Comparative Study of Various Losses for Vehicle Re-identification

Adithya Shankar[1], Akhil Poojary[1], Varghese Kollerathu[2], Chandan Yeshwanth[2], Sheetal Reddy[2], and Vinay Sudhakaran[2]

[1]Department of Engineering Design, Indian Institute of Technology Madras, Chennai, India
[2]Advanced Data Management Research Group, Siemens Corporate Research and Technologies, Siemens Technology and Services Pvt. Ltd., Bengaluru, India
{ed15b003, ed15b043}@smail.iitm.ac.in
{varghese.kollerathu, chandan.yeshwanth, sheetalreddy.arrabotu, vinay.sudhakaran}@siemens.com

Abstract

In this paper, we tackle the problem of vehicle re-identification, which has extensive applications in traffic analysis such as anomaly detection, congestion pricing and tolling. While previous methods extract visual features from the images and then use spatio-temporal regularization to further refine the results, our method focuses on extracting purely visual features from vehicle images and then further employs a re-ranking technique to improve results. We evaluate the proposed pipeline on the VeRi and CityFlow (NVIDIA AI City Challenge 2019) datasets. Experiments show that our pipeline achieves state of the art performance on the VeRi dataset. We also perform extensive analysis on each step of the pipeline and demonstrate how they increase overall performance.

1. Introduction

Vehicle re-identification (Re-ID), deals with associating the image of a vehicle obtained from one camera with the images of the same vehicle as it re-appears in the same or different non-overlapping cameras. These cameras may be situated at completely different geographical locations and the algorithm *tracks* the vehicle across different cameras. The task of Re-ID is very similar to that of image retrieval, as it deals with matching a queried object against a set of objects in the image gallery. The algorithm, which is used for retrieval, works by ranking the gallery images in the ascending order of their distances from the query image. The images which are ranked higher are expected to be the correct matches. There are several use-cases of Re-ID in crowd monitoring and anomaly detection, but more recently it has found widespread adoption in traffic analysis and monitoring. When compared to person Re-ID, vehicle Re-ID is much more challenging as demonstrated by Zhou *et al*. [21] and in Figure (1). This is primarily because in person Re-ID, there are several distinctive features available for the model - e.g. the color of the upper and lower clothing. The model can consider various features like the color combination of clothing, the texture of clothes, features attributed to a person, and so on. These features are noticeable and quite distinctive regardless of the pose of the person, as shown Figure (2).

(a) (b) (c)

Figure 1: Challenges in vehicle Re-ID:- (a) A scenario where the same car looks different due to varying illumination, viewpoints and partial occlusions. (b) The effect of shadows and sunlight causing the change in the perceived color. (c) A scenario where four distinct cars look very similar from a common viewpoint.

When it comes to vehicle re-identification, there is a noticeable similarity between different aspects of two vehicles, apart from colors. Hence, it is difficult to differentiate between cars of the same make and color. A straightforward approach is to implement a robust number plate recognition system as it would provide us with a unique identity for every vehicle. Previous contributions [5, 13] have demonstrated that automatic recognition of license plates as a glob-

This work was done as a part of an internship at Siemens Technology and Services Pvt. Ltd by the first and the second author.

Figure 2: In the task of person Re-identification, the viewpoint or pose of a person in an image does not have any major influence on the model inference. It can be observed that the color combination of discernible attributes (shirt, trousers, accessories) are prominently visible in different views (Image courtesy: Zheng *et al*. [18])

ally unique identifier provide state-of-the-art performance in vehicle identification. However, in practice it is not always feasible to extract the number plate information from all vehicles due to occlusions, variation in lighting, shadows and/or poor image quality. Thus, it is necessary to develop methods that can extract and process visual features of a vehicle in order to implement Re-ID for practical scenarios.

Figure (1) highlights some real-world obstacles faced by vehicle Re-ID. The two major ones are:

- The same vehicle, when in a different orientation/pose or viewed from different viewpoints looks considerably different.
- Different vehicles look similar from the same viewpoint, especially when the two vehicles are of the same model and color (e.g.: two different black colored sedans).

Taking these variations into consideration, our objective is to develop a model which can generate an embedding that is not only unique for each vehicle, but also invariant to illumination and the pose of the vehicle. Previous works by Zhou *et al*. [21] and Wang *et al*. [15] attempt to tackle the problem of pose variation. Zhu *et al*. [22] proposed an unconventional and novel way to pool features in order to make the embedding generated by a vehicle pose invariant.

In this paper,

- we experiment with different loss functions and compare their performance on two distinct datasets viz; VeRi [8] and CityFlow [14] datasets.
- re-ranking [19] is used as a post-processing step to further improve the performance.

2. Related Work

Convolutional Neural networks (CNNs) produce state of the art results on a variety of computer vision and pattern recognition tasks. Contributions such as Orientation Invariant Feature Embeddings[15] make use of datasets with annotated key-points to generate salient regions in an image. This approach is analogous to attention maps generated in a CNN, along the lines of making the model pose invariant. According to Wang *et al*. [15], 20 key-points were extracted from a vehicle to discriminate it from other vehicles. Based on the viewpoint of the vehicle, in any given image a set of key-point based features were extracted to produce a saliency map. Further, four different models were used for each of the 4 distinct viewpoints. Each of these four networks consist of an hourglass architecture and each of them determine the various saliency regions in the image. The feature vectors from each network are weighted based on the pose of the vehicle. This weighted feature vector is then passed to a fully connected layer to generate a final single embedding vector.

Zhou *et al*. [21] proposed a framework in which a network was trained to classify various vehicle attributes such as color, model, viewpoint to name a few. Intermediate features from the trained network were passed through an attention model in order to obtain the attentive features for five different views. These attentive features were then concatenated and trained with an adversarial loss function, where the generator network was used to generate the features of the same vehicle for different viewpoints. Zhou and Ling [20] develop on their previous work [21] which deals with predicting the features of unseen views using bi-directional LSTM. The network was trained to minimize multiple loss functions namely contrastive loss, reconstruction loss and adversarial loss.

Zhu *et al*. [22] introduced a method for pooling features (from the penultimate layer of the network) in four different directions, in order to learn a much better representation of the data. Instead of Global Average pooling, the features were pooled along the diagonal, anti-diagonal, horizontal (rows) and vertical (columns) directions. These extracted features were then fused together to obtain a single vector representation of the vehicle. Most models use the triplet loss or a variant of it to differentiate between similar looking vehicles. To minimize the effects of biased anchors, Liu *et al*. [7] proposed a method of replacing the anchor in the triplet loss with the center of the feature embedding of each class. They also trained their deep learning model in a hierarchical way, beginning with vehicle model classification (SUV, Sedan, etc.) followed by vehicle identity classification. Bai *et al*. [1] proposed sub clustering in triplet loss to handle intra-class variance. Kumar *et al*. [6] proposed a weighted triplet loss, where the positive score and the negative score used in the triplet loss are scaled in a multinomial fashion. This aided in achieving state of the art results on the VeRi dataset. Liu *et al*. [9] proposed a method to progressively refine the list of retrieved images for a given

query. Initially, a ranked list was obtained based on the similarity of visual feature embeddings. In order to obtain this embedding, a Siamese CNN was trained with contrastive loss. Based on the presence of license plates in the image the plates are identified and scanned. The list of retrievals was then refined using the extracted number plates. Additionally, to make the model more robust, spatio-temporal information was incorporated into the distance metric in order to enhance the retrievals of visually similar vehicles.

Shen *et al*. [12] used an LSTM to obtain a path proposal for each query image based on its spatio-temporal information. Wang *et al*. [15] enhance their results by modelling the camera transition probabilities using random variables. However, incorporating spatio-temporal information requires much more data which is not always available. Thus, the most simple and sophisticated model would be based on features extracted visually from an image, as they can directly be compared against the features extracted from any other vehicle image.

3. Methodology

3.1. Data

The model was evaluated on the CityFlow dataset from the NVIDIA-AI City Challenge track 2. The dataset consists of 56,277 images captured from 40 different cameras out of which 36,935 images of 333 unique vehicles are provided with annotations for training. The remaining 18,290 images of other 333 vehicles make up the test set, which serves as the 'gallery' to get the retrievals from. In the dataset provided by the organizers, a set 1052 images forms the 'query' set.

To find the optimal model and the hyper-parameters associated with it, the provided training set was split into training and validation sets. From the set of 333 unique vehicles in the training set, images corresponding to 266 vehicle identities (29,649 images) were randomly picked to form the training data. The remaining 7,286 images corresponding to 67 identities serves as the validation data for our model. A query set was made by picking out 67 vehicles (one from each vehicle identity) from the training set.

To measure the robustness of our model, the model was tested on the VeRi Dataset [8]. This dataset comprises training 37,781 images with 576 distinct vehicles. The 'gallery' comprises 200 distinct vehicles (13,257 images), while a cohort of 1678 images representing 200 vehicle ids form the 'query' set.

3.2. CNN Architecture for Vehicle Re-Identification

For the the backbone of our network, we use the Multi-Layer Factorisation Network (MLFN) as proposed by Chang *et al*. [2]. This network consists of several MLFN blocks; each block consists of multiple Factor Modules (FM) and a Factor Selection Module (FSM). All the Factor Modules are identical and consist of a set of convolutional layers. The FSM is responsible for the selection of the Factor Modules in a block. In an MLFN block, any number of Factor Modules can be activated based on the output from the Factor Selection Module. Figure (3) shows a schematic of this model.

In the MLFN architecture, Factor Modules which are on the same level in the MLFN blocks (B_1, B_2 ....B_N) can be interpreted as a separate network. Each set of FMs could be responsible for generating the features corresponding to a particular view of a vehicle. The Factor Selection Module in each block selects which FM are important based on the viewpoint of the vehicle.

We experiment by training the network with the following losses -

- Cross-Entropy Loss
- Hard Triplet Loss[4] + Cross Entropy Loss
- Center Loss[17] + Cross Entropy Loss
- Additive Angular Margin Loss[3] + Cross Entropy Loss

All the losses are fused with Cross Entropy Loss to stabilize the training process.

3.3. Cross-Entropy

We use conventional cross-entropy loss after the final softmax layer. The softmax layer outputs the log of probabilities. In Eq.[1], $x_i \in \mathbb{R}^d$ denotes the deep feature of the i^{th} sample which belongs to the y_i^{th} class. $W_j \in \mathbb{R}^d$ denotes the j^{th} column of the weight $W \in \mathbb{R}^{d \times n}$ and $b_j \in \mathbb{R}^n$ the bias term. The variables N and n are the batch size and class number, respectively. A softmax layer followed by a cross entropy loss is widely used for classification. However, this approach does not explicitly optimize for the task of obtaining a good feature embedding.

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\left(\frac{e^{(W_{y_i}^T x_i + b_{y_i})}}{\sum_{j=1}^{n} e^{(W_j^T x_i + b_j)}}\right) \tag{1}$$

3.4. Additive Angular Margin Loss (ArcFace)

Deng *et al*. [3] set the bias term b_j to 0 in Eq.1, and then transform the logit as per $\|W_j\| \|x_i\| cos(\theta_j)$, where θ_j is the angle between weight W_j and the feature x_i. The weights W_j and the embedding feature x_i were normalized using l_2 normalization. The normalized embedding feature was then re-scaled to $s = 30$ (hyperparameter). These normalizations restrict the predictions to depend only on the angle θ_j between feature vectors and the weights. The learned embedding features form a distribution on a hypersphere

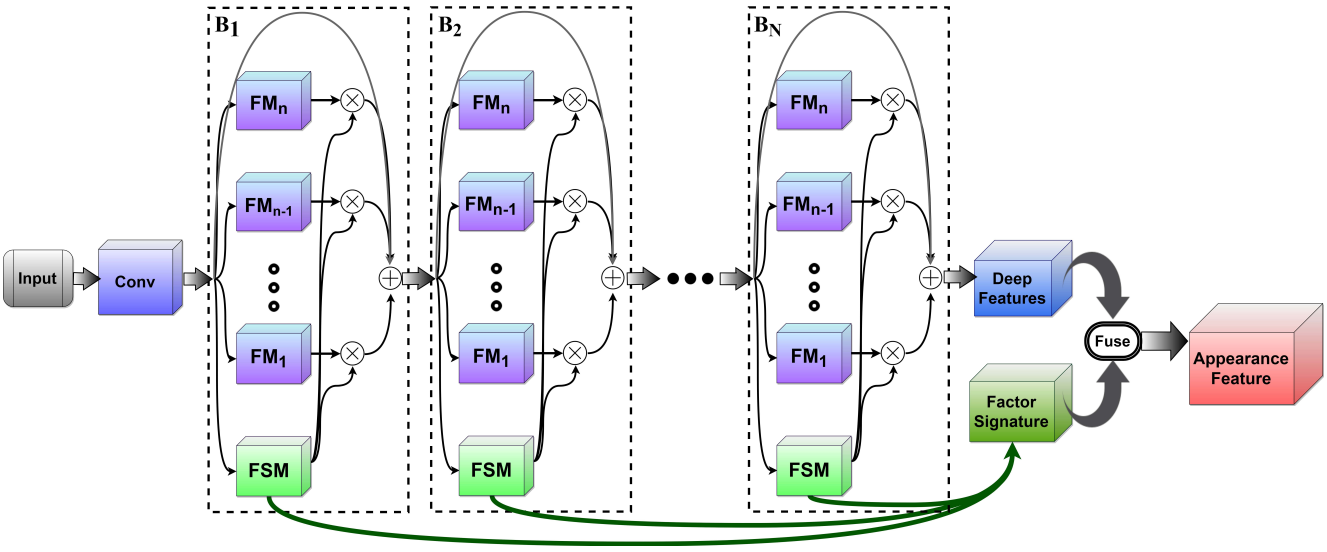

Figure 3: Multi-level Factorization Network (MLFN)

with a radius s. This forms the Additive Angular Margin Loss (ArcFace) shown in Eq. [2].

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\left(\frac{e^{scos(\theta_{y_i})}}{e^{scos(\theta_{y_i})} + \sum_{j=1, j\neq y_i}^{n} e^{(scos\theta_j)}}\right) \tag{2}$$

To increase intra-class compactness, a marginal penalty m was added to the angle between x_i and W_{y_i}, Eq.[3].

$$L = -\frac{1}{N}\sum_{i=1}^{N} log\left(\frac{e^{scos(\theta_{y_i}+m)}}{e^{(scos\theta_{y_i}+m)} + \sum_{j=1, j\neq y_i}^{n} e^{(scos\theta_j)}}\right) \tag{3}$$

3.5. Center Loss

The idea behind center loss [17] is to pull all the features of a class together and make them converge at a single point. Ideally, this is what is required for vehicle Re-ID where the features of the vehicle should be the same irrespective of the viewpoint.

$$L_c = \frac{1}{2}\sum_{i=1}^{N} \|x_i - c_{y_i}\|^2 \tag{4}$$

In Eq.4, $c_{y_i} \in \mathbb{R}^d$ denotes the center of deep features corresponding to the y_i^{th} class. This formulation is effective to model the intra class variations. Ideally, c_{y_i} should be updated once all the samples corresponding to the y_i^{th} class have been passed through the network. However, this is impractical since all images corresponding to the same class cannot be sent in a single batch. During implementation, the centers are updated after each mini batch. Using Cross Entropy Loss alone results in feature embeddings that have large intra class variation while using center loss alone could result in the learnt features and sphere centers eventually degrading to zero, i.e., the loss becomes zero without the model learning the necessary features.

3.6. Hard Triplet Loss

Schroff *et al.* [11] proposed a modification to the large margin nearest neighbor loss[16] called the triplet loss. The hypothesis behind this kind of loss formulation was that the largest intra class distance in the training samples should be lower than the smallest inter class distance. However, since it is not possible for a network to generate extremely similar features for the same vehicle in drastically different viewpoints and scenarios, this loss formulation ensures that the features of the same car are closer in the embedded vector space as compared to the features of a different car.

$$L_{tri}^{a}(a) = \sum_{a,p,n} [m + D_{a,p} - D_{a,n}] \tag{5}$$

Given an anchor point x_a, the projection of a positive point x_p (belonging to the same class as the anchor) is closer to the anchor's projection, than that of a negative point x_n (belonging to a different class). Further, the margin m in Eq.[5] ensures that these two classes are separated by at least a margin of m. We follow the approach as proposed by Hermans *et al.* [4]. Computing triplet loss is expensive, so it is important to sample effective data points for the triplet loss. If the data points given to the model for training are trivial, this would make the algorithm converge faster as it is easier to satisfy the condition, and make the model perform poorly on non-trivial data points. On the other hand, training the model only with hard (non-trivial) samples makes

it vulnerable to outliers. Thus, it is important to train the model with a mixture of trivial and non-trivial data-points. In the context of vehicle Re-ID, the trivial data points would be those where the same vehicle in the same pose is given as the positive sample point to the anchor. For a given anchor x_a, a hard positive data point corresponds to images of x_a acquired from different viewpoints.

Hermans *et al*. [4] construct a training batch by randomly sampling P identities from the training sample, and then randomly sampling K images for each identity, yielding a batch of $(P \times K)$ images. They have demonstrated the performance of two methods on a batch of images, Batch Hard and Batch All. In the Batch Hard technique, only those triplets are formed from a batch, per anchor, which consist of the hardest positive and the hardest negative corresponding to that anchor. This way, we will obtain $(P \times K)$ samples per batch. In the Batch All method, all the possible triplets for every anchor are sampled and presented to the model. This way we will end up with $\big(P \times K \times (K-1) \times (PK-K)\big)$ triplets.

3.7. Training Procedure

We use the network architecture as the one described by Chang *et al*. [2]. We stack 16 MLFN blocks with each block containing 32 Factor Modules. Correspondingly, we get a 32-D Factor Solution Module (FSM) output vector. This results in a $32 \times 16 = 512$-dimension Factor Signature. The embedding feature dimension is set to 1024. The network was trained for 80 epochs. The learning rate is set to 0.0003 and then reduced by a factor of 10 after 20 and 40 epochs. The parameters of the network were updated to minimize one of the loss functions mentioned in 3.2. For networks trained with Triplet, Center and ArcFace loss, the loss is summed with Cross-Entropy loss to stabilize the training process. The parameters of the network were learned to minimize the loss function with AMS Grad[10] as the optimizer.

The images were resized to a dimension 224×224 using bilinear interpolation. Further the images were flipped along the vertical axis. The brightness and contrast of the images were randomly varied by a factor of $\pm 0.2\%$ and $\pm 0.15\%$ respectively so as to make the model robust to illumination changes. Batch size was set to 128, with 8 instances per id.

3.8. Testing Procedure

During inference, images were resized to 224×224 and no further data augmentation techniques were used. The final classification layer was removed to attain a 1024 dimension feature vector. Cosine distance was used as the similarity metric between two feature vectors. The feature embeddings were l_2 normalized to a unit vector before computing the cosine distance.

Model	**mAP** $(lr = 0.0003)$	**mAP** $(lr = 0.00003)$
MLFN-Triplet	57	40.2
MLFN-ArcFace	36.2	19.6
MLFN-Center	40.7	18.5

Table 1: Performance on the AICity held-out dataset. lr is the learning rate and MLFN stands for Multi-Level Factorisation Net, the backbone network which was trained on Triplet, ArcFace and Center Loss.

4. Results

4.1. Evaluation Metric

Mean Average Precision was used to evaluate the model. For each query, we calculate the area under the Precision-Recall curve, which is known as average precision (AP) for that query. The mean value of APs of all the queries is the mean average precision (mAP).

4.2. Hyper-Parameter Optimization

For CNNs learning rate, loss function and the number of layers are few of the hyperparameters. In this paper, choice of the learning rate and the loss to be minimized were chosen based on the performance on the "held out data". Table 1 compares the performance of the network when trained using different learning rates and various loss functions. From the table it was observed that the network trained with a learning rate of 0.0003 with Triplet loss and Cross entropy as the loss function produced better performance than networks trained with other configurations.

Table 2 illustrates the performance of the network on VeRi dataset when trained with different hyperparameter settings. Similar to the AI city 'held out test data', it was observed that the model trained with learning rate of 0.0003 with triplet loss and cross entropy as the joint loss function produced better performance.

4.3. Performance on Challenge Data

The trained network was tested on the CityFlow dataset which was made available by the organizers of *Nvidia AI City Challenge 2019*. The resulting performance metrics were obtained by uploading our results to the online portal provided by the AI City Challenge.

It was observed that the post processing technique, i.e., re-ranking of the retrieved results aided improved mAP by 4.5%, Table 3. Figure 5 (a),(b) show the results before and after the post-processing step. Out of 84 teams, the proposed solution was placed 54^{th}.

4.4. Performance on VeRI dataset

To gauge the robustness of the model, the model was evaluated on the VeRi dataset. On the test data, with mean

Model	mAP ($lr = 0.0003$)	mAP ($lr = 0.00003$)
MLFN-Triplet	66.1	60.3
MLFN-ArcFace	19.2	35.1
MLFN-Center	34.3	49.42
MLFN-Xent	64.9	–

Table 2: Performance on VeRi dataset. lr is the learning rate and MLFN stands for Multi-Level Factorisation Net, the backbone network which was trained on Triplet, ArcFace and Center Loss.

Model	mAP
MLFN+Triplet	23.1
MLFN+Triplet+Re-Rank	27.6

Table 3: Performance of MLFN trained with Triplet loss on AICity test dataset with learning rate of 0.0003. Second row shows the model performance after Re-Ranking the results.

Method	mAP	top-1	top-5
Our Method + ReRanking	**71.78**	92.55	95.173
Batch Sampling[6]	67.55	90.23	96.42
Our Method	66.06	91.78	96.42
GSTE[1]	59.47	**96.24**	**98.97**
VAMI[21]	50.13	77.03	90.82
VAMI+ST[21]	61.32	85.92	91.84
OIFE[15]	48.00	89.43	-
OIFE+ST[15]	51.42	92.35	-
PROVID[9]	27.77	61.44	78.78
Path-LSTM[12]	58.27	83.49	90.04

Table 4: Comparing the performance of our approach with the other proposed approaches on VeRI Dataset

Average Precision as the evaluation metric, it was observed that the proposed solution achieved state of the art performance. Table 4. compares the performance of our model with the existing state of the art Re-ID techniques.

5. Discussion

Based on our experiments, it was observed that Triplet loss along with Cross Entropy was the most effective loss function for the task of vehicle Re-ID. When compared to other losses, Triplet loss aids in pulling the feature embeddings of a class together while pushing apart the feature embeddings associated to different classes. On the other hand, Additive Angular Margin Loss (ArcFace) does not explicitly formulate the loss to do this. Center loss reduces the distance between each class, but fails to push different classes apart.

Qualitatively it was observed that models trained with triplet loss produce fewer false positives in top ranks when compared to ArcFace and Center Loss, Figure (4).

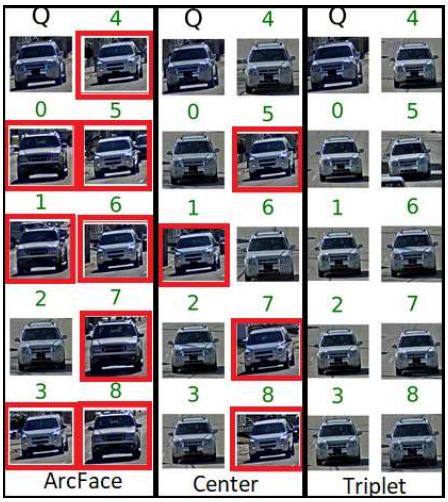

Figure 4: Illustrates the effectiveness of triplet loss. (Images with the red bounding boxes are incorrect retrievals, ones without a bounding box are correct retrievals)

Combining Triplet, Center and ArcFace loss would not result in an effective loss formulation. Center Loss incorporates the same pull factor which also appears in triplet loss. ArcFace loss projects the features produced by the CNN onto a sphere-space and separates the different classes by an angular margin, while Triplet loss separates the classes in cosine-distance space. Effectively, Triplet loss provides all the benefits which the other loss functions provide combined. Re-ranking proves to be an effective post-processing technique which improves retrieval results by **4.5% mAP on Cityflow** (AICity Challenge 2019) dataset and **5.72% mAP on VeRI dataset**. The idea behind re-ranking is to inspect if the actual query image is retrieved when each of the retrieved images are used as a query image. Such a scenario implies a high correlation between the two images (query and retrieved image). This way of refining the retrievals brings the correct matches to a higher rank in the query-retrieval set, thereby increasing the mAP value.

Based on our experimental results, it was observed that CNNs extract features based on the pose of a vehicle along with other attributes such as color and make of the vehicle. Other extraneous factors such as lighting conditions, shadows as well as occlusions in the image also contribute to the extracted features significantly. This issue is illustrated in Figure (6). For instance, two distinct cars which are relatively similar in appearance (compared to two very different cars) and in the same pose, will produce features which are closer in the distance space when compared to the same car when it is in a different pose. This is an inherent property of the convolution neural network, as it will produce similar

(a) Before Re-Ranking

(b) After Re-Ranking

Figure 5: Effect of Re-Ranking. (Images bounded by green box are correct matches whereas images bounded by red box are incorrect matches.

outputs when the distributions of pixels are comparable.

However, when the color of the queried vehicle is distinctive, the network is able to retrieve its true matches, regardless of viewpoints or poses as depicted in Figure (7). A possible explanation for this is the sparsity of such vehicles in the dataset. When compared to CityFlow dataset, the images in VeRi dataset have less variations along the lines of illumination, occlusion, perspective and scale. For the reason stated above, it was observed that our model was able to achieve state-of-the art results on VeRi dataset, while achieving comparable results on a much more challenging CityFlow dataset.

Figure 6: Cars which look similar to the Query car in the same viewpoint are retrieved before the actual queried car's images in other poses. The effect of lighting and illumination conditions can also be seen here.

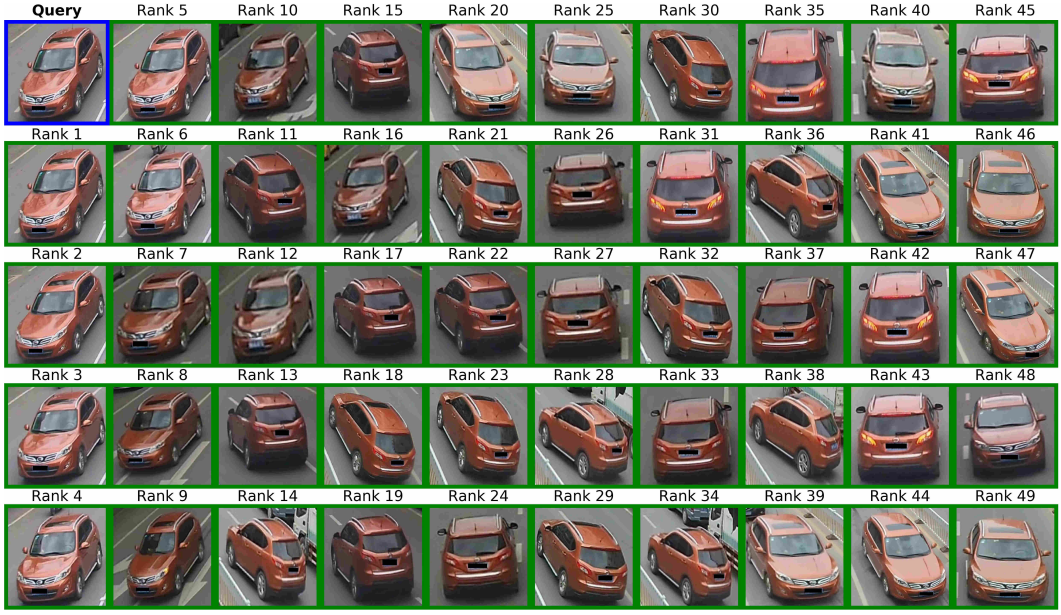

Figure 7: Better performance is observed for cars with a distinctive color.

References

[1] Yan Bai, Yihang Lou, Feng Gao, Shiqi Wang, Yuwei Wu, and Ling-Yu Duan. Group-sensitive triplet embedding for vehicle reidentification. *IEEE Trans. Multimedia*, 20(9):2385–2399, 2018.

[2] Xiaobin Chang, Timothy M. Hospedales, and Tao Xiang. Multi-level factorisation net for person re-identification. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2109–2118, 2018.

[3] Jiankang Deng, Jia Guo, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. *CoRR*, abs/1801.07698, 2018.

[4] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *CoRR*, abs/1703.07737, 2017.

[5] Vishal Jain, Zitha Sasindran, Anoop Kolar Rajagopal, Soma Biswas, Harish S. Bharadwaj, and K. R. Ramakrishnan. Deep automatic license plate recognition system. In *Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing, ICVGIP 2016, Guwahati, Assam, India, December 18-22, 2016*, pages 6:1–6:8, 2016.

[6] Ratnesh Kumar, Edwin Weill, Farzin Aghdasi, and Parthsarathy Sriram. Vehicle re-identification: an efficient baseline using triplet embedding. *CoRR*, abs/1901.01015, 2019.

[7] Hongye Liu, Yonghong Tian, Yaowei Wang, Lu Pang, and Tiejun Huang. Deep relative distance learning: Tell the difference between similar vehicles. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2167–2175, 2016.

[8] Xinchen Liu, Wu Liu, Huadong Ma, and Huiyuan Fu. Large-scale vehicle re-identification in urban surveillance videos. In *IEEE International Conference on Multimedia and Expo, ICME 2016, Seattle, WA, USA, July 11-15, 2016*, pages 1–6, 2016.

[9] Xinchen Liu, Wu Liu, Tao Mei, and Huadong Ma. PROVID: progressive and multimodal vehicle reidentification for large-scale urban surveillance. *IEEE Trans. Multimedia*, 20(3):645–658, 2018.

[10] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *CoRR*, abs/1904.09237, 2019.

[11] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 815–823, 2015.

[12] Yantao Shen, Tong Xiao, Hongsheng Li, Shuai Yi, and Xiaogang Wang. Learning deep neural networks for vehicle re-id with visual-spatio-temporal path proposals. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 1918–1927, 2017.

[13] Jakub Spanhel, Jakub Sochor, Roman Juránek, Adam Herout, Lukas Marsik, and Pavel Zemcík. Holistic recognition of low quality license plates by CNN using track annotated data. In *14th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2017, Lecce, Italy, August 29 - September 1, 2017*, pages 1–6, 2017.

[14] Zheng Tang, Milind Naphade, Ming-Yu Liu, Xiaodong Yang, Stan Birchfield, Shuo Wang, Ratnesh Kumar, David C. Anastasiu, and Jenq-Neng Hwang. Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification. *CoRR*, abs/1903.09254, 2019.

[15] Zhongdao Wang, Luming Tang, Xihui Liu, Zhuliang Yao, Shuai Yi, Jing Shao, Junjie Yan, Shengjin Wang, Hongsheng Li, and Xiaogang Wang. Orientation invariant feature embedding and spatial temporal regularization for vehicle re-identification. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 379–387, 2017.

[16] Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, 2009.

[17] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VII*, pages 499–515, 2016.

[18] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1116–1124, 2015.

[19] Zhun Zhong, Liang Zheng, Donglin Cao, and Shaozi Li. Re-ranking person re-identification with k-reciprocal encoding. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 3652–3661, 2017.

[20] Yi Zhou and Ling Shao. Vehicle re-identification by adversarial bi-directional LSTM network. In *2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018, Lake Tahoe, NV, USA, March 12-15, 2018*, pages 653–662, 2018.

[21] Yi Zhou and Ling Shao. Viewpoint-aware attentive multi-view inference for vehicle re-identification. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 6489–6498, 2018.

[22] Jianqing Zhu, Huanqiang Zeng, Jingchang Huang, Shengcai Liao, Zhen Lei, Canhui Cai, and Lixin Zheng. Vehicle re-identification using quadruple directional deep learning features. *CoRR*, abs/1811.05163, 2018.