Learning More in Vehicle Re-Identification: Joint Local Blur Transformation and Adversarial Network Optimization

Yanbing Chen 1,2, Wei Ke 1,*, Hao Sheng 3,4 and Zhang Xiong 3

1 Faculty of Applied Sciences, Macao Polytechnic University, Macao 999078, China; yanbing.chen@mpu.edu.mo
2 School of Computer Science and Technology, Zhejiang Sci-Tech University, Hangzhou 310018, China
3 State Key Laboratory of Virtual Reality Technology and Systems, School of Computer Science and Engineering, Beihang University, Beijing 100191, China; shenghao@buaa.edu.cn (H.S.);
xiongz@buaa.edu.cn (Z.X.)
4 Beihang Hangzhou Innovation Institute Yuhang, Xixi Octagon City, Yuhang District, Hangzhou 310023, China
* Correspondence: wke@mpu.edu.mo

Abstract: Vehicle re-identification (ReID) tasks are an important part of smart cities and are widely used in public security. It is extremely challenging because vehicles with different identities are generated from a uniform pipeline and cannot be distinguished based only on the subtle differences in their characteristics. To enhance the network’s ability to handle the diversity of samples in order to adapt to the changing external environment, we propose a novel data augmentation method to improve its performance. Our deep learning framework mainly consists of a local blur transformation and a transformation adversarial module. In particular, we first use a random selection algorithm to find a local region of interest in an image sample. Then, the parameter generator network, a lightweight convolutional neural network, is responsible for generating four weights and then as a basis to form a filter matrix for further blur transformations. Finally, an adversarial module is employed to ensure that as much noise information as possible is added to the image sample while preserving the structure of the training datasets. Furthermore, by updating the parameter generator network, the adversarial module can help produce more appropriate and harder training samples and lead to improving the framework’s performance. Extensive experiments on datasets, i.e., VeRi-776, VehicleID, and VERI-Wild, show that our method is superior to the state-of-the-art methods.

Keywords: vehicle re-identification; local blur transformation; filter matrix; adversarial module; convolutional neural network

1. Introduction

Vehicle ReID [1] attempts to locate the identity of a specific vehicle in a huge network of cameras quickly. It has been used in a variety of contexts. For starters, vehicle ReID [2] can help police fight crime. Moreover, it can aid city planners in gaining a better understanding of traffic flow. With a strong application background, vehicle ReID is gaining traction in computer vision tasks. Over the last decade, deep learning has become popular and a major method used for computer vision. The research community has been driven to create CNN-based [3] approaches for vehicle ReID tasks.

In order to train a model with good performance throughout the training process, neural networks typically require quite massive amount of data. Data collection and annotation, on the other hand, are quite costly. Moreover, a shortage of data is an obvious stumbling point when building a strong deep neural network. It frequently results in overfitting or poor generalization. Data limitation leads to performance reduction in the field of vehicle ReID as well. Data augmentation [4] is a good method for obtaining extra training samples without having to collect and annotate more data. Common augmentation methods typically generate more training samples by data warping. Moreover, data warping
includes random erasing [5], adversarial training [6], geometric and color transformations, and neural style transfer [7].

Unlike previous traditional methods [8], we propose a novel data augmentation strategy that increases the complexity rather than the number of samples in a dataset. In particular, our method blends traditional augmentation method with deep learning technology. A general framework has been attempted in our previous work [9] to exercise this idea. To make the framework more efficient, we adopt the architecture and reorder some of the modules. We introduce a filter matrix to blur the local region of an image sample in order to increase the difficulty of network learning. First, we use a local region selection and a light weight neural network to learn the filter matrix for further data augmentation. Here, we define this light weight neural network as the parameter generator network (PGN). The special purpose of this step is to retain essential information while increasing the difficulty of identification. Second, an adversarial module is employed in our framework. This module includes a recognizer and a parameter generator, as previously mentioned. The recognizer is intended for identifying the vehicles in datasets that belong to the same identity. The parameter generator (PGN), on the other hand, generates augmented image samples. However, the identities of these augmented images are not necessarily the same as the identities of the original images. Thus, the recognizer and PGN form an adversarial module. The two modules compete with each other and strike a balance between the consistency of vehicle identities and the variation of transformed samples. The results of augmented images are depicted in Figure 1. Extensive experiments prove that our framework has excellent generalizability.

For validation, three datasets, i.e., VeRi-776 [10], VehicleID [11], and VERI-Wild [12], are subjected to a series of asymptotic ablation experiments. Our experiments show that the framework proposed greatly surpass the baseline and other previous methods.

Our contributions are summed up as follows:

1. A data augmentation method that combines traditional data augmentation and deep learning technology;
2. A deep learning framework that combines region location selection, local blur transformation, and an adversarial framework;
3. Rather than adding more samples, the dataset is expanded by making the data samples harder. The original structure of the dataset is retained;
4. The proposed framework optimizes both the data augmentation and the recognition model without any fine-tuning. The augmented samples are created by dynamic learning.

The rest of this paper is organized as follows. Section 2 presents related work with respect to our research. Our method is then elaborated in Section 3, including some algorithm details and framework descriptions. Section 4 involves the experiments and addresses some qualitative analysis. At the end, we provide some conclusions in Section 5.

2. Related Work

2.1. Vehicle Re-Identification (ReID)

Vehicle ReID [13,14] has extensive research and important applications in computer vision. With the rapid development of deep learning techniques, using neural networks has become mainstream for vehicle ReID. Vehicle ReID [15] requires robust and discriminative image representation. Liu et al. [10] proposed a fusion of multiple features, e.g., colors, textures, and deep learned semantic features. Furthermore, Liu et al. proposed a large-scale benchmark VeRi-776 and improved the performance of their previous model FACT with a Siamese network for license plate recognition and spatio-temporal properties [10].

Deep joint discriminative learning (DJDL) for extracting exclusivity features was proposed by Li et al. [16]. They demonstrated a pipeline that mapped vehicle images into a vector space using deep relative distance learning (DRDL), and the similarity of the two vehicles could be referred to the distance. Wang et al. [17] extracted orientation information from 20 key-point locations of vehicles and presented an orientation-invariant feature-embedding module. Wei et al. [18] introduced a recurrent neural network-based hierarchical attention (RNN-HA) network for vehicle ReID, which incorporated a large number of attributes.

Bai et al. [19] suggested a group sensitive triplet embedding strategy to model interclass differences, which Bai et al. [19] found to be effective. He et al. [20] recently investigated both local and global representations to offer a valid learning framework for vehicle re-ID; however, their method is labor-intensive because it is dependent on the labeled components.

Although the previously mentioned vehicle ReID methods differ in some ways, they all need a large number of image samples. However, acquiring data is time-consuming and labor-intensive. Moreover, data augmentation is a very good way to solve this problem.

2.2. Generative Adversarial Networks (GAN)

Goodfellow et al. [21] have made great achievements in image generation, and there have been many applications [22–25]. GAN [26] is a deep learning model that has emerged as one of the most promising approaches for unsupervised learning on complex distributions. The original GAN [27] was proposed with a deconvolutional network for generating images from noise and a convolutional network for discriminating real or fake samples. To produce a reasonably good output, the model learns by playing each other through two modules of the framework: the Generative and the Discriminative.

In domains with a lack of image samples, GAN is a potent data augmentation tool [28]. It can help create artificial instances from datasets. Bowles et al. [29] discussed it as a method for extracting extra data from datasets. Denoising convolutional neural networks (DnCNN) [30] and image restoration convolutional neural networks (IRCNN) [31] are two CNN-based approaches. By employing the CNN-based end-to-end transformation, these CNN-based approaches considerably improve the performance of image denoising compared to model-based optimization methods.
However, the images generated by GAN [21] are determined by a random vector, and the results cannot be controlled. To overcome this problem, a conditional version named conditional generative adversarial nets (CGAN) [32] was proposed. CGAN conditions not only the generator but also the discriminator by introducing external cues. Inspired by GAN ideas, we add an adversarial module to the overall framework to effectively generate the augmented samples.

2.3. Data Augmentation

Data augmentation [4] is often used to help avoid overfitting [33]. In vehicle ReID tasks, problems such as view angles, illumination, overlapping shadows, complex backgrounds, and image scaling must be overcome. Nevertheless, there are not many good solutions for addressing data augmentation. It is widely assumed that larger datasets produce better training models. However, due to the manual efforts involved in collecting and labeling data, assembling massive datasets can be a scary task.

In many areas of image research, general augmentation methods such as flipping, rotation, scaling, and perspective transformation [4] are often effective. In the vehicle ReID task, we found that only using normal data augmentation is not sufficient. These methods are not flexible enough and do not fully satisfy the effect of dynamic optimization. Peng et al. [34] augmented samples jointly using adversarial learning and pre-training processes. Ho et al. [35] developed flexible augmentation policy schedules to expedite the search process. Cubuk et al. [36] used reinforcement learning to search the policy for augmentation.

Our new method becomes more advantageous, because it integrates the previously mentioned traditional methods and our deep learning module.

3. Methodology

We propose an adversarial strategy and local blur transformation to make more efficient augmented samples. In this section, the overall structure of the framework (Section 3.1) is described at first. Then, we introduce the two main modules separately—Local Blur Transformation (Section 3.2) and Transformation Adversarial Module (Section 3.3).

3.1. Overall Framework

The suggested framework contains two components, as shown in Figure 2: local blur transformation (LBT) and transformation adversarial module (TAM).

A sample, denoted as $x$, is input to framework. Firstly, LBT takes input image $x$ and selects one local region. Then, it produces four transformed augmented images $x_1, \ldots, x_4$, each with a different filter matrix. Finally, given images $x_1, \ldots, x_4$, TAM chooses the most difficult vehicle among them as the replacement. The most difficult vehicle image has the largest distance from input image $x$.

TAM consists of PGN and the recognizer. When augmented images $x_1, \ldots, x_4$ enter the recognizer, the recognizer checks if these image identities still equal to image sample $x$. The image will be discarded if it no longer belongs to the original identity.

Note that PGN is employed both in LBT and TAM. PGN is used in LBT to supply the parameters of the filter matrix needed for the blur transformation. Meanwhile, in TAM, PGN creates the most difficult augmented sample.

At the end, the most difficult sample selected by TAM replaces the original image as a training sample; then, we proceed to the next iteration.
Figure 2. Overview of the proposed framework. It consists of two major modules: local blur transformation (red part) and transformation adversarial module (green part). Transformation adversarial module is the main idea of the system. In detail, the parameter generator network learns from the selected filter matrix transformation, and the recognizer learns more to judge the augmented image sample. Both are continuously updating themselves to lift network performances. Note that parameter generator is employed both in local blur transformations and the adversarial module.

3.2. Local Blur Transformation (LBT)

LBT determines a local blur region and then generates four augmented samples. This includes three steps: local-region selection (LRS), the parameter generator network (PGN), and blur transformation. LRS produces a rectangle region for blur transformation, and then PGN outputs the weight values for the filter matrix. Finally, we obtain the augmented samples by local blur transformations.

3.2.1. Local-Region Selection (LRS)

LRS is employed to select an interest range from input sample. Algorithm 1 specifically describes the process of selecting a local rectangular region. The ratio of the selected area is initialized between $A_1$ and $A_2$. A reasonable aspect ratio is set up from $R_1$ to $R_2$ to create a selected area that is more square. Specifically, the area ratio, donated as $A_t$, is set up at a range from $A_1 \leq A_t \leq A_2$, and the aspect ratio as $R_t$ satisfies $R_1 \leq R_t \leq R_2$. With $A_t$ and $R_t$, the width $W_t$ and height $H_t$ of the selected region are calculated. Moreover, the top-left corner of the selected region is also determined by random position $P_{x,y}$ such that the region is fully located inside the image. Thus, the local region is finally represented by $(P_{x,y}, W_t, H_t)$. 
Algorithm 1 Process of Local-Region Selection

**Input:** image width $W$; image height $H$; area of image $A$; ratio of width and height $R$; area ratio ranges from $A_1$ to $A_2$; aspect ratio ranges from $R_1$ to $R_2$.

**Output:** selected rectangle region $(P_x, P_y, W_t, H_t)$.

1: repeat
2: $A_t \leftarrow \text{rand}(A_1, A_2) \times A$
3: $R_t \leftarrow \text{rand}(R_1, R_2)$
4: $H_t \leftarrow \sqrt{A_t \times R_t}$
5: $W_t \leftarrow \sqrt{A_t / R_t}$
6: $P_x \leftarrow \text{rand}(0, W)$
7: $P_y \leftarrow \text{rand}(0, H)$
8: until $P_x + W_t \leq W \land P_y + H_t \leq H$
9: return rectangle region $(P_x, P_y, W_t, H_t)$

3.2.2. Parameter Generator Network (PGN)

Once the local region is determined, it enters the PGN. Then, PGN generates the weight of a filter matrix for blur transformation. The parameters, four transformation weights $W_1, \ldots, W_4$, form the following $3 \times 3$ filter matrix.

$$ S_1 = \begin{pmatrix} 0 & W_1 & 0 \\ W_2 & 0 & W_4 \\ 0 & W_3 & 0 \end{pmatrix}. \quad (1) $$

The detailed structure of PGN is shown in Table 1. It contains convolutional, pooling, and fully connected layers. In particular, MP donates a max pooling layer, and BN stands for batch normalization. Note that the kernel size, stride, and padding size are, respectively, 3, 1, and 1. At the end, FC layer outputs four parameters, which contribute to another filter matrix.

<table>
<thead>
<tr>
<th>Description</th>
<th>Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>$1 \times h \times w$</td>
</tr>
<tr>
<td>Con16, Relu, MP</td>
<td>$16 \times 16 \times 50$</td>
</tr>
<tr>
<td>Con64, Relu, MP</td>
<td>$64 \times 8 \times 25$</td>
</tr>
<tr>
<td>Con128, BN, Relu</td>
<td>$128 \times 8 \times 25$</td>
</tr>
<tr>
<td>Con128, Relu, MP</td>
<td>$128 \times 4 \times 12$</td>
</tr>
<tr>
<td>Con64, BN, Relu</td>
<td>$64 \times 4 \times 12$</td>
</tr>
<tr>
<td>Con16, BN, Relu, MP</td>
<td>$16 \times 2 \times 6$</td>
</tr>
<tr>
<td>FLayer</td>
<td>4</td>
</tr>
</tbody>
</table>

As shown in Figure 3, we used a filter matrix with the original sample to synthesize the augmented blur image; see Section 3.2.3 for more details. To produce the four augmented samples, the other three filter matrices $S_2, S_3,$ and $S_4$ must be created. By rotating the positions of the weight parameters clockwise, we obtain the following filter matrices.

$$ S_2 = \begin{pmatrix} 0 & W_2 & 0 \\ W_3 & 0 & W_1 \\ 0 & W_4 & 0 \end{pmatrix}, S_3 = \begin{pmatrix} 0 & W_3 & 0 \\ W_4 & 0 & W_2 \\ 0 & W_1 & 0 \end{pmatrix}, S_4 = \begin{pmatrix} 0 & W_4 & 0 \\ W_1 & 0 & W_3 \\ 0 & W_2 & 0 \end{pmatrix}. \quad (2) $$

Finally, we utilize the four filter matrices $S_1, \ldots, S_4$ with original sample $x$ to transform four augmented samples $x_1, \ldots, x_4$. The four augmented samples are made by LBT. Our
innovation lies in the generation of transformation parameters by using a neural network, which greatly improves the efficiency of the framework.

![Figure 3. Overview of blur transformation.](image)

### 3.2.3. Blur Transformation

When the filter matrices $S_1, \ldots, S_4$ are generated, blur transformation is employed to change the original sample $x$ to four augmented samples $x_1, \ldots, x_4$. Blur transformation transforms the original image to a new augmented image, as shown in Figure 3.

We obtain the filter matrix by using the four weights obtained by PGN, placing them into the corresponding positions of a $3 \times 3$ filter matrix and filling the remaining positions of the matrix with 0. Then, we obtain each blurred pixel by making a dot product of this matrix and the $3 \times 3$ window anchored at the pixel of the original local image. As a result, we obtain the local blur image. In order to compensate for the extra margin of the local blur image, we add padding to the outer edge of the original local image and fill it all the positions with 0, as shown in Figure 4. After padding, the result will be a blurred and uniformly sized output image according to the matrix’s sliding calculation.

![Figure 4. Overview of padding added to the original local image.](image)

In summary, the blur transformation calculation formula is carried out specifically as follows. Suppose the original image is $H \times W$; then, the pixel value matrix of the original image is $o$, the pixel value filter matrix is $f$, and the transformed pixel value is $t$. We then obtain $t$ from $o$ and $f$ by Equation (3), where $0 \leq x \leq W$ and $0 \leq y \leq H$.

$$t[x, y] = \sum_{l=0}^{2} \sum_{k=0}^{2} o[x+k, y+l] \times f[k, l].$$

(3)
In particular, for the filtered structure, there may be negative values or values greater than 255. For this case, we truncate them directly between 0 and 255. For negative ones, the absolute value is taken to ensure that the result of the operation is positive.

\[ t[x, y] \leftarrow |t[x, y]| \mod 256. \]  

(4)

3.3. Transformation Adversarial Module (TAM)

The goal of TAM is to be able to generate augmented images that are hard enough while keeping the sample’s identity constant. If a transformed image loses its vehicle identity, we thought that it had incorporated too much noise and discarded it from the augmented dataset. The experimental results showed that this form of filtering was quite effective.

TAM is made up of three components: PGN, the recognizer, and learning target selection. Due to the adversarial module, PGN will dynamically optimize itself. The structure of TAM is shown in the green part of Figure 2. Before explaining the individual components of TAM, we elaborate the algorithm of TAM in Algorithm 2.

The algorithm contains PGN and the recognizer. PGN outputs the parameter of the filter matrix for further augmentation. The recognizer ensures that the identity of image is unchanged. The roles of PGN and the recognizer are competitive: one is to make the sample as deformed as possible, and the other is to ensure that the identity is lost after excessive augmentation.

**Algorithm 2** Adversarial process of PGN and the recognizer

| Input: | selected rectangle region \( A \), original image \( x \) |
| Output: | updated PGN' |
| 1: | parameter \( W_1, W_2, W_3, W_4 \) is obtained from PGN: \( W_1, W_2, W_3, W_4 = \text{PGN}(A) \) |
| 2: | form a 3 \( \times \) 3 filter matrix \( S_1 \) by these four parameters \( W_1, W_2, W_3, W_4 \), then follow the clockwise and fill in the four weight \( W_1, W_2, W_3, W_4 \) in the corresponding positions to generate the other three filter matrix \( S_2, S_3, S_4 \) |
| 3: | finally, obtain augmented images \( x_1, x_2, x_3, x_4 \) based on \( S_1, S_2, S_3, S_4 \) |
| 4: | recognize image: \( \text{Reg} \leftarrow \text{Recognizer}(x) \) |
| 5: | for \( 1 \leq i \leq 4 \) do |
| 6: | \( x_i \leftarrow \text{BT}(x, S_i) \) |
| 7: | recognize image: \( \text{Reg}_i \leftarrow \text{Recognizer}(x_i) \) |
| 8: | if \( \text{ID}[\text{Reg}_i] = \text{ID}[\text{Reg}] \) then |
| 9: | \( \Delta_i \leftarrow \) measure difficulty by distance between \( \text{Reg}_i \) and \( \text{Reg} \) |
| 10: | end if |
| 11: | end for |
| 12: | \( i \leftarrow \arg \max \{ \Delta_i \mid \Delta_i \text{ is defined, } 1 \leq i \leq 4 \} \) |
| 13: | if \( i \) is defined then |
| 14: | PGN' = update PGN with \( S_i \) |
| 15: | else |
| 16: | PGN' = PGN |
| 17: | end if |

3.3.1. Recognizer

The recognizer is intended for retaining the sample’s identity. It calculates the likelihood that augmented samples belong to the same identity by comparing their vector-space distances. The greater the probability, the more likely it is that they are members of the same vehicle. The recognizer is reconstructed on the basis of ResNet50 [37]. ResNet50 is a pre-trained classification network on the ImageNet [38] dataset. It is a proven good feature extraction network. ImageNet [39], containing over 20,000 different categories, is a very good dataset in the field of computer visual research. After pre-training ResNet50 on ImageNet, we then slightly modified the network’s structure, resulting in our recognizer.
Since we compare the spatial distance as a metric, the recognizer is not affected by any form of ID losses. Shi et al. [40], for example, used the CTC loss [41] for convolutional recurrent neural networks, and attentional decoders [42,43] are guided by the cross-entropy loss. As a result, our framework is amenable to various recognizers. In the following experiments, we demonstrate the overall framework’s adaptability.

Specifically, we used the recognizer to combine the augmented samples with the original sample to determine if they have the same classification identification. If the identification of an augmented sample is different, it will be abandoned immediately. Otherwise, they are saved for further steps.

3.3.2. Learning Target Selection

The most difficult augmented sample will be kept with the help of learning target selection. Meanwhile, the selected filter matrix will update the PGN parameters in turn.

As shown in Figure 5, the distances are calculated between images $x_1, \ldots, x_4$ and the original sample, $x$. The most difficult augmented image with the largest feature distance will be selected at the end. Using this strategy, we choose the augmented one with the largest distance and optimized PGN with the corresponding parameters. As a result, we select the augmented image sample $x'$ to replace the original $x$.

Based on the distance, we choose the largest filter matrix from $S_1, \ldots, S_4$ and form vector $\vec{S}$ in turn. Note that the PGN is a neural network; thus, the role of the loss is crucial. The loss is described in Equation (5), and $\vec{S}_1$ represents the predicted value, while $\vec{S}'$ represents the actual value. Moreover, $\vec{\alpha}$ is a hyperparameter that controls the loss in a flexible manner.

$$\text{Loss} \leftarrow f(\vec{S}', \vec{S}_1, \vec{\alpha}) \tag{5}$$

![Figure 5](image.png)

**Figure 5.** Overview of feature distances. The augmented images $x_1, x_2, x_3$, and $x_4$ compare the distances with the original image $x$.

4. Experiment and Explanation

In this section, we conduct a series of experiments in order to demonstrate the performance of the proposed methods.

4.1. Datasets

In order to explore the vehicle ReID problem, a number of datasets have been introduced in the last few years. A dataset should include enough data so that a vehicle ReID model can learn intra-class variations. It should also include a large amount of annotated data collected from a large network of cameras.
4.1.1. VeRi-776

These dataset vehicles were all taken in a natural state. Each image of a vehicle is recorded from numerous angles with different illuminations and resolutions, resulting in a dataset of 50,000 photographs of 776 automobiles. Furthermore, vehicle license plates and spatial temporal relationships are marked for all vehicle tracks. The dataset is commonly utilized in the V-reID problem due to its high recurrence rate and the vast number of vehicle photos recorded with varied features.

4.1.2. VehicleID

VehicleID has 221,763 photos of 26,267 automobiles, with the majority of the photographs being front and rear views. With a total of 250 manufacturer models, each photograph includes extensive information about the vehicle model, as well as vehicle identity and camera number labeling information. In addition, the vehicle’s model information is marked on 90,000 photos of 10,319 automobiles.

4.1.3. VERI-Wild

VERI-Wild was collected in suburban areas and contains 416,314 images of 40,671 vehicles captured by 174 traffic cameras. VERI-Wild vehicles are captured without excessive restrictions and in a much richer scenario where the vehicles are located, and the vehicle images are captured over a long time span with significant changes in lighting and weather. The training set consists of 277,797 images (30,671 vehicles) and the test set consists of 138,517 images (10,000 vehicles).

4.2. Implementation

We resize the image samples to $320 \times 320$ at the beginning. To verify the framework performance, we use the same settings as baseline [44]. As the backbone feature network, ResNet50 [45] is used. As shown in Figure 6, soft margin triplet loss [46] occurs training. Moreover, the SGD [47] is adopted as the optimizer. In addition, the initial learning rate begins at $10^{-2}$ and gradually decreases to $10^{-3}$ after the first ten epochs. At the 40th and 70th epoch, it decays to $10^{-3}$ and $10^{-4}$, respectively. The model runs in a total of 120 epochs at the end.

![Figure 6. Overview of baseline model.](image)

As a pre-processing module, our framework is added to the of the baseline [44] to perform our experiment. The baseline framework is detail explained in [48,49].

4.3. Ablation Study

We conducted extensive comparative experiments to verify the contribution of the different modules of our framework. As mentioned before, this overall framework consists of LBT and TAM. The two modules, LBT and TAM, can be split further. In other words, our framework can consist of only LBT and also can comprise all modules. Different modules can be composed for ablation experiments.
We then run experiments with two different combinations: LBT and LBT + TAM. The ablation experiments results are shown in Table 2. Figure 7 shows the augmentation samples results.

Table 2. Results on the three datasets (%).

<table>
<thead>
<tr>
<th>Model</th>
<th>VeRi-776</th>
<th>VehicleID</th>
<th>VERI-Wild</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R1</td>
<td>MAP</td>
<td>Small</td>
</tr>
<tr>
<td>baseline</td>
<td>95.7</td>
<td>76.6</td>
<td>83.0</td>
</tr>
<tr>
<td>LBT (Our)</td>
<td>96.7</td>
<td>80.5</td>
<td>84.3</td>
</tr>
<tr>
<td>LBT + TAM (Our)</td>
<td>96.9</td>
<td>81.6</td>
<td>84.3</td>
</tr>
</tbody>
</table>

Figure 7. Overview of Augmentation Results. **Column 1:** original samples. **Column 2:** augmented samples created by LBT. **Column 3:** augmented samples created by LBT + TAM.

4.3.1. Our Model vs. Baseline

We compare the performance on each of the three well-known public datasets. In addition to VeRi-776, the other two dataset, as we know, comprise small, medium, and large datasets. Both in VeRi-776 and other datasets, as shown in Table 2, its performance is superior. Specifically, even if only LBT modules are available, R1 and MAP exceed the baseline by 1.0% and 3.9%, respectively, on VeRi-776. Moreover, we have performed experiments on VehicleID in three groups. The R1 increases 1.3% on the smallVehicleID, 2.7% on the mediumVehicleID, and 0.7% on the largeVehicleID. MAP increased 0.8% on the smallVehicleID, 2.9% on the mediumVehicleID, and 0.4% on the largeVehicleID, respectively.

Even though the performance of those baselines was already excellent before adding TAM, our method is still superior. If we focus more on the average metric, it can be found that LBT and LBT + TAM, respectively, improve the MAP by 3.9% and 5.0% on VeRi-776, 0.8% and 0.9% on smallVehicleID, 2.8% and 3.0% on mediumVehicleID, and 0.4% and 0.6% on largeVehicleID. Note that the MAP has significant enhancements on mediumVehicleID.

As shown in Table 2, LBT and LBT + TAM improved distinctly on VERI-Wild. Note that only when all modules are combined together can the framework's performance be maximized.
4.3.2. Internal Comparison of Our Model

We know that our modules are flexible and stackable, so let us compare the performance change of our framework when only using the LBT module and when all modules are used. Results shows that MAP improves by 1.1% on VeRi-776 and 0.1%, 0.1%, and 0.2% on VehicleID. Table 2 demonstrates a significant improvement in VERI-Wild. As we can observe, the LBT module is critical to our augmentation framework. However, after adding the TAM module, it works better and renders our framework more complete and automated. For some datasets, the improvement was less obvious, but for VERI-Wild, the improvement was significant.

4.4. Comparison with the Sota

In this part, we compare results against some state-of-the-art methods. As shown in Tables 3–5, our method is also significantly superior compared to other methods. The performance results convincingly validate the effectiveness of our approach and further show that the individual modules in the framework are capable of incremental performance improvements. It is worth mentioning that, on VeRi-776, it exceeds the second-best method in MAP accuracy by 2.2%, and it also performs well on the other datasets. Our methods are also very advantageous compared to the best methods currently available.

Table 3. Comparison with the Sota on VeRi-776 (%).

<table>
<thead>
<tr>
<th>Methods</th>
<th>MAP</th>
<th>R1</th>
<th>R5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Siamese-CNN [50]</td>
<td>54.2</td>
<td>79.3</td>
<td>88.9</td>
</tr>
<tr>
<td>FDA-Net [12]</td>
<td>55.5</td>
<td>84.3</td>
<td>92.4</td>
</tr>
<tr>
<td>Siamese-CNN+ST [90]</td>
<td>58.3</td>
<td>83.5</td>
<td>90.0</td>
</tr>
<tr>
<td>PROVID [51]</td>
<td>53.4</td>
<td>81.6</td>
<td>95.1</td>
</tr>
<tr>
<td>AAVER [52]</td>
<td>66.4</td>
<td>90.2</td>
<td>94.3</td>
</tr>
<tr>
<td>BS [53]</td>
<td>67.6</td>
<td>90.2</td>
<td>96.4</td>
</tr>
<tr>
<td>CCA [54]</td>
<td>68.0</td>
<td>91.7</td>
<td>94.3</td>
</tr>
<tr>
<td>PRN [20]</td>
<td>70.2</td>
<td>92.2</td>
<td>97.9</td>
</tr>
<tr>
<td>AGNET [55]</td>
<td>71.6</td>
<td>95.6</td>
<td>96.6</td>
</tr>
<tr>
<td>PAMTRI [56]</td>
<td>71.8</td>
<td>92.9</td>
<td>97.0</td>
</tr>
<tr>
<td>VehicleX [57]</td>
<td>72.3</td>
<td>95.0</td>
<td>98.0</td>
</tr>
<tr>
<td>APAN [57]</td>
<td>73.5</td>
<td>93.3</td>
<td></td>
</tr>
<tr>
<td>MDL [44]</td>
<td>79.4</td>
<td>90.7</td>
<td></td>
</tr>
</tbody>
</table>

Our       | **81.6** | 96.9 | 99.0 |

Table 4. Comparison of the Sota on VehicleID (%).

<table>
<thead>
<tr>
<th>Methods</th>
<th>Small</th>
<th>Medium</th>
<th>Large</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R1</td>
<td>R5</td>
<td>R1</td>
</tr>
<tr>
<td>VAMI [58]</td>
<td>63.1</td>
<td>83.3</td>
<td>52.9</td>
</tr>
<tr>
<td>FDA-Net [12]</td>
<td>-</td>
<td>-</td>
<td>59.8</td>
</tr>
<tr>
<td>AGNET [55]</td>
<td>71.2</td>
<td>83.8</td>
<td>69.2</td>
</tr>
<tr>
<td>AAVER [52]</td>
<td>74.7</td>
<td>93.8</td>
<td>68.6</td>
</tr>
<tr>
<td>OIFE [17]</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CCA [54]</td>
<td>75.5</td>
<td>91.1</td>
<td>73.6</td>
</tr>
<tr>
<td>PRN [20]</td>
<td>78.4</td>
<td>92.3</td>
<td>75.0</td>
</tr>
<tr>
<td>BS [53]</td>
<td>78.8</td>
<td>96.2</td>
<td>73.4</td>
</tr>
<tr>
<td>VehicleX [57]</td>
<td>79.8</td>
<td>93.2</td>
<td>76.7</td>
</tr>
</tbody>
</table>

Our       | **84.3** | 93.5 | **83.5** | 90.6 | **79.9** | **86.5** |
Table 5. Comparison of the Sota on VERI-Wild (%).

<table>
<thead>
<tr>
<th>Methods</th>
<th>Small</th>
<th>Medium</th>
<th>Large</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAP</td>
<td>R1</td>
<td>MAP</td>
</tr>
<tr>
<td>GoogleNet [59]</td>
<td>24.3</td>
<td>57.2</td>
<td>24.2</td>
</tr>
<tr>
<td>FDA-net [12]</td>
<td>35.1</td>
<td>64.0</td>
<td>29.8</td>
</tr>
<tr>
<td>MLSL [60]</td>
<td>46.3</td>
<td>86.0</td>
<td>42.4</td>
</tr>
<tr>
<td>AAVER [52]</td>
<td>62.2</td>
<td>75.8</td>
<td>53.7</td>
</tr>
<tr>
<td>BS [53]</td>
<td>70.0</td>
<td>84.2</td>
<td>62.8</td>
</tr>
<tr>
<td>UMTS [61]</td>
<td>72.7</td>
<td>84.5</td>
<td>-</td>
</tr>
<tr>
<td><strong>Our</strong></td>
<td><strong>72.9</strong></td>
<td><strong>93.4</strong></td>
<td><strong>66.7</strong></td>
</tr>
</tbody>
</table>

4.5. Visualization of our Method

Figure 8 shows the process of local blur transformation clearly. We visualize the augmented results of orig. To show our method more clearly, we have created a few figures. On the dataset related to our experiments, four temporary augmented images were generated during the learning process, which we also show in Figure 9. As we can observe, the left column comprise the original samples, and the right column comprise the augmented images.

![Figure 8. Workflow of local blur transformation.](image)

From the results of these visualizations, it is clear that although parts of the images have been blurred, most of the features of these vehicles have been retained. Finally, we automatically selected the most suitable image to replace the original one, and the network’s performance is gradually optimized.
5. Conclusions

In this paper, we propose a novel method for augmenting sample data in vehicle ReID. A local blur transformation and an adversarial module were employed to ensure that as much noise information as possible was added to the image sample while preserving the structure of the training datasets. We increase the complexity rather than the number of samples. Because of this advantage, our method can be used as a pre-processing layer in other deep learning systems, broadening its application potential.

Unlike the previous framework, we target the local regions of the images and use convolutional operations to blur them to increase the difficulty of network, thus improving the performance of the network. In future work, we will further improve the framework by adding an attention mechanism to identify local regions more purposefully and make the model more efficient.

Based on the tradeoff between algorithm performance and resource overhead cost, the weight of the filter matrix for blur transformation has only four non-zero values. In subsequent research studies, we will consider more effective methods for further optimizing the balance, and we will perform more experiments for verification. We also have plans to perform more experiments in order to combine more baselines in order to verify the generality of our framework.

Author Contributions: Conceptualization, Y.C.; methodology, Y.C., W.K., H.S. and Z.X.; writing original draft preparation, Y.C.; writing review and editing, Y.C., W.K., H.S. and Z.X.; Funding acquisition, W.K., H.S. and Z.X. All authors have read and agreed to the published version of the manuscript.

Funding: This study is partially supported by the National Key R&D Program of China (No.2018YFB2100800), the National Natural Science Foundation of China (No.61872025), Macao Polytechnic University (Research Project RP/ESCA-03/2020), and the Open Fund of the State Key Laboratory of Software Development Environment (No. SKLSDE-2021ZX-03).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.
Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References


