

One-Shot Knowledge Transfer for Scalable Person Re-Identification

Longhua Li^{1,2} Lei Qi^{1,2*} Xin Geng^{1,2*}

¹School of Computer Science and Engineering, Southeast University, Nanjing, China

²Key Laboratory of New Generation Artificial Intelligence Technology and Its Interdisciplinary Applications (Southeast University), Ministry of Education, China

{lhli, qilei, xgeng}@seu.edu.cn

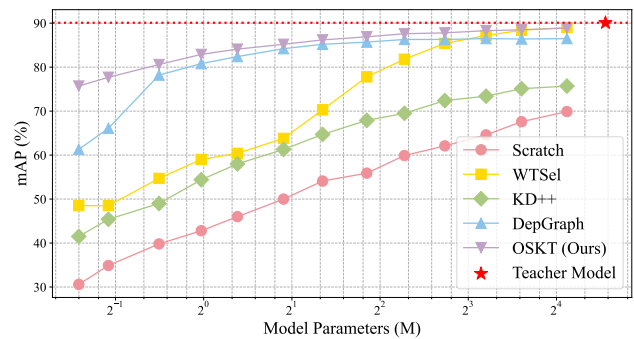
Abstract

Edge computing in person re-identification (ReID) is crucial for reducing the load on central cloud servers and ensuring user privacy. Conventional compression methods for obtaining compact models require computations for each individual student model. When multiple models of varying sizes are needed to accommodate different resource conditions, this leads to repetitive and cumbersome computations. To address this challenge, we propose a novel knowledge inheritance approach named OSKT (One-Shot Knowledge Transfer), which consolidates the knowledge of the teacher model into an intermediate carrier called a weight chain. When a downstream scenario demands a model that meets specific resource constraints, this weight chain can be expanded to the target model size without additional computation. OSKT significantly outperforms state-of-the-art compression methods, with the added advantage of one-time knowledge transfer that eliminates the need for frequent computations for each target model.

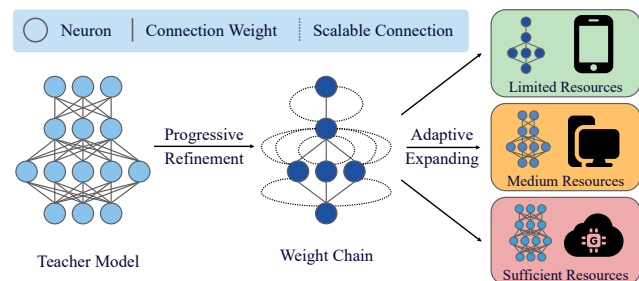
1. Introduction

Person re-identification (ReID) is pivotal for surveillance and smart city systems [2], aiming to uniquely identify individuals across disparate camera views [37, 43]. While deploying ReID models on edge devices is crucial for mitigating cloud computing burdens and safeguarding data privacy [38, 40], recent advancements in the ReID field are often encumbered by heavy parameter loads and intensive computational requirements [9, 12, 22], restricting their applicability in resource-constrained scenarios.

To achieve an optimal accuracy-cost trade-off, different edge computing scenarios require carefully customized models to maximize re-identification performance within their resource constraints. A common approach for obtaining a high-performance compact model is to compress



(a) mAP vs. Model Size (Log Scale) on Market1501



(b) Conceptual Diagram of the Proposed Method

Figure 1. (a) Transfer knowledge from a pre-trained ResNet50 to student models of different sizes. (b) Refine knowledge into a weight chain in a single pass, enabling adaptive expansion of models of varying sizes meeting downstream resource constraints.

a large, well-trained teacher model into a smaller student model through techniques such as pruning [19] and distillation [18]. Nevertheless, these methods are typically limited to generating a single model of a fixed size at a time, requiring separate and often intricate processes for each downstream scenario, thus consuming significant resources.

In this work, we propose a knowledge inheritance method suitable for general model architectures. It requires only a single computation to obtain student models of various sizes, without the need for the repetitive and computationally expensive training procedures typically required

*Corresponding authors.

by traditional compression methods for each target size. The knowledge of the teacher model is first refined into a weight chain, which acts as an intermediate knowledge carrier. When a student model of a specific size is required, the weight chain can be rapidly expanded to the target size without additional computation. The resulting model can be directly deployed or fine-tuned for downstream tasks.

Specifically, the weight chain maintains the same depth (number of layers) as the teacher model, with each layer’s weights being refined from the corresponding layer in the teacher model. However, its width (feature dimension) is substantially reduced compared to the teacher model, while maintaining the flexibility to be expanded to generate models of any intermediate width between the weight chain and the teacher model. Notably, the weight chain preserves the complete normalization layers of the original width to effectively reuse the core feature dimensions of its output, thereby providing richer feature representations. When a model of specific size is required, this can be achieved by stacking the weight matrix rows of the weight chain and correspondingly merging the affine transformation parameters of the normalization layers to attain the desired width.

The process of refining the weight chain from the teacher model involves two key steps: initialization and progressive refinement of the weight chain. Initially, we set the weight matrix rows in the weight chain as cluster centers from the teacher model’s corresponding rows, as these centers exhibit functional representativeness and facilitate subsequent refinement processes. For progressive refinement, we jointly train the teacher model and the smallest student model (with width matching the weight chain) built by the weight chain, propagating gradients back to the teacher model and the weight chain. Serving as a bridge between the teacher model and all student models, this approach enables simultaneous training of both ends while implicitly training intermediate models, eliminating the need for intensive training of individual student models of varying sizes. Furthermore, we incorporate an MSE loss between the weight chain rows and the corresponding row clusters of the teacher model to ensure that all weight rows within the same cluster are refined into a single core weight row. Ultimately, the refined weight chain can be expanded to generate models of any size between its own width and that of the teacher model without additional computation.

Our contributions are summarized as follows:

- To our knowledge, we are the first to propose a one-shot computation-based method for generating scalable person ReID models, adaptable to varying resource scenarios.
- Our method can be seamlessly integrated with general model architectures, providing highly accurate models that outperform traditional knowledge transfer methods.
- Extensive experiments demonstrate that our approach consistently performs well across diverse real-world sce-

narios, providing a robust solution for person ReID tasks.

2. Related Work

2.1. Person Re-Identification

In Person ReID, early methods [24, 42, 43] focused on handcrafted features and metric learning techniques [39]. With the rise of deep learning, Convolutional Neural Network (CNN)-based approaches [5, 26, 29, 30, 35] dominated for a long time, significantly boosting ReID performance. Recently, Vision Transformer (ViT) [8]-based methods [14, 16, 17, 27, 28, 41, 45] have shown promising results in capturing cross-view relationships. However, these methods are developed on mainstream base models and are not suitable for deployment in resource-constrained edge nodes in practical applications. Recent works [13, 20, 44] have designed lightweight models for person ReID, but these still require extensive pre-training on large-scale datasets like ImageNet [7] and LUPerson [11] to achieve high performance. Therefore, achieving high performance in different resource-limited scenarios still involves repetitive computation. In contrast, our method refines the knowledge from the teacher model into a weight chain through a one-time computation. When a model of a specific size is required, the weight chain can be instantly expanded to the target-sized model without training.

2.2. Knowledge Transfer

Knowledge transfer involves techniques that allow transferring knowledge from one model to another, enabling efficient model initialization using teacher models. We categorize these methods into two types: those that use additional guidance from teacher models and those based solely on the weights of teacher models. The first type is primarily represented by knowledge distillation [18, 32], while the second includes commonly used methods such as model pruning [10, 19, 34]. These methods often require costly distillation or pruning with large pre-training datasets, leading to significant computational overhead when initializing scenario-specific models. Other methods in the second category include Net2Net [4], DPIAT [6], Weight Selection [36], Weight Distillation [25], Learngene [31] and so on. However, these methods are either not flexible enough to generate densely-sized student models or inefficient at transferring the knowledge of teacher models, requiring extensive computations to produce student models of various sizes. Our method ingeniously leverages the weights of the teacher model and incorporates a sophisticated one-shot training strategy to derive densely-sized student models.

3. Methodology

To efficiently generate size-specific models for person ReID across various scenarios, we propose a novel framework

termed OSKT. As shown in Figure 4, OSKT consists of two core components: (1) weight chain refinement from the teacher model and (2) student model generation via the weight chain. This section is organized as follows: First, we unify CNN and ViT architectures for clarity. Then we formally introduce the weight chain concept, followed by a comprehensive technical elaboration of both components.

3.1. Unified Representation of CNNs and ViTs

Our framework is universally applicable to mainstream network architectures, including both CNNs and ViTs. To establish architectural unification, we focus on the feature *dimension* (termed as *channel* in CNNs), which serves as a common bridge between these architectures. Figure 2 provides a conceptual visualization of this unified representation. In CNNs, convolutional filters each generate a distinct output feature channel, while in ViTs, each weight matrix row produces a specific output feature dimension. We abstract these feature-generating units as *rows*. When features are propagated to the next layer, each dimension is independently processed by a corresponding CNN channel or ViT weight matrix column. We abstract these feature-processing units as *columns*. Both architectures employ normalization layers (e.g., BN in CNNs, LN in ViTs) between layers, where each feature dimension is associated with affine transformation parameters (γ, β) that scale and shift normalized features, enhancing their representational capacity.

Consider a teacher model with L layers, where N_{l-1} and N_l represent the input and output feature dimensions of the l -th layer, respectively. Let $\mathcal{F}_{l,j}$ represent the j -th row of the l -th layer, where $\mathcal{F}_{l,j} \in \mathbb{R}^{N_{l-1} \times O_l}$, with O_l denoting the number of operation parameters in the l -th layer. For example, in convolutional layers, $O_l = K \times K$ (where K is the kernel size), whereas in fully connected layers, $O_l = 1$. Then the weight matrices in the teacher model can be parameterized as $\{\mathcal{F}_l \in \mathbb{R}^{N_l \times N_{l-1} \times O_l}, 1 \leq l \leq L\}$.

3.2. Weight Chain as the Knowledge Carrier

Building on the observation of feature dimension redundancy [1, 3, 21], we propose using refined rows from the teacher model’s weight matrices as knowledge carriers. The weight chain preserves the teacher model’s depth while containing a significantly reduced number of refined rows per layer compared to the teacher’s width. Formally, the weight chain is parameterized as $\{\mathcal{F}_l^C \in \mathbb{R}^{M_l \times N_{l-1} \times O_l}, 1 \leq l \leq L\}$, where M_l denotes the number of refined rows in the l -th layer. The normalization layers in the weight chain share weights with their teacher counterparts, enabling multiple (γ, β) pairs to reuse the core feature dimensions generated by the corresponding refined rows. By leveraging the weight chain as an intermediate knowledge carrier, our method is built on three key insights:

Insight 1. Leveraging teacher model weights for ef-

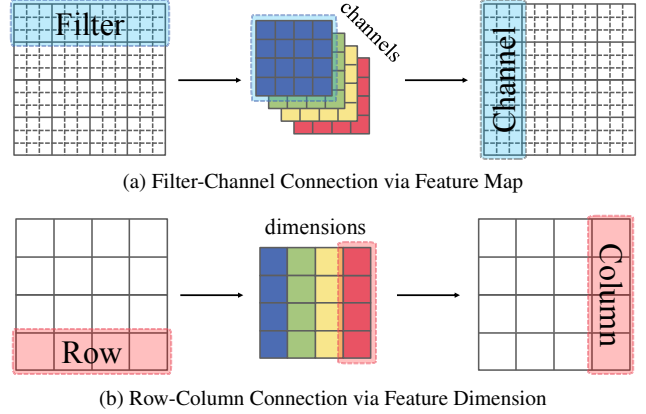


Figure 2. (a) In CNNs, a filter is abstracted as a *row*, while a channel in the weight matrix is abstracted as a *column*. (b) In ViTs, the weight matrix is also abstracted into *rows* and *columns*.

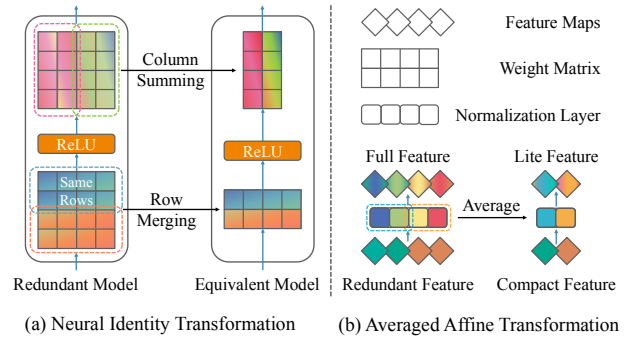


Figure 3. (a) Achieving identity transformation by merging identical rows and summing corresponding columns in the next layer. (b) Extracting lightweight principal features by averaging the normalization layer’s affine transformation parameters.

cient knowledge transfer. Well-trained model weights encapsulate substantial knowledge, and their effective utilization can significantly enhance knowledge transfer efficiency. Our analysis reveals that traditional distillation methods, which rely solely on teacher model guidance, underperform on mainstream person ReID datasets. In contrast, our approach employs cluster centers of the teacher model’s weight rows combined with a progressive strategy to efficiently transfer knowledge to the weight chain.

Insight 2. Neural identity transformation. In a neural network layer with multiple identical rows, if we ignore the subsequent normalization layers, merging these rows into one and summing their corresponding columns in the next layer preserves the network’s function, as shown in Figure 3a. This matrix multiplication property allows progressive refinement of knowledge into core weight rows, enabling the student model to approximate the teacher model’s function. In practice, normalization layers must be addressed. Each feature dimension is associated with an affine trans-

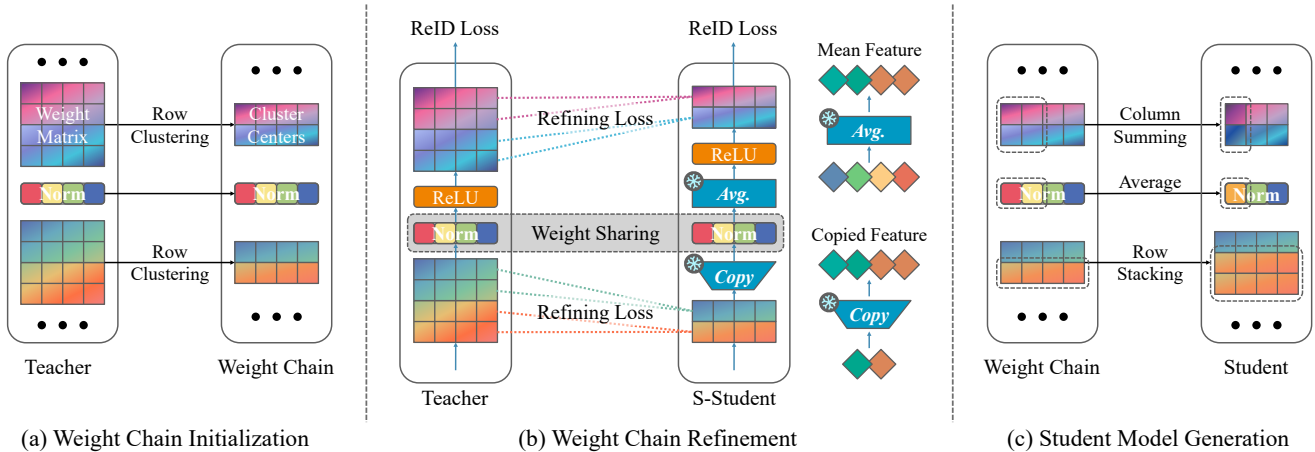


Figure 4. An overview framework of the proposed method. (a) Cluster the weight rows within each layer of the teacher model, using cluster centers as initial weight chain rows. (b) Train the teacher model and the S-Student constrained by the weight chain, using Refining loss and ReID loss to infuse core knowledge into the weight chain. (c) Reuse and stack the refined rows proportionally to achieve the required layer width for the student model. Average the (γ, β) pairs within each normalization layer and sum the column weights of each subsequent layer accordingly to generate the student model. This is an $O(1)$ operation that does not require additional computation.

formation pair (γ, β) , and identical rows may correspond to distinct pairs, representing diverse feature transformations. To reduce feature dimensions by merging identical rows, their corresponding (γ, β) pairs must also be merged. As shown in Figure 3b, averaging these pairs retains critical information, leveraging the representativeness and minimal distance within clusters to enhance fine-tuning efficiency.

Insight 3. Weight chain as a bridge between teacher and student models. The weight chain serves as a knowledge carrier, expandable to create student models ranging from the teacher model’s size at maximum to its own width at minimum. Each layer’s width in these student models can vary continuously within this range, positioning the weight chain as a bridge connecting the teacher model to a spectrum of student models. At the bridge’s ends lie the teacher model and the smallest expandable student model. By extending the weight chain to these two endpoint models and training with ReID loss, gradients propagate back to the weight chain. This approach trains only the two endpoint models while providing indirect training for all intermediate models. Like lifting a string of beads by their ends, training the endpoints elevates all intermediate models in between.

3.3. Refinement of the Weight Chain

3.3.1. Row Clustering for the Weight Chain Initialization

The weight matrix of the teacher model encapsulates rich knowledge, which effectively initializes the weight chain, accelerating the refinement process. As illustrated in Figure 4a, at each layer, we cluster the weight rows of the teacher model, setting the number of clusters to match the width of the weight chain. The cluster centers, being representative and exhibiting smaller intra-cluster distances, initialize

the corresponding rows in the weight chain, facilitating progressive refinement and enhancing knowledge transfer.

In layers with residual connections, where outputs from multiple layers are aggregated, we cluster these layers together to align the number of distinct feature dimensions with the number of weight chain rows in each layer. This enables us to initialize student models by summing the column weights in subsequent layers, effectively mimicking the functionality of the teacher model.

For normalization layers, we directly share affine transformation weights between the weight chain and the teacher model, enabling a single row in the weight chain to correspond to multiple affine transformation pairs. While these parameters are few, they are critical for generating models of varying sizes through different levels of merging. This design also prevents student models from representing identical functions and stagnating in the same local optima.

3.3.2. Progressive Refinement of the Weight Chain

After obtaining teacher weight centers as initial weight chain rows, further optimization is needed because these centers, while representative, may not fully capture shared functionalities and can be affected by noisy weights. We employ a progressive refinement process by simultaneously training the teacher model and the smallest student model (S-Student), both constrained by the weight chain. The S-Student shares the same width as the weight chain, enabling the effective transfer of shared knowledge from the teacher model to the weight chain throughout the training process. Serving as a bridge, the weight chain connects the teacher model with student models of all widths. By training only the two models at either end of this bridge—the teacher

model and the S-Student—we can achieve strong performance across all intermediate student models.

Specifically, we use Mean Squared Error (MSE) loss to tighten the clusters, aligning the weight rows more closely in the same cluster within the teacher model:

$$\mathcal{L}_{ref} = \frac{1}{L} \sum_{1 \leq l \leq L} \frac{1}{M_l} \sum_{\substack{1 \leq j \leq M_l \\ k \in \mathcal{I}_l^{(j)}}} (\mathcal{F}_{l,k} - \mathcal{F}_{l,j}^C)^2, \quad (1)$$

where $\mathcal{F}_{l,k}$ and $\mathcal{F}_{l,j}^C$ are the weight rows in the l -th layer of the teacher model and the weight chain, respectively, and $\mathcal{I}_l^{(j)}$ indexes the clustered teacher weight rows associated with the j -th row of the weight chain. To preserve essential ReID knowledge during refinement, we train the teacher model using both ID loss (\mathcal{L}_{id}) and hard triplet loss (\mathcal{L}_{tri}). The overall loss on the teacher model, \mathcal{L}_T , is given by:

$$\mathcal{L}_T = \mathcal{L}_{id}(\mathbf{p}^T) + \mathcal{L}_{tri}(\mathbf{f}^T), \quad (2)$$

where \mathbf{p}^T and \mathbf{f}^T are the logits and features generated by the teacher model, respectively. The overall loss function on the S-Student, \mathcal{L}_S , is given by:

$$\mathcal{L}_S = \mathcal{L}_{id}(\mathbf{p}^S) + \mathcal{L}_{tri}(\mathbf{f}^S), \quad (3)$$

where \mathbf{p}^S and \mathbf{f}^S are the logits and features generated by the S-Student. It is important to note that the gradients derived from the losses applied to the S-Student are backpropagated to the weight chain, thereby optimizing it.

The total loss, \mathcal{L} , integrates the refining loss \mathcal{L}_{ref} with ReID losses \mathcal{L}_T and \mathcal{L}_S to optimize the weight chain:

$$\mathcal{L} = \mathcal{L}_T + \mathcal{L}_S + \alpha \mathcal{L}_{ref}. \quad (4)$$

In ViTs, α is a progressive coefficient set as $\alpha = \frac{iter}{n.iter}$, with $iter$ being the current iteration and $n.iter$ the total iterations. For CNNs, α is fixed at 1.

3.4. Student Generation with the Weight Chain

In this section, we illustrate the student model generation process using the l -th layer of the student model as an example, as illustrated in Figure 4c. In the teacher model, the l -th layer has N_l rows, while the weight chain’s l -th layer has M_l refined rows. If the student model requires C_l rows in this layer ($M_l \leq C_l \leq N_l$), we stack refined rows in the l -th layer of the weight chain proportionally to the number of corresponding teacher weight rows to achieve C_l rows. This results in a student-weight chain-teacher rows matcher, where each student row corresponds to multiple teacher rows. We then sum the weights of the corresponding columns in subsequent layers to preserve similar functionality as before merging. For the followed normalization (γ, β) pairs associated with the same student row, we average these pairs to initialize the corresponding normalization layer in the student model. This averaging retains the most essential information from the refined features, enabling enhanced optimization in downstream scenarios.

Table 1. Results of transferring knowledge from ResNet50 to student models on the Market1501 dataset.

Method	Res-50-S1		Res-50-S3		Res-50-S5	
	mAP	R1	mAP	R1	mAP	R1
Scratch	30.6	53.3	47.4	70.9	60.7	80.9
WTSel	48.5	72.3	63.0	81.5	84.1	93.0
KD++	41.5	65.7	59.9	80.4	71.4	86.9
DepGraph	61.3	80.7	83.2	92.7	86.3	94.3
OSKT	75.7	89.4	84.7	93.3	87.6	94.5

Table 2. Results of transferring knowledge from ViT-B and ViT-S to their respective two student models on the Market-1501 dataset. Results are reported in terms of mAP/Rank-1.

Method	ViT-S-S1	ViT-S-S2	ViT-B-S1	ViT-B-S2
Scratch	13.9/23.9	18.3/31.2	22.2/37.4	21.9/36.3
WTSel	41.0/60.5	54.8/75.1	16.8/31.7	58.7/78.2
KD++	22.6/38.7	25.0/41.2	25.9/41.7	28.2/44.4
DepGraph	56.5/74.1	69.0/84.2	15.3/30.1	81.5/91.7
OSKT	74.2/87.1	77.2/89.0	81.6/91.9	82.9/92.4

4. Experiments

4.1. Experimental Designs

Datasets and evaluation metrics. We evaluate on three prominent person ReID benchmarks: Market1501 [42], MSMT17 [33], and CUHK03 [23], which are referred to as M, MS, and C, respectively. Evaluation metrics include Cumulative Matching Characteristic (CMC) and mean Average Precision (mAP), which are standard for comprehensive performance assessment.

Teacher and student models. We leverage pre-trained ResNet50 [15] from LUPerson [11] for CNN architecture and pre-trained ViT-B/ViT-S from PASS [45] for ViT architecture, fine-tuning them on person ReID datasets as teacher models. For ResNet50, we design six progressively scaled student models (Res-50-S1 to S6), each doubling its predecessor’s parameters, while establishing two student models each for ViT-B (ViT-B-S1, S2) and ViT-S (ViT-S-S1, S2), with detailed specifications provided in the appendix.

Comparison methods. Our study compares training from scratch, knowledge distillation (using KD++ [32] for feature alignment and norm enhancement), model pruning (evaluating DepGraph [10] for dependency-based group sparsity and fine-tuning), weight selection [36] (referred to as WTSel), and our OSKT, systematically assessing student model performance across these methods.

Table 3. Results of transferring knowledge from ResNet50 to student models. K.T.: knowledge transfer using the source dataset. F.T.: fine-tune on the downstream dataset. k : # of student models. m : # of weight chains refined from the teacher model. In this table, $m = 3$.

Method	Epoch		Res-50-S1		Res-50-S2		Res-50-S3		Res-50-S4		Res-50-S5		Res-50-S6		
	K.T.	F.T.	mAP	R1	mAP	R1	mAP	R1	mAP	R1	mAP	R1	mAP	R1	
MS→M	WTSel	-	100· k	41.3	64.0	54.6	76.5	60.1	80.2	73.1	88.5	77.4	90.7	86.1	94.0
	KD++	200· k	100· k	54.6	76.7	63.2	82.5	71.2	87.5	77.5	90.5	78.9	91.0	80.9	92.2
	DepGraph	100· k	100· k	61.6	81.0	75.0	89.2	77.3	90.3	82.4	92.8	82.3	92.7	82.1	92.5
	OSKT	100· m	100· k	72.3	87.5	74.4	89.6	82.6	92.6	82.9	92.7	85.6	93.9	86.1	94.1
MS→C	WTSel	-	100· k	16.6	15.5	24.8	24.6	22.2	21.3	38.9	40.1	45.1	45.5	68.9	71.8
	KD++	200· k	100· k	26.8	27.0	34.3	35.4	41.6	42.9	52.0	53.3	53.6	55.1	56.1	57.6
	DepGraph	100· k	100· k	31.3	31.7	49.0	51.5	54.8	57.8	62.4	64.4	65.0	68.7	65.8	69.9
	OSKT	100· m	100· k	45.7	47.3	49.2	52.2	62.5	65.1	63.1	65.8	68.6	71.7	69.8	71.6
M→C	WTSel	-	100· k	18.0	15.9	27.4	27.1	28.0	27.9	45.3	47.4	55.0	56.9	65.7	68.3
	KD++	200· k	100· k	21.1	21.1	31.8	32.1	37.5	37.5	46.1	47.3	50.3	52.1	55.9	58.4
	DepGraph	100· k	100· k	23.4	22.7	47.7	49.9	53.4	56.4	65.1	68.2	65.8	68.3	67.0	70.5
	OSKT	100· m	100· k	44.6	47.4	47.9	50.9	60.9	64.1	63.5	66.5	68.0	70.6	69.5	72.3

Table 4. Results of transferring knowledge from ViT-S and ViT-B to their respective student models under three across-scenario settings. Results are reported in terms of mAP/Rank-1.

Method	ViT-S-S1	ViT-S-S2	ViT-B-S1	ViT-B-S2	
MS→M	WTSel	42.8/62.9	55.4/75.3	15.8/30.5	60.0/79.1
	KD++	38.1/55.9	41.2/60.7	40.2/59.3	40.7/59.4
	DepGraph	65.5/82.1	74.6/86.9	63.7/80.7	83.3/92.1
	OSKT	76.5/88.7	79.0/89.7	82.1/91.9	83.0/92.0
MS→C	WTSel	1.6/0.9	20.9/19.9	3.0/2.5	24.2/23.1
	KD++	15.8/14.1	17.9/16.6	18.4/17.1	20.5/18.7
	DepGraph	46.7/47.9	59.4/61.3	41.1/43.1	71.0/74.5
	OSKT	61.2/63.5	63.9/66.4	69.4/72.4	70.3/73.0
M→C	WTSel	1.7/1.4	20.8/19.5	2.5/1.9	5.2/4.5
	KD++	10.4/9.0	11.6/10.4	12.2/10.1	13.9/12.1
	DepGraph	40.3/41.4	51.7/52.7	6.2/5.2	63.1/65.6
	OSKT	49.3/50.9	51.3/52.9	59.4/61.7	61.9/64.4

4.2. Knowledge Transfer in a Single Scenario

We assess the efficacy of knowledge transfer from a teacher model to student models using diverse methods on a single dataset, subsequently comparing their test performance on the same dataset. The results in Table 1 and Table 2 demonstrate that our method effectively leverages the teacher model’s weights for knowledge transfer.

4.3. Knowledge Transfer across Scenarios

Cross-scene knowledge transfer initially leverages a person re-identification dataset to perform knowledge transfer and obtain a student model, followed by fine-tuning and eval-

Table 5. Cost of obtaining student models of ViT-S and ViT-B. K.T.: # of epochs for knowledge transfer using the source dataset. F.T.: # of epochs for fine-tuning on the downstream dataset. k : # of student models. m : # of distinct-width weight chains refined from teacher models. In Table 4, $m = 1$ for both ViT-S and ViT-B.

Method	Teacher: ViT-S		Teacher: ViT-B	
	K.T.	F.T.	K.T.	F.T.
WTSel	-	120· k	-	120· k
KD++	120· k	120· k	120· k	120· k
DepGraph	120· k	120· k	120· k	120· k
OSKT	120· m	120· k	120· m	120· k

uating the student model on a distinct downstream scene. This is a common practice in real-world applications, as it is often difficult to quickly obtain large amounts of labeled data for training in downstream scenarios. Since knowledge transfer typically involves transferring from a larger dataset to a smaller one, we designed three cross-scenario transfer settings: from MSMT17 to Market1501 and CUHK03, as well as from Market1501 to CUHK03. For downstream training, we employ the BoT [26] framework for ResNet and adhere to the PASS [45] setup for ViT. As shown in Tables 3 and 4, our method consistently surpasses others, especially as the student models’ sizes decrease.

4.3.1. Convergence Speed

In Figure 5, we present the validation mAP curves for student models over the first 100 epochs on downstream datasets. Our method consistently surpasses others, offering reduced computational costs and enhanced efficiency.

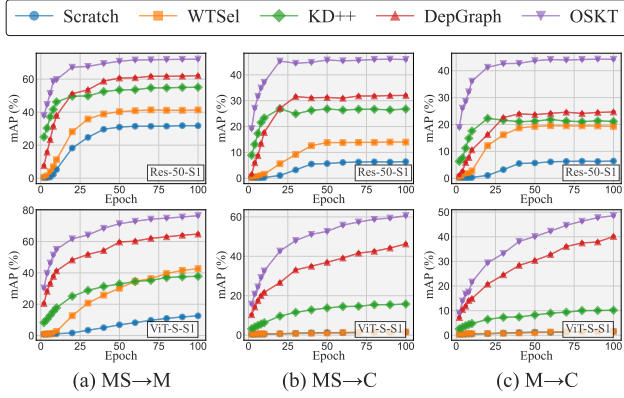


Figure 5. Convergence curves of student models generated by our OSKT, compared to other approaches across diverse scenarios.

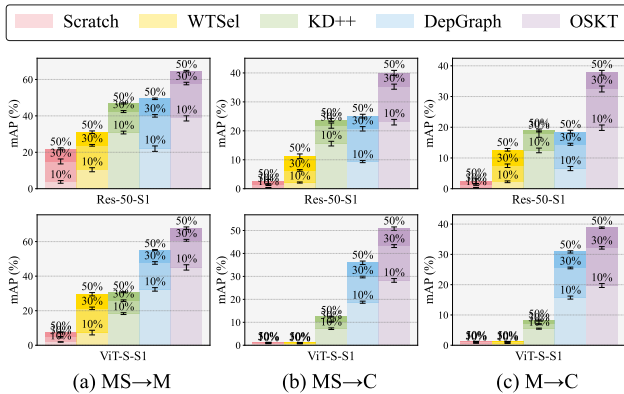


Figure 6. Performance comparison under few-shot settings, with each sampling ratio evaluated using five random seeds.

4.3.2. Few-shot Setting

In Figure 6, we present the results under settings using 10%, 30%, and 50% of IDs on downstream datasets. To ensure the robustness of the experimental results, we perform five random samplings of IDs for each few-shot setting.

4.3.3. Combined with Lightweight ReID Architectures

Our method demonstrates strong compatibility with modern lightweight ReID models. As shown in Table 6, we validate its effectiveness using two state-of-the-art architectures: OSNet [44] and MSINet [13]. We adopt the full-size models as teachers and construct two smaller variants for each model with width multipliers of 0.25 and 0.5. Pre-training is performed on MSMT17, with evaluation on Market1501 and CUHK03, comparing results from scratch training, pre-training, and our proposed method. Notably, both OSNet and MSINet employ a lightweight building block comprising a point-wise 1×1 convolution followed by a depth-wise 3×3 convolution, which we cluster as a single unit for weight chain initialization and refinement.

Table 6. Results of modern ReID models initialized with various methods. Pre-training dataset: MSMT17. β : width multiplier.

	β	Params	Method	Market1501		CUHK03	
				mAP	R1	mAP	R1
OSNet [44]	0.25	0.15M	Scratch	78.3	90.7	48.4	51.5
			Pretrain	80.0	91.6	58.0	61.8
			OSKT	81.8	92.5	60.6	63.6
	0.5	0.56M	Scratch	83.7	93.4	57.9	60.6
			Pretrain	84.7	93.4	65.9	69.1
			OSKT	85.4	94.2	66.9	69.6
MSINet [13]	0.25	0.17M	Scratch	81.2	92.7	53.3	55.7
			Pretrain	81.8	92.5	57.7	59.9
			OSKT	82.4	92.8	60.2	62.4
	0.5	0.63M	Scratch	86.0	94.4	64.1	66.6
			Pretrain	87.4	94.6	70.9	73.4
			OSKT	87.9	95.0	72.8	75.6

4.4. Ablation Study

4.4.1. Ablation Studies of Alternative Settings

We design ablation studies focusing on the initialization and refinement of the weight chain to provide further insights into our framework. The results are shown in Table 7.

Leveraging weights from the teacher model. We replace the method of utilizing weights from the teacher model with randomly initializing the parameters of the teacher model and then generating the student model according to Figure 4. The results are shown as in (a) of Table 7.

Method for initializing the weight chain. We substitute the method of employing clustering to obtain weight centers for initializing the weight chain with two alternative approaches: random clustering and inverse clustering, as presented in (b) and (c) of Table 7, respectively.

Distance metric for clustering weight rows. In the above experiments, for ResNet50, we clustered the weight rows of the teacher model based on Euclidean distance, while for ViT, we clustered based on cosine distance. By swapping these settings, we obtain the results shown in (d) of Table 7.

Progressive refinement of the weight chain. After initializing the weight chain using the teacher model, we omit the refinement step illustrated in Figure 4b, with the results shown in (e) of Table 7. During the refinement phase, we eliminate the loss term \mathcal{L}_T from Equation 4. Consequently, the weight chain is no longer influenced by the loss of the teacher model but is solely driven by the gradients of the S-Student’s loss \mathcal{L}_S . The corresponding results are presented in (f) of Table 7. In the above experiments, for ResNet50, the weight α of \mathcal{L}_{ref} is set to 1, while for ViTs, $\alpha = \frac{iter}{n_iter}$ serves as a progressive weight. By swapping these settings,

Table 7. Ablation of alternative settings in OSKT. Results are reported in terms of mAP/Rank-1.

Setting	Single Scenario: M				Across Scenarios: MS→C			
	Res-50-S1	Res-50-S2	ViT-S-S1	ViT-S-S2	Res-50-S1	Res-50-S2	ViT-S-S1	ViT-S-S2
Scratch	30.6/53.3	41.5/65.1	13.9/23.9	18.3/31.2	6.1/4.8	9.1/6.7	1.5/0.5	2.3/1.4
(a) rand teacher	54.7/76.5	58.2/78.9	33.0/50.5	33.9/52.1	28.2/28.2	30.8/30.9	14.5/13.0	15.8/14.1
(b) rand cluster	55.1/76.2	59.9/80.2	66.1/83.4	70.6/85.8	31.4/33.8	34.9/35.4	51.4/51.9	54.4/55.8
(c) inverse cluster	46.1/70.2	54.8/77.0	66.8/83.7	69.3/85.2	13.3/11.9	17.6/16.1	47.7/49.5	49.6/51.0
(d) distance metric	74.2/88.9	76.7/89.6	64.9/82.1	69.9/85.1	37.6/38.9	41.4/44.0	52.3/53.7	56.6/58.1
(e) w/o refinement	52.2/73.3	59.8/79.3	56.6/75.8	58.9/78.1	21.6/20.9	29.4/29.0	25.2/24.2	26.6/26.9
(f) w/o \mathcal{L}_T	62.7/81.2	64.0/81.5	64.4/82.0	65.4/82.9	34.2/35.1	35.4/37.3	44.8/46.0	45.9/48.7
(g) weight of \mathcal{L}_{ref}	75.6/ 89.4	77.1/89.3	73.8/ 87.3	75.2/87.9	45.5/ 47.4	48.0/51.1	59.1/61.2	60.0/61.9
OSKT	75.7/89.4	77.6/90.6	74.2/87.1	77.2/89.0	45.7/47.3	49.2/52.2	61.2/63.5	63.9/66.4

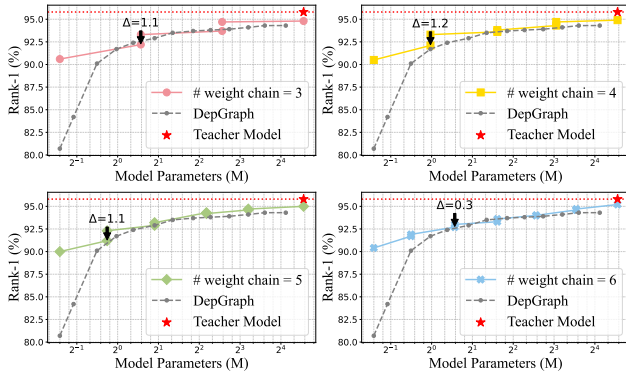


Figure 7. Scalability of weight chains refined from ResNet50 on Market1501. Each weight chain produces two models corresponding to its own width and the subsequent weight chain’s width, with the maximum inter-stage discrepancy denoted by “ Δ ”.

we obtain the results shown in (g) of Table 7.

4.4.2. Scalability Analysis of Weight Chains

When the size range of required student models is large, the weight chain’s parameter count becomes significantly smaller than the teacher model’s, leading to inefficiencies in generating larger student models. To address this, we analyze the scalability of the weight chain, determining how many chains are needed to maintain performance while covering a size range from small to teacher model size. Using ResNet50 with parameter counts spanning 1/64 to the teacher model’s size, we present experimental results in Figure 7. We explore results using multiple weight chains, each responsible for a specific width range. For a size span $[a, b]$ and s weight chains, the widths of the weight chains are generated as a geometric sequence, with the common ratio x determined by solving $a \cdot x^s = b$. For instance, given a size span $inplanes \in [8, 64]$ and $s = 3$, solving for x yields $x = 2$, resulting in widths 8, 16, and 32 for weight chains.

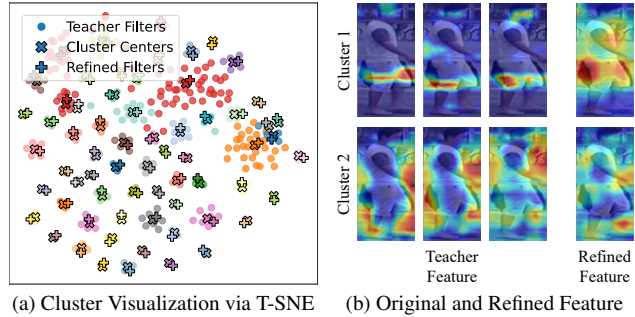


Figure 8. (a) T-SNE visualization of the clustering and refining results for the layer3.5.conv2 in ResNet50. (b) Features extracted by filters from different clusters in the teacher model and their corresponding refined filters in the weight chain.

4.4.3. Functional Analysis of Weight Chain Rows

We visualize the clustering and refining results of the weight rows, as illustrated in Figure 8a. The features extracted from the weight rows of the teacher model and their corresponding refined weight rows in the weight chain are presented in Figure 8b. Weight rows within the same cluster and their refined counterparts tend to activate similar features, as demonstrated by Cluster 1. Additionally, the refined weight rows can effectively transform noisy activation features into features containing discriminative pedestrian information, as exemplified by Cluster 2.

5. Conclusion

This study introduces a novel approach for efficiently generating high-performance person ReID models tailored to diverse resource constraints. Our OSKT meticulously refines core knowledge from the teacher model into a weight chain through a single computational pass, enabling the acquisition of resource-adaptive models without incurring additional computational costs in subsequent steps.

Acknowledgment

This research was supported by the Jiangsu Science Foundation (BK20243012, BG2024036), the National Science Foundation of China (62125602, 62206052, U24A20324, 92464301), CAAI-Lenovo Blue Sky Research Fund, and the Fundamental Research Funds for the Central Universities (2242025K30024).

References

- [1] Babajide O Ayinde, Tamer Inanc, and Jacek M Zurada. Redundant feature pruning for accelerated inference in deep neural networks. *Neural Networks*, 2019. 3
- [2] Nayan Kumar Subhashis Behera, Pankaj Kumar Sa, and Sambit Bakshi. Person re-identification for smart cities: State-of-the-art and the path ahead. *Pattern Recognition Letters*, 2020. 1
- [3] Arnav Chavan, Zhiqiang Shen, Zhuang Liu, Zechun Liu, Kwang-Ting Cheng, and Eric P Xing. Vision transformer slimming: Multi-dimension searching in continuous optimization space. In *CVPR*, 2022. 3
- [4] Tianqi Chen, Ian Goodfellow, and Jonathon Shlens. Net2Net: Accelerating Learning via Knowledge Transfer. *arXiv:1511.05641*, 2016. 2
- [5] De Cheng, Yihong Gong, Sanping Zhou, Jinjun Wang, and Nanning Zheng. Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In *CVPR*, 2016. 2
- [6] Maciej A. Czyzewski, Daniel Nowak, and Kamil Piechowiak. Breaking the Architecture Barrier: A Method for Efficient Knowledge Transfer Across Networks. *arXiv:2212.13970*, 2022. 2
- [7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 2
- [8] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv:2010.11929*, 2020. 2
- [9] Zhaopeng Dou, Zhongdao Wang, Yali Li, and Shengjin Wang. Identity-seeking self-supervised representation learning for generalizable person re-identification. In *ICCV*, 2023. 1
- [10] Gongfan Fang, Xinyin Ma, Mingli Song, Michael Bi Mi, and Xinchao Wang. DepGraph: Towards Any Structural Pruning. In *CVPR*, 2023. 2, 5
- [11] Dengpan Fu, Dongdong Chen, Jianmin Bao, Hao Yang, Lu Yuan, Lei Zhang, Houqiang Li, and Dong Chen. Unsupervised pre-training for person re-identification. In *CVPR*, 2021. 2, 5
- [12] Dengpan Fu, Dongdong Chen, Hao Yang, Jianmin Bao, Lu Yuan, Lei Zhang, Houqiang Li, Fang Wen, and Dong Chen. Large-scale pre-training for person re-identification with noisy labels. In *CVPR*, 2022. 1
- [13] Jianyang Gu, Kai Wang, Hao Luo, Chen Chen, Wei Jiang, Yuqiang Fang, Shanghang Zhang, Yang You, and Jian Zhao. MSINet: Twins Contrastive Search of Multi-Scale Interaction for Object ReID. In *CVPR*, 2023. 2, 7
- [14] Wenxuan Guo, Zhiyu Pan, Yingping Liang, Ziheng Xi, Zhicheng Zhong, Jianjiang Feng, and Jie Zhou. Lidar-based person re-identification. In *CVPR*, 2024. 2
- [15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5
- [16] Shuting He, Hao Luo, Pichao Wang, Fan Wang, Hao Li, and Wei Jiang. Transreid: Transformer-based object re-identification. In *ICCV*, 2021. 2
- [17] Weizhen He, Yiheng Deng, Shixiang Tang, Qihao Chen, Qingsong Xie, Yizhou Wang, Lei Bai, Feng Zhu, Rui Zhao, Wanli Ouyang, et al. Instruct-reid: A multi-purpose person re-identification task with instructions. In *CVPR*, 2024. 2
- [18] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv:1503.02531*, 2015. 1, 2
- [19] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. In *NeurIPS*, 1989. 1, 2
- [20] Hanjun Li, Gaojie Wu, and Wei-Shi Zheng. Combined Depth Space Based Architecture Search for Person Re-Identification. In *CVPR*, 2021. 2
- [21] Jiafeng Li, Ying Wen, and Lianghua He. Sconv: Spatial and channel reconstruction convolution for feature redundancy. In *CVPR*, 2023. 3
- [22] Siyuan Li, Li Sun, and Qingli Li. Clip-reid: exploiting vision-language model for image re-identification without concrete text labels. In *AAAI*, 2023. 1
- [23] Wei Li, Rui Zhao, Tong Xiao, and Xiaogang Wang. Deep-ReID: Deep Filter Pairing Neural Network for Person Re-identification. In *CVPR*, 2014. 5
- [24] Shengcai Liao, Yang Hu, Xiangyu Zhu, and Stan Z Li. Person re-identification by local maximal occurrence representation and metric learning. In *CVPR*, 2015. 2
- [25] Ye Lin, Yanyang Li, Ziyang Wang, Bei Li, Quan Du, Tong Xiao, and Jingbo Zhu. Weight Distillation: Transferring the Knowledge in Neural Network Parameters. *arXiv:2009.09152*, 2021. 2
- [26] Hao Luo, Youzhi Gu, Xingyu Liao, Shenqi Lai, and Wei Jiang. Bag of tricks and a strong baseline for deep person re-identification. In *CVPR workshops*, 2019. 2, 6
- [27] Hao Luo, Pichao Wang, Yi Xu, Feng Ding, Yanxin Zhou, Fan Wang, Hao Li, and Rong Jin. Self-supervised pre-training for transformer-based person re-identification. *arXiv:2111.12084*, 2021. 2
- [28] Yang Qin, Yingke Chen, Dezhong Peng, Xi Peng, Joey Tianyi Zhou, and Peng Hu. Noisy-correspondence learning for text-to-image person re-identification. In *CVPR*, 2024. 2
- [29] Yifan Sun, Liang Zheng, Yi Yang, Qi Tian, and Shengjin Wang. Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline). In *ECCV*, 2018. 2
- [30] Guanshuo Wang, Yufeng Yuan, Xiong Chen, Jiwei Li, and Xi Zhou. Learning discriminative features with multiple granularities for person re-identification. In *MM*, 2018. 2
- [31] Qiu-Feng Wang, Xin Geng, Shu-Xia Lin, Shi-Yu Xia, Lei Qi, and Ning Xu. Learngene: From open-world to your learning task. In *AAAI*, 2022. 2

- [32] Yuzhu Wang, Lechao Cheng, Manni Duan, Yongheng Wang, Zunlei Feng, and Shu Kong. Improving knowledge distillation via regularizing feature norm and direction. *arXiv:2305.17007*, 2023. 2, 5
- [33] Longhui Wei, Shiliang Zhang, Wen Gao, and Qi Tian. Person Transfer GAN to Bridge Domain Gap for Person Re-Identification. *arXiv:1711.08565*, 2018. 5
- [34] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *NeurIPS*, 2016. 2
- [35] Kunlun Xu, Xu Zou, Yuxin Peng, and Jiahuan Zhou. Distribution-aware knowledge prototyping for non-exemplar lifelong person re-identification. In *CVPR*, 2024. 2
- [36] Zhiqiu Xu, Yanjie Chen, Kirill Vishniakov, Yida Yin, Zhiqiang Shen, Trevor Darrell, Lingjie Liu, and Zhuang Liu. Initializing Models with Larger Ones. *arXiv:2311.18823*, 2023. 2, 5
- [37] Mang Ye, Jianbing Shen, Gaojie Lin, Tao Xiang, Ling Shao, and Steven CH Hoi. Deep learning for person re-identification: A survey and outlook. *IEEE TPAMI*, 2021. 1
- [38] Mang Ye, Wei Shen, Junwu Zhang, Yao Yang, and Bo Du. Securereid: Privacy-preserving anonymization for person re-identification. *IEEE TIFS*, 2024. 1
- [39] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Deep metric learning for person re-identification. In *ICPR*, 2014. 2
- [40] Junwu Zhang, Mang Ye, and Yao Yang. Learnable privacy-preserving anonymization for pedestrian images. In *MM*, 2022. 1
- [41] Quan Zhang, Lei Wang, Vishal M Patel, Xiaohua Xie, and Jianhaung Lai. View-decoupled transformer for person re-identification under aerial-ground camera network. In *CVPR*, 2024. 2
- [42] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable Person Re-identification: A Benchmark. In *ICCV*, 2015. 2, 5
- [43] Liang Zheng, Yi Yang, and Alexander G Hauptmann. Person re-identification: Past, present and future. *arXiv:1610.02984*, 2016. 1, 2
- [44] Kaiyang Zhou, Yongxin Yang, Andrea Cavallaro, and Tao Xiang. Omni-Scale Feature Learning for Person Re-Identification. In *CVPR*, 2019. 2, 7
- [45] Kuan Zhu, Haiyun Guo, Tianyi Yan, Yousong Zhu, Jinqiao Wang, and Ming Tang. Pass: Part-aware self-supervised pre-training for person re-identification. In *ECCV*, 2022. 2, 5, 6