Progressive Local Filter Pruning for Image Retrieval Acceleration

Xiaodong Wang, Zhedong Zheng, Yang He, Fei Yan, Zhiqiang Zeng, and Yi Yang, *Senior Member, IEEE*

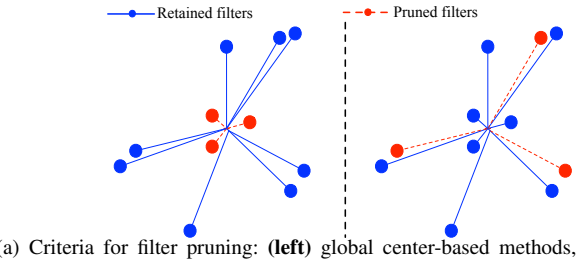
***Abstract*—Most image retrieval works aim at learning discriminative visual features, while little attention is paid to the retrieval efficiency. The speed of feature extraction is key to the real-world system. Therefore, in this paper, we focus on network pruning for image retrieval acceleration. Different from the classification models predicting discrete categories, image retrieval models usually extract continuous features for retrieval, which are more sensitive to network pruning. Such different characteristics of the retrieval and classification models make the traditional pruning method sub-optimal for image retrieval acceleration. Two points are critical for pruning image retrieval models: preserving the local geometry structure of filters and maintaining the model capacity during pruning. In view of the above considerations, we propose a Progressive Local Filter Pruning (PLFP) method. Specifically, we analyze the *local* geometry of filter distribution in every layer and select redundant filters according to one new criterion that the filter can be replaced locally by other similar filters. Furthermore, to preserve the model capacity of the original model, the proposed method *progressively* prune the filter by decreasing the scale of filter weights gradually. We evaluate our method on four scene retrieval datasets, *i.e.*, Oxford5K, Oxford105K, Paris6K, and Paris106K, and one person re-identification dataset, *i.e.*, Market-1501. Extensive experiments show that the proposed method (1) preserves the original model capacity while pruning (2) and achieves superior performance to other widely-used pruning methods.**

***Index Terms*—Image Retrieval, Person Re-identification, Network Pruning, Local Geometry, Deep Learning**

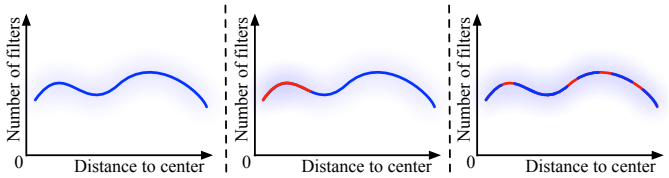
I. Introduction

IMAGE representation learned by Convolutional Neural Network (CNN) has become dominant in many retrieval fields, such as person re-identification and scene retrieval, due to the discriminative power [1], [2], [3], [4], [5], [6], [7]. However, training and testing the CNN-based model usually demands expensive computation resources for the fast calculation [8], [9], [10]. It remains challenging for CNN-based applications to the platforms with limited resources, *e.g.*, cellphones and self-driving cars [11]. To accelerate the inference, researchers resort to pruning the redundant filters and simplifying the CNN-based model to find a trade-off between performance and efficiency [8], [12], [13], [14], [15]. Nevertheless, the existing network pruning methods mostly pay attention to the image classification problem. Pruning image retrieval models remains under-explored.

In this paper, we intend to fill the gap, and focus on network pruning methods for image retrieval acceleration. Compared with the image classification model, of which the output is a discrete category, the image retrieval task aims at extracting continuous features. Therefore, the image retrieval model is sensitive to the pruning operation in nature, which would lead to unexpected feature distribution changes and degrade the subsequent image matching performance. This paper addresses two main challenges in pruning image retrieval models. First,



(a) Criteria for filter pruning: **(left)** global center-based methods, (**right**) our method.



(b) Filter distributions of different methods: (**left**) original filter distribution, (**middle**) global center-based methods, (**right**) our method.

Fig. 1: A comparison between the conventional global center-based methods and the proposed method. **(a)** Circles denote different filters, and lines indicate the distance of filters to the geometric center. For global center-based methods (left), red filters close the geometric center tend to be removed, while our method (right) tends to prune filters according to local relationships. **(b)** Furthermore, we show pseudo filter distributions of different methods. The red curve denotes redundant filters to be pruned. The global center-based methods (middle) will largely change the original filter distribution (left) if the red part is removed, while our method (right) properly maintains the original filter distribution.

Work done during the visiting at University of Technology Sydney. This paper was supported by China Scholarship Council (No. 201908350025), Fuxiaquan National Independent Innovation Demonstration Zone collaborative platform project (No. 3502ZCQXT2021009), National Natural Science Foundation of Fujian Province (Nos. 2021J011186, 2020J01266), Natural Science Foundation of Xiamen (No. 3502Z20227073), Scientific Research Fund of Fujian Provincial Education Department (No. JAT200486), the University Industry Research Fund of Xiamen (No. 2022CXY0416), A*STAR SERC Central Research Fund (SC23/21-1073CI).

Xiaodong Wang is with the college of Computer and Information Engineering, Xiamen University of Technology, Xiamen 361024, China (e-mail:xdwangjsj@xmut.edu.cn)

Zhedong Zheng, Yang He, and Yi Yang were with ReLER Lab, University of Technology Sydney, NSW 2007, Australia. Zhedong Zheng is with the School of Computing, National University of Singapore, Singapore 118404. Yang He is with the Centre for Frontier AI Research (CFAR), A*STAR, Singapore 138632. Yi Yang is with the College of Computer Science and Technology, Zhejiang University, Hangzhou 310058, China.

Fei Yan and Zhiqiang Zeng are with the college of Computer and Information Engineering, Xiamen University of Technology, Xiamen 361024, China.

it is critical to preserve the local geometry structure of lters in the original model . In other words, the lter distribution of the pruned model should be the same as the original model. The existing methods mainly adopt the global center-based pruning approaches, which rank the lters according to the distance to the geometric center [8], [12], [16]. However, as shown in Figure 1(b), they suffer from destroying the original lter distribution. The pruning procedure of our method and global center-based methods are illustrated in Figure 1(a). We could observe that the global center-based methods tend to prune lters closed to the geometric center, resulting in a signi cant change in lter distributions (Figure 1(b), middle). In contrast, our method (Figure 1(b), right) can adequately maintain the original lter distributions (Figure 1(b), left). Second,maintaining the model capacity during pruning remains challenging The "lottery" mechanism [14], [17] proposes to drop speci c channels and argues that the key is the weight initialization. Iterative training and rewinding is necessary for this line of methods. In this way, the models of different iterations usually have totally different feature spaces from the original model. In parallel, some pruning techniques [8], [16] argue that the essential lies in the training process and drop the weight by deploying dynamic masks. The masks directly set redundant lters to zeros, which also affects the learned feature space largely. In this paper, we start from a well-trained model and remove redundant lters by decreasing the weight scale gradually. It could be viewed as one mild pruning method instead of setting the lter to zeros immediately. The model capacity, therefore, is preserved especially for the high-proportion pruning demands.

In an attempt to overcome the above-mentioned challenges, we propose Progressive Local Filter Pruning (PLFP) to accelerate CNN models for image retrieval. 1) To maintain the distribution of the original model, we propose the local lter pruning criterion, which is different from conventional global-based criteria. 2) Unlike existing methods, we do not prune the model with zero masks or retrain the model from lottery initialization. Instead, we leverage the pre-trained knowledge and adopt one progressive pruning policy by gradually decreasing the weight scales. To summarize, this paper has the following contributions.

- We propose a Progressive Local Filter Pruning (PLFP) method to leverage the local geometry structure of the lter distribution. According to the local similarity, we iteratively nd the redundant lters which could be replaced by the neighbor lters. We intend to preserve the lter distribution in the original model.
- To better utilize the priori knowledge in the pre-trained model, we propose a lter weight decreasing strategy to progressively prune the redundant lters. The progressive pruning allows the pruned model to recover the model capacity, especially when dropping a large proportion of lters. The ablation study veri es the effectiveness of the progressive pruning strategy.
- The extensive experiments on several benchmark datasets demonstrate the effectiveness of the proposed method. The proposed method has achieved the best performance on all four scene retrieval datasets, comparing to the compared pruning methods. Besides, we also arrive at the best pruning performance after removing 88.9% FLOPs on the person re-id benchmark, and surpass the second-best compared methods by over 8.4% mAP and 6.0% Rank-1.

The rest of this paper is organized as follows. Section II reviews the related work. Section III introduces the proposed progressive local lter pruning framework in detail. Experimental results are conducted and discussed in Section IV, followed by the conclusions in Section V.

II. Related Work

Image retrieval. Recently, owing to the ef ciency in feature extraction, Convolutional Neural Networks (CNN) has been widely applied to learn the visual representation from data in image retrieval communities [18], [19], [20], [21], [22]. Babenko et al. [18] propose to leverage the well-trained CNN model to explore the common knowledge for image retrieval tasks. Gu et al. [20] further promote the performance of image retrieval by global CNN representations and a joint loss network, while Linet al. [23] add a learnable key point module to mine salient areas. Besides, CNN also demands large-scale training data [24]. Therefore, Radenovi et al. [25] propose to utilize the 3D model to help data collection, which could greatly save the cost of manual annotation.

Besides, in the eld of person re-identi cation [26], [27], [28], [29], [30], CNN also largely improves the performance. Similar to image retrieval, CNN-based person re-identi cation targets at generating discriminative person representations using metric learning [27], [31] or classi cation losses [26]. Some studies introduce multi-scale deep learning frameworks to explore the discriminative features at different locations and scales [29], [32]. For instance, Shi et al. [29] apply a similarity constraint to the Siamese network, which is able to learn local and global representations simultaneously. Inspired by the recent development of the Generative Adversarial Networks (GANs) [33], some works propose to improve the re-identi cation embeddings by leveraging the augmented data [28], [34], [35], [36], domain adaption [37], [38], [39], [40]. Unfortunately, extracting feature CNN descriptors suffers from huge computation costs on million-scale even billion-scale data, hampering its practicability for real-world applications [41]. Although the CNN-based representations have shown effectiveness on image retrieval tasks, they usually consume expensive computation resources, and little attention is paid to the computational cost of the image retrieval model.

Network acceleration. There are several previous works on accelerating CNNs, such as matrix decomposition [42], dynamic inference [43], structure design [44], quantization methods [45], [46], knowledge distillation [47], [48], [49], [50], and network pruning methods [8], [51], [52], [53], [54], [55], [56], [57], [58], [59], [60]. Among them, pruning methods have attracted much attention as they can properly reduce computation cost and accelerate the inference by removing unnecessary lters or connections. According to the pruning policy, previous pruning methods can be roughly categorized

Fig. 2: The geometry interpretation of the proposed method. Given a pre-trained model, for a speci c layer (the second layer in this example), we rst calculate the local geometry for each lter and select redundant lters (orange lled circles) with small local property (more details are in Section III). Then we deploy the lter weight decreasing policy to adjust the weight scale of the selected lters (orange lled squares). After that, we ne-tune the model to give the potential important lters chances to be recti ed (green lled square), maintaining the model capacity. This " lter selection, lter decreasing, and ne-tuning" process is iteratively performed till the weight of redundant lters gradually converges to zeros. Finally, during inference, we could obtain a slim model after removing the zero-weight lters.

into hard-pruning and soft-pruning methods. The early research on hard-pruning starts at the brain damage [61] and brain surgeon [62] techniques. Han et al. remove connections with low-weight [63], while Li et al. determine the importance of lters by the l_1-norm values [12]. Some researchers propose to rank lters according to the contributions to the objective loss function [64] or the reconstruction error on the feature maps [13]. Srinivas et al. [65] and He et al. [16] propose to analyze network redundancy according to the mutual information among different lters. Another line of pruning method, soft-pruning, is a sort of training compatibility method. Soft-pruning does not drop the pruned lters immediately but utilizes a dynamic mask to set candidate lters to zeros temporally. The critical lters could be recovered later [8], [66]. The method could also be smoothly embedded into the training framework of traditional CNN models. Furthermore, He et al. determine the redundancy of each lter by its global closeness to other lters [16]. However, the soft-pruning methods still drop the lter via setting the weights to zeros and may largely compromise the well-trained weight from the original model, especially when pruning a large number of lters.

III. Progressive Local Filter Pruning

We let W be a set of lters of the whole network with L convolutional layers and denote $W^l = \{w_1^l; w_2^l; \cdots; w_{C^l}^l\}$ as a set of lters of the l-th convolutional layer, where w_i^l is the i-th lter and C^l is the number of the lters. Let W_{pr} be the lter set of the pre-trained network. Most previous hard-pruning methods [12], [64] generally remove the lters or channels from the pre-trained models. Given a dataset $X = \{(x_i; y_i)\}_{i=1}^N$ with N training samples and the pruning rate P^l for the l-th layer, hard-pruning methods aim at optimizing the following objective function:

$$\min_W \frac{1}{N} \sum_{i=1}^{N} \ell(W; (x_i; y_i)); \quad s.t. |W| \leq \sum_{l=1}^{L} P^l |W_{pr}^l| \tag{1}$$

where $\ell(\cdot)$ is the loss function (we adopt the contrastive loss and triplet loss in this paper), $|\cdot|$ represents the cardinalities of the lter set. Typically, hard-pruning methods directly remove the selected lters from the pre-trained model. Then they ne-tune the pruned model to complement the performance degradation. Noticing that once the lters are pruned, they will not be updated during the following ne-tuning procedure. Such a coarse pruning strategy will seriously reduce the model capacity [8].

Intuitively, some lters may be less important in the pre-trained model, but their importance may rise after the network structure changes, e.g., removing some lters or connections. Therefore, pruning methods should give lters more chances to adjust themselves according to network structure alternations dynamically. Motivated by the above observation and soft-pruning technology [8], we proposed to prune lters in a soft type manner by solving the following problem:

$$\min_W \frac{1}{N} \sum_{i=1}^{N} \ell(W; (x_i; y_i)) \quad s.t. \|W\|_0 \leq \sum_{l=1}^{L} P^l |W_{pr}^l|; \; |W| = |W_{pr}|; \tag{2}$$

where $\|W\|_0$ denotes the number of non-zero lters in W. $|W| = |W_{pr}|$ indicates that the soft-pruned lter set W still has a one-to-one mapping relation with the initial lter set W_{pr}. Note that we do not remove any lters during training; instead we gradually set them to zeros. In this way, we could achieve a similar pruning effect as Eq.(1).

To optimize the objective function in Eq.(2), soft-pruning methods rst select redundant lters by some global center-based importance criteria, such as l_1-norm, l_2-norm, and l_p-norm. Then they zero the selected lters and ne-tune the model again, making it possible to recover the potential important lters. However, as discussed in Section I, such global center-based criteria may select lters in the high-density regions, resulting in a volatile lter distribution. Besides, the zero-pruning strategy in these soft-pruning methods may largely compromise the kept weight in the pruned model from that in the pre-trained model.

To solve the above-mentioned problems, we propose a

Algorithm 1 Local Filter Selection

Require: The original filter set W^l, the prune rate P^l for the l-th layer and the nearest neighbor number k

Ensure: The selected filter subset $\hat{W}^l$ with the best local property preservation.

Initialize the selected filter subset $\hat{W}^l = \{\}$
Initialize the rest filter subset $W^l = \{\underline{w}_1^l, \underline{w}_2^l, \ldots, \underline{w}_{C^l}^l\}$ by W^l
Construct an undirected graph $G = \langle V, E \rangle$ with the vertex set $V = \{\underline{w}_1^l, \underline{w}_2^l, \ldots, \underline{w}_{C^l}^l\}$ and the edge set E are defined as: $\forall w_i, w_j \in V, 1 \le i, j \le |V|$,

$$E_{ij} = \begin{cases} d(w_i, w_j), & \text{if } i \ne j \\ 0, & \text{Otherwise} \end{cases}$$

while $|\hat{W}^l| < P^l C^l$ **do**
  $T = \arg\min_{i; \underline{w}_i^l \in W^l} \Omega(\underline{w}_i^l)$
  if $|T| == 1$ **then**
    $i = T(1)$
  else
    $i = \arg\min_{i; i \in T} \sum_{j=1}^{|W^l|} E_{ij}$
  end if
  $\hat{W}^l \leftarrow \hat{W}^l \cup \underline{w}_i^l$, $W^l \leftarrow W^l \setminus \underline{w}_i^l$
  Delete node $\underline{w}_i^l$ in V and the incident edges from G
end while
return $\hat{W}^l$

progressive local filter pruning method. Our method contains two components: The first is the Local Filter Selection to locate the redundant filters. The second is the Filter Weight Decreasing method to prune filters. Specifically, we identify redundant filters by their local geometry relationships. The candidate filters are not immediately dropped. Instead, we adopt the weight decreasing policy to decrease the weight scale of the selected filters. In this way, we effectively inherit prior knowledge in pre-trained model and progressively decrease the impact of the candidate filters on the network. Meanwhile, some potential important filters could have chances of recovering to the original scale in the following training process. After several iterations of training, the weight of redundant filters gradually converges to zeros. Finally, we can obtain the compressed model by removing the zero-weight filters. A brief illustration of the pruning process for a single convolutional layer is shown in Figure 2.

1) Local filter selection. We intend to find redundant filters according to the mutual information. Inspired by previous works on the local manifold learning [67], [68], which aims at locality structure exploring, we argue that the local geometry of filters could reflect more accurate mutual information. Instead of considering the relationship between all filters in [16], we focus on the local geometry between neighbor filters. Intuitively, if one filter shares the similar local property with the neighbors, the filter could be replaced by the neighbor filters. For the filter W^l of the l-th layer, we could formulate its local property as the sum of the local relation across every single filter:

$$\Psi(W^l) = \sum_{w_i^l \in W^l} \Omega(w_i^l) = \sum_{w_i^l \in W^l} \frac{1}{|U(w_i^l)|} \sum_{w_j^l \in U(w_i^l)} d(w_i^l, w_j^l); \quad (3)$$

where $d(w_i^l, w_j^l)$ denotes the distance function to model the local relationship between w_i^l and w_j^l; $U(w_i^l)$ represents the neighborhood of w_i^l, such as k nearest neighbors and k nearest neighbors. In this paper, we deploy the Euclidean distance $d(w_i^l, w_j^l) = \|w_i^l - w_j^l\|_2$ and k nearest neighbors for $U(w_i^l)$.

Based on the previous analysis, we intend to prune filters that obtain the small local property to preserve the original filter distribution of the network. Concretely, given the pruning rate P^l for the l-th layer, we want to obtain the optimal filter subset after pruning $P^l C^l$ filters, keeping the original local property. The layer-wise filter selection criterion could be formulated as:

$$\min_{W^l} \Psi(W^l) - \Psi(W_{pr}^l) \quad \text{s.t.} \sum_{l=1}^{L} \|W^l\|_0 \le P^l |W_{pr}^l|; \; |W| = |W_{pr}|; \quad (4)$$

Note that every time we change the filter W, some filters are pruned and the local neighbor relationship is also changed. To solve the Eq. (4), one naive way is to sort filters according to the local geometry distance $\Omega(w_i^l)$ in Eq.(3), and then select $P^l C^l$ filters with the smallest local property. However, this may lead to an elimination of filters in the densest area. Meanwhile, the pruned filter W may be sub-optimal. To optimize the objective, we propose to update the filter selection in an iterative way. Specifically, we sample one filter in W^l according to Eq. (4), re-evaluate the local property of each remaining filter, and repeat the optimization procedure till $P^l C^l$ filters are sampled. If two or more filters are obtaining the same local geometry distance, we calculate their global distance in [16] (the global distance of filter w_i^l is the total distance between w_i^l and the other filters in the same layer) and sample the filter with the minimum global distance score. The detailed algorithm is provided in Algorithm 1.

Advantages of local filter selection. The conventional pruning methods usually apply the global center-based criterion, e.g., l_1-norm and l_p-norm to the clustering center. We argue that the global center-based criteria may prune the critical filters which are close to the center, and change the original filter distribution. In contrast, the proposed method focuses on keeping filter diversity as well as preserving the filter distribution of the original model. We, therefore, leverage the local geometry as the indicator to search the candidate filters. The candidates with small local property could be replaced by their neighbors. In this way, after pruning, the model could recover the original filter distribution.

2) Filter weight decreasing. We could obtain the candidate filter subset $\hat{W}^l$ for pruning by Algorithm 1. One way is to remove these filters directly following the hard-pruning methods [12], [64]. However, the operation may impact the capacity of the well-trained model, especially when a large proportion of filters is dropped. Although several soft-pruning methods, e.g., [8], propose to leverage the dynamic mask, which does not drop the filter until the training completion, the side effect caused by setting the filter to zeros also compromises the model training process. To solve this problem, we propose an effective strategy to gradually reduce the value of candidate filters by multiplying a constant decay factor

(0 1). Formally, for the l-th layer, we update each selected filter w_i^l as w_i^l, where $w_i^l \in \hat{W}^l$. After the scale of the candidate filter is decreased, we then fine-tune the network for one epoch. The local geometry score between filters is calculated again and is used to re-select the pruned filters. In other words, the wrongly-selected filters are provided more chances to recover the original scale. This "decreasing and fine-tuning" process is iteratively performed, till the weights of redundant filters gradually converge to zeros. By integrating the filter weight decreasing strategy with the selected filters according to Eq.(4), we could re-write our training objective function in Eq.(2) as follows:

$$\min_{W} \frac{1}{N} \sum_{i=1}^{N} \ell(W; (x_i, y_i))$$
$$\text{s.t. } W = \{\bar{W}, \hat{W}\}; \quad \hat{W} \in \{0\}; \quad 0 < \gamma < 1; \quad \|W\|_0 < \sum_{l=1}^{L} P^l |W_{pr}^l|; \quad (5)$$
$$|W| = |W_{pr}|;$$

where $\hat{W}$ and $\bar{W}$ refer to the selected filters and the rest filters in W across the whole network, respectively.

It is worth noting that since we iteratively update the model in every epoch, some weights are multiplied by γ multiple times and will gradually decrease to zeros.

Advantages of filter weight decreasing. According to the convolution operation, the filter weight decreasing actually decreases the contribution of the selected filter. For instance, the original output is $w * x$, and the output of the weight decreasing filter is $\gamma w * x$, where $*$ denotes the convolution operation. The contribution of the selected filter is also decreased by γ times. If $\gamma = 0$, the filter weight decreasing will equal to the conventional hard-pruning methods. If $\gamma = 1$, the network will not be pruned. The proposed pruning, therefore, could be viewed as one mild strategy of the hard-pruning method. We progressively decrease the contribution of the selected filters, while providing chances of recovering the potential important filters to keep the model capacity. In Section IV-C, we further provide the ablation study on the effectiveness of the filter weight decreasing.

IV. Experiments

We apply our pruning method on two sorts of CNN-based image retrieval applications, i.e., scene retrieval and person re-identification. The following experiments are conducted with two kinds of networks, i.e., VGG-16 [69] and ResNet-50 [70], on five benchmarks, i.e., Oxford5K [71], Oxford105K [71], Paris6K [72], Paris106K [72], and Market-1501 [73]. On the selected datasets, the main task is the cross-view image matching, which could be formulated as a metric learning problem. The target is to map the images of different cameras to shared space. In this space, the embeddings of the same location should be close, while the embeddings of different locations should be apart. We impose two kinds of networks with the triplet loss [35] and contrastive loss [25] for re-identification and scene retrieval, respectively. Note that this paper mainly focuses on network pruning for image retrieval. In other words, we try to keep the pruned model as small as possible with the least performance degradation.

1). Datasets: The datasests used in our experiment are described as follows:

Scene retrieval data: 1) training dataset: Following [25], we impose the Structure-from-Motion 3D (SfM3D) dataset used in [74] for our training samples, which are derived from Flickr and contains 7.4 million images. The initial dataset SfM3D contains overlapping classes, such as images in Oxford5K and Paris6K datasets. After removing overlapping classes, there are nearly 120k images from 713 classes, where 551 and 162 classes are randomly selected for training and validation, respectively. We select around 2,000 and 1,700 images for training and validation queries per epoch, respectively. 2) testing datasets: We evaluate the proposed method on four prevailing scene retrieval datasets, namely Oxford5K [71], Paris6K [72], Oxford105K [71], and Paris106K [72]. The Oxford5K dataset contains 5062 Oxford building images collected from Flickr. Similar to Oxford5K, the Paris6K dataset consists of 6412 images from Paris landmarks. Both of these two datasets have 55 query images corresponding to 11 buildings/landmarks. The Oxford105K and Paris106K datasets are generated by adding 1 million distractor images from Flickr to Oxford5K and Paris6K, respectively.

Person re-identification data: The Market-1501 dataset is used to evaluate the performance of the proposed method in terms of person re-identification. This dataset contains 32,668 images with 1,501 different individuals, and each individual is captured in a university by at most six cameras and is present in at least two cameras. All images are automatically detected by the Deformable Part Model (DPM) detector [75]. The misalignment problem is common, and the dataset is close to the realistic settings. 751 individuals are selected for the training set and 750 individuals are chosen for the testing set without overlapping. On average, every individual in the training set has 17.2 photos. In total, there are 19732 testing images, 3368 query images, and 12936 training images.

2). Compared methods: To verify the effectiveness, we compare the proposed method with the competitive pruning approaches, including (1) HFP, the hard type filter pruning method, which selects and deletes redundant filters according to the l_1-norm criterion [12]. (2) SFP, the soft type filter pruning method. Redundant filters are set to zeros instead of direct deletion [8]. (3) FPGM, the filter pruning via geometric median [16]. (4) Taylor. Filters are ranked by Taylor expansion, which is built on the contribution of filters on network loss [64]. The pruned model is based on widely-used retrieval code, such as (1) Person re-identification baseline: Our implementation is similar to [35], which imposes a ResNet-50 as the backbone network. (2) Scene retrieval baseline, which uses VGG-16 as the backbone network with the contrastive loss function [25].

A. Results on Person Re-identification

Setting: Three state-of-the-art pruning methods, that is, SFP [8], FPGM [16], and HFP [12], are introduced for comparison. Following [8], all the convolutional layers are pruned with the same pruning rate P at the same time. We test all the comparison pruning methods with P varying from

TABLE I: Performance evaluation of the compared pruning methods on Market-1501 using ResNet-50 initialized on the Imagenet dataset. For each layer, the larger pruning rate indicates more filters will be dropped. `FLOPs(%)' and `Parameters(%)' denote the FLOPs and parameters drop between the pruned model and the baseline model. Our method outperforms other comparison methods, especially when a large proportion of filters is pruned. The best results are highlighted in bold.

Pruning rate(%)	Methods	mAP(%)	Rank-1(%)	Rank-5(%)	Rank-10(%)	FLOPs(%)	Parameters(%)
0	Baseline [35]	70.81	86.63	93.74	96.05	0	0
10	SFP [8]	69.08	85.45	93.85	95.78	14.50	12.92
	FPGM [16]	70.23	86.40	93.97	95.81		
	HFP [12]	69.08	85.33	93.88	96.17		
	Ours (k=1)	70.06	86.22	94.18	95.99		
	Ours (k=5)	70.38	86.25	94.18	96.26		
	Ours (k=10)	**70.53**	**86.58**	**94.39**	**96.29**		
20	SFP [8]	69.03	84.77	93.74	95.69	28.25	24.83
	FPGM [16]	70.07	86.07	93.79	95.81		
	HFP [12]	67.93	85.51	93.97	96.14		
	Ours (k=1)	69.60	85.60	93.91	96.08		
	Ours (k=5)	69.76	85.78	93.94	95.93		
	Ours (k=10)	**70.23**	**86.07**	**93.59**	**95.69**		
50	SFP [8]	65.85	83.05	92.96	95.52	62.45	55.57
	FPGM [16]	65.31	83.02	91.98	94.66		
	HFP [12]	57.22	77.82	90.29	93.74		
	Ours (k=1)	**66.32**	**83.85**	**93.29**	**95.64**		
	Ours (k=5)	66.14	83.64	92.84	94.93		
	Ours (k=10)	66.09	83.88	93.11	95.55		
90	SFP [8]	48.02	71.17	86.88	90.91	88.85	74.28
	FPGM [16]	45.31	68.26	84.74	89.64		
	HFP [12]	47.24	70.57	85.09	89.85		
	Ours (k=1)	**56.47**	**77.25**	**89.46**	**92.84**		
	Ours (k=5)	50.98	72.06	87.11	91.33		
	Ours (k=10)	49.29	70.31	86.40	91.24		

$f10\%; 20\%; 50\%; 90\%g$. For the soft-pruning methods, i.e., SFP, FPGM, and our method, we embed the pruning operation into training procedure, and prune filters while fine-tuning for 100 epochs. To conduct a fair comparison, for the hard-pruning method, i.e., HFP, we prune network once and fine-tune it with the same number of epochs as the soft-pruning methods. Our method contains two hyper-parameters, i.e., k and . We tune k from $f1; 5; 10g$ and set $= 1 \quad 10^{-2}$ when $P \; 6 \; 50\%$ and $= 3 \quad 10^{-1}$ when $P > 50\%$, respectively.

Following [35], we deploy ResNet-50 as the backbone network for our person re-identification baseline, and replace the last average pooling layer and fully-connected layer with an adaptive max-pooling layer. We employ the widely used triplet loss function [76] and SGD to train the network with an initial learning rate $_0 = 0:001$, momentum 0.9, margin 0.3, the pool size of 128, and the batch size of 32.

Results: As shown in Table I, in most cases the soft-pruning methods, i.e., SFP, FPGM, and our method, can greatly improve the accuracy than the hard-pruning method, i.e., HFP, indicating that keeping the connections of redundant filters while pruning is quite important for retaining the model capacity. When fewer filters are pruned, the global-based geometric pruning method, i.e., FPGM, is slightly better than SFP by preserving crucial filters with small magnitude. However, FPGM tends to degrade the accuracy with the pruning rate increasing. Particularly, when 90% of filters are pruned, FPGM fails SFP. The reason may be that FPGM globally measures the importance of filters according to their distance to the geometric center, which tends to hurt the filter distribution when a large proportion of filter is pruned. Our method is consistently better than the other methods for almost all the pruning rate P. In particular, our method brings a bigger advantage in getting smaller models over the other pruning methods. For example, when pruning 50% of filters, our method outperforms the second best method, i.e., SFP, nearly 0.5% mAP, and just increases the Rank-5 error by 0.45% than the baseline. As the pruning rate increasing, our proposed local pruning method obtains much better results than the comparison methods. For instance, our method with $k = 1$ surpasses the second best method, i.e., SFP, over 8% mAP and 6% Rank-1 accuracy respectively, when pruning 90% filters and 88.9% FLOPs[1] have been reduced. This demonstrates that the local geometry criterion is beneficial to preserve the original filter distribution, which is consistent with observations in Section I. In addition, our method contains two critical components, i.e., local ranking and weight decreasing. We present a detailed discussion of them in Section IV-C.

B. Results on Scene Retrieval

Setting: We employ the same backbone network, i.e., VGG-16, as [25] for the scene retrieval baseline, which is pre-trained on the building dataset, i.e., Structure-from-Motion 3D (SfM3D) [74]. The pre-trained model is provided by the author[2]. We follow the conventional pipeline in [25] and adopt the contrastive loss function as the objective function. Adam [77] is used to train the model with an initial learning rate $_0 = 5 \quad 10^{-6}$, exponential decay $r = _0 \exp(\; 0:01t)$ over

[1] https://github.com/sovrasov/ ops-counter.pytorch
[2] https://github.com/ lipradenovic/cnnimageretrieval-pytorch

TABLE II: Performance evaluation of the compared pruning methods using VGG-16 on various scene retrieval datasets, e.g., Oxford5K, Oxford105K, Paris6K, and Paris106K in terms of mAP (%). The proposed method obtains the best results comparing with other pruning methods. The best results are highlighted in bold.

Methods	Datasets				FLOPs(%)#
	Oxford5K	Oxford105K	Paris6K	Paris106K	
Baseline [25]	82.45	77.86	81.37	74.82	-
HFP [12]	75.25	68.89	71.78	60.61	50.70
Taylor [64]	76.15	71.07	72.51	62.36	52.39
SFP [8]	75.26	70.65	74.02	62.15	51.59
FPGM [16]	73.62	68.83	72.23	62.54	52.99
Ours	77.67	72.34	74.12	64.34	57.37

64:34% mAP on Paris106K, where 57:37% FLOPs is reduced. The performance drop of our method is limited with 4:78% mAP from the baseline [25] on Oxford5K. Similar results are observed on other datasets, where our method saves more FLOPs and still achieves a better performance over other pruning methods, which demonstrates the effectiveness of the proposed method.

Fig. 3: The pruning sensitivity of different methods with varying pruning rates. One low-level convolutional layer (left) and one high-level convolutional layer (right) are selected for comparison. We could observe that our method has advantages of exploring the potential redundancy of each layer and can achieve relatively higher mAP than other pruning methods.

epoch t, momentum 0:9, margin 0:85, the batch size of 5, and the pool size for hard-negative mining is set to 2,000. During testing, we extract the multi-scale representations from the images of different scale factors without whitening operation, i.e., $f p\frac{1}{2}; 1; \sqrt{2}g$.

Three state-of-the-art pruning methods are chosen for comparison, i.e., HFP [12], Taylor [64], and SFP [8]. Following [12], we investigate the pruning sensitivity of each convolutional layer and manually choose the best pruning rates. Concretely, for each layer, we tune the pruning rate from $f10\%; 20\%; \ldots; 90\%g$. For each method, we first record the border performance value, where the mAP relatively drops over 3%. Then, we calculate the pruning threshold by averaging the border performance values across the comparison methods. The best pruning rate for each method is the maximum value in the tuning range with larger mAP than the pruning threshold. The pruning sensitivity results over two convolutional layers, i.e., one low-level and one high-level layer, are shown in Figure 3. For Taylor [64], which globally ranks filters through all convolution layers, we set the pruning rate to 40%. For our method, the hyper-parameters k and ϵ are set to 10 and 5×10^{-1}, respectively. Similar to the comparison strategy in Section IV-A, for soft-pruning methods, i.e., SFP and our method, we train network and progressively prune filters for 100 epochs. For the hard-pruning methods, i.e., HFP and Taylor, we prune network once and train the compressed model with the same number of epochs as the soft-pruning methods.

Results: As shown in Table II, we summarize the comparison results on the Oxford5K, Oxford105K, Paris6K, and Paris106K datasets. Our method achieves the best results on all the selected datasets, e.g., 77:67% mAP on Oxford5K and

C. Ablation Study

Our method contains two components, i.e., local filter selection (controlled by hyper-parameter k) and filter weight decreasing (controlled by hyper-parameter ϵ). A large k indicates that more neighbor filters are taken into consideration. Meanwhile, the value of ϵ affects the speed of the weight decreasing. We provides detailed case studies to help understand these components and to determine the hyper-parameters.

TABLE III: Performance comparison with our proposed two components, i.e., local filter selection and weight decreasing, on Market-1501 using ResNet-50. We observe that these components jointly works well for filter pruning and can boost the pruning performance.

Methods	Performance			
with local filter selection?	7	X	7	X
with weight decreasing?	7	7	X	X
Rank-1(%)	68.44	71.06	74.50	77.25
mAP(%)	45.18	48.08	54.03	56.47

Are local filter selection and weight decreasing necessary? In our PLFP, local filter selection and weight decreasing are imposed to preserve the original filter distribution and utilize the model capacity, respectively. To study the effectiveness of them, we first remove these two components in PLFP and conduct a new pruning method with the global-geometric filter framework in FPGM [16], namely GP. Then, we manually add one component at a time into GP to study their influence. Table III shows the results on Market-1501 with 90% filters pruned. We can observe that GP with local filter selection (the third column) and GP with weight decreasing (the fourth column) constantly outperform GP, indicating the importance of leveraging local geometry information and weight decreasing for filter pruning. Besides, GP with our proposed two components (the fifth column) achieves the best mAP over the others. This demonstrates that our proposed two components jointly work well for filter pruning and can obtain much encouraging pruning performance.

How to determine the value of k? From Table I, we could observe that our method is sensitive to the parameter k. To

Fig. 4: The parameter sensitivity analysis of k on the Market-1501 dataset. We observe that our method performs slightly better with a large k (i.e., $k \geq 10$) when fewer filters (i.e., pruning rate $P < 50\%$) are pruned. In contrast, it tends to achieve higher performance with a small k (i.e., $k < 10$) when more filters are removed (i.e., $P \geq 50\%$).

investigate the importance of parameter k, we chose different k values from $\{1, 5, 10, \ldots, 1000, 5000\}$ on Market-1501 with pruning rate P varying from $\{10\%, 20\%, 50\%, 90\%\}$. We set $= 1 \times 10^{-2}$ when $P \leq 50\%$ and $= 3 \times 10^{-1}$ when $P > 50\%$ respectively. The results are shown in Figure 4. Concretely, if P is relatively small, e.g., $P < 50\%$, a large k value will lead to better performance. In contrast, a small k value is the first choice for the high-proportion pruning demande.g., $P \geq 50\%$. We speculate that when a small proportion of filters is selected, a large k could utilize more local information without damaging the original filter structure. When we intend to remove a large proportion of filters, the large k may lead to dropping the entire points in the local geometry, which could be avoided by searching the limited local geometry. A small k value, therefore, performs well in this condition, where only the closest neighbor is taken into consideration. Besides, we can also observe in Figure 4 that when $k > 10$, the performance improvements are subtle with the increase of k value. Therefore, in practice, to reduce the cost of hyper-parameter tuning, we can set $k = 1$ when $P \geq 50\%$, and $k = 10$ when $P > 50\%$.

Fig. 5: The parameter sensitivity analysis of on the Market-1501 dataset. We observe that our method is insensitive when pruning rate $P < 60\%$. In contrast, a large (i.e., $= 3 \times 10^{-1}$) is the better choice when $P \geq 60\%$.

Effect of weight decreasing. To study the sensitivity with respect to parameter, we evaluate different values in the range from $\{6 \times 10^{-1}; 3 \times 10^{-1}; 1 \times 10^{-1}; 6 \times 10^{-2}; 3 \times 10^{-2}; 1 \times 10^{-2}; 0\}$ on the Market-1501 dataset with different pruning rates P varying from $\{10\%, 20\%, 60\%, 90\%\}$. We fix $k = 1$ in the ablation study. The results are shown in Figure 5.

We observe that our method is not sensitive to when P is small, e.g., $P = 10\%$ and $P = 20\%$. In contrast, when a large proportion of parameters is dropped e.g., $P \geq 60\%$, the model with larger , which decreases the filter scale slowly, significantly performs well. It is consistent with our intuition that the filter weight decreasing helps the model adapt to the large prune rate. Note that if the value of is too large, e.g., $= 6 \times 10^{-1}$, the network may converge very slowly, and limited redundant filters are decreased to zeros. To compare the results fairly, we use the hard-pruning method to drop the final redundant filters, so the model with $= 6 \times 10^{-1}$ does not perform well.

Fig. 6: Weight distribution of different methods before and after pruning. The global-center method (i.e., SFP (blue), prunes filters close to the geometric center (zero here), largely changing the original distribution (red). In contrast, PLFP (green) evaluates the redundancy of filters according to their local similarity and is capable of maintaining the original filter distribution.

D. Filter Distribution Preservation Analysis

Our method introduces the local filter selection to preserve the original filter distribution in pre-trained models. In this section, we verify the filter distribution preservation ability of our method by analyzing the weight distribution of different methods before and after filter pruning without fine-tuning. The pruning rate is set to 50%. The weight is from the last convolutional layer of ResNet-50. We can observe that the global-center method (i.e., SFP (blue), tends to prune filters with small-scale weight value (close to the geometric center), leading to a relatively flat curve and a large disparity with the original distribution (red). In contrast, our PLFP (green) locally evaluates the redundancy of filters with their neighbors. As can be seen in Figure 6, the numbers of the deleted filters in our PLFP are relatively evenly distributed than SFP across the original range of weight value. Therefore, PLFP is capable of maintaining the original filter distribution.

To further show how the proposed method works, we visualize the first convolutional layer feature maps in our person re-identification network. The results are shown in Figure 7, where we have pruned 10% of filters and marked the corresponding pruned feature maps with red boxes. These pruned feature maps contain the outlines of the input image, such as straps (2, 31), T-shirts (13, 59), shorts (28), and hat (57). Clearly, our method prunes filters according to the local redundancy criterion, where the pruned feature maps can be replaced by the remaining ones. For example, straps, T-shirts, shorts, and hats can be replaced by feature maps: (43,

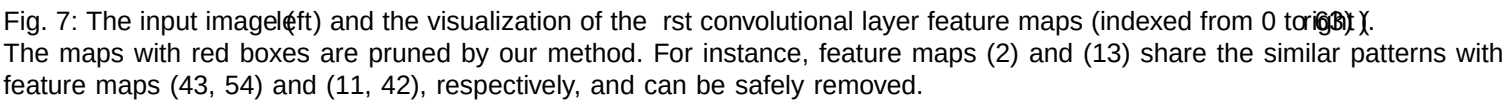


Fig. 7: The input image (left) and the visualization of the first convolutional layer feature maps (indexed from 0 to 63) of the pre-trained model. The maps with red boxes are pruned by our method. For instance, feature maps (2) and (13) share the similar patterns with feature maps (43, 54) and (11, 42), respectively, and can be safely removed.

54, 55, et al.), (11, 42, et al.), (20, 22 et al.), and (1, 34, 53, et al.), respectively.

E. Model Capacity Preservation Analysis

TABLE IV: The mean Euclidean distance on Market-1501 between the embedding features extracted from the original model and the pruned models. It shows that our method is more capable of retaining the model capacity of the original model than the other pruning methods.

Methods	Distance (10^{-2})	mAP(%)	Rank-1(%)
FPGM [16]	35.28	45.31	68.26
HFP [12]	35.17	47.24	70.57
SFP [8]	35.13	48.02	71.17
Ours ($k = 1$)	34.70	56.47	77.25

As demonstrated in Section IV-C, our method is adept at preserving the model capacity while pruning. In other words, the pruned model could keep the original pre-trained knowledge as much as possible. To verify this, we collect and analyze the outputs of the last convolutional layer of the pre-trained and pruned models. Intuitively, if the pruned models maintain a large model capacity, they can generate close outputs with the pre-trained model. Following the experimental setting in Section IV-A, for all the compared pruning methods, we prune and fine-tune the pre-trained model for 100 epochs. Then, we compute the Euclidean distance between the outputs of the original model and pruned models with 90% filters removed on Market-1501. The results are shown in Table IV. We find that our model generates the most similar output with the baseline model in [35] and achieves over 8% and 6% than the second best method in terms of mAP and Rank-1, respectively, which indicates that the proposed method can recover the model capacity while fine-tuning by keeping connections of the redundant filters. Therefore, it is suitable for pruning image retrieval networks.

F. Pruning Efficiency Analysis

For the pruning efficiency investigation, we record the execution (including pruning and fine-tuning) time and inference time of the comparison methods with 90% filters are pruned on Market-1501. All of the pruned methods are trained on an Intel (R) Core (TM) i9-9980XE CPU @ 3.00GHz PC with 64G memory and a GPU (Nvidia 2080Ti). After obtaining the slim model, we test them on a CPU (1.3 GHz Dual-Core i5). As can be observed in Table V, our method outperforms other pruning methods more than 8% mAP and accelerates the baseline model (186.28 microseconds per image) by over 4.0 (i.e., 46.28 microseconds per image), demonstrating the effectiveness of the proposed method. For execution, it takes 5.83 hours for PLFP, which consumes 0.55 hours more than FPGM (i.e., 5.28 hours. The main reason is that PLFP introduces an iterative pruning strategy and needs more computation resources. However, once PLFP is converged, we can safely remove the redundant filters (i.e., the filters that converge to zero) to obtain the slim model with the equivalent inference time as FPGM.

TABLE V: Runtime comparison of different pruning methods on ResNet-50 with 90% filters are pruned. `ms' refers to millisecond. The proposed method has saved more than 75% inference time with respect to the baseline and achieved better retrieval performance over other pruning methods.

Methods	Type	mAP(%)	Inference time (ms/image)
Baseline [35]	-	70.81	186.28
HFP [12]	hard	47.24	46.28
SFP [8]	soft	48.02	46.28
FPGM [16]	soft	45.31	46.28
Ours	soft	56.47	46.28

V. Conclusion and Future Work

In this paper, we propose a progressive local filter pruning strategy for image retrieval acceleration. Different from existing global center-based pruning works, the proposed approach resorts to remain the filter distribution by considering local relations between filters. Besides, the progressive filter weight decreasing allows the pruned model to preserve the model capacity, even when dropping a large proportion of filters. Compelling results on four scene retrieval benchmarks and one person re-id dataset show the effectiveness of our method on accelerating the image retrieval models. We hope this work can pave the way for fast image retrieval in real-world environments.

In future, we will extend the proposed PLFP to other image retrieval related applications like 3D point cloud matching,

object detection, *etc*. In addition, we would like to further improve PLFP by combining the filter correlations among different layers. One possible direction is to design a multi-task pruning framework, where each layer is treated as one pruning task. In this framework, multi-task learning and sparse regularization techniques can be considered to extract the inter-layer filter correlations.

REFERENCES

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

[2] X. Yang, C. Deng, F. Zheng, J. Yan, and W. Liu, "Deep spectral clustering using dual autoencoder network," in *CVPR*, 2019, pp. 4061–4070.

[3] Z. Liu, J. Wang, S. Gong, H. Lu, and D. Tao, "Deep reinforcement active learning for human-in-the-loop person re-identification," in *ICCV*, 2019, pp. 6122–6131.

[4] C. Deng, E. Yang, T. Liu, J. Li, W. Liu, and D. Tao, "Unsupervised semantic-preserving adversarial hashing for image search," *TIP*, vol. 28, no. 8, pp. 4032–4044, 2019.

[5] Z. Zheng, T. Ruan, Y. Wei, Y. Yang, and T. Mei, "Vehiclenet: Learning robust visual representation for vehicle re-identification," *IEEE Transactions on Multimedia*, pp. 1–1, 2020, doi:10.1109/TMM.2020.3014488.

[6] Y. Yang, Y. Zhuang, and Y. Pan, "Multiple knowledge representation for big data artificial intelligence: framework, applications, and case studies," *Frontiers Inf. Technol. Electron. Eng.*, vol. 22, no. 12, pp. 1551–1558, 2021, doi:10.1631/FITEE.2100463.

[7] D. Xie, C. Deng, C. Li, X. Liu, and D. Tao, "Multi-task consistency-preserving adversarial hashing for cross-modal retrieval," *TIP*, vol. 29, pp. 3626–3637, 2020.

[8] Y. He, G. Kang, X. Dong, Y. Fu, and Y. Yang, "Soft filter pruning for accelerating deep convolutional neural networks," in *IJCAI*, 2018.

[9] P. Ren, Y. Xiao, X. Chang, P. Huang, Z. Li, X. Chen, and X. Wang, "A comprehensive survey of neural architecture search: Challenges and solutions," *ACM Comput. Surv.*, vol. 54, no. 4, pp. 76:1–76:34, 2022.

[10] C. Yan, X. Chang, Z. Li, W. Guan, Z. Ge, L. Zhu, and Q. Zheng, "Zeronas: Differentiable generative adversarial networks search for zero-shot learning," *TPAMI*, vol. 44, no. 12, pp. 9733–9740, 2022.

[11] J. Fang, F. Wang, P. Shen, Z. Zheng, J. Xue, and T.-s. Chua, "Behavioral intention prediction in driving scenes: A survey," *arXiv:2211.00385*, 2022.

[12] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," in *ICLR*, 2017.

[13] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," *ICCV*, 2017.

[14] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Training pruned neural networks," *CoRR*, vol. abs/1803.03635, 2018.

[15] J. Lin, L.-Y. Duan, S. Wang, Y. Bai, Y. Lou, V. Chandrasekhar, T. Huang, A. Kot, and W. Gao, "Hnip: Compact deep invariant representations for video matching, localization, and retrieval," *IEEE Transactions on Multimedia*, vol. 19, no. 9, pp. 1968–1983, 2017.

[16] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang, "Filter pruning via geometric median for deep convolutional neural networks acceleration," in *CVPR*, 2019.

[17] T. Sun, Y. Shao, X. Li, P. Liu, H. Yan, X. Qiu, and X. Huang, "Learning sparse sharing architectures for multiple tasks," *AAAI*, 2019.

[18] A. Babenko, A. Slesarev, A. Chigorin, and V. S. Lempitsky, "Neural codes for image retrieval," vol. abs/1404.1777, 2014.

[19] H. Lin, Y. Fu, P. Lu, S. Gong, X. Xue, and Y.-G. Jiang, "Tc-net for isbir: Triplet classification network for instance-level sketch based image retrieval," in *ACMMM*, 2019, pp. 1676–1684.

[20] Y. Gu, C. Li, and Y.-G. Jiang, "Towards optimal cnn descriptors for large-scale image retrieval," in *ACMMM*, 2019, pp. 1768–1776.

[21] Z. Zheng, L. Zheng, and Y. Yang, "Pedestrian alignment network for large-scale person re-identification," *TCSVT*, 2017, doi:10.1109/TCSVT.2018.2873599.

[22] E. Yang, T. Liu, C. Deng, and D. Tao, "Adversarial examples for hamming space search," *TCYB*, vol. 50, no. 4, pp. 1473–1484, 2020.

[23] J. Lin, Z. Zheng, Z. Zhong, Z. Luo, S. Li, Y. Yang, and N. Sebe, "Joint representation learning and keypoint detection for cross-view geo-localization," *IEEE Transactions on Image Processing*, 2022.

[24] C. Deng, Z. Chen, X. Liu, X. Gao, and D. Tao, "Triplet-based deep hashing network for cross-modal retrieval," *TIP*, vol. 27, no. 8, pp. 3893–3903, 2018.

[25] F. Radenović, G. Tolias, and O. Chum, "Fine-tuning CNN image retrieval with no human annotation," *TPAMI*, 2018.

[26] L. Zheng, Y. Yang, and A. G. Hauptmann, "Person re-identification: Past, present and future," *CoRR*, vol. abs/1610.02984, 2016.

[27] Z. Zheng, L. Zheng, and Y. Yang, "A discriminatively learned CNN embedding for person reidentification," *ACM TOMM*, vol. 14, no. 1, pp. 13:1–13:20, 2018, doi:10.1145/3159171.

[28] X. Qian, Y. Fu, T. Xiang, W. Wang, J. Qiu, Y. Wu, Y.-G. Jiang, and X. Xue, "Pose-normalized image generation for person re-identification," in *ECCV*, 2018, pp. 650–667.

[29] C. Shen, Z. Jin, W. Chu, R. Jiang, Y. Chen, G. Qi, and X. Hua, "Multi-level similarity perception network for person re-identification," *ACM TOMM*, vol. 15, no. 2, pp. 32:1–32:19, 2019.

[30] X. Qian, Y. Fu, T. Xiang, Y.-G. Jiang, and X. Xue, "Leader-based multi-scale attention deep architecture for person re-identification," *TPAMI*, 2019.

[31] S. Zhou, J. Wang, R. Shi, Q. Hou, Y. Gong, and N. Zheng, "Large margin learning in set-to-set similarity comparison for person reidentification," *IEEE Transactions on Multimedia*, vol. 20, no. 3, pp. 593–604, 2018.

[32] C. Deng, X. Xu, H. Wang, M. Yang, and D. Tao, "Progressive cross-modal semantic network for zero-shot sketch-based image retrieval," *TIP*, vol. 29, pp. 8892–8902, 2020.

[33] G. Ding, S. Zhang, S. Khan, Z. Tang, J. Zhang, and F. Porikli, "Feature affinity-based pseudo labeling for semi-supervised person re-identification," *IEEE Transactions on Multimedia*, vol. 21, no. 11, pp. 2891–2902, 2019.

[34] Z. Zheng, L. Zheng, and Y. Yang, "Unlabeled samples generated by gan improve the person re-identification baseline in vitro," in *ICCV*, Oct 2017.

[35] Z. Zheng, X. Yang, Z. Yu, L. Zheng, Y. Yang, and J. Kautz, "Joint discriminative and generative learning for person re-identification," in *CVPR*, 2019.

[36] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation." in *AAAI*, 2020, pp. 13 001–13 008.

[37] J. Wang, X. Zhu, S. Gong, and W. Li, "Transferable joint attribute-identity deep learning for unsupervised person re-identification," in *CVPR*, 2018, pp. 2275–2284.

[38] X. Zhang, J. Cao, C. Shen, and M. You, "Self-training with progressive augmentation for unsupervised cross-domain person re-identification," in *ICCV*, 2019, pp. 8222–8231.

[39] F. Yang, K. Li, Z. Zhong, Z. Luo, X. Sun, H. Cheng, X. Guo, F. Huang, R. Ji, and S. Li, "Asymmetric co-teaching for unsupervised cross-domain person re-identification." in *AAAI*, 2020, pp. 12 597–12 604.

[40] X. Jin, C. Lan, W. Zeng, Z. Chen, and L. Zhang, "Style normalization and restitution for generalizable person re-identification," in *CVPR*, 2020, pp. 3143–3152.

[41] C. Deng, E. Yang, T. Liu, and D. Tao, "Two-stream deep hashing with class-specific centers for supervised image search," *TNNLS*, vol. 31, no. 6, pp. 2189–2201, 2020.

[42] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Speeding up convolutional neural networks with low rank expansions," in *BMVC*, 2014.

[43] C. Li, G. Wang, B. Wang, X. Liang, Z. Li, and X. Chang, "Ds-net++: Dynamic weight slicing for efficient inference in cnns and vision transformers," *TPAMI*, pp. 1–16, 2022.

[44] Z. Zheng, X. Wang, N. Zheng, and Y. Yang, "Parameter-efficient person re-identification in the 3d space," *IEEE Transactions on Neural Networks and Learning Systems*, 2022, doi:10.1109/TNNLS.2022.3214834.

[45] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding," in *ICLR*, 2016.

[46] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *ECCV*, 2016.

[47] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *NIPS Deep Learning and Representation Learning Workshop*, 2015.

[48] T. Chen, I. Goodfellow, and J. Shlens, "Net2net: Accelerating learning via knowledge transfer," in *ICLR*, 2016.

[49] J. Kim, S. Park, and N. Kwak, "Paraphrasing complex network: Network compression via factor transfer," in *NeurIPS*, 2018.

[50] S. Lin, R. Ji, C. Yan, B. Zhang, L. Cao, Q. Ye, F. Huang, and D. Doermann, "Towards optimal structured cnn pruning via generative adversarial learning," in *CVPR*, 2019, pp. 2790–2799.