Progressive Transfer Learning for Person Re-identification

Zhengxu Yu[1], **Zhongming Jin**[2], **Long Wei**[1], **Jishun Guo**[4], **Jianqiang Huang**[2], **Deng Cai**[1], **Xiaofei He**[1,3] and **Xian-Sheng Hua**[2]
[1] State Key Lab of CAD&CG, Zhejiang University, Hangzhou, China,
[2] DAMO Academy, Alibaba Group, Hangzhou, China,
[3] Fabu Inc., Hangzhou, China,
[4] GAC R&D Center, Guangzhou, China
yuzxfred@gmail.com, zhongming.jinzm@alibaba-inc.com,
longwei@zju.edu.cn, guojishun@gacrnd.com, jianqiang.hjq@alibaba-inc.com,
{dengcai, xiaofeihe}@cad.zju.edu.cn, huaxiansheng@gmail.com

Abstract

Model fine-tuning is a widely used transfer learning approach in person Re-identification (ReID) applications, which fine-tuning a pre-trained feature extraction model into the target scenario instead of training a model from scratch. It is challenging due to the significant variations inside the target scenario, e.g., different camera viewpoint, illumination changes, and occlusion. These variations result in a gap between the distribution of each mini-batch and the distribution of the whole dataset when using mini-batch training. In this paper, we study model fine-tuning from the perspective of the aggregation and utilization of the global information of the dataset when using mini-batch training. Specifically, we introduce a novel network structure called Batch-related Convolutional Cell (BConv-Cell), which progressively collects the global information of the dataset into a latent state and uses this latent state to rectify the extracted feature. Based on BConv-Cells, we further proposed the Progressive Transfer Learning (PTL) method to facilitate the model fine-tuning process by joint training the BConv-Cells and the pre-trained ReID model. Empirical experiments show that our proposal can improve the performance of the ReID model greatly on MSMT17, Market-1501, CUHK03 and DukeMTMC-reID datasets. The code will be released later on at https://github.com/ZJULearning/PTL

1 Introduction

Person re-identification (ReID) is to re-identify the same person in different images captured by different cameras or at different time. Due to its wide applications in surveillance and security, person ReID has attracted much interest from both academia and industry in recent years.

With the development of deep learning methods and the newly emerged person ReID datasets, the performance of person ReID has been significantly boosted recently. However, several open problems remain. First, training a feature extraction model from scratch need a large volume of annotation data, but the annotated data is hard to acquire in person ReID tasks due to the poor quality of the image and the privacy concerns of pedestrians. Hence, making use of the existing datasets to help training the feature extractor have attracted great attention in the community. Second, the significant variations between different scenarios and within the same scenario make the person ReID task challenging. A noticeable performance degradation often occurs if we directly apply a pre-trained model on the target dataset without fine-tuning it into the target scenario.

Most of the recently proposed works [Deng *et al.*, 2018; Ma *et al.*, 2018] have focused on mitigating the impact of variations between different datasets. Most of these works focus on transferring the image style of the target domain and the source domain to the same by using Generative Adversarial Networks (GANs) based models. However, the imperfect style transferring models can bring in noises and potentially change the data distribution of the whole dataset. Meanwhile, the person ID in the generated images is not guaranteed to be the same as in the real images.

As for variations inside the dataset, which we focused in this work, it is less mentioned in recently proposed works. The distribution difference between each mini-batch and the entire dataset caused by internal variations has a significant influence on the model fine-tuning process. This difference leads to a deviation of gradient estimation and thus affect the effect of model fine-tuning. The most straightforward approach to mitigate this problem is increasing the batch size. However, Keskar et al. [Keskar *et al.*, 2016] and our experiments revealed that using a large-batch setting tends to converge to sharp minimizers, and further leads to poorer performance.

Moreover, most of the state-of-the-art deep learning methods in person ReID task have used an off-the-shelf network, like DenseNet [Huang *et al.*, 2017] and ResNet [He *et al.*, 2016], as backbone network. However, Deep CNNs are difficult to initialize and optimize with limited training data. Therefore, model fine-tuning is widely used to mitigate shortages of annotated training data in person ReID tasks, which

make the study of how to mitigate the impact of the internal variation more critical. For instance, most of the off-the-shelf models used in ReID tasks are pre-trained on a relatively larger dataset like ImageNet [Russakovsky *et al.*, 2015] and then fine-tuning into the target dataset.

In this paper, we study how to mitigate the impact of internal variations from the viewpoint of aggregation and utilization of the global information of the dataset. First, we propose a novel CNN building block, which we call the Batch-related Convolutional Cell (BConv-Cell). The BConv-Cell progressively aggregates the global information of the dataset into a latent state in a batch-wise manner. The latent state aggregated in previous batches will be used to mitigate the impact of the internal variations in the following batches. Based on the BConv-Cells, we further propose the Progressive Transfer Learning (PTL) method to fine-tune the pre-trained model by integrating it with the BConv-Cells. We conduct extensive experiments on MSMT17 [Wei *et al.*, 2018], Market-1501 [Zheng *et al.*, 2015], CUHK03 [Li *et al.*, 2014] and DukeMTMC-reID [Zheng *et al.*, 2017] datasets to show that our proposal can effectively promote the ReID performance.

We summarize the contributions of this work as follows:

1. We propose a novel network structure called the Batch-related Convolutional Cell (BConv-Cell). In mini-batch training, the BConv-Cells can progressively aggregate the global information of the dataset, and then use this information to help optimize model in the next batches.
2. Based on the BConv-Cells, we then propose the Progressive Transfer Learning (PTL) method to fine-tune a pre-trained model into the target scenario by integrating the BConv-Cells.
3. The experimental results show that the model fine-tuned by using our proposal can achieve state-of-the-art performance on four persuasive person ReID datasets.

2 Batch-related Convolutional Cell

The BConv-Cell is based on a straightforward thought that making use of the global information of the dataset to mitigate the adverse influence caused by internal variation.

The BConv-Cell is inspired by the Conv-LSTMs [Xingjian *et al.*, 2015]. However, there are several fundamental difference between the BConv-Cells and the Conv-LSTMs. First, there is no time concept and explicit temporal connections between inputs in the BConv-Cells. Meanwhile, the BConv-Cells is not designed to handle sequential inputs but single segmented images. Second, the BConv-Cells have a different architecture from the Conv-LSTMs. The BConv-Cells only maintain a latent state which contained the aggerated global information, but the Conv-LSTMs reserved both the hidden state and the cell state. Moreover, the BConv-Cells is not designed to conduct prediction.

By using the memory mechanism, the BConv-Cells can progressively collect global information and use it to facilitate the parameter optimization process during fine-tuning. More than that, different from other LSTM based methods like meta-learners, the output of the BConv-Cells can be directly used as the extracted feature. Meanwhile, the nature of the BConv-Cells is a stack of Conv-layers, so it can be used as a building block of a multi-layer feature extraction network.

The key equations of the BConv-Cell have shown as follow:

$$\begin{aligned} i_b &= \sigma(W_{xi} * x_b + b_i) \\ f_b &= \sigma(W_{xf} * x_b + b_f) \\ o_b &= \sigma(W_{xo} * x_b + b_o) \\ C_b &= f_b \circ C_{b-1} + i_b \circ tanh(W_{xc} * x_b + b_c) \\ y_b &= o_b \circ tanh(C_b), \end{aligned} \tag{1}$$

where $*$ denotes the convolution operator, $\circ$ denotes the Hadamard product, σ denotes a sigmoid function, x_b is the input of the BConv-Cell in b-th batch. i_b, f_b and o_b is the output of input gate i, forget gate f and output gate o respectively, C_b is the latent state reserved after b-th batch, W is the weight of the corresponding convolutional layer in the BConv-Cell and y_b is the output of the BConv-Cell. All the input x_b, latent state C_b and gate output i_b, f_b, o_b are 3-dimensional tensors.

As shown in Eq. 1, the output y_b is determined by the latent state C_b and the input x_b. The latent state C_b is determined by the input x_b and C_{b-1}. From the fourth formula of Eq. 1, we can notice that the C_b maintains part of the information of all the historical input batches. The iteration formula of latent state C_b as:

$$C_b = g(x_1, x_2, ..., x_b), \tag{2}$$

where g is the simplified notation of the composition of functions $\{g_i | 1 \leq i \leq b\}$.

3 Progressive Transfer Learning Network

3.1 Progressive Transfer Learning

Given an off-the-shelf CNN as the backbone network, we pair up the BConv-Cells with its Conv-blocks to form a new network, and we name it as the progressive transfer learning (PTL) network. A sketch of the PTL network has shown in Figure 1. The red dotted box denotes a building block of the PTL network, which formed by a BConv-Cell, a 1x1 Conv-layer and the Conv-block of the backbone network. Formally, we define this building block as:

$$\begin{aligned} x_b^i &= F_{conv}(x_b^{i-1}) \\ y_b^i &= F_{bconv}(F_{1\times1}(x_b^i, y_b^{i-1}), C_{b-1}^i) \\ C_b^i &= g(x_1^i, x_2^i, ..., x_b^i), \end{aligned} \tag{3}$$

where x_b^0 indicate the input image of b-th batch, x_b^i is the output of the i-th ($i >= 1$) Conv-block in b-th batch, y_b^i is the output of the i-th BConv-Cell. Eq. 3 only contains the second and the third equation when $i = 0$. The function F_{conv} and F_{bconv} represent the mapping function learned by Conv-block and BConv-Cell respectively. $F_{1\times1}$ is the 1x1 Conv-layer as shown in Figure 1. C_b^i is the latent state of the i-th BConv-Cell after the b-th batch. The structure of the Conv-block is flexible, which can be replaced by Conv-block of many Deep CNNs like DenseNet or ResNet.

As shown in Eq. 3, the BConv-Cell learn the mapping function from input to feature space while collecting global information and updating the latent state. We can notice from

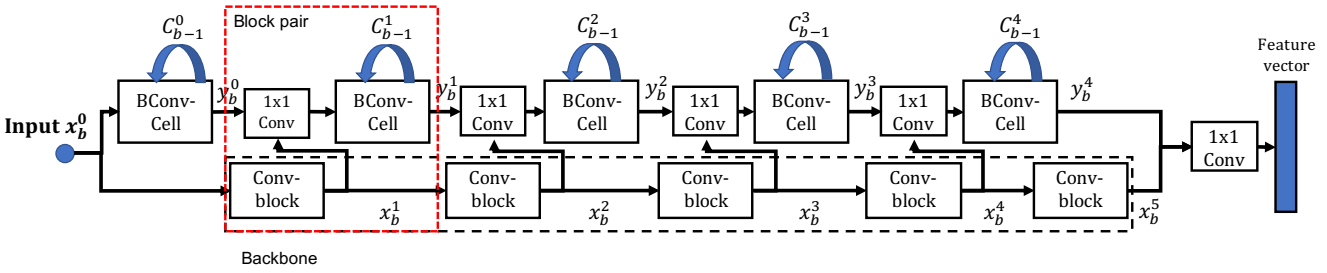

Figure 1: Sketch of the PTL network. The black dotted box indicates the backbone network. x_b^i and y_b^i are the outputs of the i-th Conv-block of the backbone and the related BConv-Cell respectively, x_b^0 denotes the input image, b indicates the b-th input batch, C_{b-1}^i is the latent state of the i-th BConv-Cell after the last batch. The red dotted box denotes the block pair of the Conv-block and the BConv-Cell. In each block pair, x_b^i and y_b^{i-1} are concatenated and feeding into a 1x1 Conv layer before feed into the BConv-Cell. The output of the last BConv-Cell and that of backbone network are concatenated before feeding into the 1x1 Conv-layer. The latent state of the BConv-Cell is stored after every batch and feedback to the same BConv-Cell when next batch coming.

Eq. 3 that the discriminative knowledge of the past batches is progressively aggregated into the latent state.

3.2 Network Architecture

We have tested the PTL method with several different structures of backbone networks, including DenseNets and ResNets. We use the DenseNet-161 as backbone network to describe the construction of the PTL network.

The DenseNet-161 consists of five Conv-blocks, we use four BConv-Cells to pair up with the top four Conv-blocks as shown in the Figure 1. At the top of the network, we use a BConv-Cell to capture the low-level feature of the input image, which is shown in the left of Figure 1. At the bottom of the network, the output of the last BConv-Cell is concatenated with the output of the last Conv-block and then feed into a 1x1 Conv-layer to get the feature vector. During training, the feature vector is then fed into a classifier which contains three Fully connection layers. For simplicity, the classifier is not shown in Figure 1. During evaluating, we directly use the feature vector conduct image retrieve.

As we can see in the Figure 1, feature maps transmit along two dimensions in the PTL network. The first is batch iteration, BConv-Cells evolve the latent states with each input batch and transmit it to the next batch. The second is the depth of the network, in which feature maps transmit from the first layer to the last layer.

During testing, we set all the latent states as zeros. To simulate the test condition, all the latent states are set to zeros at the beginning of each epoch during training. This setting ensures that historical knowledge is progressively collected and aggregated only once in each epoch.

As we mentioned above, the backbone in Figure 1 can be replaced by most of the commonly used feature extraction networks. In this work, we use ResNet-50, DenseNet-161 and MGN [Wang *et al.*, 2018] as backbone network.

3.3 Parameter Optimization Procedure

Our proposal facilitate parameter optimization by using the BConv-Cells to cooperate with the backbone network, which does not limit the selection of the optimization method. Hence, the combined model still can be optimized by using commonly used optimizers like SGD and SGD-M.

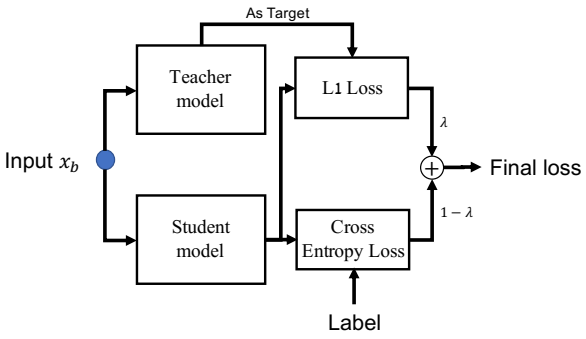

Figure 2: The implementation of STD method. The teacher model is set to evaluation mode during the whole process.

We argue that the PTL method can make up for two shortcomings of SGD-M optimizer. First, in SGD-M, the historical gradient is aggregated in a linear sum roughly by using humanly pre-defined weights, which make it inflexible and not optimized. Second, the loss after each batch only determined by the current input batch, which has a strong bias and leading to performance oscillation during training.

By using the PTL method, the historical gradient aggregation is replaced by calculating the gradient of a composition function recursively with learnable weights. More than that, the sample bias of current batch can be mitigated by using the historical knowledge carried by the learned latent states C_b.

4 Student-Teacher Distillation Method

Compared with the backbone network, the parameter number of the PTL network grew up inevitably. To fairly compare with baselines, we introduce an improved model distillation method called Student-Teacher Distillation (STD) method to fine-tune a backbone model (like ResNet-50) by using the fine-tuned PTL model. The STD method is not essential for the PTL method in practical applications.

As the prerequisite, we assume we have obtained a fine-tuned model by using our proposed PTL method. We then introduce a new objective function for model distillation. The objective function consists of two parts. First is a cross en-

tropy loss between the output predictions of the student model and the ground truth. The second is a L1 loss between the output feature vector of the student model and that of the teacher model. The new objective function is given by:

$$L_{distill} = (1 - \lambda)L_{CE} + \lambda L_{l1}, \quad (4)$$

where λ is a hyper-parameter to adjust the ratio of the cross-entropy loss and the L1 loss. This new object function combines both supervision information and merit of the PTL network to extract discriminative feature.

The implementation of STD method has shown in Figure 2. We set the teacher model to evaluation mode during the whole process. The input image feeds into teacher and student model at the same time. After which, the parameter of the student network will be updated according to the proposed objective function in Eq. 4. After training, the teacher model can be abandoned.

5 Experiments

We first carried out model fine-tuning experiments with our proposal on four convincing ReID datasets and compared it with both the state-of-the-art ReID methods and several transfer-learning methods. We then conduct model transferring experiments among multiple datasets to evaluate the performance of the PTL method when handling multiple step transferring.

5.1 Dataset

We selected four persuasive ReID datasets to evaluate our proposal, including Market-1501, DukeMTMC-reID, MSMT17 and CUHK03.

Market-1501. The Market-1501 dataset contains 32,668 annotated bounding boxes of 1,501 identities.

DukeMTMC-reID. The DukeMTMC-reID dataset contains 1,404 identities. 702 IDs are selected as the training set and the remaining 702 IDs as the testing set.

MSMT17. The raw video on the MSMT17 dataset is recorded in 4 days with different weather conditions in a month using 12 outdoor cameras and three indoor cameras. The MSMT17 dataset contains 126,441 bounding boxes of 4,101 identities. We followed the same dataset split by Wei et al. [Wei *et al.*, 2018], and we also used the evaluation code provided by them (https://github.com/JoinWei-PKU/MSMT17_Evaluation).

CUHK03. The CUHK03 dataset consists of 14,097 images of 1,467 persons from 6 cameras. Two types of annotations are provided in this dataset: manually labeled pedestrian bounding boxes and DPM-detected bounding boxes. We followed the same dataset split as used in the [Wang *et al.*, 2018]. For all experiments on Market-1501, DukeMTMC-reID and CUHK03, we used the evaluation code provided in Open-ReID (https://github.com/Cysu/open-reid).

Market-Duke. We use the training sets of the two datasets Market-1501 and DukeMTMC-reID to form a new dataset called the Market-Duke dataset. We further use this dataset to train the models to compare the difference between one-step model fine-tuning and multi-step model fine-tuning. All validation, query and gallery set of these two datasets are abandoned.

Method	#Param.	mAP	CMC-1
GoogLeNet [Wei *et al.*, 2018]	-	23.00	47.60
PDC [Wei *et al.*, 2018]	-	29.70	58.00
GLAD [Wei *et al.*, 2018]	-	34.00	61.40
ResNet-50	28m	28.63	59.77
ResNet-50+PTL	35m	32.58	62.76
DenseNet-161	32m	38.60	70.80
DenseNet-161+PTL	42m	**42.25**	72.65
DenseNet-161+PTL+STD	32m	41.38	**73.12**

Table 1: Results on the MSMT17 dataset. #Param. indicates parameter number, m indicates million.

5.2 Experiment Setting

We select the DenseNet-161 model and ResNet-50 model both pre-trained on the ImageNet dataset as backbone model. As for state-of-the-art model in ReID tasks, we select the MGN [Wang *et al.*, 2018] model, which also use a ResNet-50 as backbone network. We modified the backbone network by using our proposed PTL method, and name these models as DenseNet-161+PTL, ResNet-50+PTL and MGN+PTL respectively. We then use the STD method to train the DenseNet-161 model (DenseNet-161+PTL+STD) by using the DenseNet-161+PTL as teacher model.

All images have been reshaped into 256x128 (height x width) before feeding into the network except for the experiments of MGN and MGN+PTL, which use image size 384x128. We take out the output of the 1x1 Conv-layer as the discriminative feature. The initial learning rate is set to 0.01 and decay the learning rate ten times every ten epochs. Models are fine-tuned for 50 epochs. Unless otherwise stated, in all of our experiments, we use SGD-M as the optimizer. The hyper-parameter λ is set to 0.8 by practicing in the following experiments. For experiments involved MGN and MGN+PTL, we followed the experiment setting in [Wang *et al.*, 2018]. Meanwhile, we use the single-query setting in all experiments.

To the best of our knowledge, the official implementation of the MGN has not been released. Hence, we use the reproduction version published in https://github.com/GNAYUOHZ/ReID-MGN. The results obtained by this reproduction code are noted as MGN (reproduced) in all the tables. Although the MGN model uses the ResNet-50 as the backbone network, the parameter number of the MGN model (66m) is much more than the ResNet-50 model (28m). Due to the GPU usage limitation, we have not conducted experiments about the MGN+PTL+STD.

We use the cross-entropy loss in all of the fine-tuning processes in our experiments, except for the MGN+PTL. The MGN use a combined loss function insists of cross-entropy loss and triplet loss, we use the same loss function in all the experiments of the MGN+PTL.

Method	mAP	CMC-1
DML [Zhang *et al.*, 2018]	70.51	89.34
HA-CNN [Li *et al.*, 2018]	75.70	91.20
PCB+RPP [Sun *et al.*, 2018]	81.60	93.80
MGN [Wang *et al.*, 2018]	86.90	95.70
DenseNet-161*	69.90	88.30
DenseNet-161	76.40	91.70
DenseNet-161+PTL	77.50	92.50
DenseNet-161+PTL+STD	77.50	92.20
MGN (reproduced)	85.80	94.60
MGN+PTL	**87.34**	**94.83**

Table 2: Results on the Market-1501 dataset. DenseNet-161* used a batch size of 90, other experiments involving DenseNet-161 used a batch size of 32. MGN (reproduced) is our reproduction of the MGN [Wang *et al.*, 2018].

5.3 MSMT17

We first evaluate the PTL method on the MSMT17 dataset. The detailed evaluation statistics has shown in Table 1. We can see a significant performance promotion by using the PTL method. The performance of the DenseNet-161+PTL+STD outperforms not only the backbone model but also all of the baseline methods. We also can notice that the DenseNet-161+PTL+STD model can achieve higher CMC-1 score than the DenseNet-161+PTL model. We attribute the success to the combined loss function of the STD method. By combining the cross-entropy Loss with the L1 loss, the student model can learn the discriminative knowledge from the teacher model while imposing restrictions on the learned knowledge.

5.4 Market-1501

We use DenseNet-161 and MGN as backbone model to evaluate the performance of the PTL method on Market-1501 dataset. We select several state-of-the-art person ReID methods as baselines. Among these methods, the DML [Zhang *et al.*, 2018] is also pre-trained on ImageNet and transferred to Market-1501.

The results has summarized in Table 2. We can notice that by using the PTL method and the STD method, a simple DenseNet-161 model can outperform the state-of-the-art transfer learning based person ReID methods on Market-1501 dataset. Meanwhile, the MGN+PTL outperforms all the state-of-the-art methods.

Moreover, we can notice that using a large batch size (DenseNet-161*) is not an effective way to narrow the gap between the distribution of each mini-batch and the distribution of the dataset. In contrast, large batch size can result in poor performance.

5.5 CUHK03

We then conduct model fine-tuning experiments on CUHK03 dataset. We compare the performance of MGN+PTL with several state-of-the-art methods. The results has summarized in Table 3. We can notice that by using the PTL method, the ReID performance of the MGN model has promoted tremendously, and outperforms all the state-of-the-art methods.

Methods	Detected		Labelled	
	mAP	CMC-1	mAP	CMC-1
HA-CNN [Li *et al.*, 2018]	38.60	41.70	41.00	44.40
PCB [Sun *et al.*, 2018]	54.20	61.30	-	-
PCB+RPP [Sun *et al.*, 2018]	57.50	63.70	-	-
MGN [Wang *et al.*, 2018]	66.00	66.80	67.40	68.00
MGN (reproduced)	69.41	71.64	72.96	74.07
MGN+PTL	**74.22**	**76.14**	**77.31**	**79.79**

Table 3: Results on the CUHK03 dataset.

Method	mAP	CMC-1
HA-CNN [Li *et al.*, 2018]	63.80	80.50
PCB [Sun *et al.*, 2018]	69.20	83.30
MGN [Wang *et al.*, 2018]	78.40	88.70
MGN (reproduced)	77.07	87.70
MGN+PTL	**79.16**	**89.36**

Table 4: Results on the DukeMTMC-reID dataset.

5.6 DukeMTMC-reID

We then conduct experiments on DukeMTMC-reID dataset. As for baselines, we compare the performance with several state-of-the-art methods, including HA-CNN, PCB and MGN. The results have shown in Table 4, we can notice that by using our method, the MGN+PTL model can outperforms all state-of-the-art methods.

5.7 Transfer among Multiple Datasets

In real-world applications, ReID model needs to transfer among a sort of datasets to take advantage of all available data. Therefore, we conduct multiple dataset transferring experiments to evaluate the performance of our proposal when dealing with model fine-tuning among multiple datasets. Similar to the experiment on MSMT17 dataset, we also use the STD method to train a DenseNet-161 model to compare with the baselines fairly.

The detailed results have shown in Table 5, from which we can see that the PTL method achieves better performance compared with baseline models. We also notice that the performance of two-step transferring achieves better performance compare with one step transferring. For instance, fine-tuning follow the order 'Duke to Market to MSMT17' outperforms 'Market-Duke to MSMT17'. We argue that it is caused by the substantial style variation in the Market-Duke dataset is richer than either Market-1501 or DukeMTMC-reID dataset.

More than that, we can see that the order of fine-tuning can influence the performance of the final model. The model fine-tuned by 'Duke to Market to MSMT17' can achieve highest score in both mAP and CMC-1.

5.8 Evaluate STD Method on MSMT17

In this subsection, we evaluate the STD method on MSMT17 dataset. We conduct a series of comparative experiments by adjusting the ratio of the cross-entropy loss and L1 loss.

Method	Fine-tuning list	mAP	CMC-1
DenseNet-161	Market-Duke to MSMT17	41.12	72.49
DenseNet-161+PTL+STD	Market-Duke to MSMT17	42.53	74.11
DenseNet-161	Market to Duke to MSMT17	41.22	72.78
DenseNet-161+PTL+STD	Market to Duke to MSMT17	42.34	73.60
DenseNet-161	Duke to Market to MSMT17	41.80	73.00
DenseNet-161+PTL+STD	Duke to Market to MSMT17	**42.73**	**74.31**
DenseNet-161+PTL	Duke to Market	76.00	91.30
DenseNet-161+PTL+STD	Duke to Market	75.50	91.10
DenseNet-161+PTL	Duke to MSMT17 to Market	**77.90**	91.60
DenseNet-161+PTL+STD	Duke to MSMT17 to Market	77.40	91.60

Table 5: Results of transfer among multiple datasets. Fine-tuning list indicates the order of fine-tuning, e.g., 'Duke to Market' means the model is fine-tuned on DukeMTMC-reID before fine-tune it on Market-1501. Market and Duke denote the Market-1501 dataset and the DukeMTMC-reID dataset respectively.

Method	#Param.	λ	mAP	CMC-1
DenseNet-161+PTL	42m	-	42.45	72.48
DenseNet-161+PTL+STD	32m	0.00	38.60	70.80
DenseNet-161+PTL+STD	32m	0.30	41.26	72.52
DenseNet-161+PTL+STD	32m	0.50	42.27	**73.49**
DenseNet-161+PTL+STD	32m	0.80	**42.51**	73.37
DenseNet-161+PTL+STD	32m	1.00	41.66	72.32

Table 6: Results of the STD method on the MSMT17 dataset. The λ denotes the hyper-parameter in Eq. 4. The λ denotes the proportion of L1 loss in the combined loss function. $\lambda = 0$ means use a SGD-M optimizer to fine-tune a DenseNet-161 model on MSMT17 without using the STD method.

We use the DenseNet-161+PTL model transferring from Market-1501 to MSMT17 in Table 5 as the teacher model. The student model is a DenseNet-161 model which has been transferred from ImageNet to Market-1501 using a SGD-M optimizer.

The detailed results are shown in Table 6. From this table, we can see that by using the STD method, the performance of the DenseNet-161 model is promoted significantly. Meanwhile, we can see that the score of the DenseNet-161+PTL+STD grows up when λ grows up. However, when λ bigger than 0.8, the score no longer increases anymore. We argue that it is because the cross-entropy loss in the combined loss function is essential.

6 Related Works

6.1 Transfer Learning Methods

Many transfer learning methods have been proposed recently. Zhong et al. [Zhong *et al.*, 2018] proposed a domain adaption approach which transfers images from one camera to the style of another camera. Fan et al. [Fan *et al.*, 2018] proposed an unsupervised fine-tuning approach which used an IDE model trained on DukeMTMC-reID as start point and fine-tuned it on target dataset. Different from these approaches, our method is based on model fine-tuning, which is more flexible and easy to conduct.

As for optimization methods used in transfer learning, training a meta-learner to learn how to update the parameters of the backbone model have attracted lots of attention recently [Ha *et al.*, 2016; Finn *et al.*, 2017]. In these approaches, parameters are updated using a learned update algorithm. For instance, Finn et al. [Finn *et al.*, 2017] proposed a meta-learning method MAML by using a LSTM network to update parameters. Our proposal is distinct from these approaches in several aspects. First, the goal of these meta-learning works is to find a better parameter optimization route which can efficiently optimize the model parameter. Differently, the PTL network is designed to mitigate the distribution difference between mini-batch and the whole dataset. Meanwhile, the BConv-Cells can be directly participating in the feature extraction.

6.2 Person Re-identification Networks

With the prosperity of deep learning, using deep learning networks as feature extractor has become a common practice in person ReID tasks. Many deep learning based person ReID methods [Varior *et al.*, 2016; Zhang *et al.*, 2017; Li *et al.*, 2014] have been proposed. As for transfer learning based deep person ReID method, Geng et al. [Geng *et al.*, 2016] proposed a deep transfer learning model to address the data sparsity problem.

6.3 Knowledge Distillation Methods

Our proposed STD method is a special case of knowledge distillation [Hinton *et al.*, 2015]. More generally, it can be seen as a special case of learning with privileged information. Using distillation for model compression is mentioned by Hinton et al. [Hinton *et al.*, 2015]. Wu et al. [Wu, 2016] used the distillation method to improve the accuracy of binary neural networks on ImageNet.

7 Conclusion

In this paper, we propose a Batch-related Convolutional Cell (BConv-Cell) to mitigate the impact of the bias of each mini-batch caused by internal variations. The BConv-Cells can progressively collect the global information of the dataset during training while participating in the feature extraction. This global information will be used to mitigate the bias of each mini-batch in the next iterations. Based on the BConv-Cells, we propose the Progressive Transfer Learning (PTL) method to fine-tune the pre-trained model into the target dataset. Extensive experiments show that our method can improve the performance of the backbone network significantly and achieved state-of-the-art performance on four datasets, including Market-1501, MSMT17, CUHK03, and DukeMTMC-reID datasets.

Acknowledgements

This work was supported in part by the National Nature Science Foundation of China (Grant Nos: 61751307), in part by the National Youth Top-notch Talent Support Program and in part by Alibaba-Zhejiang University Joint Institute of Frontier Technologies.

References

[Deng *et al.*, 2018] Weijian Deng, Liang Zheng, Qixiang Ye, Guoliang Kang, Yi Yang, and Jianbin Jiao. Image-image domain adaptation with preserved self-similarity and domain-dissimilarity for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 994–1003, 2018.

[Fan *et al.*, 2018] Hehe Fan, Liang Zheng, Chenggang Yan, and Yi Yang. Unsupervised person re-identification: Clustering and fine-tuning. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 14(4):83, 2018.

[Finn *et al.*, 2017] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org, 2017.

[Geng *et al.*, 2016] Mengyue Geng, Yaowei Wang, Tao Xiang, and Yonghong Tian. Deep transfer learning for person re-identification. *arXiv preprint arXiv:1611.05244*, 2016.

[Ha *et al.*, 2016] David Ha, Andrew M. Dai, and Quoc V. Le. Hypernetworks. *CoRR*, abs/1609.09106, 2016.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Hinton *et al.*, 2015] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[Huang *et al.*, 2017] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, volume 1, page 3, 2017.

[Keskar *et al.*, 2016] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *CoRR*, abs/1609.04836, 2016.

[Li *et al.*, 2014] Wei Li, Rui Zhao, Tong Xiao, and Xiaogang Wang. Deepreid: Deep filter pairing neural network for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 152–159, 2014.

[Li *et al.*, 2018] Wei Li, Xiatian Zhu, and Shaogang Gong. Harmonious attention network for person re-identification. In *CVPR*, volume 1, page 2, 2018.

[Ma *et al.*, 2018] Liqian Ma, Qianru Sun, Stamatios Georgoulis, Luc Van Gool, Bernt Schiele, and Mario Fritz. Disentangled person image generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 99–108, 2018.

[Russakovsky *et al.*, 2015] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[Sun *et al.*, 2018] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 480–496, 2018.

[Varior *et al.*, 2016] Rahul Rama Varior, Mrinal Haloi, and Gang Wang. Gated siamese convolutional neural network architecture for human re-identification. In *European Conference on Computer Vision*, pages 791–808. Springer, 2016.

[Wang *et al.*, 2018] Guanshuo Wang, Yufeng Yuan, Xiong Chen, Jiwei Li, and Xi Zhou. Learning discriminative features with multiple granularities for person re-identification. In *2018 ACM Multimedia Conference on Multimedia Conference*, pages 274–282. ACM, 2018.

[Wei *et al.*, 2018] Longhui Wei, Shiliang Zhang, Wen Gao, and Qi Tian. Person trasfer gan to bridge domain gap for person re-identification. In *Computer Vision and Pattern Recognition, IEEE Conference on*, 2018.

[Wu, 2016] Xundong Wu. High performance binarized neural networks trained on the imagenet classification task. *CoRR*, abs/1604.03058, 2016.

[Xingjian *et al.*, 2015] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015.

[Zhang *et al.*, 2017] Xuan Zhang, Hao Luo, Xing Fan, Weilai Xiang, Yixiao Sun, Qiqi Xiao, Wei Jiang, Chi Zhang, and Jian Sun. Alignedreid: Surpassing human-level performance in person re-identification. *arXiv preprint arXiv:1711.08184*, 2017.

[Zhang *et al.*, 2018] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4320–4328, 2018.

[Zheng *et al.*, 2015] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1116–1124, 2015.

[Zheng *et al.*, 2017] Zhedong Zheng, Liang Zheng, and Yi Yang. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[Zhong *et al.*, 2018] Zhun Zhong, Liang Zheng, Zhedong Zheng, Shaozi Li, and Yi Yang. Camera style adaptation for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5157–5166, 2018.