Supplementary Material for “IDM: An Intermediate Domain Module for Domain Adaptive Person Re-ID”

Yongxing Dai[1] Jun Liu[2] Yifan Sun[3] Zekun Tong[4] Chi Zhang[3] Ling-Yu Duan[1,5*]
[1] Institute of Digital Media (IDM), Peking University, Beijing, China [2] Singapore University of Technology and Design, Singapore
[3] Megvii Technology [4] National University of Singapore, Singapore [5] Peng Cheng Laboratory, Shenzhen, China
{yongxingdai, lingyu}@pku.edu.cn, jun_liu@sutd.edu.sg, sunyf15@tsinghua.org.cn

In this supplementary material, we provide more details that could not be presented in the regular paper due to the space limitation. In Section 1, we show the overall training and testing procedure. In Section 2, we provide more details about the implementation of our method. In Section 3, we compare our method with the state-of-the-arts on other two real $\rightarrow$ real UDA re-ID benchmarks. In Section 4, we analyse the effects of the hyper-parameters.

1. Overall Training and Testing Procedure

The overall training procedure of our method is shown in Algorithm 1, where we use XBM [14] as the memory bank to implement our Strong Baseline method. If using XBM, it means we implement our proposed IDM module in Strong Baseline; If not using XBM, it means we implement the IDM module in Naive Baseline. More details about XBM can be seen at Section 2.3 below. Our proposed IDM module is only used for training and is discarded for testing. In the testing procedure, we use the L2-normalized features after the global average pooling (GAP) layer followed by a batch normalization (BN) layer.

2. Implementation Details

ResNet-50 [6] pretrained on ImageNet is adopted as the backbone network. Domain-specific BNs [1] are used in the backbone network to narrow the domain gaps. Following [9], we resize the image size to 256×128 and apply some common image augmentation techniques, including random flipping, random cropping, and random erasing [25]. We perform DBSCAN [2] clustering on the unlabeled target data to assign pseudo labels at the beginning of each training epoch, in a manner like the existing UDA re-ID methods [3, 12, 5]. The mini-batch size is 128, including 64 source images of 16 identities and 64 target images of 16 pseudo identities. We totally train 50 epochs and each epoch contains 400 iterations. The initial learning rate is set as 3.5×10^{-4} which will be divided by 10 at the 20th and 40th epoch respectively. The Adam optimizer with weight decay 5×10^{-4} and momentum 0.9 is adopted in our training. The loss weights μ_1, μ_2, μ_3 are set as 0.7, 0.1, 1 respectively. In the IDM module, the FC1 layer is parameterized by $W_1 \in \mathbb{R}^{c\times 2c}$ and MLP is composed of two fully connected layers which are parameterized by $W_2 \in \mathbb{R}^{(c/r)\times c}$ and $W_3 \in \mathbb{R}^{2\times(c/r)}$ respectively, where c is the representations' channel number after the m-th stage and r is the reduction ratio. If not specified, we plug the IDM module after the stage-0 of ResNet-50 and set r as 2. The IDM module is only used in training and will be discarded in testing. Our method is implemented with Pytorch, and four NVIDIA RTX 2080Ti GPUs are used for training and only one GPU is used for testing.

2.1. Clustering on the Target Domain

As shown in Algorithm 1, we perform DBSCAN [2] clustering on the features of all the target domain samples to assign pseudo labels for the target domain samples, which is similar to the the existing clustering-based UDA re-ID methods [3, 12]. Specifically, we use the the Jaccard distance [24] as the metric in DBSCAN, where the k-reciprocal nearest neighbor set is used to calculate the pairwise similarity. We set k as 30 in our experiments. In DBSCAN, we set the maximum distance between neighbors as 0.6 and the minimal number of neighbors for a dense point as 4.

2.2. The structure of our IDM module

Our proposed IDM module is very easy to implement, including a FC1 layer and a MLP (Muti-Layer Perception) followed by a softmax function. Specifically, the FC1 layer is a fully connected layer parameterized by $W_1 \in \mathbb{R}^{c\times 2c}$. The MLP contains two fully connected layers which are parameterized by $W_2 \in \mathbb{R}^{(c/r)\times c}$ and $W_3 \in \mathbb{R}^{2\times(c/r)}$ respectively. We denote c as the channel number of the feature map at the m-th stage of ResNet-50, and denote r as the reduction ratio for the dimension reduction. In ResNet-50, the

*Corresponding Author.

Algorithm 1: The overall training procedure

```
Input: Source labeled dataset $\{(x_i^s, y_i^s)\}$ and target
       unlabeled dataset $\{x_i^t\}$;
Output: The trained backbone network $f(\cdot)$ and
        classifier $\varphi(\cdot)$;
1  Initialize the backbone network $f(\cdot)$ with the
     ImageNet-pretrained ResNet-50;
2  Initialize the XBM memory as an empty queue $M$;
3  Plug our IDM module after the $m$-th stage of the
     backbone $f(\cdot)$ and randomly initialize it.
4  for epoch $= 1$ to MaxEpochs do
5      Use the backbone $f(\cdot)$ to extract features $\{f_i^t\}$ for the
         target dataset $\{x_i^t\}$;
6      Assign pseudo labels $\{y_i^t\}$ for target domain samples
         $\{x_i^t\}$ by performing DBSCAN clustering on the
         target features $\{f_i^t\}$;
7      for iter $= 1$ to MaxIters do
8          Sample a mini-batch of samples including $n$
             source samples $\{(x_i^s, y_i^s)\}_{i=1}^n$ and $n$ target
             samples $\{(x_i^t, y_i^t)\}_{i=1}^n$;
9          Feed forward the batch into the network to obtain
             the features and predictions for the source,
             target, and intermediate domains: $(f^s, \varphi^s)$,
             $(f^t, \varphi^t)$, and $(f^{\text{inter}}, \varphi^{\text{inter}})$;
10         if using XBM then
11             Enqueue($M$, $\{(f^t, y^t)\}$, $\{(f^s, y^s)\}$);
12             if M is full then
13                 Dequeue($M$);
14             end
15             Use all entries in $M$ for the hard negatives
                 mining in the triplet loss in $\mathcal{L}_{\text{ReID}}$ from
                 Eq. (9);
16         end
17         Calculate the overall training loss by Eq. (9);
18         Update $f(\cdot)$, $\varphi(\cdot)$, and our IDM module together
             by back-propagating the gradients of Eq. (9);
19     end
20 end
```

channel number c is 64/256/512/1024/2048 after the stage-0/1/2/3/4 respectively. In Figure 1, we evaluate the effectiveness of different values on the reduction ratio r.

2.3. Implementation of XBM

Similar to many joint training UAD re-ID methods [26, 27, 13, 8, 5, 21] that use the memory bank [18, 17, 14] to improve the discriminability on the target domain, we also use the memory bank to implement our Strong Baseline method. Specifically, we use the memory bank to mine hard negatives of both source and target domains to calculate the triplet loss, similarly to XBM [14]. Following XBM [14], the memory bank is maintained and updated as a queue: for each mini-batch, we enqueue the features and (pseudo) labels of samples in this current mini-batch, and

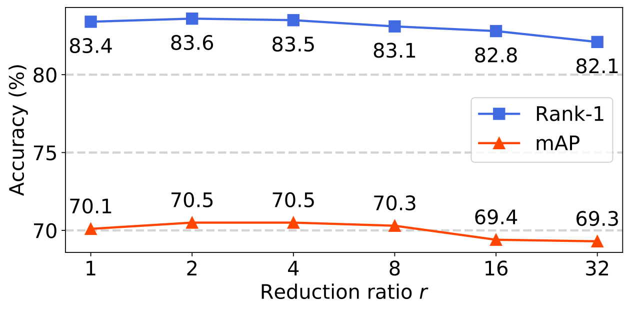

Figure 1. Performance of our method with different values of the reduction ratio r in our IDM module. Evaluating on Market $\rightarrow$ Duke when our IDM is plugged after the stage-0 of ResNet-50.

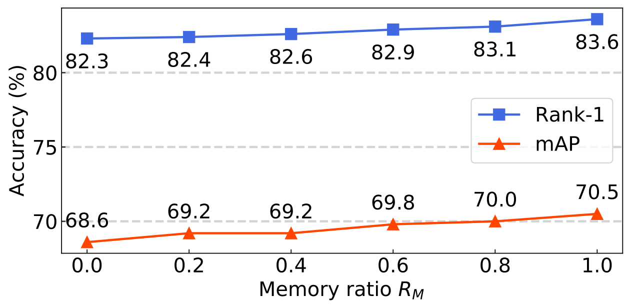

Figure 2. Performance of our method with different values of the memory ratio R_M in XBM.

dequeue the entites of the earlist mini-batch if the queue is full. For each mini-batch, we use all the entites in the memory bank to mine hard negatives for the triplet loss [7]. The procedure of maintaining and updating the XBM can be seen in Algorithm 1. Besides, we set the memory ratio as $R_M = N^M/(N^s + N^t)$, where N^M is the size the memory bank and N^s (N^t) is the number of all the source (target) training samples. If not specified, we set R_M as 1 in our experiments, *i.e.,* the size of the memory is the same as the size of the whole training dataset (including both source and target domains). We also evaluate the effectiveness on different values of R_M in Figure 2.

3. Additional Experimental Results

Some other real $\rightarrow$ real tasks are used to evaluate the UDA re-ID performances in the existing UDA re-ID methods [16, 20, 19, 28, 10, 5], where they use MSMT17 [15] as the source dataset, and use Market-1501 [22] and DukeMTMC-reID [11, 23] as the target datasets respectively. As shown in Table 1, our method can outperform the state-of-the-arts methods on these two real $\rightarrow$ real tasks by a large margin. From all the results in our regular paper and this supplementary material, our method can significantly outperform the state-of-the-arts methods in all the existing UDA re-ID benchmarks.

Table 1. Comparison with the state-of-the-art UDA re-ID methods on other real $\rightarrow$ real tasks.

Methods	Reference	MSMT17 $\rightarrow$ Market-1501				MSMT17 $\rightarrow$ DukeMTMC-reID			
		mAP	R1	R5	R10	mAP	R1	R5	R10
CASCL [16]	ICCV 2019	35.5	65.4	80.6	86.2	37.8	59.3	73.2	77.8
MAR [20]	CVPR 2019	40.0	67.7	81.9	87.3	48.0	67.1	79.8	84.2
PAUL [19]	CVPR 2019	40.1	68.5	82.4	87.4	53.2	72.0	82.7	86.0
DG-Net++ [28]	ECCV 2020	64.6	83.1	91.5	94.3	58.2	75.2	73.6	86.9
D-MMD [10]	ECCV 2020	50.8	72.8	88.1	92.3	51.6	68.8	82.6	87.1
MMT-dbscan [4]	ICLR 2020	75.6	89.3	95.8	97.5	63.3	77.4	88.4	91.7
SpCL [5]	NeurIPS 2020	77.5	89.7	96.1	97.6	69.3	82.9	91.0	93.0
IDM (Ours)	ICCV 2021	**82.1**	**92.4**	**97.5**	**98.4**	**71.9**	**83.6**	**91.5**	**93.4**

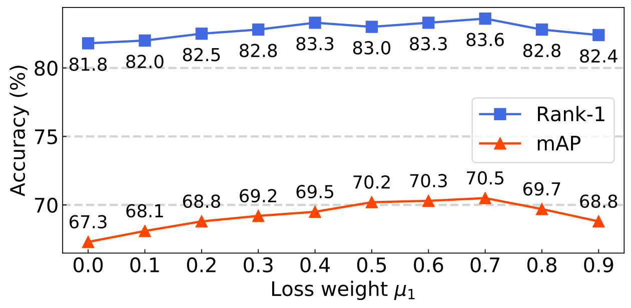

Figure 3. Performance of our method with different values of the loss weight μ_1.

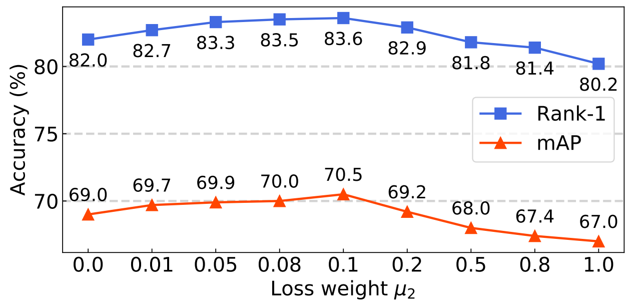

Figure 4. Performance of our method with different values of the loss weight μ_2.

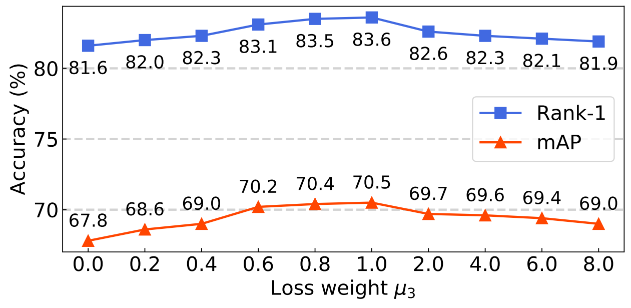

Figure 5. Performance of our method with different values of the loss weight μ_3.

4. Parameter Analysis

We tune the hyper-parameters on the task of Market $\rightarrow$ Duke, and apply the tuned hyper-parameters to all the other UDA re-ID tasks in our regular paper.

4.1. The reduction ratio r in our IDM module

In Figure 1, we evaluate the effectiveness of different values on the reduction ratio r when we plug our IDM module after the stage-0 of ResNet-50. When r gets larger, the performance gets slightly lower because the larger reduction ratio will make the IDM harder to learn. From the results in Figure 1, we set r as 2 for our method in all the other UDA re-ID tasks.

4.2. The memory ratio R_M for the XBM in our Strong Baseline.

We implement XBM [14] in our Strong Baseline, where the memory bank is set as a queue of the size N^M. We use the memory ratio $R_M = N^M/(N^s + N^t)$ to control the size of the memory bank, where N^s (N^t) is the size of the source (target) domain training dataset. We evaluate the performance on Market $\rightarrow$ Duke in Figure 2. When $R_M = 0$, it means we implement our method based on Naive Baseline, *i.e.,* "Baseline1 + Our IDM (full)" in Table 1 in our regular paper. When $R_M = 1$, it means we implement our method based on Strong Baseline, *i.e.,* "Baseline2 + Our IDM (full)" in Table 1 in our regular paper. When the memory size gets larger, the performance will get better because the larger memory bank can mine more effective negatives for the target domain. However, whether we use XBM or not, our method can outperform the baseline method by a large margin.

4.3. The loss weight μ_1

We tune the value of the loss weight μ_1 in Figure 3, where μ_1 is the weight to balance the bridge loss $\mathcal{L}^{\varphi}_{\text{bridge}}$ in Eq. (9) in our regular paper. When $\mu_1 = 0$, it means "Baseline2 + Our IDM w/o $\mathcal{L}^{\varphi}_{\text{bridge}}$". We use this bridge loss $\mathcal{L}^{\varphi}_{\text{bridge}}$ to enforce on intermediate domains' prediction space. As shown in Figure 3, the performance gets better when μ_1 ranges from 0 to 0.7. If μ varies from 0.7 to 0.9, the performance will get a little degradation because more penalization on intermediate domains' prediction space will affect the learning of the source and target domains. Thus, we set μ_1 as 0.7 in all the other experiments in our regular paper.

4.4. The loss weight μ_2

We compare the performance of different values of the loss weight μ_2 in Figure 4. The weight μ_2 is used to balance the bridge loss $\mathcal{L}^f_{\text{bridge}}$ in Eq. (9). We use $\mathcal{L}^f_{\text{bridge}}$ to enforce on intermediate domains' feature space to keep the right distance between intermediate domains to the source and target domains. When $\mu_2 = 0$, it is the same as "Baseline2 + Our IDM w/o $\mathcal{L}^f_{\text{bridge}}$" in Table 1 in our regular paper. When μ_2 gets close to 0.1, the performance gets better. If setting a large weight value of μ_2, it will bring a little performance degradation because the overall loss will penalize more on the intermediate domains' feature space while penalize less on the source and target domains. From Figure 4, we set μ_2 as an appropriate value 0.1 to better balance the bridge loss $\mathcal{L}^f_{\text{bridge}}$ in Eq. (9).

4.5. The loss weight μ_3

In Figure 5, we evaluate the performances of our method with different values of the loss weight μ_3. We use the weight μ_3 to balance the diversity loss $\mathcal{L}_{\text{div}}$ in Eq. (9). When $\mu = 0$, it means the performance of "Baseline2 + Our IDM w/o $\mathcal{L}_{\text{div}}$" in Table 1 in our regular paper. As the results reported in Figure 5, we set μ_3 as 1.0 for the experiments in all the other UDA re-ID tasks.

References

[1] Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain-specific batch normalization for unsupervised domain adaptation. In *CVPR*, pages 7354–7362, 2019. 1

[2] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, volume 96, pages 226–231, 1996. 1

[3] Yang Fu, Yunchao Wei, Guanshuo Wang, Yuqian Zhou, Honghui Shi, and Thomas S Huang. Self-similarity grouping: A simple unsupervised cross domain adaptation approach for person re-identification. In *ICCV*, pages 6112–6121, 2019. 1

[4] Yixiao Ge, Dapeng Chen, and Hongsheng Li. Mutual mean-teaching: Pseudo label refinery for unsupervised domain adaptation on person re-identification. In *ICLR*, 2020. 3

[5] Yixiao Ge, Dapeng Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Self-paced contrastive learning with hybrid memory for domain adaptive object re-id. In *NeurIPS*, 2020. 1, 2, 3

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 1

[7] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017. 2

[8] Jianing Li and Shiliang Zhang. Joint visual and temporal consistency for unsupervised domain adaptive person re-identification. In *ECCV*, pages 483–499, 2020. 2

[9] Hao Luo, Wei Jiang, Youzhi Gu, Fuxu Liu, Xingyu Liao, Shenqi Lai, and Jianyang Gu. A strong baseline and batch normalization neck for deep person re-identification. *IEEE TMM*, 22(10):2597–2609, 2019. 1

[10] Djebril Mekhazni, Amran Bhuiyan, George Ekladious, and Eric Granger. Unsupervised domain adaptation in the dissimilarity space for person re-identification. In *ECCV*, pages 159–174, 2020. 2, 3

[11] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *ECCV*, pages 17–35, 2016. 2

[12] Liangchen Song, Cheng Wang, Lefei Zhang, Bo Du, Qian Zhang, Chang Huang, and Xinggang Wang. Unsupervised domain adaptive re-identification: Theory and practice. *Pattern Recognition*, 102:107173, 2020. 1

[13] Dongkai Wang and Shiliang Zhang. Unsupervised person re-identification via multi-label classification. In *CVPR*, pages 10981–10990, 2020. 2

[14] Xun Wang, Haozhi Zhang, Weilin Huang, and Matthew R Scott. Cross-batch memory for embedding learning. In *CVPR*, pages 6388–6397, 2020. 1, 2, 3

[15] Longhui Wei, Shiliang Zhang, Wen Gao, and Qi Tian. Person transfer gan to bridge domain gap for person re-identification. In *CVPR*, pages 79–88, 2018. 2

[16] Ancong Wu, Wei-Shi Zheng, and Jian-Huang Lai. Unsupervised person re-identification by camera-aware similarity consistency learning. In *ICCV*, pages 6922–6931, 2019. 2, 3

[17] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, pages 3733–3742, 2018. 2

[18] Tong Xiao, Shuang Li, Bochao Wang, Liang Lin, and Xiaogang Wang. Joint detection and identification feature learning for person search. In *CVPR*, pages 3415–3424, 2017. 2

[19] Qize Yang, Hong-Xing Yu, Ancong Wu, and Wei-Shi Zheng. Patch-based discriminative feature learning for unsupervised person re-identification. In *CVPR*, pages 3633–3642, 2019. 2, 3

[20] Hong-Xing Yu, Wei-Shi Zheng, Ancong Wu, Xiaowei Guo, Shaogang Gong, and Jian-Huang Lai. Unsupervised person re-identification by soft multilabel learning. In *CVPR*, pages 2148–2157, 2019. 2, 3

[21] Kecheng Zheng, Cuiling Lan, Wenjun Zeng, Zhizheng Zhan, and Zheng-Jun Zha. Exploiting sample uncertainty for domain adaptive person re-identification. In *AAAI*, 2021. 2

[22] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *ICCV*, pages 1116–1124, 2015. 2

[23] Zhedong Zheng, Liang Zheng, and Yi Yang. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In *ICCV*, pages 3754–3762, 2017. 2

[24] Zhun Zhong, Liang Zheng, Donglin Cao, and Shaozi Li. Re-ranking person re-identification with k-reciprocal encoding. In *CVPR*, pages 1318–1327, 2017. 1

[25] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *AAAI*, volume 34, pages 13001–13008, 2020. 1

[26] Zhun Zhong, Liang Zheng, Zhiming Luo, Shaozi Li, and Yi Yang. Invariance matters: Exemplar memory for domain adaptive person re-identification. In *CVPR*, pages 598–607, 2019. 2

[27] Zhun Zhong, Liang Zheng, Zhiming Luo, Shaozi Li, and Yi Yang. Learning to adapt invariance in memory for person re-identification. *IEEE TPAMI*, 2020. 2

[28] Yang Zou, Xiaodong Yang, Zhiding Yu, BVK Kumar, and Jan Kautz. Joint disentangling and adaptation for cross-domain person re-identification. In *ECCV*, 2020. 2, 3