ReMix: Training Generalized Person Re-identification on a Mixture of Data

Supplementary Material

Algorithm 1 ReMix

```
Input:
    Encoder $\theta_e$,
    Momentum encoder $\theta_m$,
    Mini-batch size $B$,
    Number of epochs $E$,
    Number of iterations in epoch $I$,
    Labeled multi-camera data $\mathcal{D}_m$,
    Unlabeled single-camera data $\mathcal{D}_s$.
Output: Trained momentum encoder $\theta_m$.
for $epoch = 1$ to $E$ do
    Obtain embeddings $\mathcal{M}_m$ from the momentum encoder $\theta_m$ for multi-camera data $\mathcal{D}_m$;
    Calculate centroids and camera centroids for multi-camera data $\mathcal{D}_m$ using embeddings $\mathcal{M}_m$;
    Get pseudo labeled part $\widetilde{\mathcal{D}}_s$ of single-camera data $\mathcal{D}_s$, as well as embeddings $\mathcal{M}_s$ from the momentum encoder $\theta_m$ and centroids using Algorithm 2;
    for $iter = 1$ to $I$ do
        Train $\theta_e$ with the general loss in Eq. (1):
            $\mathcal{L}_{cc}$ is calculated only for $\mathcal{D}_m$,
            $\mathcal{L}_{ins}$, $\mathcal{L}_{aug}$ and $\mathcal{L}_{cen}$ for $\mathcal{D}_m$ and $\widetilde{\mathcal{D}}_s$;
        Update $\theta_m$ using $\theta_e$ by Eq. (2);
    end for
end for
```

Algorithm 2 Single-camera Data Pseudo Labeling

```
Input:
    Momentum encoder $\theta_m$,
    Unlabeled single-camera data $\mathcal{D}_s$,
    Mini-batch size $B$,
    Number of iterations in epoch $I$.
Output: pseudo labeled dataset $D$, embeddings $E$ and centroids $C$.
$D \leftarrow \emptyset$                ▷ initialize a pseudo labeled dataset
$E \leftarrow \emptyset$                ▷ initialize a list of embeddings
$C \leftarrow \emptyset$                ▷ initialize a list of centroids
$counter \leftarrow 0$                  ▷ pseudo labeled images counter
$limit \leftarrow B * I$                ▷ number of images for pseudo labeling
while $counter < limit$ do
    Randomly select a video $\mathcal{V}$ from $\mathcal{D}_s$;
    Obtain embeddings $\widetilde{E}$ from the momentum encoder $\theta_m$ for images from the video $\mathcal{V}$;
    Generate a pseudo labeled dataset $\widetilde{\mathcal{D}}$ using embeddings $\widetilde{E}$ and DBSCAN;
    Calculate centroids $\widetilde{C}$ for the pseudo labeled dataset $\widetilde{\mathcal{D}}$ using embeddings $\widetilde{E}$;
    Update the pseudo labeled dataset $D$, the list of embeddings $E$ and the list of centroids $C$ using $\widetilde{\mathcal{D}}$, $\widetilde{E}$ and $\widetilde{C}$, respectively;
    Update $counter$ using $\widetilde{\mathcal{D}}$;
end while
```

6. Detailed Analysis

6.1. Clustering

In ReMix, we use two types of training data — labeled multi-camera and unlabeled single-camera data (see Algorithm 1). Since our method uses unlabeled single-camera data, pseudo labels are obtained for part of it at the beginning of each epoch. The pseudo labeling procedure occurs according to Algorithm 2. As we can see, our method uses DBSCAN [9] for clustering, which has several parameters. One of the main parameters is the distance threshold, which regulates the maximum distance between two instances in order to consider them neighbors.

If a small distance threshold is set, then DBSCAN marks more hard positive instances as different classes. In contrast, a large distance threshold causes DBSCAN to mark more hard negative instances as the same class. Therefore, it is necessary to find the optimal value of this parameter for specific data.

In our main paper, the distance threshold is set to 0.8, which is justified by the results of the experiments presented in Tab. 8. Additionally, Fig. 4 shows examples of single-camera data clusters obtained during ReMix training.

Threshold	0.65	0.70	0.80	0.85
$Rank_1$	76.3	76.3	**76.9**	76.0
mAP	60.2	60.5	**60.7**	60.1

Table 8. Comparison of different distance thresholds in DBSCAN. We train the algorithm on MSMT17-merged and single-camera data from LUPerson, and test it on DukeMTMC-reID.

6.2. Mini-batch Size

In our method, we compose a mini-batch from a mixture of images from multi-camera and single-camera datasets. Let $B_m = N_P^m \times N_K^m$ be the number of images from multi-camera data in a mini-batch, and $B_s = N_P^s \times N_K^s$ be the number of images from single-camera data in a mini-batch. So, the mini-batch has a size of $B = B_m + B_s = N_P^m \times N_K^m + N_P^s \times N_K^s$ images (Sec. 3.2). Here, N_P^m (N_P^s) is the number of labels (pseudo labels) from multi-

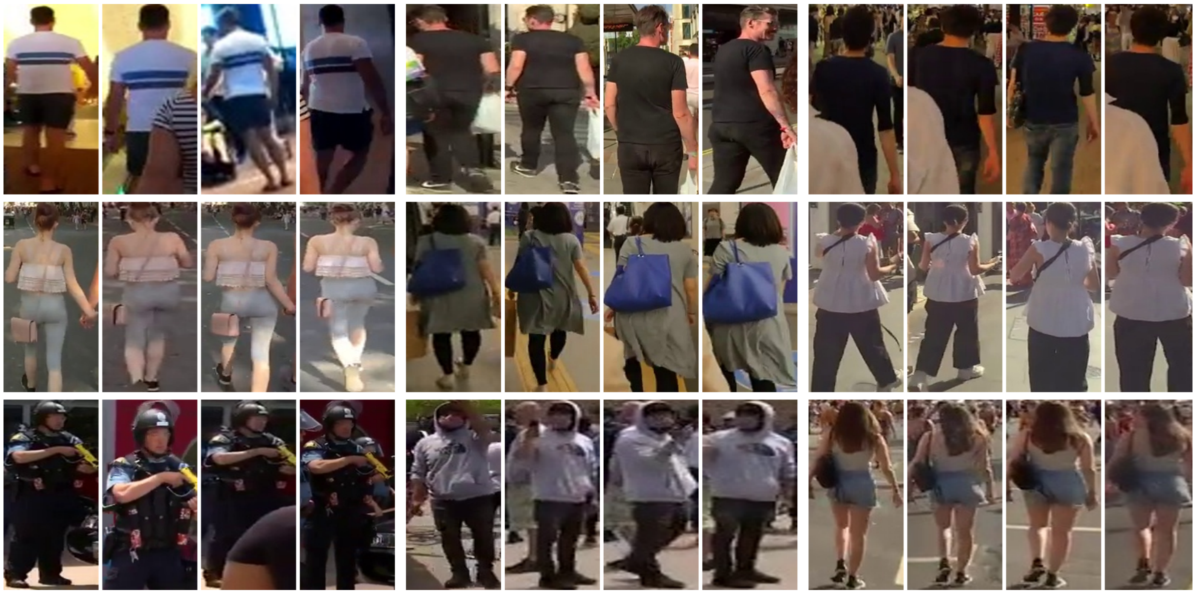

Figure 4. Examples of single-camera data clusters obtained during ReMix training. Four random images from each arbitrary cluster are selected for visualization.

camera (single-camera) data, and N_K^m (N_K^s) is the number of images for each label (pseudo label) from multi-camera (single-camera) data.

In our main paper, we set $N_P^m = N_P^s = 8$ and $N_K^m = N_K^s = 4$. Thus, the size of each mini-batch is 64 (that is, $B_m = B_s = 32$ and $B = B_m + B_s = 64$). We conduct several experiments to determine the impact of mini-batch size on the accuracy of ReMix. As can be seen from Tab. 9, the values for parameters B_m and B_s selected in our main work are among the optimal ones. The experimental results given in Tab. 10 show a relationship between the values for parameters N_K^m and N_K^s and the quality of the algorithm.

Separately, it is worth noting the influence of the value for parameter N_P^m on the quality of our algorithm. Tab. 9a shows how much the accuracy of the algorithm decreases when $B_m = 16$. A similar decrease in accuracy occurs with $N_K^m = 8$ (see Tab. 10a). This is because in both cases $N_P^m = 4$ (in the first case, $N_P^m = B_m/N_K^m = 16/4 = 4$; in the second case, $N_P^m = B_m/N_K^m = 32/8 = 4$). Thus, we can conclude that the quality of ReMix is significantly affected by the number of different labels in the mini-batch.

6.3. Input Image Size

Most works devoted to the person re-identification task use input images of size 256 × 128 pixels. Input images of the same size are used in our method. However, after studying other state-of-the-art methods in detail, we noticed that [24–26] use larger input images — 384 × 128 pixels.

B_m	$Rank_1$	mAP
16	69.1	49.0
32	**75.8**	58.7
64	75.0	**58.9**

(a) Multi-camera data.

B_s	$Rank_1$	mAP
16	77.3	61.4
32	**77.6**	**61.6**
64	77.1	61.4

(b) Single-camera data.

Table 9. Comparison of different numbers of images for each data type in a mini-batch. In ”multi-camera data” experiments, we use only MSMT17-merged for training ($N_K^m = 4$, $N_P^m = B_m/N_K^m$ and $B_s = 0$). In ”single-camera data”, we train the algorithm on MSMT17-merged and single-camera data from LUPerson ($N_K^s = 4$, $N_P^s = B_s/N_K^s$ and $B_m = 32$). The DukeMTMC-reID dataset is used for testing in all these experiments.

N_K^m	$Rank_1$	mAP
2	**76.0**	58.5
4	75.8	**58.7**
8	70.6	51.0

(a) Multi-camera data.

N_K^s	$Rank_1$	mAP
2	76.6	61.0
4	**77.6**	61.6
8	77.5	**62.1**

(b) Single-camera data.

Table 10. Comparison of different values for parameters N_K^m and N_K^s. In ”multi-camera data” experiments, we use only MSMT17-merged for training ($B_m = 32$, $N_P^m = B_m/N_K^m$ and $B_s = 0$). In ”single-camera data”, we train the algorithm on MSMT17-merged and single-camera data from LUPerson ($B_s = 32$, $N_P^s = B_s/N_K^s$ and $B_m = 32$, $N_K^m = 4$). The DukeMTMC-reID dataset is used for testing in all these experiments.

We conducted several experiments to analyze the quality of ReMix with this size of the input images. The results of these experiments are shown in Tab. 11. As can be seen, the accuracy of our method improves as the size of the in-

Image Size	Single-camera	Inference Time*	Market-1501		DukeMTMC-reID	
			$Rank_1$	mAP	$Rank_1$	mAP
256 × 128	✗	90 ms	78.4	51.7	75.8	58.7
	✓		84.0	61.0	77.6	61.6
384 × 128	✗	149 ms	79.2	51.3	76.2	59.3
	✓		**85.1**	**62.7**	**78.4**	**63.3**

* Inference speed is estimated in a single-core test on the Intel Core i7-9700K.

Table 11. Comparison of different input image sizes. We train the algorithm on MSMT17-merged and single-camera data from LUPerson (where applicable), and test it on DukeMTMC-reID.

Architecture	Single-camera	Inference Time*	Market-1501		DukeMTMC-reID	
			$Rank_1$	mAP	$Rank_1$	mAP
ResNet50-IBN	✗	90 ms	78.4	51.7	75.8	58.7
	✓		**84.0**	**61.0**	**77.6**	**61.6**
ResNet50	✗	82 ms	76.0	46.8	72.4	53.5
	✓		78.4	53.8	73.6	56.4

* Inference speed is estimated in a single-core test on the Intel Core i7-9700K.

Table 12. Comparison of different encoder architectures. We train the algorithm on MSMT17-merged and single-camera data from LUPerson (where applicable), and test it on DukeMTMC-reID.

put images increases. It is worth noting that the joint use of labeled multi-camera and unlabeled single-camera data for training also has a beneficial effect on the quality of Re-ID with larger input images. This further confirms the effectiveness of the proposed ReMix method.

Obviously, the use of larger input images can significantly increase the computational costs of the algorithm. This is confirmed by the estimates given in Tab. 11. Therefore, in our main work, we choose to prioritize method performance and resize all input images to 256 × 128.

Separately, we note that according to Tab. 6, ReMix using 256 × 128 input images outperforms others (including those methods that use 384 × 128 input images) in the cross-dataset scenario. Thus, our method achieves high accuracy while also being computationally efficient, which is important for practical applications.

6.4. Encoder Architecture

In [15, 33, 59] it was shown that using combinations of Batch Normalization and Instance Normalization improves the generalization ability of neural networks. Therefore, we compare two encoder architectures in ReMix: ResNet50 [13] and ResNet50-IBN (ResNet50 with IBN-a layers) [33]. ResNet50-IBN differs from ResNet50 only in that the former uses Instance Normalization in addition to Batch Normalization. The results of our comparison presented in Tab. 12 also demonstrate the effectiveness of ResNet50 with IBN-a layers in the cross-dataset scenario.

Moreover, our experiments show that joint training on a mixture of multi-camera and single-camera data significantly improves the accuracy of the algorithm, even when ResNet50 is used as the encoder and the momentum encoder. Additionally, according to the speed estimation of our algorithm with different encoder architectures, ResNet50-IBN is slower than ResNet50 by less than 10 ms. Therefore, the use of ResNet50 with IBN-a layers in our main paper is justified, as this architecture represents a trade-off between quality and speed.

7. Standard Person Re-ID

In our main paper, we aim to improve the generalization ability of person Re-ID methods. Our experiments in the cross-dataset and multi-source cross-dataset scenarios show that our ReMix method has a high generalization ability and outperforms state-of-the-art methods in the generalizable person Re-ID task (Sec. 4.5). We choose these test protocols because they are the closest to real-world applications of Re-ID algorithms. Indeed, in real-world scenarios, we do not have prior information about the features of capturing environments in an arbitrary scene. Therefore, person Re-ID methods should have a high generalization ability and work with acceptable accuracy in almost all possible scenes.

Even so, as we can see from Tab. 13, our method shows competitive accuracy in the standard person Re-ID task (when trained and tested on separate splits of the same dataset). It is worth noting that the other methods in this

Method	Reference	Market-1501		DukeMTMC-reID		MSMT17	
		$Rank_1$	mAP	$Rank_1$	mAP	$Rank_1$	mAP
ISP [61]	ECCV20	94.2	84.9	86.9	75.6	—	—
RGA-SC [54]	CVPR20	96.1	88.4	—	—	80.3	57.5
FlipReID [31]	EUVIP21	95.3	88.5	89.4	79.8	83.3	64.3
CAL [35]	ICCV21	94.5	87.0	87.2	76.4	79.5	56.2
CDNet [20]	CVPR21	95.1	86.0	88.6	76.8	78.9	54.7
LTReID [45]	TMM22	95.9	89.0	90.5	80.4	81.0	58.6
DRL-Net [16]	TMM22	94.7	86.9	88.1	76.6	78.4	55.3
Nformer [43]	CVPR22	94.7	91.1	89.4	**83.5**	77.3	59.8
CLIP-ReID [21]	AAAI23	95.7	89.8	90.0	80.7	84.4	63.0
AdaSP [60]	CVPR23	95.1	89.0	**90.6**	81.5	84.3	64.7
SOLIDER* [4]	CVPR23	96.1	**91.6**	—	—	**85.9**	**67.4**
ReMix (w/o s-cam.)	Ours	94.7	84.9	87.9	75.8	83.9	62.8
ReMix	Ours	**96.2**	89.8	89.6	79.8	84.8	63.9

* This is a transformer-based method.

Table 13. Comparison of our ReMix method with others in the standard person Re-ID task. In this comparison, we use two versions of the proposed method: without using single-camera data and with using single-camera data during training. Here, we use the LUPerson dataset as single-camera data to train ReMix. In this table, bold and underlining fonts suggest the best and the second-best performance, respectively.

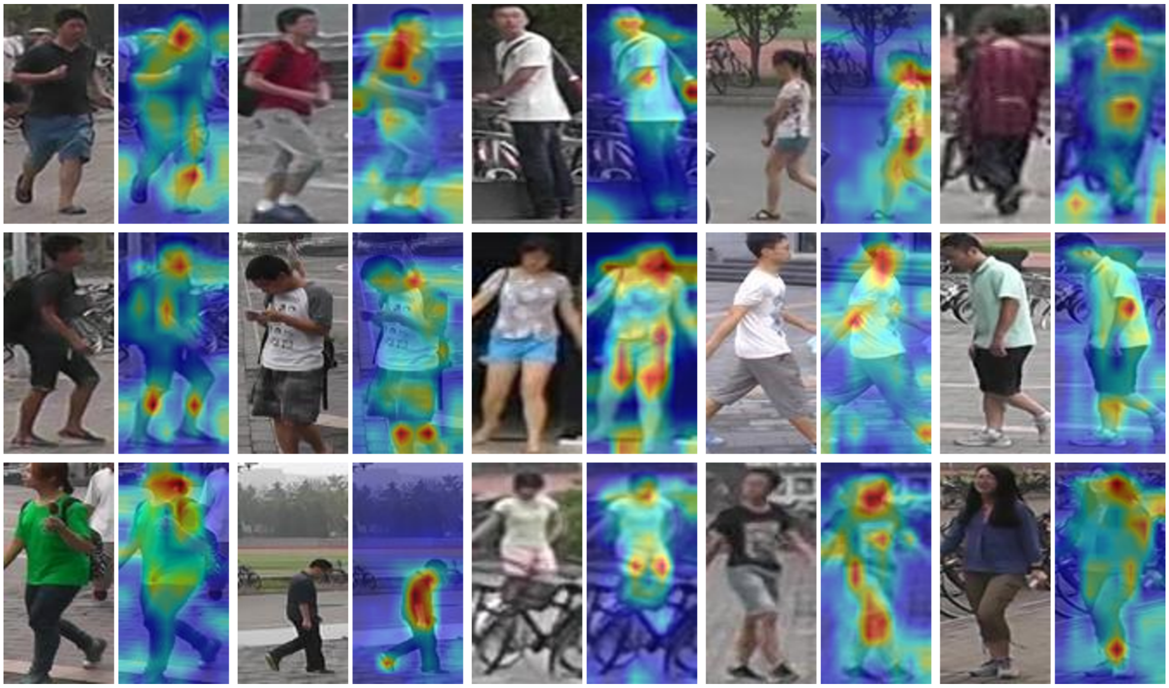

Figure 5. Visualization of activation maps of ReMix on the Market-1501 dataset.

comparison are designed specifically for standard person Re-ID scenario. At the same time, ReMix is intended as a method with high generalization ability, which should perform well in various scenes. In other words, our ReMix method is not adapted to work with a specific scene, unlike competitors. Thus, such a strong performance in this task clearly indicates the consistency and flexibility of ReMix, as well as the effectiveness of using single-camera data in addition to multi-camera data during training.

Hz	S-cam.	MOT15		MOT17	
		$MOTA$	$IDsw$	$MOTA$	$IDsw$
2	✗	83.8	70	73.8	249
	✓	**84.6**	**66**	**76.9**	**219**
4	✗	85.8	105	80.5	333
	✓	**88.0**	**90**	**83.1**	**288**
8	✗	91.6	120	88.6	375
	✓	**93.2**	**99**	**90.6**	**308**

Table 14. Impact of using single-camera data in ReMix in the tracking task. In these experiments, we use MSMT17-merged and single-camera data from LUPerson (where applicable) for ReMix training. The Deep SORT algorithm is used as a tracking method.

8. Tracking

Re-ID methods are often used as components of more practical applications, such as tracking. For example, in Deep SORT [48], the Re-ID algorithm is used to bind detections from different frames into tracks. We conduct experiments to study the impact of using single-camera data in addition to multi-camera data in ReMix not only on the quality of person Re-ID, but also on tracking.

In this study, we apply our implementation of the Deep SORT algorithm as a tracking method, using two versions of the proposed Re-ID method: without using single-camera data and with using single-camera data during training. We employ the training parts of the MOT15 [19] and MOT17 [28] benchmarks as the tracking test datasets (important: these datasets are not used to train ReMix). Since the tracking quality depends on many factors (e.g., the object detector), we use public detections from MOT15 and MOT17 to demonstrate the effectiveness of our Re-ID algorithm. In our experiments, we use Multi-Object Tracking Accuracy ($MOTA$) [1] and Number of Identity Switches ($IDsw$) [23] metrics to evaluate tracking performance. Additionally, to demonstrate the effectiveness of ReMix for binding detections from different frames into tracks, we test Deep SORT with different frame rates: 2, 4, and 8 Hz.

As can be seen from Tab. 14, the use of single-camera data in addition to multi-camera data in ReMix has a beneficial effect not only on the quality of person Re-ID, but also on tracking. With different frame rates on both benchmarks, the tracking algorithm with the proposed Re-ID method using single-camera data during training performs best. This further demonstrates the effectiveness and flexibility of ReMix. It is also important to note that in this study, we do not aim to achieve state-of-the-art results in the tracking task, but rather to demonstrate the effectiveness of our Re-ID method.