashing methods), but also is significantly faster than existing hashing ReID methods (even compared to the same code length used by other hashing methods). Overall, hashing ReID methods usually perform much worse than non-hashing methods. For example, best non-hashing ReID methods achieves 94.1% and 86.4% Rank-1 scores on Market-1501 and DukeMTMC-reID respectively. But the best hashing ReID method only obtains 81.4% and 82.5% Rank-1 scores. Moreover, existing hashing ReID models can increase accuracy by using longer code length and compromising speed. For example, ABC with 512-dimensional binary codes achieves 69.4%/69.9% Rank-1 scores and $9.8/7.5 \times 10^{-2}s$ query time per probe image. When using

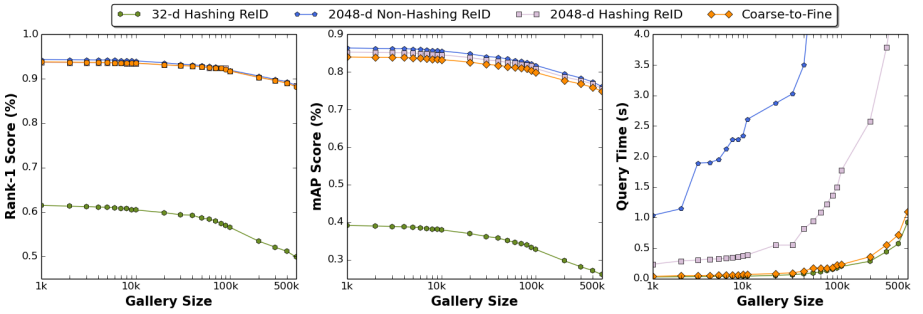

Fig. 3. Experimental results on large-scale ReID dataset Market-1501+500k. Our Coarse-to-Fine (CtF) get a high accuracy comparable with non-hashing ReID method of long code and fast speed comparable with hashing ReID method of short code.

2048 binary codes, its Rank-1 scores increase to 81.4%/82.5% with query time slow down to $2.8/2.0 \times 10^{-1}s$. This observation is also verified with our method CtF (with AiO) using different code lengths. Importantly, our method CtF (with AiO) significantly outperforms all existing hashing ReID methods in terms of both accuracy (R1 12.3% or 5.1% better) and speed (5× faster). Specifically, CtF with AiO achieves high accuracy very close to AiO without CtF using 2048 code length, but yields significant speed advantage that is comparable to much shorter 128 binary code length. CtF obtains 93.7% and 87.6% Rank-1 scores, similar to AiO without CtF of a fixed 2048 length at 93.7% and 87.7%.

4.5 Evaluation on Large-Scale ReID

Gallery size affects significantly ReID search accuracy and speed. To show the effectiveness of our proposed Coarse-to-Fine (CtF) search strategy, we evaluated it on a large-scale ReID dataset Market1501+500k. The dataset is based on the Market-1501 and enlarged with 500,000 distractors. The experimental results are shown in Figure 3. We can observe the following phenomenons.

Firstly, with the increase of gallery size, for all methods, the Rank-1 and mAP scores decrease, and the ReID speed per probe image slows down gradually. The reason is that more gallery images is more likely to contain more difficult samples. They make ReID search more challenging. Also, the extra gallery images significantly increase the time for computing all the distance comparisons and sorting required for ReID each probe image. Secondly, the non-hashing method with 2048-D real-value feature achieves the best accuracy but the worst time. This is because the real-value feature is more discriminative but slow to compute and sort. Thirdly, for hashing ReID methods, the 2048-D binary code obtains comparable ReID accuracy to that of the non-hashing model, but 10× faster. This is because Hamming distances and counting sort are faster to compute. ReID speed of 32-D binary code is 5× faster than that of 2048-D binary codes, but its accuracy drops dramatically. Finally, the proposed CtF model achieves

a comparable accuracy to that of the non-hashing method but the advantage of similar speed to that of a hashing ReID method of 32-D binary code. Critically, the advantage is independent of the gallery size. Overall, these experiments demonstrate the effectiveness of CtF for a large-scale ReID task.

4.6 Model Analysis

AiO	CP	SD	Feature Type	Rank-1(%) 32	128	512	2048	CtF	mAP(%) 32	128	512	2048	CtF
×	×	×	**B**										
✓	×	×	**B**	25.5	84.8	92.3	93.8	92.5	33.9	67.5	81.4	85.3	75.1
✓	✓	×	**B**	54.4	87.8	92.7	93.8	93.0	35.0	72.2	81.7	85.3	80.2
✓	✓	✓	**B**	60.0	88.9	92.9	93.8	93.7	37.7	71.0	82.0	85.3	84.0
upper bound			**R**	82.7	90.9	93.4	94.2	-	66.7	78.9	84.3	85.4	-

Table 5. Analysis of the All-in-One (AiO) framework. **CP**: learn multiple codes in a pyramid structure, otherwise separate models. **SD**: enhance binary codes via self-distillation. **B** and **R** mean binary codes and real-value features, respectively.

Analysis of AiO. The All-in-One (AiO) framework aims to learn and enhance multiple codes of different lengths in a single model. It uses code pyramid (CP) structure and self-distillation (SD) learning. Results are in Table 5. Firstly, longer codes contribute to better accuracy. This can be seen in all settings no matter whether CP or SD is used and what code type is. Secondly, when using short codes, real-value features is much better than binary ones. But for long codes, they obtain similar accuracy. For example, the 32-dimensional real-value feature obtains 82.7% Rank-1 score, outperforming the 32-dimensional binary code by 60%, where the latter achieved only 25.5%. But when using 2048 code length, binary codes and real-valure features both achieve approx. Rank-1 94% and mAP 84%. This suggests that the quantization loss of short codes is significantly worse than that of longer codes. Thirdly, learning with code pyramid (CP) structure or self-distillation (SD) improves short codes significantly. For example, CP+SD boosts the 32-dimensional binary codes from 25.5% to 60.0% in Rank-1 score, upto 35% gain. It is evident that both code pyramid (CP) structure and self-distillation (SD) learning contribute to the effectiveness of the coarse-to-fine (CtF) search strategy, and significantly improve model performance.

Analysis of DTO. We further analyzed parameter β of the Distance Threshold Optimization (DTO) algorithm, which controls the balance between ReID accuracy and speed. Figure 4 show the model accuracy and speed using different β value on Market-1501 and DukeMTMC-reID. Firstly, it is evident that the value of β has a good control of accuracy and speed, increasing β slows down the speed but improves accuracy. For example, when $\beta = 10^{-2}$, ReID is fastest at approx. 0.03 and 0.02 seconds to ReID each probe image on Market-1501

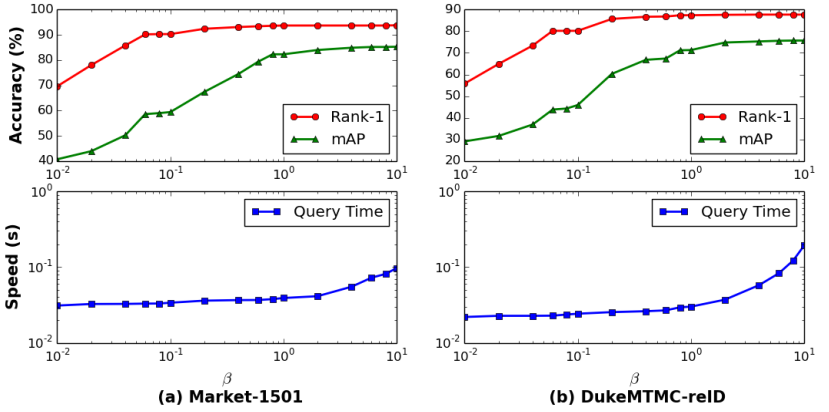

Fig. 4. Accuracy and Speed controlled by β. With the increase of β, the accuracy increases and speed becomes slow gradually.

and DukeMTMC-reID, but with mAP scores only at 40% and 30%. In contrast, $\beta = 10^1$ gives high mAP 85% and 75%, but the query speed is 5× slower at approx. 0.1 and 0.2 seconds. Secondly, when β is close to 10^0, Rank-1 and mAP are almost peaked with a good balance on speed.

5 Conclusion

In this work, we proposed a novel Coarse-to-Fine (CtF) search strategy for faster person re-identification whilst also improve accuracy on conventional hashing ReID. CtF first coarsely ranks a gallery using shorter binary codes, then iteratively utilises longer binary codes to further refine on ranking selected top candidates with increasing accuracy. In order to implement the CtF strategy, a novel All-in-One (AiO) framework together with a Distance Threshold Optimization (DTO) algorithm are formulated. The former simultaneously learns and enhances multiple binary codes of different lengths in a single model. The latter solves the complex parameter search task by a simple optimization process. The balance between search accuracy and speed is easily controlled by a single parameter. Extensive experiments show that our method is 5× faster than existing hashing ReID methods but achieves comparable accuracy with non-hashing ReID models that are 50× slower.

Acknowledgement

This work was partially supported by the Youth Innovation Promotion Association of CAS (2020140), the Alan Turing Institute Turing Fellowship, and Vision Semantics Ltd.

References

1. Bajpai, K., Kots, A.: Implementing and analyzing an efficient version of counting sort (e-counting sort). International Journal of Computer Applications **98**(9) (2014)
2. Cao, Y., Long, M., Liu, B., Wang, J.: Deep cauchy hashing for hamming space retrieval. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1229–1237 (2018)
3. Cao, Z., Long, M., Wang, J., Yu, P.S.: Hashnet: Deep learning to hash by continuation. In: 2017 IEEE International Conference on Computer Vision (ICCV). pp. 5609–5618 (2017)
4. Chen, J., Wang, Y., Qin, J., Liu, L., Shao, L.: Fast person re-identification via cross-camera semantic binary transformation. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5330–5339 (2017)
5. Chen, J., Wang, Y., Wu, R.: Person re-identification by distance metric learning to discrete hashing. In: 2016 IEEE International Conference on Image Processing (ICIP). pp. 789–793 (2016)
6. Datar, M., Immorlica, N., Indyk, P., Mirrokni, V.S.: Locality-sensitive hashing scheme based on p-stable distributions. In: Scg ?4: Proceedings of the Twentieth Symposium on Computational Geometry. pp. 253–262 (2004)
7. Fang, W., Hu, H.M., Hu, Z., Liao, S., Li, B.: Perceptual hash-based feature description for person re-identification. Neurocomputing **272**(1), 520–531 (2018)
8. Gong, S., Cristani, M., Yan, S., Loy, C.C.: Person Re-Identification (2014)
9. Goutte, C., Gaussier, E.: A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In: European Conference on Information Retrieval. pp. 345–359. Springer (2005)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016)
11. Hermans, A., Beyer, L., Leibe, B.: In defense of the triplet loss for person re-identification. arXiv preprint arXiv:1703.07737 (2017)
12. Hinton, G.E., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
13. Hoare, C.A.: Quicksort. The Computer Journal **5**(1), 10–16 (1962)
14. Kalayeh, M.M., Basaran, E., Gökmen, M., Kamasak, M.E., Shah, M.: Human semantic parsing for person re-identification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1062–1071 (2018)
15. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
16. Koestinger, M., Hirzer, M., Wohlhart, P., Roth, P.M., Bischof, H.: Large scale metric learning from equivalence constraints. In: 2012 IEEE conference on computer vision and pattern recognition. pp. 2288–2295. IEEE (2012)
17. Kulis, B., Darrell, T.: Learning to hash with binary reconstructive embeddings. In: International Conference on Neural Information Processing Systems. pp. 1042–1050 (2009)
18. Lai, H., Pan, Y., Liu, Y., Yan, S.: Simultaneous feature learning and hash coding with deep neural networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2015)
19. Lazebnik, S.: Iterative quantization: A procrustean approach to learning binary codes. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 817–824 (2011)

20. Li, W.J., Wang, S., Kang, W.C.: Feature learning based deep supervised hashing with pairwise labels. In: International Joint Conference on Artificial Intelligence. pp. 1711–1717 (2016)
21. Liao, S., Hu, Y., Zhu, X., Li, S.Z.: Person re-identification by local maximal occurrence representation and metric learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2197–2206 (2015)
22. Liao, S., Li, S.Z.: Efficient psd constrained asymmetric metric learning for person re-identification. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3685–3693 (2015)
23. Liu, W., Wang, J., Ji, R., Jiang, Y.G.: Supervised hashing with kernels. In: Computer Vision and Pattern Recognition. pp. 2074–2081 (2012)
24. Liu, Z., Qin, J., Li, A., Wang, Y., Gool, L.V.: Adversarial binary coding for efficient person re-identification. In: 2019 IEEE International Conference on Multimedia and Expo (ICME). pp. 700–705 (2019)
25. Luo, H., Gu, Y., Liao, X., Lai, S., Jiang, W.: Bag of tricks and a strong baseline for deep person re-identification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 0–0 (2019)
26. Ma, B., Su, Y., Jurie, F.: Covariance descriptor based on bio-inspired features for person re-identification and face verification. Image and Vision Computing **32**(6-7), 379–390 (2014)
27. Qian, X., Fu, Y., Xiang, T., Wang, W., Qiu, J., Wu, Y., Jiang, Y.G., Xue, X.: Pose-normalized image generation for person re-identification. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 661–678 (2018)
28. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M.S., Berg, A.C., Fei-Fei, L.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision **115**(3), 211–252 (2015)
29. Sarfraz, M.S., Schumann, A., Eberle, A., Stiefelhagen, R.: A pose-sensitive embedding for person re-identification with expanded cross neighborhood re-ranking. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 420–429 (2018)
30. Shen, F., Shen, C., Liu, W., Shen, H.T.: Supervised discrete hashing. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 37–45 (2015)
31. Sun, Y., Xu, Q., Li, Y., Zhang, C., Li, Y., Wang, S., Sun, J.: Perceive where to focus: Learning visibility-aware part-level features for partial person re-identification. pp. 393–402 (2019)
32. Sun, Y., Zheng, L., Yang, Y., Tian, Q., Wang, S.: Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline). In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 480–496 (2018)
33. Tung, F., Mori, G.: Similarity-preserving knowledge distillation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1365–1374 (2019)
34. Wang, G., Hu, Q., Cheng, J., Hou, Z.: Semi-supervised generative adversarial hashing for image retrieval. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 469–485 (2018)
35. Wang, G., Yang, S., Liu, H., Wang, Z., Yang, Y., Wang, S., Yu, G., Zhou, E., Sun, J.: High-order information matters: Learning relation and topology for occluded person re-identification. arXiv preprint arXiv:2003.08177 (2020)
36. Wang, G., Yang, Y., Cheng, J., Wang, J., Hou, Z.: Color-sensitive person re-identification. In: IJCAI'19 Proceedings of the 28th International Joint Conference on Artificial Intelligence. pp. 933–939 (2019)

37. Wang, G., Zhang, T., Cheng, J., Liu, S., Yang, Y., Hou, Z.: Rgb-infrared cross-modality person re-identification via joint pixel and feature alignment. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 3622–3631 (2019)
38. Wang, G., Zhang, T., Yang, Y., Cheng, J., Chang, J., Hou, Z.: Cross-modality paired-images generation for rgb-infrared person re-identification. In: AAAI 2020 : The Thirty-Fourth AAAI Conference on Artificial Intelligence (2020)
39. Wang, J., Kumar, S., Chang, S.F.: Semi-supervised hashing for large-scale search. IEEE Transactions on Pattern Analysis and Machine Intelligence **34**(12), 2393–2406 (2012)
40. Weiss, Y., Torralba, A., Fergus, R.: Spectral hashing. In: International Conference on Neural Information Processing Systems. pp. 1753–1760 (2008)
41. Wu, L., Wang, Y., Ge, Z., Hu, Q., Li, X.: Structured deep hashing with convolutional neural networks for fast person re-identification. Computer Vision and Image Understanding **167**, 63–73 (2017)
42. Xia, R., Pan, Y., Lai, H., Liu, C., Yan, S.: Supervised hashing for image retrieval via image representation learning (2014)
43. Yang, Y., Yang, J., Yan, J., Liao, S., Yi, D., Li, S.Z.: Salient color names for person re-identification. In: European conference on computer vision. pp. 536–551. Springer (2014)
44. Zhang, R., Lin, L., Zhang, R., Zuo, W., Zhang, L.: Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification. IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society **24**(12), 4766 (2015)
45. Zhao, F., Huang, Y., Wang, L., Tan, T.: Deep semantic ranking based hashing for multi-label image retrieval. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1556–1564 (2015)
46. Zhao, Y., Luo, S., Yang, Y., Song, M.: Deepssh: Deep semantic structured hashing for explainable person re-identification. In: 2018 25th IEEE International Conference on Image Processing (ICIP). pp. 1653–1657 (2018)
47. Zheng, F., Shao, L.: Learning cross-view binary identities for fast person re-identification. In: IJCAI'16 Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence. pp. 2399–2406 (2016)
48. Zheng, L., Huang, Y., Lu, H., Yang, Y.: Pose-invariant embedding for deep person re-identification. IEEE Transactions on Image Processing **28**(9), 4500–4509 (2019)
49. Zheng, L., Shen, L., Tian, L., Wang, S., Wang, J., Tian, Q.: Scalable person re-identification: A benchmark. In: Proceedings of the IEEE international conference on computer vision. pp. 1116–1124 (2015)
50. Zheng, L., Yang, Y., Hauptmann, A.G.: Person re-identification: Past, present and future. arXiv preprint arXiv:1610.02984 (2016)
51. Zheng, W.S., Gong, S., Xiang, T.: Reidentification by relative distance comparison. IEEE transactions on pattern analysis and machine intelligence **35**(3), 653–668 (2013)
52. Zheng, Z., Zheng, L., Yang, Y.: Unlabeled samples generated by gan improve the person re-identification baseline in vitro. arXiv preprint arXiv:1701.07717 (2017), `https://academic.microsoft.com/paper/2949257576`
53. Zhong, Z., Zheng, L., Kang, G., Li, S., Yang, Y.: Random erasing data augmentation. arXiv preprint arXiv:1708.04896 (2017)
54. Zhong, Z., Zheng, L., Zheng, Z., Li, S., Yang, Y.: Camera style adaptation for person re-identification. In: 2018 IEEE/CVF Conference on Computer Vision and

Pattern Recognition. pp. 5157–5166 (2018), https://academic.microsoft.com/paper/2963289251
55. Zhu, F., Kong, X., Zheng, L., Fu, H., Tian, Q.: Part-based deep hashing for large-scale person re-identification. IEEE Transactions on Image Processing **26**(10), 4806–4817 (2017)
56. Zhu, X., Wu, B., Huang, D., Zheng, W.S.: Fast open-world person re-identification. IEEE Transactions on Image Processing **27**(5), 2286–2300 (2018)