An Attention-driven Two-stage Clustering Method for Unsupervised Person Re-Identification Supplementary Materials

Zilong Ji[1], Xiaolong Zou[2], Xiaohan Lin[2], Xiao Liu[3], Tiejun Huang[2], and Si Wu[2,3]

[1] State Key Laboratory of Cognitive Neuroscience and Learning, Beijing Normal University, China. jizilong@mail.bnu.edu.cn
[2] School of Electronics Engineering & Computer Science, Peking University, Beijing, China.
[3] IDG/McGovern Institute for Brain Research,Peking-Tsinghua Center for Life Sciences, Academy for Advanced Interdisciplinary Studies, Peking University, Beijing, China.
{xiaolz,Lin.xiaohan,xiaoliu23,tjhuang,siwu}@pku.edu.cn

1 Hyper-parameters used in our ADTC model

Here we list the hyper-parameters we used in our model. Results reported in the main text is based on the settings in Table 1 here. The results of different settings of hyper-parameters are given in Sec. 2.

2 Robustness analysis of the model

We carry out experiments to explore the robustness of our model with respect to hyper-parameters. Here we mainly focus on four hyper-parameters: the number of clusters, the margin in the triplet loss, the number of hidden units (bottleneck units) in the channel branch of the voxel attention module and the size of dimension reduction of the features for clustering. Each time, we vary one hyper-parameter in a range each time and the others are fixed. First we show the effect of the number of clusters, as this is unknown in advance but needs to pre-specified[4]. We check the model performance on Market1501 by setting different numbers of clusters, varying from 300 to 1500, in two-stage clustering. Table 2 shows that the model performance is quite robust over a wide range of cluster numbers, even when the data is largely under-segmented or over-segmented.

Results of varying other hyper-parameters are given in Table 3. We see that the performances of our model are rather robust with respect to the triplet margin, the number of hidden units (bottleneck units) in the channel branch of the voxel attention module and the size of dimension reduction of the features for clustering. It seems that the size of nearest neighbours p in the two-stage clustering has an impact on the model performance. The reason is when p is small ($p = 5$), the number of selected images is small,

[4] Although some methods such as density-based clustering do not require specifying the number of clusters, they do need to pre-define the minimum number of points for the neighborhood and the maximum distance for two points to be considered as in the same cluster. To our best knowledge, there is no clustering algorithm which does not impose any pre-defined constraint on clustering.

Data pre-processing:	
Image size	256×128
Random horizontal flip	True
Color jitter	False
Image crop	False
Normalize to (0,1)	True
Standardization	True
RandomErasing	False
Optimization related parameters:	
Optimizer	Adam
Learning rate	0.0001
Epochs per round	10
Exponentially decay at epochs	5
Weight decay	0.0005
IDs per batch	32
Images per ID	4
Voxel attention related parameters:	
Number of units in bottleneck (d)	800
Two-stage clustering related parameters:	
Number of clusters (M)	1000
Number of PCs (for dimension reduction)	50
The size of nearest neighbours (p)	20
Loss function related parameters:	
Triplet margin	0.3
Hard example mining	True
Identification loss (softmax loss)	False

Table 1. Hyper-parameters used in our ADTC model.

which may lead to under-fitting of our model. Table 3 also shows that when the number of update epochs per round is too large (corresponding to a small clustering frequency), the model performance is degraded considerably. The underlying reason is also straightforwardly understandable. If the number of parameters update at each round is too large, the model tends to overfit to the current pseudo labeled data. This is dangerous, since at the beginning of training, the feature points are intertwined with each other which means that the pseudo labels are unreliable , and over-training on these unreliable data will lead to degenerate performances. In practice, we can avoid this by not setting the number of updating epochs at each round too large.

M	300	500	700	900	1100	1300	1500
mAP	46.1	51.8	53.8	56.5	58.3	55.5	53.6
rank1	70.0	74.5	75.5	77.1	78.3	76.8	75.9

Table 2. The model performance vs. the cluster number. Experiments are carried out on Market1501. The ground truth for the number of clusters is 751.

	mAp	rank1	rank5	rank10
the triplet margin (m):				
$m = 0.1$	55.4	76.8	88.9	92.5
$m = 0.2$	58.5	78.3	90.6	93.5
$m = 0.3$	59.7	**79.3**	90.8	94.1
$m = 0.4$	**60.1**	79.1	**90.9**	**94.4**
$m = 0.5$	59.4	78.7	89.9	93.4
the number of units in bottleneck (d):				
$d = 700$	58.9	**79.3**	**90.9**	93.9
$d = 800$	**59.7**	**79.3**	90.8	**94.1**
$d = 900$	59.1	78.4	89.9	93.2
$d = 1000$	57.5	76.8	89.7	93.1
the number of PCs (for dimension reduction):				
$PCs = 50$	**59.7**	**79.3**	90.8	**94.1**
$PCs = 100$	57.6	78.1	90.4	93.6
$PCs = 150$	59.5	79.1	**90.9**	94.0
$PCs = 200$	57.7	77.2	89.7	93.3
the size of nearest neighbours (p):				
$p = 5$	45.9	70.6	85.2	89.9
$p = 10$	49.4	72.7	87.7	92.0
$p = 15$	49.7	73.7	87.7	91.7
$p = 20$	**59.7**	**79.3**	**90.8**	**94.1**
the number of updated epochs per round (E):				
$E = 2$	54.2	76.1	89.3	92.5
$E = 5$	53.9	75.1	88.7	92.4
$E = 10$	**59.7**	**79.3**	**90.8**	**94.1**
$E = 20$	41.3	66.3	82.0	87.1
$E = 30$	31.7	56.7	75.4	80.7
$E = 40$	24.8	49.1	68.3	75.7

Table 3. Model robustness to hyper-parameters. Experiments are carried out on the Market1501 dataset.

3 Model performance under an unbalanced data distribution

The unlabeled data distribution is an important factor which influences the model performance under the unsupervised setting. The benchmark datasets we have used are well balanced for different classes. In reality, however, the collection of pedestrian images is typically biased. For example, the people working nearby and the children playing around the cameras tend to have higher chances of being included. It is therefore important to check the robustness of our model for unbalanced data. Using Market1501 (the result for DukeMTMC is similar), we artificially generate a set of unbalanced data and a set of balanced data having equal numbers. Here we first describe the process of unbalanced data generation.

Denote $\boldsymbol{P}_i$ the set of images of a pedestrian i in the dataset X, which is written as,

$$\boldsymbol{P_i} = \{P_i^1\} \cup \{P_i^2\} ... \cup \{P_i^R\}, \tag{1}$$

where the subset $\{P_i^r\}$ represents the images of the pedestrian collected from the camera r, $1 \leq r \leq R$. R is the number of cameras that $\boldsymbol{P}_i$ collected from. To generate the unbalanced data for each pedestrian, we select images from $0 \leq S \leq R$ cameras, with S a random number uniformly distributed in the range $[0, R]$. These S cameras are randomly sampled from all cameras. By this, the numbers of images from different pedestrians become highly unbalanced (see Fig. 1A). The obtained unbalanced data is written as:

$$\boldsymbol{P}_{unbalanced,i} = \{P_i^{r_1}\} \cup \{P_i^{r_2}\} ... \cup \{P_i^{r_s}\}, \tag{2}$$

where $r_1, r_2, ..., r_s$ is the indexes of the selected cameras, and $r_s \leq R$. We only create an unbalanced dataset for Market1501. The model performance for unbalanced data of DukeMTMC is similar.

For the generated unbalanced dataset, the total number of images (6624) is about a half of the original images (12936). To make a fair comparison, we trained the model with a balanced dataset having the similar size as the unbalanced data. Since the original Market 1501 is well-balanced when collected, for each pedestrian, we simply randomly sample half images from each $\{P_i^r\}$ to get the balanced dataset. The total number of images in the created balanced dataset is 6473. We observe that our model achieves a slightly degraded but still comparable performance on the unbalanced data compared to that on the balanced one (see Fig. 1B).

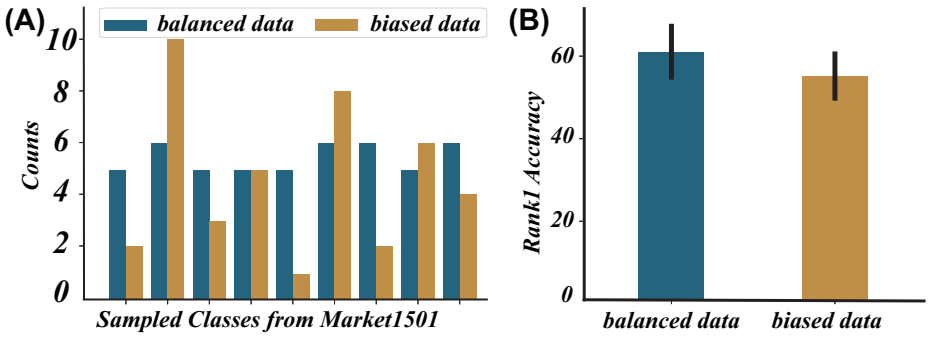

Fig. 1. (A) Illustrating the balanced and biased data distributions used for robustness analysis. (B) The model performances in the two data distributions. Market1501 is used.

4 Model performances on the labeled dataset

Following the convention, before training our model on the unlabeled dataset (the target dataset), we initialize the model on a labeled dataset (the source dataset). Table 4 shows the hyper-parameters we used when trained on the source dataset. The model architecture is the same as we described in Sec.4.2, i.e., a Resnet50 pre-trained on ImageNet as

data pre-processing:	
Image size	256×128
Random horizontal flip	True
Color jitter	False
Image crop	False
Normalize to (0,1)	True
Standardization	True
RandomErasing	True
Optimization related parameters:	
Optimizer	Adam
Learning rate	0.0002
Training epochs	200
Exponentially decay at epochs	100
Weight decay	0.0005
IDs per batch	32
Images per ID	4
Loss function related parameters:	
Triplet margin	0.3
Hard example mining	True
Identification loss (softmax loss)	False

Table 4. Hyper-parameters used in supervised training on the source dataset.

	mAp	rank1	rank5	rank10
Market1501	72.7	88.3	95.4	97.6
Market1501 to DukeMTMC	18.2	34.0	49.1	55.9
DukeMTMC	62.8	79.2	89.9	93.1
DukeMTMC to Market1501	18.8	44.0	62.1	69.4

Table 5. The performances of the initialized model on the source dataset and on the target dataset before training.

the backbone followed by the global pooling layer and a batch normalization layer. Notably, we only initialize the model on the source labeled dataset and then train it without any auxiliary label information in the unlabeled domain (as described in Sec.4.3).

Table 5 presents the performance of the model trained on the source dataset and the performance of the trained model directly applied on the unlabelled target dataset, which can be treated as the baseline for comparison in Table 1 and Table 2 in the main text. Although the baseline performance is quite low (mAp/rank1: $18.2\%/34.0\%$ on DukeMTMC and $18.8\%/44.0\%$ on Market1501) at the beginning (serving as a warmup), it is important for our method to exploit this weak signal to bootstrap the discriminating power of the proposed ADTC model.

5 Improving the clustering quality with voxel attention

Here, we demonstrate that voxel attention contributes to improving the clustering quality by visualizing the learned features. As shown in Fig. 2, with voxel attention, data

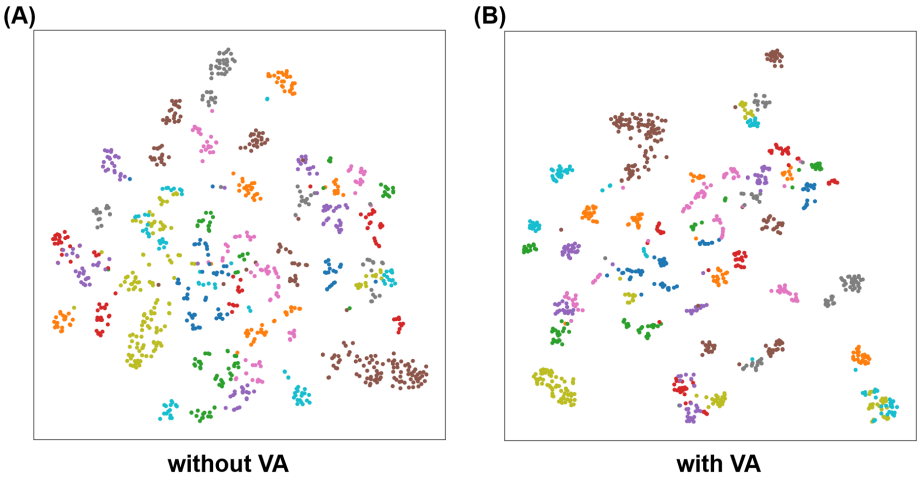

Fig. 2. T-SNE visualization of input images after training. 50 classes of person images are selected from DukeMTMC for visualization. Different colors represent different classes.

points belonging to the same class are better aggregated than that without voxel attention. Specifically, with voxel attention, the margin between different classes are enlarged, making the retrieval much easier.

6 More examples with the two-stage clustering strategy

The two-stage clustering strategy we used serves as a refinement of the original kmeans clustering algorithm (see Sec.3.2 and Sec.4.5), which is crucial for our model (see Sec.4.7). Here, we present more examples displaying the superior performance of two-stage clustering compared to the kmeans algorithm (Fig. 3).

Fig. 3. (A-B): two clusters from Market1501. (C-D): two clusters from DukeMTMC. Upper panels: results without two-stage clustering. Lower panels: results with two-stage clustering. Images in red box: wrongly assigned images in the cluster. Distance to the cluster centroid increases from left to right.