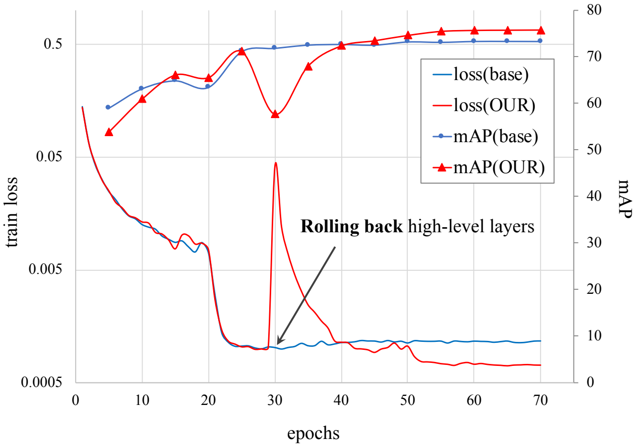
Backbone Cannot Be Trained at Once: Rolling Back to Pre-Trained Network for Person Re-Identification

Youngmin Ro,[1] Jongwon Choi,[2] Dae Ung Jo,[1] Byeongho Heo,[1] Jongin Lim,[1] Jin Young Choi[1]
{treeoflife, mardaewoon, bhheo, ljin0429, jychoi}@snu.ac.kr, jw17.choi@samsung.com
[1]Department of ECE, ASRI, Seoul National University, Korea
[2]Samsung SDS, Korea

Abstract

In person re-identification (ReID) task, because of its shortage of trainable dataset, it is common to utilize fine-tuning method using a classification network pre-trained on a large dataset. However, it is relatively difficult to sufficiently fine-tune the low-level layers of the network due to the gradient vanishing problem. In this work, we propose a novel fine-tuning strategy that allows low-level layers to be sufficiently trained by rolling back the weights of high-level layers to their initial pre-trained weights. Our strategy alleviates the problem of gradient vanishing in low-level layers and robustly trains the low-level layers to fit the ReID dataset, thereby increasing the performance of ReID tasks. The improved performance of the proposed strategy is validated via several experiments. Furthermore, without any add-ons such as pose estimation or segmentation, our strategy exhibits state-of-the-art performance using only vanilla deep convolutional neural network architecture.



Rolling back high-level layers

Figure 1: Training loss and mAP graph changed by introducing our learning strategy. ‘base’ means that the network is trained by basic strategy. In our method, the training loss escapes from local minimum and the mAP accuracy increases by utilizing the rolling-back scheme.

Introduction

Person re-identification (ReID) refers to the tasks connecting the same person, for instance, a pedestrian, among multiple people detected in non-overlapping camera views. Different camera views capture pedestrians in various poses with different backgrounds, which interferes with the ability to correctly estimate the similarity among pedestrian candidates. These obstacles makes it difficult to recognize the identities of numerous pedestrians robustly by comparing them with a limited number of person images with known identities. Furthermore, it is infeasible to obtain large training datasets sufficient to cover the appearance variation of pedestrians, making the ReID problem difficult to be solved. When sufficient training data is not available, it is a common approach to fine-tune the network pre-trained by another large dataset (*e.g.*, ImageNet) which contains abundant information. The fine-tuning approach results in better performance than the approaches in which networks are trained from randomly initialized parameters. This is a practical approach used in many research areas (Ren et al. 2015; Long, Shelhamer, and Darrell 2015) to avoid the problem of overfitting. Likewise, the previous ReID algorithms (Chang, Hospedales, and Xiang 2018; Si et al. 2018; Sun et al. 2017) have utilized the fine-tuning approach. Most of recent works in ReID research have attempted to utilize semantic information such as pose estimation (Zhao et al. 2017; Xu et al. 2018; Sarfraz et al. 2018), segmentation mask (Song et al. 2018), and semantic parsing (Kalayeh et al. 2018) to improve the accuracy of ReID by considering the additional pedestrian contexts.

In contrast to the previous studies, we are interested in incrementally improving the performance of ReID by enhancing the basic fine-tuning strategy applied to the pretrained network. A few attempts have been made to improve learning methods by the ways designing a new loss function or augmenting data in a novel way (Zhang et al. 2017; Chen et al. 2017; Zhong et al. 2017b; Sun et al. 2017). However, there has been no research on improving the learning method to consider the characteristics of each layer filter.

Before suggesting our novel fine-tuning strategy for ReID, we first empirically analyze the importance of finetuning low-level layers for ReID problems. According to related research (Zeiler and Fergus 2014; Mahendran and Vedaldi 2015), the low-level layers concentrate on details of appearance to discriminate between samples while the high-level layers contain semantic information. Thus, we need to sufficiently fine-tune the low-level layers to improve the discriminant power for the specific class ‘person’ in ReID because the low-level layers of the pre-trained network include detailed information on numerous classes. However, since the gradients delivered from high-level layers to low-level layers are reduced through back-propagation, the low-level layers suffer from a gradient-vanishing problem, which causes early convergence of the entire network before the low-level layers are trained sufficiently.

To solve this problem, we propose a novel fine-tuning strategy in which a part of the network is intentionally perturbed when learning slows down. The proposed fine-tuning strategy can recover the vanished gradients by rolling back the weights in the high-level layers to their pre-trained weights, which provides an opportunity for further tuning of weights in the low-level layers. As shown in Figure 1, the proposed fine-tuning strategy allows the network to converge to a minimum in a basin with better generalization performance than the conventional fine-tuning method. We validate the proposed method that uses no add-on schemes via a number of experiments, and the method outperforms state-of-the-art ReID methods appending additional context to the basic network architecture. Furthermore, we apply the proposed learning strategy to the fine-grained classification problem, which validates its generality for various computer vision tasks.

Related Work

Traditionally, the ReID problem has been solved by using a metric learning method (Koestinger et al. 2012) to narrow the distance among the images of the same person. Clothing provides an important hint in the ReID task, and some approaches (Pedagadi et al. 2013; Kuo, Khamis, and Shet 2013) have used color-based histograms. With the development of deep learning, many ReID methods to learn discriminative features by deep architectures appear, which dramatically increases the ReID performance (Sun et al. 2017; Li, Zhu, and Gong 2018; Hermans, Beyer, and Leibe 2017). Recently, the state-of-the-art approaches (Si et al. 2018; Song et al. 2018; Zhong et al. 2018) have also used the advanced deep architecture, especially pre-trained on ImageNet (Deng et al. 2009), as a backbone network.

Add-on semantic information method in ReID To increase the performance, many recent works based on the deep architectures have tried to consider additional semantic information such as poses of pedestrians and attention masks. One of the most popular approaches is to use the off-the-shelf pose estimation algorithms (Cao et al. 2017; Insafutdinov et al.) to tackle the misaligned poses of the candidate pedestrians. In (Su et al. 2017), using the pose information, Su *et al* aligned each part of a person, producing pose-normalized input to deal with the problem of the deformable variation of the ReID object. Sarfraz *et al.* (Sarfraz et al. 2018) proposed a view predictor network that distinguishes the front, back, and sides of a person using pose information. In addition to using the pose estimation algorithms, there was a method (Song et al. 2018) which embeds a 4-channel input by concatenating 3-channels of RGB input image and one channel of segmentation mask. Likewise, an algorithm (Kalayeh et al. 2018) uses semantic parsing masks rather than whole body mask. In (Qian et al. 2018), they generate a realistic pose-normalized image. The synthesized image can be used as training data because the label is preserved. (Xu et al. 2018) proposed attention-aware composition network. They pointed out the conventional methods using pose information based on rigid body regions such as rectangular RoI. They obtained non-rigid parts through connectivity information between the human joints and matched them individually. In contrast to the previous ReID methods, we target on improving the training method itself without any additional semantic information or extra architecture.

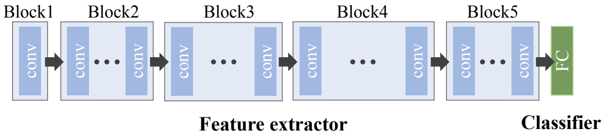


Figure 2: The description of the network: ResNet-34, ResNet-50 and ResNet-101 are utilized as a feature extractor. The classifiers are re-defined for each ReID dataset.

Advanced fine-tuning methods There are other studies to improve learning methods on pre-trained networks. Li and Hoiem (Li and Hoiem 2017) suggested a method which can learn a new task without forgetting the existing tasks in transfer learning. In (Kornblith, Shlens, and Le 2018), Kornblith *et al.* analyzed a conventional fine-tuning method, which concluded that the state-of-the-art ImageNet architecture yields state-of-the-art results over many tasks. In the ReID task, several methods have improved learning strategy on pre-trained networks. The quadruplet loss was proposed in (Chen et al. 2017). In this research, Chen *et al.* have developed an improved version of triplet losses, which does not only make the inter-class close but also add a negative sample, making the distance in the intra-class much longer. In (Zhang et al. 2017), Zhang *et al* were inspired by the distillation method (Hinton, Vinyals, and Dean 2015) between teacher and student networks and proposed a learning method based on co-student networks which can be trained without teacher network. However, there has been no research considering the fine-tuning characteristics for the ReID problem. In this paper, we propose a novel fine-tuning strategy adapted to the ReID task, which takes into account the layer-by-layer characteristic of the network.

Methodology

In this section, we first analyze the conventional fine-tuning strategy to determine which layer is insufficiently trained for ReID problems. Based on the analysis, we propose a new fine-tuning strategy that alleviates the vanishing gradient in the poorly trained layers, consequently improving the generalization performance of the fine-tuned network.

Overall framework

Before describing the empirical analysis and the proposed fine-tuning strategy, we first introduce an overall framework including a network architecture with its training and testing processes. The notations defined in this section are used in the following sections.

Architecture In this paper, we use a classification-based network (Zheng, Yang, and Hauptmann 2016) that determines the entire identity label as a class. We assume that the deep convolutional neural network consists of two components: a feature extractor and a classifier. The feature extractor is composed of multiple convolutional layers and the classifier consists of several fully-connected (FC) layers. As the feature extractor, we utilize convolutional layers of pre-trained ResNet (He et al. 2016), which are widely used in many ReID algorithms (Sun et al. 2017; Zhong et al. 2018; Qian et al. 2018). The three structures ResNet-34, ResNet-50, and ResNet-101 are used for the feature extractor to show the generality of the proposed fine-tuning strategy. According to the resolution of the convolutional layers, the feature extractor can be partitioned into five blocks where each block contains several convolutional layers of the same resolution. The five blocks of ResNet-34, ResNet-50, and ResNet-101 contain $\{1, 6, 8, 12, 6\}$, $\{1, 9, 12, 18, 9\}$, and $\{1, 9, 12, 69, 9\}$ convolutional layers, respectively. Following feature extraction, a feature vector is obtained by a global average pooling layer that averages the channel-wise values of the feature map resulting from the last convolutional layer. The resulting feature vector is a 2048-D vector for ResNet-50 and ResNet-101 and a 512-D vector for ResNet-34. The network infers the identity of the input sample by feeding the feature vector obtained from the feature extractor into the classifier. The classifier is newly defined in the order of 512-D FC layer, batch normalization, leaky-rectified linear unit, and FC layer with L-dimension, where L is the number of identities in the training set and varies between datasets. Following the last FC layer, a soft-max layer is located.

Training process We train the network to classify the identities of training samples based on cross-entropy loss. The weight parameters to be trained are denoted by $\theta \equiv \{\theta_1, ..., \theta_N, \theta_{FC}\}$, where θ_n and θ_{FC} are weight parameters of n-th block and FC layers, respectively. Given N training samples $\{\mathbf{x}_i\}_{i=1}^{N}$ with L identities and the corresponding one-hot vectors $\{\mathbf{y}_i\}_{i=1}^{N}$ where $\mathbf{y}_i \in \{0, 1\}^{L\times 1}$, the probability that $\mathbf{x}_i$ corresponds to each label is calculated as:

$$p(\mathbf{x}_i|\theta) = \mathcal{C}(\mathcal{F}(\mathbf{x}_i|\theta_1, ., \theta_N)|\theta_{FC}), \tag{1}$$

where $p(\mathbf{x}_i|\theta) \in \mathbb{R}^{L\times 1}$, $\mathcal{F}(\mathbf{x}_i|\theta_1, ., \theta_N)$ denotes feature extractor for $\mathbf{x}_i$ with $\theta_1, ., \theta_N$, and $\mathcal{C}(\,\cdot\,|\theta_{FC})$ denotes a classifier with θ_{FC}. The cross-entropy loss between the estimated $p(\mathbf{x}_i|\theta)$ and $\mathbf{y}_i$ is calculated as follows:

$$\mathcal{L}(\mathbf{x}_i, \mathbf{y}_i, \theta) = -\frac{1}{N}\sum_{i=1}^{N} \mathbf{y}_i^T \log p(\mathbf{x}_i|\theta). \tag{2}$$

In the training process, a stochastic gradient descent method is used to train θ by minimizing Eq. 2.

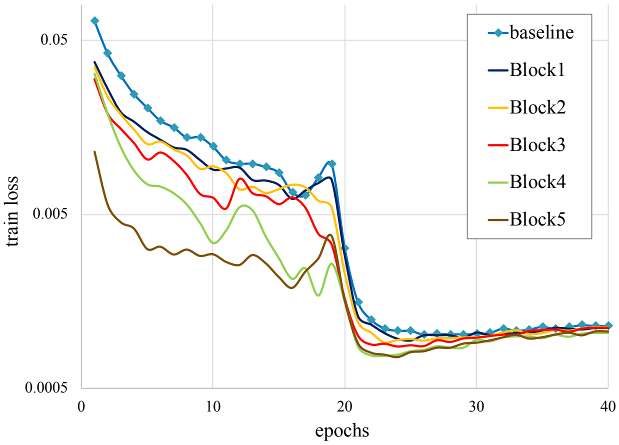


Figure 3: The training loss convergences by ordinary fine-tuning (baseline) and rolling-back schemes where block i is continuously tuned and the other blocks are rolled back to the pre-trained one.

Table 1: The generalization performance of each scheme in Figure 3. Bold numbers show the best performance.

remain layers	mAP	rank-1	rank-5	rank-10
baseline	73.16	89.43	96.35	97.77
Block1	74.08	89.49	**96.50**	97.62
Block2	**74.37**	**89.96**	**96.50**	97.62
Block3	73.87	89.90	96.20	**97.83**
Block4	73.82	89.64	95.81	97.62
Block5	71.17	88.45	95.61	97.42

Testing process The identities given to the testing set are completely different than the identities in the training set. Thus, the classifier trained in the training process cannot be used for the testing process. To find correspondence between pedestrian candidates without using the classifier, we estimate the similarity of two pedestrians based on the distance between the feature vectors of each pedestrian extracted from the trained feature extractor. To evaluate the performance, the testing set is divided into a query set and a gallery set with M_q and M_g samples, respectively. The samples of the query and gallery sets are denoted by $\{\mathbf{x}_{q,i}\}_{i=1}^{M_q}$ and $\{\mathbf{x}_{g,j}\}_{j=1}^{M_g}$, respectively. Each sample in the query set is a person of interest, which should be matched to the candidate samples in the gallery set.

The distance between $\mathbf{x}_{q,i}$ and $\mathbf{x}_{g,j}$ is calculated by L-2 norm as follows:

$$\boldsymbol{q}^{(i)} = \mathcal{F}(\mathbf{x}_{q,i}|\theta_1, ..., \theta_N) \tag{3}$$

$$\mathbf{g}^{(j)} = \mathcal{F}(\mathbf{x}_{g,j}|\theta_1, ..., \theta_N) \tag{4}$$

$$s_{i,j} = ||\boldsymbol{q}^{(i)} - \mathbf{g}^{(j)}||_2^2. \tag{5}$$

The identity of the gallery sample with the lowest distance $s_{i,j}$ is determined as the identity of the i-th query sample.

Algorithm 1 Re-fine learning

```
Parameter: N: Number of total block , M: Number of lower block
Parameter: $\theta_1^{(0)}, .., \theta_N^{(0)}$ : weights of pre-trained network
Input: $\theta_1, .., \theta_N$ , $\theta_{FC}$, $X, Y$(dataset)
$\theta_i^{(1)} = \theta_i^{(0)}, \forall\, i = 1, .., N$    ▷ Initialize weights to pre-trained one
$\theta_{FC}^{(1)} \leftarrow$ random initialization
$\hat{\theta}_1^{(1)}, .., \hat{\theta}_N^{(1)}, \hat{\theta}_{FC}^{(1)} \leftarrow$ FINE-TUNE$(X, Y, \theta_1^{(1)}, .., \theta_N^{(1)}, \theta_{FC}^{(1)})$    ▷ First fine-tune on ReID dataset X,Y
for p = 2 to M do
    $\theta_i^{(p)} = \begin{cases} \hat{\theta}_i^{(p-1)} & i < p \\ \theta_i^{(0)} & i \geq p \end{cases}$    ▷ Remain certain layers and roll back others
    $\theta_{FC}^{(p)} = \hat{\theta}_{FC}^{(p-1)}$    ▷ Do not roll back FC layers
    $\hat{\theta}_1^{(p)}, .., \hat{\theta}_N^{(p)}, \hat{\theta}_{FC}^{(p)} \leftarrow$ FINE-TUNE$(X, Y, \theta_1^{(p)}, .., \theta_N^{(p)}, \theta_{FC}^{(p)})$    ▷ Refine-tune on ReID dataset X,Y
end for
```

Analysis of fine-tuning method

This section determines which layer converges insufficiently by conventional fine-tuning. Figure 3 shows the convergence, supporting the key ideas of the proposed fine-tuning strategy. ‘baseline’ denotes the conventional fine-tuning, while ‘Block i’ indicates the refine-tuning wherein every block except ‘Block i’ is rolled back after the ‘baseline’ fine-tuning. Table 1 shows the generalization performance of each scheme. **A meaningful discovery** is that a rolling-back scheme with remaining low-level blocks (Block1, Block2, Block3) shows slower convergence than applying the rolling-back scheme to the remaining high-level blocks (Block3, Block4). However, as shown in Table 1, the scheme that maintains the low-level blocks gives better generalization performance than the scheme preserving the high-level blocks. This indicates that the ’baseline’ fine-tuning causes the low-level layers to be converged at a premature. This gives us an insight that rolling back of the high-level layers except the low-level layers might give the low-level layers an opportunity to learn further. As **additional consideration**, all the weights cannot be given in pre-trained states. This is because the output layer of a deep network for a new task is usually different from the backbone network. Hence, the FC layers must be initialized in a random manner. Rolling back the FC layers to random states does not provide any benefit. Thus, in our rolling-back scheme, FC layers are excluded from rolling back, although it is a high-level layer, to keep a consistent learning of the low-level layers.

Refine-tuning with rolling back

The aforementioned analysis shows that a premature convergence degrades performance and rolling back high-level layers can be a beneficial strategy to mitigate the premature convergence problem in the low-level layers. For further tuning of the low-level layers, we designed a rolling-back refine-tuning scheme that trains the low-level layers incrementally from the front layer along with rolling back the remaining high-level layers. The detailed rolling back scheme is described in the following.

1. In the first fine-tuning period ($p = 1$), the weights, $\theta_1, .., \theta_N$, are initialized with the pre-trained weights, $\theta_i^{(0)}$.

$$\theta_i^{(1)} = \theta_i^{(0)}, \ \ \forall i = 1, ..., N. \tag{6}$$

   The weights (θ_{FC}) in FC layer are initialized with the random scratch (He et al. 2015). Then the first period of fine-tuning is performed on the target dataset by Eq. (1), Eq. (2). The updated weight of the i-th block is denoted by $\hat{\theta_i}^{(1)}$, which is obtained by minimizing the loss from Eq. (2).

2. From the refine-tuning period with rolling back ($p \geq 2$), we roll-back the high-level layers as in the following procedure. First, Block1 (θ_1) is maintained in the state of previous period and all the remaining blocks ($\theta_2, ..., \theta_N$) are rolled back to their pre-trained states $\theta_i^{(0)}$. In other words, Block1 continues the learning, and the other blocks restart the learning from the beginning with the pre-trained initial weights. In the the incremental manner, the next low-level block is added one-by-one to the set of blocks continuing the learning, while the remaining ones are rolled back. The rolling-back refine-tuning is repeated until all layers are included in the set of blocks continuing the learning. In summary, in the p-th refine-tuning period, the weights of the network are rolled back as

$$\theta_i^{(p)} = \begin{cases} \hat{\theta_i}^{(p-1)} & i < p \\ \theta_i^{(0)} & i \geq p, \end{cases} \tag{7}$$

   where $\hat{\theta_i}^{(p-1)}, i = 1, ..., N$ are the updated weights in the $(p-1)$-th refine-tuning period. During the refine-tuning process, the (θ_{FC}) is not rolled back as mentioned above.

$$\theta_{FC}^{(p)} = \hat{\theta}_{FC}^{(p-1)}. \tag{8}$$

The detailed procedure of the refine-tuning scheme with rolling-back is summarized in Algorithm 1.

Table 2: Results of our rolling-back scheme on different ReID dataset

		ResNet-50							
	continuously	Market-1501		DukeMTMC		CUHK03-L		CUHK03-D	
Period	tuned blocks	mAP	rank-1	mAP	rank-1	mAP	rank-1	mAP	rank-1
1	none	73.16	89.43	63.26	80.83	45.17	50.07	44.05	48.00
2	B1+ FC	75.65	90.95	66.09	81.96	47.69	51.21	45.76	50.50
3	B1+B2+FC	76.54	91.12	**66.57**	**82.41**	49.98	54.36	46.20	51.36
4	B1+B2+B3+FC	**77.01**	**91.24**	66.39	82.32	**50.72**	**55.64**	**47.43**	**52.93**

Table 3: Results of our rolling-back scheme for different network types.

		ResNet-34				ResNet-101			
	continuously	Market-1501		DukeMTMC		Market-1501		DukeMTMC	
Period	tuned blocks	mAP	rank-1	mAP	rank-1	mAP	rank-1	mAP	rank-1
1	none	70.65	86.93	60.06	78.69	75.91	90.80	66.00	82.27
2	B1+ FC	73.63	89.13	63.45	81.10	77.21	90.77	69.27	83.62
3	B1+B2+FC	74.85	90.02	65.16	82.18	78.17	91.27	**70.24**	**85.19**
4	B1+B2+B3+FC	**74.97**	**90.05**	**65.44**	**83.08**	**79.95**	**92.49**	69.88	84.43

Experiment

Dataset

Market-1501 Market-1501 (Zheng et al. 2015) is widely used dataset in person ReID. Market-1501 contains 32,668 images of 1,501 identities. All the bounding box images are results of detection by the DPM detector (Felzenszwalb et al. 2010). The dataset is divided into a training set of 751 identities and a test set of 750 identities.

DukeMTMC-ReID(DukeMTMC) Based on the multi-target and multi-camera tracking dataset, DukeMTMC (Zheng, Zheng, and Yang 2017) has been specially designed for person ReID. DukeMTMC contains 36,411 images of 1,402 identities which are divided into a training set and a testing set of 702 and 702 identities, respectively.

CUHK03-np CUHK03-np (Zhong et al. 2017a) is a modified version of the original CUHK03 dataset. The hand-labeled (CUHK03-L) and DPM-detected (Felzenszwalb et al. 2010) bounding boxes (CUHK03-D) are offered. CUHK03-np contains 14,096 images of 1,467 identities. The new version is split into two balanced sets containing 767 and 700 identities for training and testing, respectively.

Implementation detail

Our method was implemented using PyTorch (Paszke et al. 2017) library. All inputs are resized to 288×144 and the batch size was set to 32. No other augmentation is used except horizontal flip in our training process. The initial learning rate was set to 0.01 and 0.1 for the feature extractor and the classifier, respectively. The learning rates were multiplied by 0.1 at every 20 epoch and we trained for 40 epochs as one refine-tuning period. In our experiment, the proposed refine-tuning strategy has been rolled back three times and four epochs have been trained for four refine-tuning periods, and so a total of 160 epochs are have been repeated for all fine-tuning. The learning rates of rolling back blocks are restored to 0.01 at the beginning of every period. In contrast, the blocks that do not roll back begin with the low learning rate of 0.001 since a high learning rate of the sufficiently trained blocks might yield sudden exploding. The optimizer used in this study was stochastic gradient descent (SGD) with nesterov momentum (Nesterov 1983). For the optimizer, the momentum rate and the weight decay were set to 0.9 and 5×10^{-4}, respectively. In every rolling back, the momentum of gradient was reset to 0. In the test process, the additional feature vector was used to add the feature vector of the horizontal flipped input pairwise. We report rank-1 accuracy of Cumulative Matching Characteristics (CMC) curve and the mean Average Precision (mAP) for performance evaluation.

Ablation tests

The network trained with the proposed strategy was verified via ablation tests on Market-1501, DukeMTMC, CUHK03-L and CUHK03-D. The proposed refine-tuning strategy is applied to a network over four periods. As the refine-tuning periods progress, the continuously tuned blocks are cumulative (e.g., B1+B2+FC in the third period). The other blocks are rolled back to their original pre-trained states. As shown in Table 2, the performance increases as the refine-tuning periods progress with the exception of DukeMTMC in the fourth period. However, even in this case, the gap was negligible. The improvement is most prominent in the second refine-tuning period during which the first rolling back is performed. To verify the generality of our refine-tuning scheme, we conducted additional experiments with other networks including ResNet-34 and ResNet-101 (He et al. 2016) under the same settings. Table 3 shows the performance of each network in Market-1501 and DukeMTMC. The proposed refine-tuning scheme also showed a consistent improvement in ResNet-34 and ResNet-101. The abla-

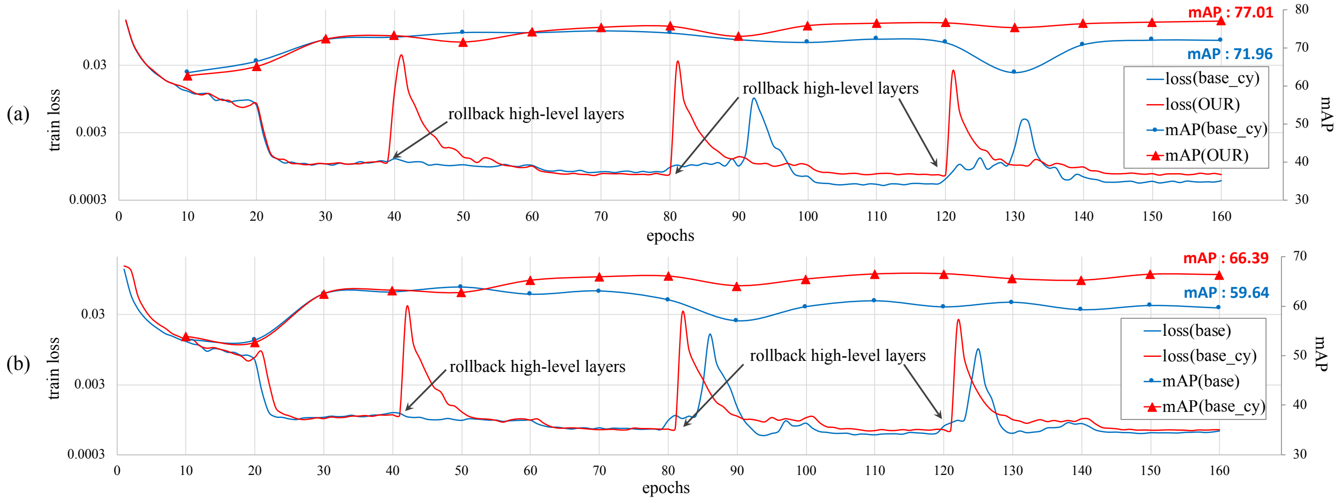


Figure 4: The train loss and mAP graph for comparison of our rolling-back scheme and the conventional fine-tuning at once. (a) is the results on Market-1501 and (b) is the results on DukeMTMC.

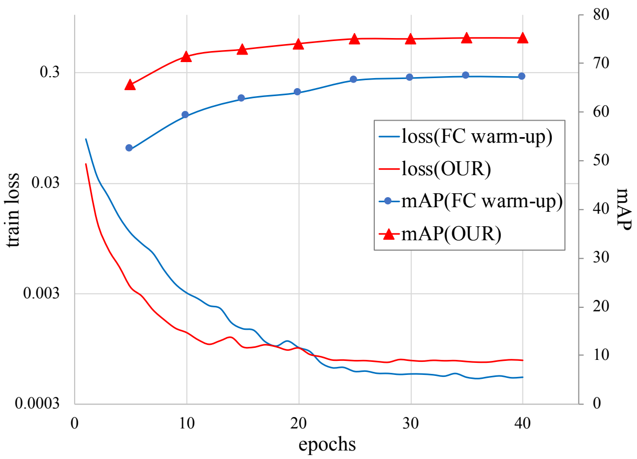


Figure 5: The results of comparison with FC warm-up training method

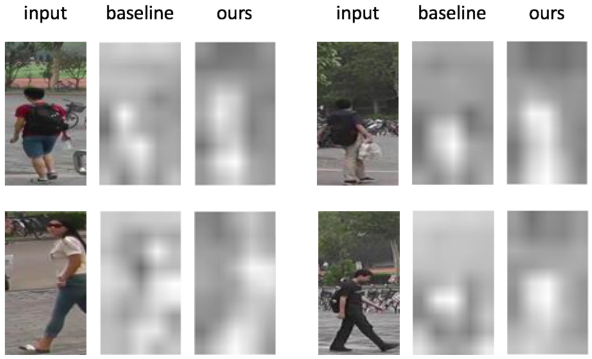


Figure 6: The attention maps formed by the last feature layer trained by our rolling-back scheme and the baseline

tion test results demonstrate that the proposed refine tuning scheme has a significant advantage as a general method to enhance the generalization performance in the ReID problem in which only a limited amount of data is available.

Effect of rolling back as a perturbation

To evaluate the effect of our rolling-back scheme, it is compared with 'base_cy' method that does roll back none of the block but merely adjusts the learning rate with the same timing as ours for a perturbation driving to other local basins. The 'base_cy' is similar to other studies (Loshchilov and Hutter 2016; Smith 2017) that perturb only the learning rate. Figure 4 shows the change in training loss and mAP of the whole processes of the proposed refine-tuning and the base_cy fine-tuning. After the first rolling-back at 40 epochs, the training loss from the rolling-back scheme converges to a value that is better than the value of the base_cy in the 70-80 epochs. After the second and third rolling-backs, the training loss of the base_cy converges to a lower value than that of the proposed method, but the base_cy shows a worse generalization performance (mAP) than the proposed method.

Comparison to FC warm-up training

In this section, we discuss the difference between our method and FC warm-up training (He et al. 2016). As mentioned previously, the new FC layers start randomly from scratch. FC warm-up is a way to freeze the pre-trained weights in all hidden layers except for the FC layers and train the FC layers before starting the main fine-tuning. In the comparison experiment, the baseline was warmed up for 20 epochs. In our proposed method, period 1 (see Table 2) is similar to FC warm-up where FC layers start from random scratch. However, the proposed method does not freeze the pre-trained weights in period 1. The training loss and mAP for FC warm-up and our methods are depicted in Figure 5. FC warm-up and our methods start fine/refine-tuning

Table 4: Comparison with State-of-the-art methods on Market-1501, DukeMTMC and CUHK03-L/D

		Market-1501		DukeMTMC		CUHK03-L		CUHK03-D		
Method	Backbone	mAP	rank-1	mAP	rank-1	mAP	rank-1	mAP	rank-1	Add-on
PT-GAN (2018)	ResNet-50	58.0	79.8	48.1	68.6	30.5	33.8	28.2	30.1	pose+GAN
SVDNet (2017)	ResNet-50	62.1	82.3	56.8	76.7	37.8	40.9	37.3	41.5	-
PDC (2017)	Inception	63.4	84.1	-	-	-	-	-	-	pose
AACN (2018)	GoogleNet	66.9	85.9	59.3	76.8	-	-	-	-	pose
HAP2S_P (2018)	ResNet-50	69.4	84.6	60.6	75.9	-	-	-	-	-
PSE (2018)	ResNet	69.0	87.7	62.0	79.8	-	-	-	-	pose
CamStyle (2018)	ResNet-50	71.6	89.2	57.6	78.3	-	-	-	-	GAN
PN-GAN (2018)	ResNet-50	72.6	89.4	53.2	73.6	-	-	-	-	pose+GAN
MGCAM (2018)	MSCAN	74.3	83.8	-	-	50.2	50.1	46.7	46.9	mask
MLFN (2018)	Original	74.3	90.0	62.8	81.0	49.2	54.7	47.8	52.8	-
HA-CNN (2018)	Inception	75.7	91.2	63.8	80.5	41.0	44.4	38.6	41.7	-
DuATM (2018)	DenseNet-121	76.6	91.4	64.6	81.8	-	-	-	-	-
Ours	ResNet-34	75.0	90.1	65.4	83.1	48.6	53.0	45.6	51.3	-
Ours	ResNet-50	77.0	91.2	66.6	82.4	50.7	55.6	47.4	52.9	-
Ours	ResNet-101	**79.9**	**92.5**	**70.2**	**85.2**	**55.7**	**59.8**	**50.5**	**55.6**	-

Table 5: Comparison with State-of-the-art methods using same backbone network ResNet-50

	Market-1501		DukeMTMC		
Method	mAP	rank-1	mAP	rank-1	Add-on
PT-GAN	58.0	79.8	48.1	68.6	GAN
SVDNet	62.1	82.3	56.8	76.7	-
HAP2S_P	69.4	84.6	60.6	75.9	-
CamStyle	71.6	89.2	57.6	78.3	GAN
PN-GAN	72.6	89.4	53.2	73.6	GAN
Ours	**77.0**	**91.2**	**66.6**	**82.4**	-

Table 6: Results our rolling-back scheme on fine-grained dataset

	CUB-2011	FGVC Aircraft	Food-101
baseline	74.59	83.89	80.21
Ours	**79.12**	**86.80**	**81.89**

Table 7: Result our rolling-back scheme for Inception V3 on Market-1501

	mAP	rank-1
baseline	70.97	87.86
Ours	**73.04**	**89.40**

after training the FC layers. The FC warm-up converges to a lower training loss than the proposed method, but the proposed method shows better performance in terms of generalization.

Attention performance of our refine-tuning method

To learn discriminative features for the ReID task, it is important to distinguish the foreground from the background. Figure 6 shows that our method can generate a more distinguishable feature map in the last convolutional layer than the baseline of the conventional fine-tuning method.

Comparisons with state-of-the-art methods

We also compared the proposed method with state-of-the-art methods. Table 5 shows the comparison results when using ResNet-50. The proposed rolling-back refine-tuning scheme shows the best performance even though our method does not use any add-on scheme. Furthermore, compared to other methods without add-on scheme (SVDNet, HAP2S_P), our method outperforms them by more than 7% mAP improvement for Market-1501. Table 4 summarizes the results compared with the state-of-the-art methods on Market-1501, DukeMCMT, and CUHK03-L/D. According to the results, the rolling-back refine-tuning scheme makes a meaningful contribution to the enhancement of any backbone networks so that it outperforms state-of-the-art algorithms utilizing add-on schemes.

Generality of rolling back scheme

We conducted additional experiments to show the generality of our method. Our method is effective for the problems that require detailed features for discrimination. To verify this, several experiments have been conducted for the fine-grained classification datasets such as CUB-200-2011 (Wah et al. 2011), FGVC-Aircraft (Maji et al. 2013), and food-101 (Bossard, Guillaumin, and Van Gool). As represented in Table 6, our method improves the performance against the baseline for all the datasets.

Additionally, to show that our method can be used in general networks, we conducted an experiment based on Inception V3 network (Szegedy et al. 2016) on Market-1501 dataset. We defined the convolutional layers {`Conv_`, `Mixed_5, Mixed_6, Mixed_7`} in Inception V3 as {B1,B2,B3,B4}. B1, B2 were selected for the low-level layers in the experiment. As shown in Table 7, our method improves more than mAP: 2% from the baseline using Inception V3 network.

Conclusion

In this paper, we proposed a refine tuning method with a rolling-back scheme which further enhances the backbone network. The key idea of the rolling-back scheme is to restore the weights in a part of the backbone network to the pre-trained weights when the fine-tuning converges at a premature state. To escape from the premature state, we adopt an incremental refine tuning strategy by applying the fine tuning repeatedly, along with the rolling-back. According to the experimental results, the rolling-back scheme makes a meaningful contribution to enhancement of the backbone network where it derives the convergence to a local basin of a good generalization performance. As a result, our method without any add-on scheme could outperform the state-of-the-arts with help of add-on scheme.

Acknowledgement

This work was supported by Next-Generation ICD Program through NRF funded by Ministry of S&ICT [2017M3C4A7077582], ICT R&D program MSIP/IITP [2017-0-00306, Outdoor Surveillance Robots].

References

Bossard, L.; Guillaumin, M.; and Van Gool, L. Food-101–mining discriminative components with random forests. In *ECCV, pages=446–461, year=2014, organization=Springer*.

Cao, Z.; Simon, T.; Wei, S.-E.; and Sheikh, Y. 2017. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*.

Chang, X.; Hospedales, T. M.; and Xiang, T. 2018. Multi-level factorisation net for person re-identification. In *CVPR*.

Chen, W.; Chen, X.; Zhang, J.; and Huang, K. 2017. Beyond triplet loss: a deep quadruplet network for person re-identification. In *CVPR*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*. Ieee.

Felzenszwalb, P. F.; Girshick, R. B.; McAllester, D.; and Ramanan, D. 2010. Object detection with discriminatively trained part-based models. *IEEE Trans. on PAMI*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Hermans, A.; Beyer, L.; and Leibe, B. 2017. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Insafutdinov, E.; Pishchulin, L.; Andres, B.; Andriluka, M.; and Schiele, B. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *ECCV*.

Kalayeh, M. M.; Basaran, E.; Gokmen, M.; Kamasak, M. E.; and Shah, M. 2018. Human semantic parsing for person re-identification. In *CVPR*.

Koestinger, M.; Hirzer, M.; Wohlhart, P.; Roth, P. M.; and Bischof, H. 2012. Large scale metric learning from equivalence constraints. In *CVPR*.

Kornblith, S.; Shlens, J.; and Le, Q. V. 2018. Do better imagenet models transfer better? *arXiv preprint arXiv:1805.08974*.

Kuo, C.-H.; Khamis, S.; and Shet, V. 2013. Person re-identification using semantic color names and rankboost. In *WACV*.

Li, Z., and Hoiem, D. 2017. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Li, W.; Zhu, X.; and Gong, S. 2018. Harmonious attention network for person re-identification. In *CVPR*.

Liu, J.; Ni, B.; Yan, Y.; Zhou, P.; Cheng, S.; and Hu, J. 2018. Pose transferrable person re-identification. In *CVPR*.

Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. In *CVPR*.

Loshchilov, I., and Hutter, F. 2016. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.

Mahendran, A., and Vedaldi, A. 2015. Understanding deep image representations by inverting them. In *CVPR*.

Maji, S.; Rahtu, E.; Kannala, J.; Blaschko, M.; and Vedaldi, A. 2013. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*.

Nesterov, Y. 1983. A method for unconstrained convex minimization problem with the rate of convergence o (1/kˆ 2). In *Doklady AN USSR*.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in pytorch.

Pedagadi, S.; Orwell, J.; Velastin, S.; and Boghossian, B. 2013. Local fisher discriminant analysis for pedestrian re-identification. In *CVPR*.

Qian, X.; Fu, Y.; Wang, W.; Xiang, T.; Wu, Y.; Jiang, Y.-G.; and Xue, X. 2018. Pose-normalized image generation for person re-identification. In *ECCV*.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*.

Sarfraz, M. S.; Schumann, A.; Eberle, A.; and Stiefelhagen, R. 2018. A pose-sensitive embedding for person re-identification with expanded cross neighborhood re-ranking. In *CVPR*.

Si, J.; Zhang, H.; Li, C.-G.; Kuen, J.; Kong, X.; Kot, A. C.; and Wang, G. 2018. Dual attention matching network for context-aware feature sequence based person re-identification. *arXiv preprint arXiv:1803.09937*.

Smith, L. N. 2017. Cyclical learning rates for training neural networks. In *WACV*.

Song, C.; Huang, Y.; Ouyang, W.; and Wang, L. 2018. Mask-guided contrastive attention model for person re-identification. In *CVPR*.

Su, C.; Li, J.; Zhang, S.; Xing, J.; Gao, W.; and Tian, Q. 2017. Pose-driven deep convolutional model for person re-identification. In *ICCV*.

Sun, Y.; Zheng, L.; Deng, W.; and Wang, S. 2017. Svdnet for pedestrian retrieval. In *ICCV*.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2818–2826.

Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The caltech-ucsd birds-200-2011 dataset.

Xu, J.; Zhao, R.; Zhu, F.; Wang, H.; and Ouyang, W. 2018. Attention-aware compositional network for person re-identification. *arXiv preprint arXiv:1805.03344*.

Yu, R.; Dou, Z.; Bai, S.; Zhang, Z.; Xu, Y.; and Bai, X. 2018. Hard-aware point-to-set deep metric for person re-identification. In *ECCV*.

Zeiler, M. D., and Fergus, R. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*.

Zhang, Y.; Xiang, T.; Hospedales, T. M.; and Lu, H. 2017. Deep mutual learning. *arXiv preprint arXiv:1706.00384*.

Zhao, L.; Li, X.; Zhuang, Y.; and Wang, J. 2017. Deeply-learned part-aligned representations for person re-identification. In *ICCV*.

Zheng, L.; Shen, L.; Tian, L.; Wang, S.; Wang, J.; and Tian, Q. 2015. Scalable person re-identification: A benchmark. In *Computer Vision, IEEE International Conference on*.

Zheng, L.; Yang, Y.; and Hauptmann, A. G. 2016. Person re-identification: Past, present and future. *arXiv preprint arXiv:1610.02984*.

Zheng, Z.; Zheng, L.; and Yang, Y. 2017. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In *ICCV*.

Zhong, Z.; Zheng, L.; Cao, D.; and Li, S. 2017a. Re-ranking person re-identification with k-reciprocal encoding. In *CVPR*.

Zhong, Z.; Zheng, L.; Kang, G.; Li, S.; and Yang, Y. 2017b. Random erasing data augmentation. *arXiv preprint arXiv:1708.04896*.

Zhong, Z.; Zheng, L.; Zheng, Z.; Li, S.; and Yang, Y. 2018. Camera style adaptation for person re-identification. In *CVPR*.