

Deep Feature Fusion with Multiple Granularity for Vehicle Re-identification

Peixiang Huang*, Runhui Huang*, Jianjie Huang, Rushi Yangchen, Zongyao He,
Xiyi Li†, Junzhou Chen†

School of Intelligent Systems Engineering, Sun Yat-sen University, Guangzhou, China

{huangpx3, huangrh9, huangjj67, yangchrsh, hezy28}@mail2.sysu.edu.cn,

{stslxy, chenjunzhou}@mail.sysu.edu.cn

Abstract

Vehicle re-identification (*Re-Id*) plays a significant role in modern life. We found that Vehicle Re-Id and Person Re-Id are two very similar tasks in the field of Re-Id. To some extent, the Person Re-Id Networks can be transplanted to the Vehicle Re-Id tasks.

In this paper, a Deep Feature Fusion with Multiple Granularity (DFFMG) method for Vehicle Re-Id is proposed for integrating discriminative information with various granularity. DFFMG is based on the Multiple Granularity Network (MGN), the state-of-the-art method from Person Re-Id. We pondered on the discrimination between Vehicle Re-Id and Person Re-Id. And we carefully designed DFFMG: a multi-branch deep network architecture which consists of one branch for global feature representations, two for vertical local feature representations and other two horizontal ones. Besides, several re-ranking methods were tested in our experiments and achieved higher scores. This network is adopted to train and test on the 2019 NVIDIA AI City Dataset [16].

1. Introduction

With the development of technology, more and more kinds of vehicles appear in our lives. There is a growing demand for Vehicle Re-Id. To be specific, Vehicle Re-Id aims to spot a vehicle from multiple non-overlapping cameras. The Person Re-Id task is similar to Vehicle Re-Id, so the design of Vehicle Re-Id network structure takes it as reference. Based on the perfect performance of various neural network model used in the Person Re-Id tasks, many researchers take the available designed model for the baseline in preparing the initial studies, such as FACT [4], NuFACT [5], DRDL [6], *etc.* But persons are different from vehicles. For the most time, one vehicle captured in different cameras has dramatically varied visual appearances.



Figure 1. Classical vehicle samples from the 2019 NVIDIA AI City Dataset. Each row denotes the same vehicle captured by different cameras from different viewpoints.

What’s more, two different vehicles of both the same color and type have a similar appearance from the same viewpoint, as shown in Fig. 1. Thus, the accuracy in Vehicle Re-Id falls short and designing a network model combining efficiency with strong robustness for Vehicle Re-Id is still a problem. Then it drives researchers to improve the model structure.

To fulfill the task, the classical method utilized some well-known deep feature learning architectures to extract features of vehicles [21]. For instance, AlexNet [1], GoogleNet [2] and VGGNet [3] are respectively employed as extractors for FACT, NuFACT and DRDL. Our model uses the backbone network ResNet50 and adds different distinct Conv layers after the backbone. Like other neural network tasks, it requires us to train it based on the provided data at first. When training the model, we attempted to assemble different loss functions and optimizers. Then we used the saved model parameters to extract features. Finally, different distance functions were employed to calculate the distance. According to the result, we sorted distance to find the top-100 most similar images. Besides, we found the features in a series of the same type even can be different from each other. Hence, to address this problem, the deep feature fusion method is proposed

* This first two authors contributed equally to this paper.

† Corresponding author: Xiyi Li and Junzhou Chen.

in our model to increase the interclass variance and improve the robustness of the model. The deep feature fusion method uses the feature map extracted by the backbone and put it into the Branch Part. Branch Part is employed to get the global feature and local feature. It consists five branches, one branch for global feature representations, two branches for vertical local feature representations and two branches for horizontal local feature representations. Fusing various directions and sizes of features supports us with multiple granularity information to classify accurately.

What's more, we employed different original re-ranking methods to improve our performance and it worked indeed.

In conclusion, our work is innovative. First, we used both global feature and part feature fusion, partitioned vehicle images along with two directions, *i.e.*, vertically and horizontally and integrated discriminative information with various granularity. It combines advantages of the state-of-the-art methods for Person Re-Id to fit Vehicle Re-Id.

Second, we employed several original re-ranking strategies for the retrieval part of the Re-Id evaluation. It's our creative work and has brought about a significant improvement in the mAP.

Through experiments, it has shown that the proposed approaches can improve Vehicle Re-Id accuracy significantly.

The rest of this paper is organized as follows. Part 2 introduces the related work. Part 3 describes the proposed Deep Feature Fusion with Multiple Granularity for Vehicle Re-Id and what we followed and introduces re-ranking strategies what we took for experiments. Part 4 briefly analyses the 2019 NVIDIA AI City Dataset, introduces our implementation details, presents the experimental results to validate the superiority of the proposed method by comparative analysis, and identifies both strengths and weaknesses of the DFFMG. Part 5 makes a summary, then prospects for the future and concludes this paper.

2. Related Work

By searching for papers about the object identification, we found that most previous researches were mainly for either human face or person, which has achieved fruitful research results in computer vision field [6]. Though the face and the person identification have been mature and have popular topics, there is no excellent achievement in Vehicle Re-Id. The existing Vehicle Re-Id mainly focused on three aspects: feature extraction, similarity metric and re-ranking.

Feature extraction. Many Re-Id algorithms mainly focused on fine-grained classification model for feature extraction instead of identifying the objects with the same or different identities.

For Person Re-Id, Ahmed *et al.* [7] designed a pairwise verification CNN model, which took a pair of cropped images as input and employed a binary verification loss function for training. Zhao *et al.* [8] came up with a method that makes use of mid-level features from automatically discovered hierarchical patch cluster trees for view invariant and discriminative feature extraction. Ding *et al.* [9] and Cheng *et al.* [10] trained CNN with triplet samples. They minimized feature distances between the same person and maximized the distances between different people.

For Vehicle Re-Id, Dominik *et al.* [11] utilized 3D bounding boxes for rectifying car images and then concatenated color histogram features of pairs of vehicle images. A binary linear SVM is trained to verify whether a couple of images have the same identity or not. And Zhu *et al.* [12] proposed quadruple directional deep learning network to extract quadruple directional deep learning features of vehicle images to perform better. Their ideas inspired us when we designed our network.

Similarity Metric. Liu *et al.* [4] proposed FACT, which is based on feature fusion for Vehicle Re-Id tasks, that applies the Euclidean or Cosine distance between a vehicle feature pair which is extracted by the designed network to measure the similarity. S. Chopra *et al.* [13] presented a method for training a similarity metric from data to learn a function that maps input patterns into target space. It can be used for recognition or verification applications where either the number of categories is very large or very small.

In particular, Liu *et al.* [5] propose the NuFACT which cannot only learn the discriminative representation of vehicle appearance from different viewpoints but also reduce the feature redundancy to guarantee. It adopts a Null Foley-Sammon Transform (NFST) based metric learning approach for fusion of multi-level features.

Re-ranking. As we known, re-ranking also proves to be useful for improving object retrieval accuracy. Li *et al.* [14] develop a re-ranking model by analyzing the relevant information and direct information of near neighbors of each pair of images.

A popular re-ranking approach is re-ranking with the k -reciprocal encoding that Zhong *et al.* [15] proposed. A k -reciprocal feature is calculated by encoding its k -reciprocal nearest neighbors into a single vector, which is used for re-ranking under the Jaccard distance. Then it combines the original distance with the Jaccard distance as the final distance to be ranked. Our work is similar to the idea, and we added some additional functions to make better performance according to the dataset.

3. Deep Feature Fusion with Multiple Granularity Network

In this section, we describe several frameworks we used and the modifications we made to get richer features:

(1) We drew on the latest researches about person re-identification (*i.e.* PCB and MGN) that perform the state-of-the-art results in several Person Re-Id data sets.

(2) We compared Vehicle Re-Id with Person Re-Id and combined the hallmark of these two networks we mentioned above, proposed the DFFMG fit for the Vehicle Re-Id.

3.1. PCB: A Part-based Convolutional Baseline

PCB [17] employs the ResNet50 as the backbone. The change of the backbone network is the structure. Before the original global average pooling (GAP) layer, the structure is maintained. But the GAP and the following after GAP are eliminated. And the output \mathbf{T} of the remained backbone network is divided into p horizontal stripes, each stripe undergoes average pooling then generates p local feature maps g . Afterwards, each g employs 1×1 Conv to reduce the dimension and generate corresponding 256-dimension vectors h . Finally, after each h passes through a fully-connected (FC) layer, it will be sent into a classifier separately and a softmax layer to predict the identity of the input. It also follows the idea of SSD and R-FCN of object detection to remove the down-sampling of the backbone network to enlarge the tensor \mathbf{T} .

After the partition, the content within each part should be consistency. But the uniform partition of PCB will cause many outliers exit in each part. These outliers are closer to the content in other part, implying within-part inconsistency. In order to relocate these outliers and eliminate inconsistency phenomenon, PCB proposed the refined part pooling.

3.2. RPP: Refined Part Pooling

RPP is proposed in [17]. Uniform partition for PCB is simple and effective. But it was proved to cause the inconsistency phenomenon, either. Column vectors f in the same part of tensor \mathbf{T} should be similar to each other and is dissimilar to other column vectors that represent for other parts, otherwise the phenomenon of within-part inconsistency occurs. In order to eliminate this inconsistency phenomenon and relocate the outliers on the correct part, RPP applies the re-sampling with the similarity S between f and g , *i.e.*, the average-pooled column vector of each stripe. RPP uses a linear layer followed by a classification S which is a softmax

activation. And each S calculates the similarity between f and P_i :

$$S(f \leftrightarrow P_i) = \text{softmax}(W_i^T f) = \frac{\exp(W_i^T f)}{\sum_{j=0}^p \exp(W_j^T f)} \quad (1)$$

where $S(f \leftrightarrow P_i)$ means the similarity between f and P_i , W_i is the trainable weight matrix of the classification.

$$P_i = \{S(f \leftrightarrow P_i) \times f, \forall f \in F\} \quad (2)$$

where F is the complete set of column vectors in tensor \mathbf{T} . The “ $\{\}$ ” denotes re-sampling operation.

In the Eq. (2), a column vector f in \mathbf{T} and the predicted probability of f for P_i are given. And Eq. (2) assigns f to part P_i with S as the confidence and sampling weight. Meanwhile, each part P_i is sampled from all column vectors f with S . After the Eq. (2), the previous S is updated and we keep doing Eq. (2) until convergence.

By doing this, RPP achieves a “soft” and adaptive partition instead of the original “hard” and uniform partition. Additionally, RPP replaces the average pooling from the original network.

3.3. MGN: Multiple Granularity Network

MGN [18] also employs the ResNet50 as the backbone network and modifies the Conv4_1. As a result of Conv4, it will develop three branches. The structures of these branches are similar, but the down-sampling rates are different.

The first branch is the global branch which employs down-sampling with a stride-2 Conv in the Conv5_1 block. Then the global max pooling is used to generate a 2048-dimension feature vector on the feature map that comes from the last operation. A 1×1 Conv is attached to Conv5 for reducing the feature vector’s dimension to 256. The second and the third branch are going to learn the representations of the local feature. To maintain the proper receptive field for local feature, these two branches do not employ the down-sampling. Through dividing the feature map into N uniform stripes in the horizontal direction, these N stripes respectively receive the global max pooling and 1×1 convolution layer to generate the corresponding local feature.

The bigger N means the greater the granularity is. The middle branch, $N=2$, can be understood as dividing the picture into two parts; the following branch, $N=3$, can be understood as dividing the picture into upper, middle and lower parts. Finally, the global feature vectors from the first branch and the local feature vectors from the other two branches will concatenate into a 2048-dimension feature vector for similarity searching.

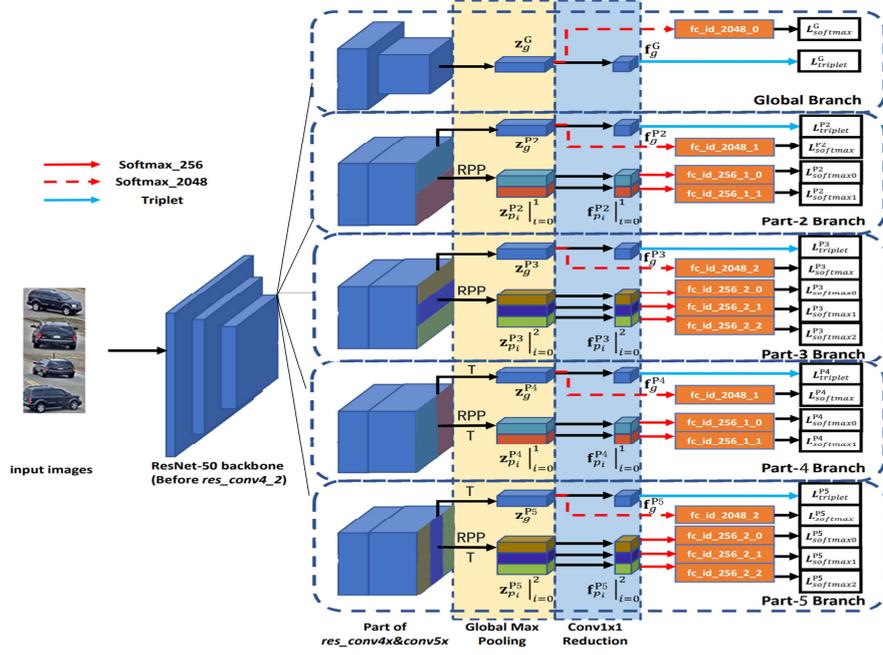


Figure 3. The DFFMG Network Structure. T presents the transpose operation.

3.3. Triplet Loss

In terms of re-identification, triplet network-based architectures are widely used because of promising results. It can learn an embedding space in where features of same vehicles are pulled closer and features of different vehicles are pushed far away. The basic theory realization of triplet loss [6, 20] is intuitively shown in Fig. 2.



Figure 2. Triplet Loss learning process.

To explain it in detail by mathematical formulas, Marin-Reyes *et al.* [20] suppose that, the input images are a batch of triplet units $\{x_A, x_P, x_N\}$. They represent the anchor sample, the positive sample whose identity is the same as the anchor one, and the negative sample whose identity is different from the anchor one. Following [6, 20], let the triplet units satisfy the following constraint Eq. (3) to get ideal features:

$$\|f(x_A) - f(x_P)\| + \varepsilon \leq \|f(x_A) - f(x_N)\| \quad (3)$$

It can also be expressed as Eq. (4):

$$\|f(x_A) - f(x_P)\|^2 + \varepsilon \leq \|f(x_A) - f(x_N)\|^2 \quad (4)$$

Among them, $f(x)$ is used to denote a mapping function for feature extracting. And ε is a positive constant parameter we predefined presenting the minimum margin between matched and mismatched pairs. By deep learning, it can gradually satisfy Eq. (4) to get better results.

Thus, the triple loss function what we should minimize when training can be defined as Eq. (5):

$$L = \sum_{i=0}^n \max\{0, \|f(x_A) - f(x_P)\|^2 + \varepsilon - \|f(x_A) - f(x_N)\|^2\} \quad (5)$$

The idea of triplet loss will improve the network performance to some extent for re-identification tasks.

3.4. The DFFMG Network Structure

Before we proposed our designed network, DFFMG, we compared and analyzed PCB and MGN. They are kind of similar in some parts and they all put the network to focus on the part feature representation. The following makes a comparative analysis of these two networks and lists their similarities and differences.

Similarities.

(1) They all focus on extracting the part feature and they all uniformly divided the tensor into several parts to get the local element.

(2) They all employ the softmax function for these part feature vectors.

(3) They all remove the down-sampling when extracting the local feature.

Differences.

(1) PCB only extracts part feature, but MGN not only does the things similar to PCB but also has two branches to acquire the local feature vectors. Moreover, MGN has one more branch for global feature and all branches consider the global feature. MGN combines these three branches to get more features.

(2) PCB proposes RPP for the cursory uniform part feature and eliminate within-part inconsistency. However, MGN still exits inconsistency phenomenon.

(3) MGN employs the triplet loss for global feature. In the meantime, the local feature vectors operate the softmax function.

According to the analysis above, we propose DFFMG which combines the merits of PCB and MGN.

First of all, we compare the Vehicle Re-Id and Person Re-Id. Vehicles are more extensive than persons in spatial size and the spatial scale. It means the features of vehicles are richer than that of persons. The multiple-branches local feature extraction is more needed in vehicles Re-Id. In the meantime, the motorcycle type is diverse which is not like human only have one outline. Vehicle Re-Id needs a global feature for helping the classification. Thus, we keep MGN as the backbone network. Based on MGN, we add more two branches for getting the vertical feature. After the dissection of the output tensor of the Conv4_2, we employ RPP for the excellent discrimination of image's local feature.

Loss functions we use in DFFMG are softmax loss and triplet loss. The softmax loss is employed after each local feature vectors, and the triplet loss is employed after each global feature vectors, which are the same as MGN's triplet loss.

The overall network structure diagram is shown in Fig. 3 which borrows from MGN.

3.5. Re-ranking

Following the previous research [15, 22, 23, 24], we adopt the method of calculating the distance between query image feature and gallery image feature. The original distance is aggregated with Jaccard distance to revise the initial ranking list.

Let d_o denote the original distance and d_j denotes the Jaccard distance. After that, the original distance between

two vehicles, one from the query dataset (q) and the other from the gallery dataset (g_i), can be measured by Mahalanobis distance as Eq. (6) shows:

$$d_o = (x_q - x_{g_i})^T M (x_q - x_{g_i}) \quad (6)$$

where x_q and x_{g_i} represent the feature of query q and gallery g_i , respectively, and M is a positive semidefinite matrix.

We could have sort the original distance to get a list of top 100 images which are most similar to the query image. But it will be closer to what we expect when we combine with Jaccard distance. Jaccard distance can be calculated by the Jaccard metric of their k -reciprocal sets as Eq. (7) shows:

$$d_j = 1 - \frac{|R^*(q,k) \cap R^*(g_i,k)|}{|R^*(q,k) \cup R^*(g_i,k)|} \quad (7)$$

where $|\cdot|$ denotes the number of candidates in the set, and R is defined as the k -reciprocal nearest neighbors.

Then the final distance is D , as Eq. (8) shows:

$$D = \alpha d_o + (1 - \alpha) d_j \quad (8)$$

where α denotes the penalty factor penalizing galleries far away from the query q , and its value ranges from 0 to 1. We can ascend sort of the final distance D to obtain the ranking list.

Besides, we have made use of the track data from the 2019 NVIDIA AI City Dataset, especially for re-ranking. The track data shows that a series of the same identity vehicle images shot by a camera in time. This is also the breakthrough point of our design. We have three ideas to refine the final ranking list. The followings are our ideas based on our designed DFFMG network.

Re-ranking_v1. In the dataset, each line in track_id.txt provides a sequence of ids for the same kind of vehicle. We name track id for each line in order. And we can know each vehicle image belongs to which track. For example, the first line consists of {29, 18199, 6332, ...}, then the NO.29, NO.18199, NO.6332 images belong to the first line named NO.1 track and these images' track id is 1. Base on this, we transform the ranking list $\{g_1, g_2, g_3, \dots, g_{100}\}$ to the track list $\{t_1, t_2, t_3, \dots, t_{100}\}$ where each t is each image's track id. In the track list, there are many identical track ids due to some of images in the initial ranking list are from the same kind. Thus we can count the frequency of every track's appearance for the following work.

We set different weights for different intervals, such as $[(0, 1], (0, 5], (0, 10]$, then we obtain $\{w(0, 1], w(0, 5], w(0, 10], w(0, 15], w(0, 20], w(0, 30], w(0, 100]\}$ where w denotes weight. Reasonable weights help to get better performance, so we unceasingly revised the weights in our

experiments. Meanwhile, for each interval, we count up the number of each track and rank them in order of frequency, respectively. Give an example of the method, for the interval $(0, i]$, the initial ranking list is $\{g_1, g_2, \dots, g_i\}$, the corresponding track list is $\{t_1, t_2, \dots, t_i\}$. In the track list, suppose that there are non-repetitive track ids $\{x, y, z\}$, and the numbers of tracks the same as x, y, z are a, b, c . So the frequencies of x, y, z are expressed as $f(0, i]$: $\{x: \frac{a}{i}, y: \frac{b}{i}, z: \frac{c}{i}\}$. For every interval, we can get different frequencies of different tracks and the $f(0, i]$ is a dictionary type storing each track's frequency in the interval $(0, i]$. Then we obtain $\{f(0, 1], f(0, 5], f(0, 10], f(0, 15], f(0, 20], f(0, 30], f(0, 100]\}$ for each non-redundant track id and each interval. And we multiply each f by each interval's weight we set before as Eq. (9) shows,

$$T_{score} = \sum_{i=0}^n f(T|(0, i]) \times w_{(0, i]} \quad (9)$$

where T denotes track, $f(T|(0, i])$ denotes the frequency of the track T when the interval is $(0, i]$.

The result of Eq. (9) becomes each track's score. Take x, y, z referred above for example, the score of the track x equals:

$$\frac{a_1}{1} \times w(0, 1] + \frac{a_2}{5} \times w(0, 5] + \dots + \frac{a_7}{100} \times w(0, 100],$$

and so on. If the score is higher, the track is more possible to be the same kind of vehicle set as the query vehicle. So, we sort the scores in descending order to get a final non-repeated track list $\{y, z, x\}$. Each track has many images, such as $x: \{x_0, x_1, x_2, \dots, x_n\}$, where x_i denotes the image id belonging to the track x . To make use of track labels, in the new ranking list, we bring all images from the same track which has higher score forward. Finally, we constrain it to the end when the number of the ranking list is 100. It means some images in the previous ranking list are probably to be abandoned. Then we can obtain the re-ranking list $\{y_0, y_1, y_2, \dots, y_n, z_0, z_1, z_2, \dots, z_n, x_0, x_1, x_2, \dots, x_n\}$, and the length of list is 100.

Re-ranking_v2. This version is based on the first version. Similarly, we get a final non-repeated track list $\{y, z, x\}$ following above steps at first. The difference is that we do not abandon any image in the original ranking list in case the wrong tracks were brought forward. Or it will result in the decrease of mAP. So we bring images which belong to a high score track and are in the original ranking list ahead in the new ranking list. For example, we may get $\{g_{30}, g_{22}, g_{34}, \dots, g_{23}, g_{12}, g_4, \dots, g_{10}, g_{62}, g_{29}\}$, the length of the list is 100 and $\{g_{30}, g_{22}, g_{34}, \dots\}$ belongs to the track y , other two belong to z, x , respectively.

As thus, we make sure that all the image ids from re-ranking list are from the initial ranking list and no one is abandoned, they just change positions with each other. So not only the CMC of top 100 will not be changed, but also the mAP will increase.

Re-ranking_v3. Version 3 is based on version 2 to fine-tune it. The final ranking list this version obtains abnegates some images which are low-ranking and let high-ranking images on the front. The idea is also smoothly comprehensive. First of all, we also get a final non-repeated track list $\{y, z, x\}$, and we bring images belonging to the top 1 and top 2 track $\{y, z\}$ to front, and then the rest images which belong to a high score track and are in the original ranking list are put after them in the new ranking list, like what we do in re-ranking_v2. Then we may get $\{g_{30}, g_{22}, g_{34}, \dots, g_{23}, g_{12}, g_4, \dots, g_{98}, g_{99}, g_{100}\}$, the length of the list is 100 and $\{g_{30}, g_{22}, g_{34}, \dots\}$ belongs to the track y , $\{g_{23}, g_{12}, g_4, \dots\}$ belongs to the track z . And $\{\dots, g_{98}, g_{99}, g_{100}\}$ don't belong to either y or z , the images in it are rest images reserved in order.

As thus, images from low-score track and are low-ranking in previous ranking list will be given up, and more images from the high-score track will appear in the front of the final ranking list simultaneously.

4. Experiments

4.1. The 2019 NVIDIA AI City Dataset

To better assist different vehicle-related experiments, this dataset splits into two parts for model training and testing and one part for the query to check the mAP (mean average precision) and CMC (cumulative matching characteristic). According to CityFlow [16], we know that the dataset contains 56,277 bounding boxes in total, where 36,935 of them from 333 object identities form the training set, and the test set consists of 18,290 bounding boxes from the other 333 identities.

Fig. 4 intuitively shows the exact numbers of each vehicle model for train dataset and test tracks. The rest of the 1,052 images are the queries. On average, each vehicle has 84.50 image signatures from 4.55 camera views. In the appendix, we can get labels for train and indeed no labels for a test.

4.2. Implementation details

In our experiments, we use the popular deep learning framework "Pytorch" to train and test our network model. Training data consists of its labels. Specifically, in our experiments, we were accurately positioning the vehicle in an image and crop it by Faster-RCNN [25].

The cropped images are the input. Then we transform every image's size to $(224, 224)$. We use momentum of $\mu = 0.9$ and weight decay $\lambda = 1 \times 10^{-5}$. Batch size is set to 32. We start with a base learning rate of $l(0) = 0.01$ and then drops by repeatedly multiplying 0.1 after every 50 epoch iterations.

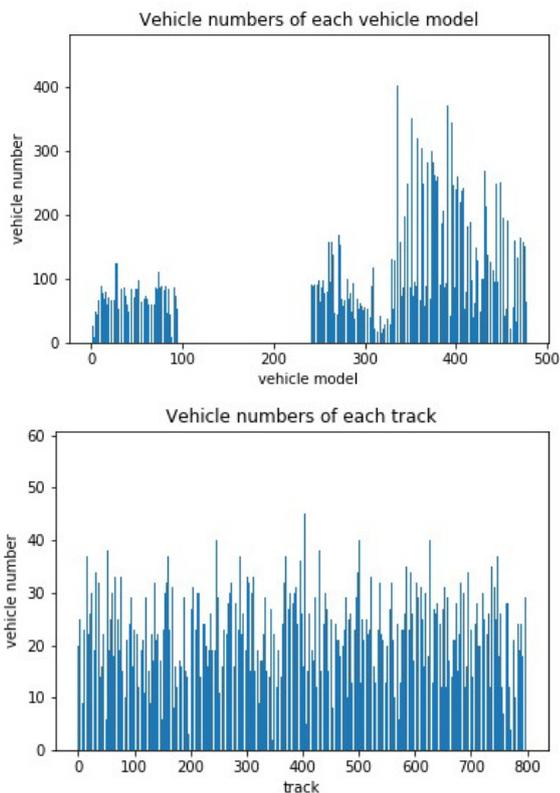


Figure 4. The first figure’s horizontal axis presents vehicle model labels for train. The second figure’s horizontal axis presents track ids of gallery. We can easily draw a conclusion that the number of every gallery track’s images is relatively balanced and as for train dataset, it shows a little sample unbalanced.

4.3. Results

When we did our tests, we recorded the whole changing process of loss decreasing as Fig. 5 shows. Overall, the loss declined regularly.

Moreover, we have fine-tuned and visualized the result. We let each image shown in a specific size to tile the interface. The result is illustrated in Fig. 6 and Fig. 7.

In Fig. 6 and Fig. 7, the left is the query image and the right are the top-30 nearest neighborhoods from the gallery which are resized to (224, 224). The red box and the green box correspond to the negatives and the positives, respectively. These figures show that our network achieves a good result.

By submitting the results in the online evaluation server, the CMC curve is shown in Fig. 8. And Tab. 1 displays the results of the state-of-the-art metric learning method and our networks for Vehicle Re-Id on our dataset.

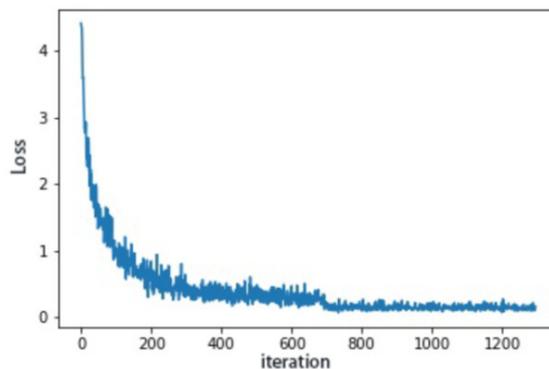


Figure 5. Loss changes by the number of iterations.



Figure 6. For the NO.2 query image, our network performs excellently. It clearly shows that all the matching images are accurate. It owes to the obvious features and feature fusion with multiple granularity.

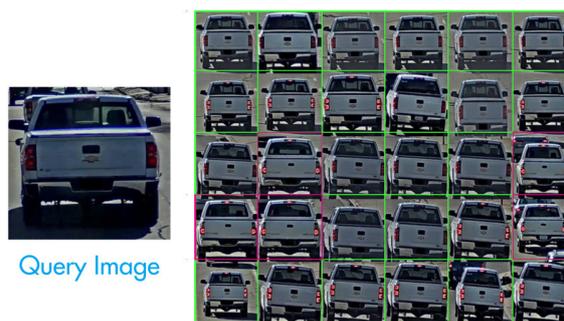


Figure 7. For the NO. 629 query image, our network performs also well. But a few images are not accurate and all the positives mostly show the same perspective of the vehicle. It’s challenging to tell all images of the same type from vehicles similar in brands, tires, colors and so on.

Method	mAP	Rank-1	Rank-5	Rank-10	Rank-15	Rank-20	Rank-30	Rank-100
MoV1+BA[19]	0.313	0.496	0.650	0.712	----	----	----	----
MoV1+BH[19]	0.320	0.484	0.652	0.714	----	----	----	----
MoV1+BS[19]	0.313	0.490	0.631	0.709	----	----	----	----
MoV1+BW[19]	0.308	0.501	0.649	0.714	----	----	----	----
PCB	0.198	0.425	0.537	0.594	0.646	0.676	0.709	0.773
MGN	0.229	0.377	0.451	0.476	0.499	0.551	0.621	0.688
DFFMG	0.253	0.480	0.600	0.661	0.693	0.726	0.777	0.855
DFFMG+v1	0.353	0.465	0.465	0.465	0.471	0.486	0.560	0.623
DFFMG+v2	0.285	0.482	0.491	0.497	0.505	0.528	0.604	0.709
DFFMG+v3	0.304	0.478	0.478	0.480	0.503	0.556	0.600	0.711

Table 1. These are results of different methods for Vehicle Re-Id with the 2019 NVIDIA AI City Dataset. In the first column, the last three methods are our DFFMG model with three re-ranking versions.

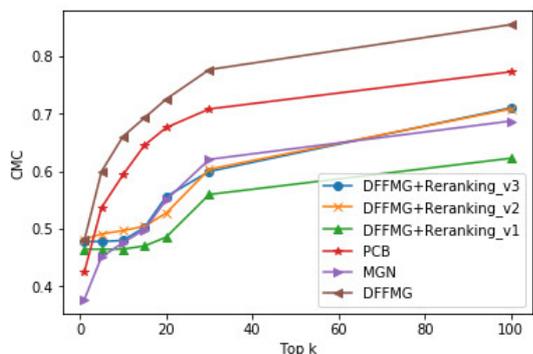


Figure 8. CMC Curve

From Tab. 1 and Fig. 8 we can learn about that our model DFFMG has a significant advantage in CMC compared to any other our models. It proves that the Deep Feature Fusion with Multiple Granularity indeed helps Re-Id tasks. However, the result is slightly less than the state-of-the-art metric learning method. Thus we should update our model by combining with some metric learning methods.

Even though, our model DFFMG refined by the first version of the re-ranking method achieves the highest mAP [26] among any other approaches, including the state-of-the-art metric learning method. This is our idea's breakthrough and strength.

The other two versions of re-ranking method perform better than the baseline MGN and original DFFMG, but not as good as the first version.

Nevertheless, the re-ranking methods also have apparent shortcomings that it is CMC has declined overall despite its high mAP. We think that our re-ranking approaches bring positive result set forward. In this way,

the recall rate will rise, and then on the contrast, the precision rate will decline. It is challenging to gain both simultaneously.

5. Conclusions

In this paper, we proposed a Deep Feature Fusion with Multiple Granularity (DFFMG) model to accomplish a Vehicle Re-Id task for 2019 AICITY CHALLENGE, which is a still not well-explored problem. Based on the baseline, PCB and MGN, we exploited a five-branch network for feature representations, one branch for the global feature representation, two branches for vertical local feature representations and two branches for horizontal local feature representations to obtain multiple features so that we could identify more precisely. Besides, we employed a newly original re-ranking method to get better performance.

In conclusion, our design has its innovation and strength. We combined multiple granularity with neoteric re-ranking methods and achieved good results which are over baseline and surpasses the state-of-art metric learning methods. Overall, the results can be improved in the future because there are still some queries matching wrong. We will take the essence of the state-of-the-art metric learning method and discard the dross to refine our model in the future to rise to the occasion to improve scientific research in the field of re-identification and promote the maturity and development of Vehicle Re-Id.

Acknowledgements

This work is supported by National Natural Science Foundation of China under Grant No. U1611461.

References

- [1] Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in Proc. Annu. Conf. Neural Inf. Process. Syst., Stateline, NV, USA, 2012, pp. 1097–1105.
- [2] Szegedy *et al.*, "Going deeper with convolutions," in Proc. *IEEE* Conf. Comput. Vis. Pattern Recognit., Boston, MA, USA, Jun. 2015, pp. 1–9.
- [3] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: <https://arxiv.org/abs/1409.1556>
- [4] X. Liu, W. Liu, H. Ma, and H. Fu, "Large-scale vehicle re-identification in urban surveillance videos," in Proc. *IEEE* Int. Conf. Multimedia Expo, Seattle, WA, USA, Jul. 2016, pp. 1–6.
- [5] X. Liu, W. Liu, T. Mei, and H. Ma, "PROVID: Progressive and multimodal vehicle reidentification for large-scale urban surveillance," *IEEE Trans. Multimedia*, vol. 20, no. 3, pp. 645–658, Mar. 2018. [6].
- [6] H. Liu, Y. Tian, Y. Wang, L. Pang, and T. Huang, "Deep relative distance learning: Tell the difference between similar vehicles," in Proc. *IEEE* Conf. Comput. Vis. Pattern Recognit., Las Vegas, NV, USA, Jun. 2016, pp. 2167–2175.
- [7] Ahmed, E.; Jones, M.; Marks, T.K. An improved deep learning architecture for person re-identification. In Proceedings of the *IEEE* Conference on Computer Vision and Pattern Recognition, Hynes Convention Center, Boston, MA, USA, 7–12 June 2015; pp. 3908–3916.
- [8] Zhao, Rui, W. Ouyang, and X. Wang. "Learning Mid-level Filters for Person Re-identification." 2014 *IEEE* Conference on Computer Vision and Pattern Recognition (*CVPR*) *IEEE* Computer Society, 2014.
- [9] S. Ding, L. Lin, G. Wang, and H. Chao. Deep feature learning with relative distance comparison for person reidentification. PR, 2015. 2
- [10] D. Cheng, Y. Gong, S. Zhou, J. Wang, and N. Zheng. Person Re-identification by multi-channel parts-based cnn with improved triplet loss function. In *CVPR*, 2016. 2
- [11] Zapletal, Dominik, and A. Herout. " [*IEEE* 2016 *IEEE* Conference on Computer Vision and Pattern Recognition Workshops (*CVPRW*) - Las Vegas, NV, USA (2016.6.26-2016.7.1)] 2016 *IEEE* Conference on Computer Vision and Pattern Recognition Workshops (*CVPRW*) - Vehicle Re-identification for Automatic Video Traffic Surveillance." 2016 *IEEE* Conference on Computer Vision and Pattern Recognition Workshops (*CVPRW*) *IEEE* Computer Society, 2016:1568-1574.
- [12] Zhu, Jianqing, *et al.* "Vehicle Re-identification Using Quadruple Directional Deep Learning Features." (2018).
- [13] S. Chopra, R. Hadsell and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," 2005 *IEEE* Computer Society Conference on Computer Vision and Pattern Recognition (*CVPR*), San Diego, CA, USA, 2005, pp. 539-546.
- [14] W. Li, Y. Wu, M. Mukunoki, and M. Minoh. Common-nearneighbor analysis for person re-identification. In *ICIP*, 2012.
- [15] Zhong, Zhun, *et al.* "Re-ranking Person Re-identification with k-reciprocal Encoding." (2017).
- [16] Tang, Zheng, *et al.* "CityFlow: A City-Scale Benchmark for Multi-Target Multi-Camera Vehicle Tracking and Re-identification." (2019).
- [17] Sun Y, Zheng L, Yang Y, *et al.* Beyond Part Models: Person Retrieval with Refined Part Pooling (and a Strong Convolutional Baseline)[J]. 2017.
- [18] Wang G, Yuan Y, Chen X, *et al.* Learning Discriminative Features with Multiple Granularities for Person Re-identification[C]// 2018 *ACM* Multimedia Conference. *ACM*, 2018.
- [19] Ratnesh Kumar, Edwin Weill, Farzin Aghdasi, and Parthasarathy Sriram. Vehicle Re-identification: An efficient baseline using triplet embedding. In Proc. *IJCNN*, 2019. 5, 7, 12
- [20] Marin-Reyes, Pedro Antonio, *et al.* "Unsupervised Vehicle Re-identification Using Triplet Networks." 2018 *IEEE/CVF* Conference on Computer Vision and Pattern Recognition Workshops (*CVPRW*) *IEEE*, 2018.
- [21] "Joint Feature and Similarity Deep Learning for Vehicle Re-identification." *IEEE* Access (2018):1-1.
- [22] D. Qin, S. Gammeter, L. Bossard, T. Quack, and L. Van Gool. Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In *CVPR*, 2011. 1, 2, 3
- [23] S. Bai and X. Bai. Sparse contextual activation for efficient visual re-ranking. *IEEE TIP*, 2016. 2, 4.
- [24] M. Ye, C. Liang, Y. Yu, Z. Wang, Q. Leng, C. Xiao, J. Chen, and R. Hu. Person Re-identification via ranking aggregation of similarity pulling and dissimilarity pushing. *IEEE TMM*, 2016. 1, 2, 3, 4.
- [25] Ren, Shaoqing, *et al.* "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks." (2015).
- [26] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian. Scalable person re-identification: A benchmark. *ICCV*, pages 1116–1124, 2015.