Relation Preserving Triplet Mining for stabilising the Triplet Loss in Re-identification Systems

Abstract

Curriculum Learning (CL) is a training strategy that gradually introduces the machine learning model to simpler and progressively more complex data, enhancing its learning effectiveness. In previous vehicle re-identification methods, the tendency to rely solely on hard samples for triplet loss computation has led to the neglect of managing training datasets and triple sampling patterns. This oversight can cause the model to overfit to challenging instances, limiting its ability to generalize to other sample types. In this paper,we suggest the training dataset should encompass examples ranging from easy to challenging,and it is akin to mimicking the sequential progression in human learning during a course. This leads us to introduce Easy to Difficult Vehicle Re-identification (ETDVR) method,which respects the human learning patterns. We use such a training scheme to establish a human-like, well-conditioned training process by adjusting the difficulty of training samples at different stages. This allows a single network to evolve from a simple structure to a more complex one,simultaneously providing state-of-the-art results.

Keywords: Vehicle Re-identification, Curriculum Learning, Triplet Mining, Human Learning Pattern.

1 Introduction

The task of vehicle re-identification (V-ReID) involves matching images of the same vehicle from a large gallery set when presented with a query image, as depicted in Fig 1.V-ReID has significant practical applications in fields such as self-driving vehicles, smart cities, and traffic monitoring, and can provide a reliable alternative to other methods such as sensors and license plate recognition.

Despite many progresses have been made in the recent years due to deep learning, vehicle Re-ID is still face many challenges, such as large intra-class variance and small inter-class variance caused by different

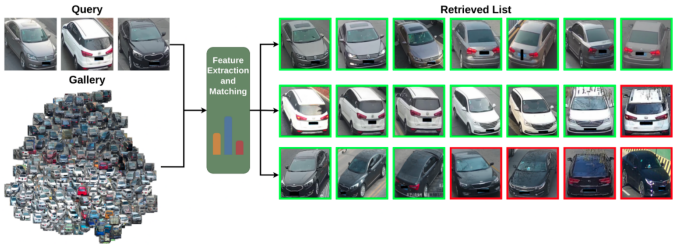

Figure 1. The example of vehicle re-identificaiton

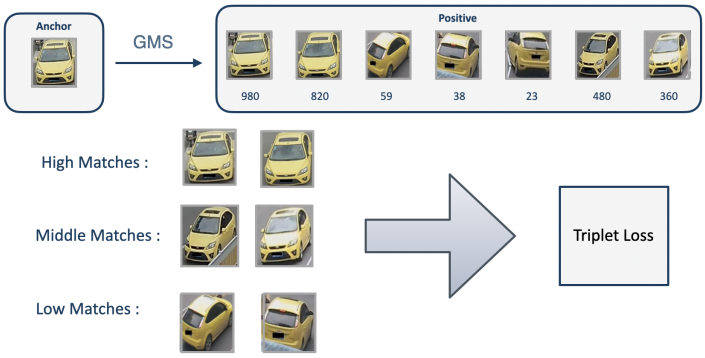

Figure 2. The training process for sample difficulty measurement based on matching values of traditional features.

perspectives. The past methods [11] [14] typically use a deep neural network to extract the vehicle visual representation.Some methods propose to enhance the feature representation by using multi-head module to extract multi-scale information, such as Huynh et at. [14]. In [9], [21], the separate use of detecting local regions and segmenting vehicles has been employed to obtain better local representations, thereby improving the overall network performance. However, the use of detection and segmentation networks may lead to a more cumbersome network, and all methods are not aware of the importance of triplets for training, simply employing traditional hard sample mining for triplet selection.

Thus in [8], author use a simple but effective triplet mining methods called Relation Perserving Triplet Mining(RPTM), which uses traditional matching algorithms to calculate matching values between all positive samples, and then selecting triplet positive samples based on these matching values.However, the entire training process is too fixed, and the triplets of images trained in each epoch are the same. This ignored the various complexities of data samples and the learning status of the current model. However, If the model is primarily trained on samples of moderate difficulty, it may lack robustness to extreme situations or edge cases. This could lead to poor model performance in real-world applications, especially when dealing with extreme difficulty or unusual scenarios.

Recently, the curriculum learning [5] approach has begun to make its way into the field of computer vision. Curriculum learning involves the model starting with simple samples and gradually increasing the sample difficulty through a difficulty regulator, which aligns with the daily learning pattern of humans, enabling the network to evolve from a relatively simple structure to a robust outcome gradually. However, solely training on samples from easy to hard does not yield satisfactory results.Therefore,we use the matching values obtained by traditional feature matching methods to measure the difficulty.Such as Figure 2, in the first stage, we train the triplet loss with positive samples that have larger matching values. As the training loss gradually converges, we reintroduce positive samples with moderate matching values, and finally proceed with training on the entire dataset.In addition, we added an attention mechanism head to address the issue where traditional networks fail to focus on specific details within an image.Finally, we employed a new adaptive adjuster for adjusting the loss weight for better performance. In summary, our paper contributions are:

1. We adopt curriculum learning [5] as a module adjusting the training difficulty of triplet positive samples to enhance the generalization capacity and convergence rate of models.

2. Multi-head with attention mechanism are attached to the backbone to help the model learn more detailed features. The features are then automatically grouped into sub-features, each help narrows down the search space of the target identity.

3. We use a novel adaptive loss weight to adjusting the ratio between Cross Entropy Loss and Triplet Loss.

2 Related works

2.1 Re-identification.

The demand for urban surveillance applications has led to s surge of interest in person and vehicle re-identification.Challenge benchmarks such as vehicleID [17], Veri-776 [18] and others [28] [19] have been established; and many new algorithms have been proposed.In reID, many algorithms achieve good results by estimating vehicular pose. Notably, Tang et al. [27] ceated a sythetic datat set for pose estimation and Meng et al. [22] used a parse model to split vehicles into four parts for pose-aware feature embedding. Recently, Vision Transformers (ViT) for reID [11] were proposed for attention learning. We suggest that the root problem encountered by most of these techniques lies in their definition of triplet loss.By replacing traditional triplet losses with our ETD technique, we show that it is possible to achieve state-of-the-art results by training process from simplicity to difficulty. This stands out from the trend towards ever more complex reID techniques.

2.2 Triplet Loss.

The triplet loss was first introduced in the context of face identification [26]. Since then, it has undergone many refinements [3]. Such triplet-based formulations implicitly assume that the given IDs correspond to meaningful groups.We suggest that this assumption is often wrong and that triplets should be defined with respect to naturally occurring groups rather than the given labels. This perspective on triplet loss differs significantly from that used in most papers. To our knowledge, the research most similar to ours is Ghosh et al. [8] who acknowledge the importance of naturally occurring groups within an ID.However, Ghosh et al. attempts to only use the images with moderate matching values, fighting rather than all the images.Another problems for triplet loss works like Ghosh et al. [8]'s, is that variations often have no naturally occurring cluster boundaries.This is not a problem for ETDNet which trains from easy to hard involves starting with simpler examples and gradually introducing more complex ones, mimicking the progressive learning order seen in human curricula, rather than simply using easy or difficult samples.

2.3 Curriculum Learning.

The original concept of CL is first proposed by Bengio at al. [5].In short, curriculum learning means "training from easier data to harder data".More specifically, the basic idea is to "start small",train the network with easier data subsets and then gradually increase the difficulty level of data until the whole training dataset. To begin with, from the perspective of optimization problem,Bengio et al. [5] initially point out that CL can be seen as a particular continuation method.Intuitively, continuation methods [1] are optimization strategies for non-convex criterion which first optimize a smoother version of the problem to reveal the "global picture",and then gradually consider less smoothing versions, until the target objective of interest.This strategy also shares

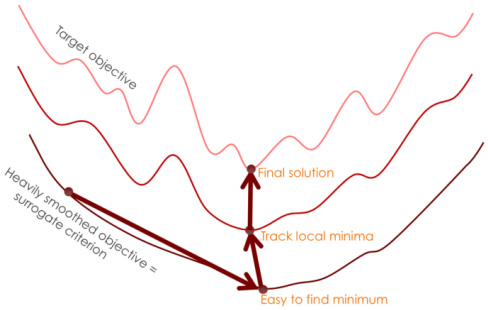

Figure 3. Illustration of the continuation method from [4], which is the essence of the CL [5]. It starts from optimizing a heavily smoothed version of the objective, and gradually moves to the target objective.Tracking the local minima throughout the training guides the model towards better parameter space and makes it more generalizable.

the same spirit with simulated annealing. As illustrated in Fig 3, continuation methods provide a sequence of optimization objectives, starting with a heavily smoothed objective for which it is easy to find a global minimum, and tracking the local minima throughout the training. In this way, continuation methods guide the training towards better regions in parameter space, as shown in Fig 3, the local minima learned from easier objectives have better generalization ability and are more likely to approximate global minima. Moreover, from the view of transfer learning, this continuation strategy can also be regarded as a sequence of unsupervised pretraining [5]: training on the preceding objectives could act as a pre-training process which both helps optimization and provides regularization on succeeding objectives.

3 Method

To prevent training pipelines from stagnating, it is important to implement a good triplet mining scheme. Triplet mining is part of a larger framework which views features as the key to machine learning. For example, NetVLAD [2] and many other domain transfer works, show that adapting features significantly improves performance. Somewhat similarly, knowledge distillation [23] tries to compress unwieldy networks into more compact features for practical deployment. We focus on triplet mining, as most related works in reID use some form of triplet loss and also to effectively highlight easy to difficult triplet mining.

Naively incorporating every possible triplet into the loss yields poor results [12]. Instead, training algorithms employ triplet-mining, a process which aims to incorporate only the most relevant triplets into the triplet-cost. Unfortunately, there is no consensus on how relevance can be measured; thus, triplet mining relies on heuristics. The two most popular heuristics are: hard-negative mining and semi-hard negative mining. Hard-negative mining focuses on triplets whose negatives are very similar to the anchor. Semi-hard negative mining shifts the focus from the hardest negatives to negatives close to the decision boundary. Both heuristics seem sensible and often perform well; however, closer inspection suggests something may be amiss.

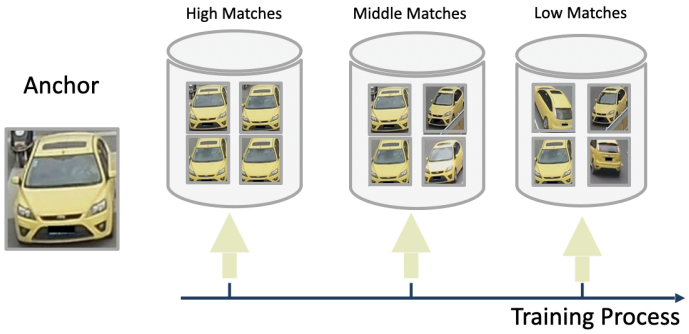

Figure 4. Training Process of Curriculum Learning.

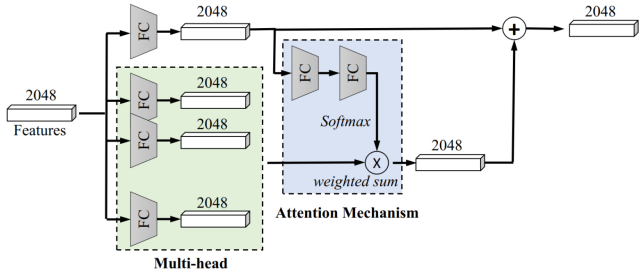

Figure 5. Multi-head with attention mechanism.

3.1 Curriculum Learning Triplet Mining

GMS [6] is a modern feature matcher that uses coherence to validate hypothesised feature matches.The coherence scheme assumes that a true match hypothesis will be strongly supported by many other match hyposthesis will be strongly supported by many other match hypothesis between neignbouring region pairs, while a false match hypotheses will not.The coherence-based validation is notably better than the traditional ratio test [20].This allows GMS to reliably match features across significant viewpoint changes while simultaneously ensuring few matches between image pairs with nothing in common. Thus,we use the GMS [6] to obtain the matching values between samples of a certain ID. These matching values are then used to assess the difficulty of the samples. The difficulty of the samples is adjusted by monitoring the variance of the loss for each epoch.Figure 4 show the traning process.Initially, training starts with samples having larger matching values. As the training progresses and the loss decreases, the difficulty of samples is adjusted based on the loss monitor, gradually increasing the difficulty until all samples are included.

3.2 Multi-head with Attention Mechanism.

Distinguishing thousands of vehicles with multiple views is challenging. Instead of using only one head, using multi-head encourages the re-id model to learn more diverse features from different vehicle characteristics. Thus, we adopted the multiples heads architecture [15] to further enhance the quality of the visual representation for vehicle re-id. Figure 5 shows the architecture of the multiple head with attention mechanism. In particular, the 2048-dim feature obtained from the backbone is fed into multiple parallel fully connected (FC) layers. Following [15], each FC layer is considered as one head and expected to learn distinct features which take into account different vehicle characteristics. Additionally, the attention mechanism determines which

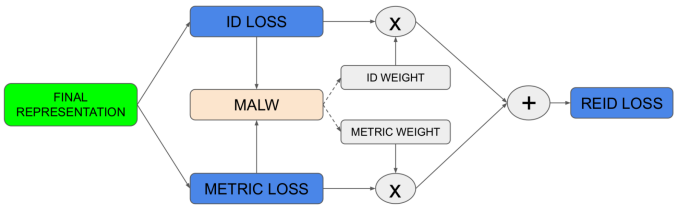

Figure 6. Momentum Adaptive Loss Weight.

head's features are more important to the final encoding feature.

3.3 Momentum Adaptive Loss Weight

Figure 6 describe how the MALW updates the weights during training progress. Let λ_{ID} and λ_{metric} be the loss weights for ID Loss and Metric Loss, respectively. Initially, the ratio between λ_{ID} and λ_{metric} is set to 1:1. After K iterations training, the ID loss weight λ_{ID} is updated based on the standard deviation of the recorded ID Loss L_{ID} and Metric Loss L_{Metric} with a momentum factor. The MALW method [14] improves our model performance by balancing the training losses without adding any computation cost to the inference step.

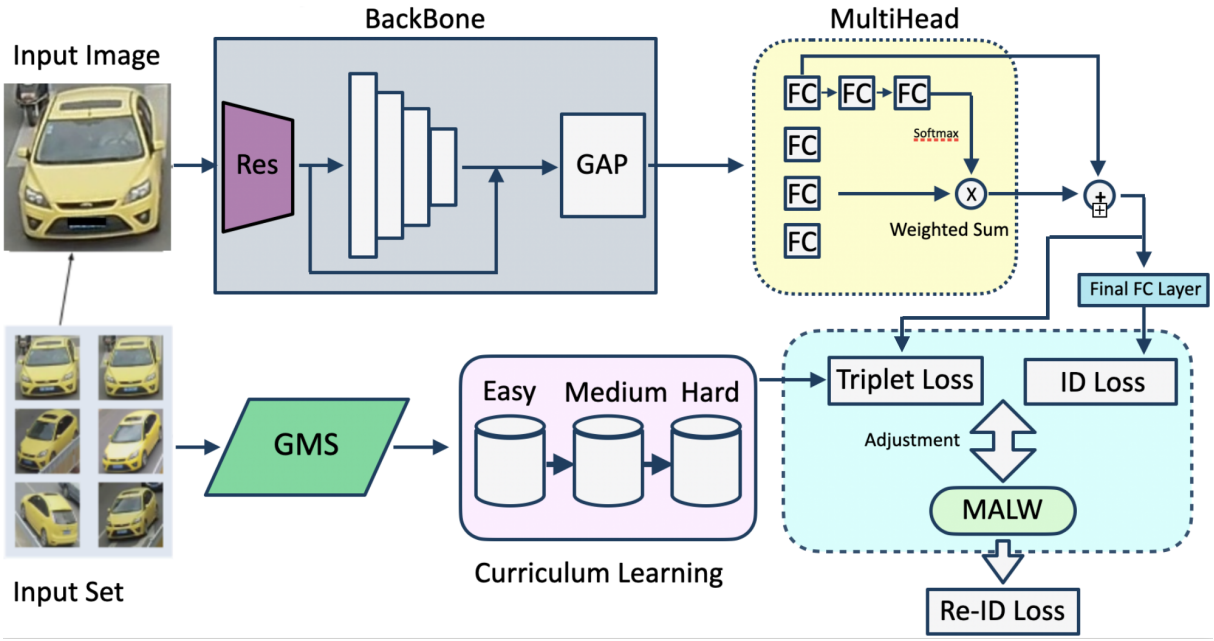

Figure 7. The structure of our method

4 Implementation details

A schematic of the network architecture is provided in Figure 7. In this section we discuss the model layout, first, we will introduce the comparison between released source codes in Section 4.1.Then we will elaborate on the comparative easy to difficult pipeline in Section 4.2 and the model structure with ETDVR in

Section 4.3.In the Section 4.4, we will introduce the main contributions.

4.1 Comparing with the released source codes

```python
class FC(nn.Module):
    def __init__(self, inplanes, outplanes):
        super(FC, self).__init__()
        self.fc = nn.Linear(inplanes, outplanes)
        self.bn = nn.BatchNorm1d(outplanes)
        self.act = nn.PReLU()

    def forward(self, x):
        x = self.fc(x)
        return self.act(x)
2 usages
class MultiHeads(nn.Module):
    def __init__(self, feature_dim=2048, groups=4, mode='S', backbone_fc_dim=2048):
        super(MultiHeads, self).__init__()
        self.mode = mode
        self.groups = groups
        # self.Backbone = backbone[resnet]
        self.instance_fc = FC(backbone_fc_dim, feature_dim)
        self.GDN = GDN(feature_dim, groups)
        self.group_fc = nn.ModuleList([FC(backbone_fc_dim, feature_dim) for i in range(groups)])
        self.feature_dim = feature_dim

    def forward(self, x):
        B = x.shape[0]
        # x = self.Backbone(x)  # (B,4096)
        instacne_representation = self.instance_fc(x)

        # GDN
        group_inter, group_prob = self.GDN(instacne_representation)
        # print(group_prob)
        # group aware repr
        v_G = [Gk(x) for Gk in self.group_fc]  # (B,512)

        # group ensemble
        group_mul_p_vk = list()
        if self.mode == 'S':
            for k in range(self.groups):
                Pk = group_prob[:, k].unsqueeze(dim=-1).expand(B, self.feature_dim)
                group_mul_p_vk.append(torch.mul(v_G[k], Pk))
            group_ensembled = torch.stack(group_mul_p_vk).sum(dim=0)
        # instance , group aggregation
        final = instacne_representation + group_ensembled
        return final

2 usages
class GDN(nn.Module):
    def __init__(self, inplanes, outplanes, intermediate_dim=256):
        super(GDN, self).__init__()
        self.fc1 = FC(inplanes, intermediate_dim)
        self.fc2 = FC(intermediate_dim, outplanes)
        self.softmax = nn.Softmax()

    def forward(self, x):
        intermediate = self.fc1(x)
        out = self.fc2(intermediate)
        # return intermediate, self.softmax(out)
        return intermediate, torch.softmax(out, dim=1)
```

```python
class ft_net_SE(nn.Module):

    def __init__(self, class_num, droprate=0.5, stride=2, pool='avg', init_model=None):
        super().__init__()
        model_name = 'se_resnext101_32x4d' # could be fbresnet152 or inceptionresnetv2
        # model_ft = pretrainedmodels.__dict__[model_name](num_classes=1000, pretrained='imagenet
        model_ft = se_resnext101_32x4d(num_classes=1000, pretrained='imagenet')

        if stride == 1:
            model_ft.layer4[0].conv2.stride = (1,1)
            model_ft.layer4[0].downsample[0].stride = (1,1)
        if pool == 'avg':
            model_ft.avg_pool = nn.AdaptiveAvgPool2d((1,1))
        elif pool == 'max':
            model_ft.avg_pool = nn.AdaptiveMaxPool2d((1,1))
        elif pool == 'avg+max':
            model_ft.avg_pool2 = nn.AdaptiveAvgPool2d((1,1))
            model_ft.max_pool2 = nn.AdaptiveMaxPool2d((1,1))
        else:
            print('UNKNOW POOLING!!!!!!!!!!!!!!!!!!!!!!!!!!!')
        #model_ft.dropout = nn.Sequential()
        model_ft.last_linear = nn.Sequential()
        self.model = model_ft
        self.pool  = pool
        # For DenseNet, the feature dim is 2048
        self.multihead = MultiHeads(feature_dim=2048, groups=8, mode="S", backbone_fc_dim=2048)
        if pool == 'avg+max':
            self.classifier = ClassBlock(4096, class_num, droprate)
        else:
            self.classifier = ClassBlock(2048, class_num, droprate)

    def forward(self, x):
        x = self.model.features(x)
        if self.pool == 'avg+max':
            v1 = self.model.avg_pool2(x)
            v2 = self.model.max_pool2(x)
            v = torch.cat((v1,v2), dim = 1)
        else:
            v = self.model.avg_pool(x)
        v = v.view(v.size(0), v.size(1))
        if not self.training:
            return v
        multi_v = self.multihead(v)
        v = self.classifier.add_block(v+multi_v)
        y = self.classifier.classifier(v)
        return y,v
```

Figure 8. Multi-head with attention mechanism.

```
if indexx > len(gms[labelx]) - 1:
    indexx = len(gms[labelx]) - 1
a = gms[labelx][indexx]
if cfg.MODEL.RPTM_SELECT == 'min':
    threshold = np.arange(10)
elif cfg.MODEL.RPTM_SELECT == 'mean':
    threshold = np.arange(np.amax(gms[labelx][indexx]) // 2)
elif cfg.MODEL.RPTM_SELECT == 'max':
    threshold = np.arange(np.amax(gms[labelx][indexx]))
else:
    threshold = np.arange(np.amax(gms[labelx][indexx]) // 2)  # defaults to mean
```

```
# if epoch%20 <= 20 and epoch%20 >16:
#     threshold = np.arange(np.amax(gms[labelx][indexx]) // 3)
# elif epoch%20 > 10 and epoch%20 <= 16:
#     threshold = np.arange(np.amax(gms[labelx][indexx]) // 1.5)
# elif epoch%20 <= 10:
#     threshold = np.arange(np.amax(gms[labelx][indexx]))
# else:
#     threshold = np.arange(np.amax(gms[labelx][indexx]) // 1.5)
# try:
#     masked_array = ma.masked_where(np.isin(a, threshold, invert=True), a)
#     valid_indices = np.where(~masked_array.mask)[0]
#     minpos = np.random.choice(valid_indices)  # 从没有遮罩的索引中随机选择一个
# except Exception:
#     minpos = np.random.choice(range(len(gms[labelx][indexx])))
```

Figure 9. Curriculum Learning Module

```
# 改动
#  L_ID = lambda_ID * xent_loss
#  L_metric = lambda_metric * htri_loss
#  S_ID.append(L_ID.detach().cpu().numpy())
#  S_metric.append(L_metric.detach().cpu().numpy())
#  if (iter % 500) == 0:
#      ID_std = np.std(list(S_ID))
#      metric_std = np.std(list(S_metric))
#      S_ID.clear()
#      S_metric.clear()
#      if ID_std > metric_std:
#          new_lambda_ID = 1 - (ID_std - metric_std) / ID_std
#          lambda_ID = 0.9 * lambda_ID + (1 - 0.9) * new_lambda_ID
#      print("此时LambdaID为: {}".format(lambda_ID))
#  iter += 1
#  loss = lambda_ID * htri_loss + lambda_metric * xent_loss
```

```
loss = cfg.LOSS.LAMBDA_HTRI * htri_loss + cfg.LOSS.LAMBDA_XENT * xent_loss
```

Figure 10. Momentum Adaptive Loss Weight

The code on the left of Figure 8 represents the implementation of the multi-head attention mechanism. The neural network from the original paper is shown on the right side of Figure 1, with the modifications highlighted in red. Figure 9 shows the code for the curriculum learning module and the original code. I have marked the modified code with red boxes.The figure 10 shows the code and source code for Momentum Adaptive Loss Weight. The modified code is also highlighted in red boxes.

4.2 Neural Network

For fair comparison of our results with established benchmarks, we chose ResNet-50 and ResNet-101 pretrained on ImageNet as our backbone. Our ETDVR module includes instance-batch-normalization and a squeeze-excitation layer [13]. The weights of this network are trained by minimising the loss function in Eq. 1. This network is trained using triplets defined through our Relation Preserving Triplet Mining (RPTM) in Section 4.2.

$$E = \lambda_{ent} E_{ent} + \lambda_{tri} E_{tri} \tag{1}$$

The images are resized to (240,240) for vehicle reID and (300,150) for person reID. Data augmentation is applied, with random flipping, random padding, random erasing and colour jitter (randomly changing contrast, brightness, hue and saturation) all activated. Stochastic Gradient Descent(SGD) is used as the optimiser for the model. The initial learning rate is initialised at 0.005 and is set to decay by a factor of 0.1 every 20 epochs. The model is trained for 80 epochs with a batch size of 24. Training parameters are fixed for all datasets.

4.3 Main contributions

1. We utilize GMS to detect matching values among all samples under a specific ID, evaluating the difficulty of samples. This facilitates the curriculum learning process. We start training with samples that have higher matching values, gradually increasing sample difficulty as the loss converges. This process continues until training on the complete dataset. Through curriculum learning [5] triplet mining, it reduces the network's dependency on specific sample types, preventing overfitting, and significantly improves the network's generalization ability, achieving better re-identification results.

2. Analyzing the visualization results of the original network, we observe that the original network, due to triplet training only on samples with moderate matching values, faces extreme difficulty in recognizing challenging samples. By incorporating a multi-head attention mechanism [15] into the network, the overall system improves its ability to identify local regions, resulting in better representations. This enhancement contributes to an overall improvement in the network's robustness.

3. Conventionally, the loss weights are set equally,i.e. 1:1 ratio.However, in practice, ID Loss is relatively much larger than Metric Loss, which causes the imbalance and affects the training performance.So, we employed a small trick to adjust the relationship between ID loss and triplet loss,called Momentum Adaptive Loss Weight (MALW).

5 Results and analysis

Figure 11. Heatmap Analysis

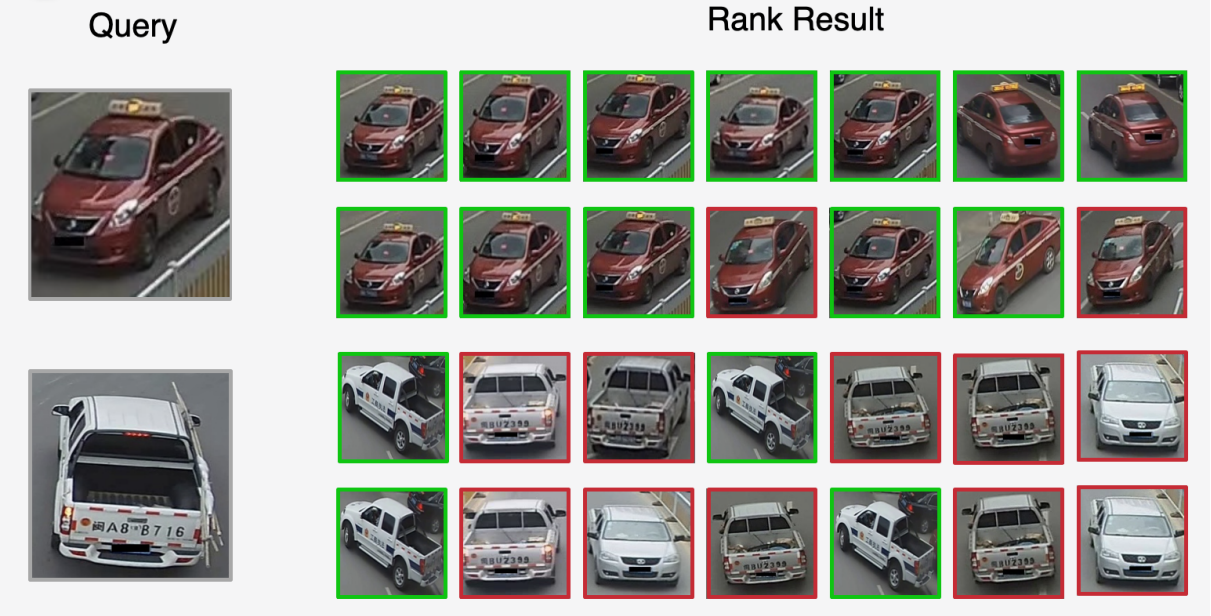

Figure 12. Ranking Results Analysis

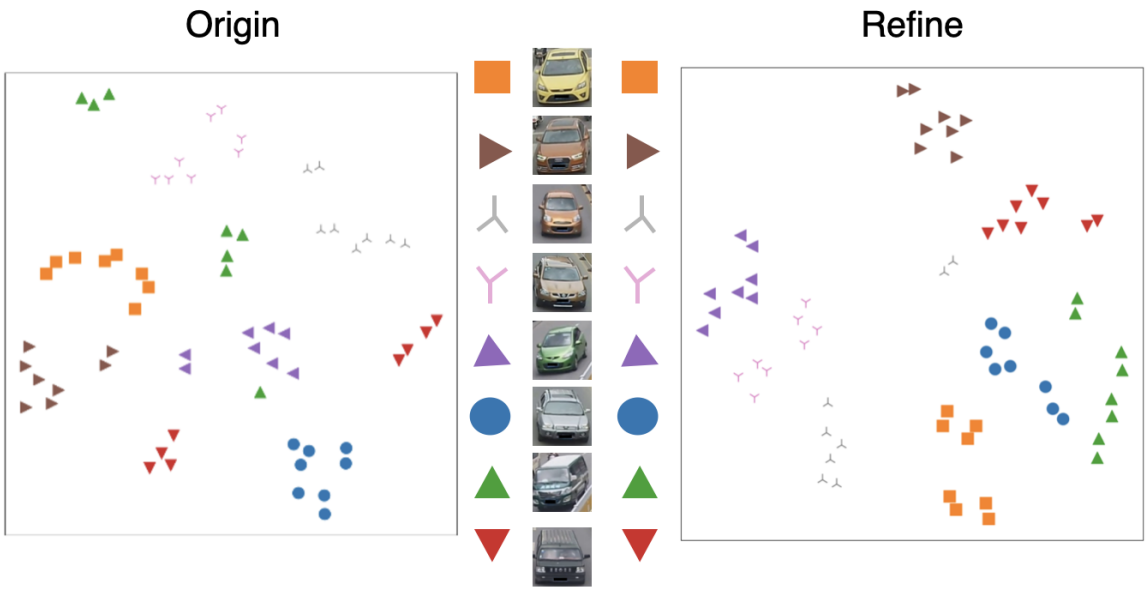

Figure 13. two-Dimensional Feature Vector Analysis

By analyzing the heat map in Figure 11, we can clearly see that the previous network focuses more on aspects such as network architecture, while the improved network pays more attention to reflective content. Enhancing local perception capabilities contributes to improving the overall re-identification ability of the network. Furthermore, we can obtain the rank analysis results in Figure 12. Obviously, for small cars and similar vehicles, our model performs better in re-identification tasks due to its enhanced ability to observe details. However, for trucks in this category, neither model achieves satisfactory results. This could be attributed to the scarcity of data for such vehicles, making it challenging for the models to learn the correct focal points.Finally, we performed dimensionality reduction on the image features using t-SNE. The resulting images, as shown in the Figure 13, demonstrate that the improved feature vectors exhibit enhanced clustering capabilities.

6 Conclusion and future work

6.1 Conclusion

In this work, we have shown that the easy to difficult method can help significantly improve triplet mining, not only can fully utilize the training dataset but also, consequently, creating a more smooth optimisation procedure that leads to better generalisation.To that end, we introduced Easy to Difficult Vehicle Re-identification (ETDVR) method, a network improved by leveraging the matching values between positive samples for traditional triplet mining.We showed how feature matches could be used to judge the difficulty of a sample, leading to a better conditioned triplet loss, creating feature learners with enhanced training stability.Morevover, we highlighted that ETDVR outperforms recent reID methods while .Finally, we believe our research can be extended to Unsupervised Domain Adaptation for even better scalabilty.

6.2 Future work

Integrated Structure for Detection, Tracking, and Re-identification Currently, re-identification often involves using detected vehicle bounding boxes to further calculate the similarity of vehicle images. In tracking, both the detected bounding boxes and re-identified similarity scores are required. The lack of an integrated structure, such as utilizing the feature vectors obtained during detection for tracking (e.g., YOLO [25] captures feature vectors corresponding to detected bounding boxes), poses significant challenges in accurate tracking and re-identification.

Due to the scarcity of vehicle images from different perspectives in existing datasets and the limited variety of vehicle types, developing accurate vehicle re-identification models can be challenging. Generative Adversarial Models (GANs) serve as a powerful deep learning technique for generating synthetic data, including vehicle images from different viewpoints, offering a potential solution to the challenges in vehicle re-identification. Researchers have focused on unsupervised GANs, such as CycleGAN [29], for style transfer between two classes of photos, with style transformations like day-to-night aiding in better feature extraction for re-identification models, thus improving accuracy in vehicle re-identification tasks. However, addressing the issue of imbalanced dataset samples in vehicle re-identification using GANs, especially given the lower image quality produced by CycleGAN, remains a challenge.

With the recent popularity of chatGPT [24], researchers in computer vision have shifted focus towards applying unsupervised training methods from natural language processing to computer vision. This has led to the development of models like Vision Transformer [7], Clip [16], and MoCo [10]. Vision Transformer divides images into smaller patches, treating them as a sequence for transformer processing. Clip aligns text with corresponding images and uses contrastive learning to bring text describing an image closer and text not from the image further away, enabling models to train without relying on image annotations. MoCo, following a similar contrastive learning approach, brings closer the representations of the same image under different augmentations and pushes apart representations of different images.

References

[1] Eugene L Allgower and Kurt Georg. *Numerical continuation methods: an introduction*, volume 13. Springer Science & Business Media, 2012.

[2] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307, 2016.

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

[4] Yoshua Bengio. Evolving culture versus local minima. In *Growing adaptive machines: Combining development and learning in artificial neural networks*, pages 109–138. Springer, 2014.

[5] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[6] JiaWang Bian, Wen-Yan Lin, Yasuyuki Matsushita, Sai-Kit Yeung, Tan-Dat Nguyen, and Ming-Ming Cheng. Gms: Grid-based motion statistics for fast, ultra-robust feature correspondence. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4181–4190, 2017.

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. arxiv 2020. *arXiv preprint arXiv:2010.11929*, 2010.

[8] Adhiraj Ghosh, Kuruparan Shanmugalingam, and Wen-Yan Lin. Relation preserving triplet mining for stabilising the triplet loss in re-identification systems. *arXiv preprint arXiv:2110.07933*, 2021.

[9] Bing He, Jia Li, Yifan Zhao, and Yonghong Tian. Part-regularized near-duplicate vehicle re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3997–4005, 2019.

[10] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[11] Shuting He, Hao Luo, Pichao Wang, Fan Wang, Hao Li, and Wei Jiang. Transreid: Transformer-based object re-identification. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 15013–15022, 2021.

[12] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

[13] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[14] Su V Huynh. A strong baseline for vehicle re-identification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4147–4154, 2021.

[15] Yonghyun Kim, Wonpyo Park, Myung-Cheol Roh, and Jongju Shin. Groupface: Learning latent groups and constructing group-based representations for face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5621–5630, 2020.

[16] Yangguang Li, Feng Liang, Lichen Zhao, Yufeng Cui, Wanli Ouyang, Jing Shao, Fengwei Yu, and Junjie Yan. Supervision exists everywhere: A data efficient contrastive language-image pre-training paradigm. *arXiv preprint arXiv:2110.05208*, 2021.

[17] Hongye Liu, Yonghong Tian, Yaowei Yang, Lu Pang, and Tiejun Huang. Deep relative distance learning: Tell the difference between similar vehicles. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2167–2175, 2016.

[18] Xinchen Liu, Wu Liu, Tao Mei, and Huadong Ma. A deep learning-based approach to progressive vehicle re-identification for urban surveillance. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 869–884. Springer, 2016.

[19] Yihang Lou, Yan Bai, Jun Liu, Shiqi Wang, and Lingyu Duan. Veri-wild: A large dataset and a new method for vehicle re-identification in the wild. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3235–3243, 2019.

[20] David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee, 1999.

[21] Dechao Meng, Liang Li, Xuejing Liu, Yadong Li, Shijie Yang, Zheng-Jun Zha, Xingyu Gao, Shuhui Wang, and Qingming Huang. Parsing-based view-aware embedding network for vehicle re-identification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7103–7112, 2020.

[22] Dechao Meng, Liang Li, Xuejing Liu, Yadong Li, Shijie Yang, Zheng-Jun Zha, Xingyu Gao, Shuhui Wang, and Qingming Huang. Parsing-based view-aware embedding network for vehicle re-identification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7103–7112, 2020.

[23] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3967–3976, 2019.

[24] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[25] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[26] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[27] Zheng Tang, Milind Naphade, Stan Birchfield, Jonathan Tremblay, William Hodge, Ratnesh Kumar, Shuo Wang, and Xiaodong Yang. Pamtri: Pose-aware multi-task learning for vehicle re-identification using highly randomized synthetic data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 211–220, 2019.

[28] Zheng Tang, Milind Naphade, Ming-Yu Liu, Xiaodong Yang, Stan Birchfield, Shuo Wang, Ratnesh Kumar, David Anastasiu, and Jenq-Neng Hwang. Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8797–8806, 2019.

[29] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.