DAS: Densely-Anchored Sampling for Deep Metric Learning (Supplementary Materials)

Lizhao Liu[1,2], Shangxin Hunag[1], Zhuangwei Zhuang[1],
Ran Yang[1], Mingkui Tan[1,3†], and Yaowei Wang[2†]

[1]South China University of Technology [2]PengCheng Laboratory
[3]Key Laboratory of Big Data and Intelligent Robot, Ministry of Education
{selizhaoliu,sevtars,z.zhuangwei,msyangran}@mail.scut.edu.cn,
mingkuitan@scut.edu.cn, wangyw@pcl.ac.cn

We organize our supplementary materials as follows. In Section A, we provide the detailed formulations of both pair-based and proxy-based DML loss functions. In Section B, we detail the formulations of DML sampling methods. In Section C, we provide more implementation details of DAS. In Section D, we analyze the overhead of DAS, In Section E, we provide experiment results of DAS on widely used proxy-based losses. In Section F, we provide results on DAS w/o image augmentation. In Section G, we study the effect of batch size on DAS. In Section H, we study the effect of embedding dimension on DAS. In Section I, we visualize and analyze the frequency recorder matrix. In Section J, we provide the evolution of training process w.r.t. more DML losses. In Section K, we investigate the effect of hyper-parameters r_s, r_b, T. In Section L, we provide qualitative results w.r.t. DFS and MTS. In Section M, we provide more qualitative results on different loss functions.

A Detailed Formulations of Loss Function in DML

A.1 Pair-based Loss Function

Contrastive Loss [2]. The goal of contrastive loss is simply pulling the embeddings of the same class as close as possible and separating the embeddings of different classes at least of a given margin. Specifically, contrastive loss requires the index set of the sampled embedding pairs $\mathcal{P} = \{(i,j)\}$ and the pair-wise euclidean distance is calculated as $\mathbf{D}_{ij} = \|\mathbf{v}_i - \mathbf{v}_j\|$. Then the formulation of contrastive loss is as follows

$$\mathcal{L}_{\text{Contrastive}} = \sum_{(i,j)\in\mathcal{P}} \mathbb{I}\{y_i = y_j\}\ \mathbf{D}_{ij} + \mathbb{I}\{y_i \neq y_j\}\ [\gamma - \mathbf{D}_{ij}]_+ \,, \qquad \text{(I)}$$

where $\mathbb{I}\{\cdot\}$ is the indicator function, γ (set to 1.0 in this paper) is the margin.

Triplet Loss [8]. Triplet loss extends the contrastive loss by converting the absolute distance relationship between embeddings into a relative distance relationship (*i.e.,* ranking): the distance between embeddings of different classes

[†] Corresponding authors.

should be farther away than any embeddings of the same class. Specifically, triplet loss requires sampling a set of embedding triplets $\mathcal{T} = \{(a, p, n)\}$, where $y_a = y_p \neq y_n$ and a, p, n are the index of the anchor, positive and negative, respectively. The formulation of triplet loss is as follows

$$\mathcal{L}_{\text{Triplet}} = \sum_{(a,p,n)\in\mathcal{T}} \left[\mathbf{D}_{ap} - \mathbf{D}_{an} + \gamma\right]_+ , \tag{II}$$

where γ (set to 0.2 in this paper) is the margin.

Margin Loss [11]. Margin loss introduces a more flexible optimization paradigm into the triplet loss. Specifically, a adjustable and learnable margin $\boldsymbol{\beta} \in \mathbb{R}^C$ is proposed to replace the fixed margin (*i.e.,* 0) between embedding of different classes, which converts the triplet ranking problem into a relative ordering of pairs. The formulation of margin loss is as follows

$$\mathcal{L}_{\text{Margin}} = \sum_{(i,j)\in\mathcal{P}} \mathbb{I}\{y_i = y_j\} \left[\gamma + \mathbf{D}_{ij} - \boldsymbol{\beta}_{y_i}\right]_+ + \mathbb{I}\{y_i \neq y_j\} \left[\gamma + \boldsymbol{\beta}_{y_i} - \mathbf{D}_{ij}\right]_+ , \tag{III}$$

where γ (set to 0.2 in this paper) is the margin in the triplet loss and $\boldsymbol{\beta}_{y_i}$ is the learnable margin for class y_i. Each element in $\boldsymbol{\beta}$ is initialized with 1.2 and the learning rate for $\boldsymbol{\beta}$ is set to 5e^{-4}.

Generalized Lifted Structure Loss [3]. Generalized lifted structure loss extends the standard lifted structure loss [6] by considering all embeddings from the same class w.r.t. the anchor during intra-class distance minimization. Generalized lifted structure loss pulls embeddings of the same class w.r.t. the anchor close while pushing embeddings of different classes apart. To save computation cost, each embedding in a batch is used as the anchor once. To be specific, the index set of the sampled embeddings is $\mathcal{P} = \{(a, \mathcal{Q}, \mathcal{R})\}$, where $a \notin \mathcal{Q}, \mathcal{R}$, and $y_a = y_q \neq y_r, q \in \mathcal{Q}, r \in \mathcal{R}$. Then the formulation of generalized lifted structure loss is as follows

$$\mathcal{L}_{\text{GenLifted}} = \sum_{(a,\mathcal{Q},\mathcal{R})\in\mathcal{P}} \left[\log \sum_{q\in\mathcal{Q}} \exp\left(\mathbf{D}_{aq}\right) + \log \sum_{r\in\mathcal{R}} \exp\left(\gamma - \mathbf{D}_{ar}\right)\right]_+ + \nu \left\|\mathbf{v}_a\right\|^2 , \tag{IV}$$

where γ (set to 1.0 in this paper) is the margin to avoid pushing the embeddings of different classes too large and ν (set to 5e^{-3} in this paper) regularizes the embeddings. Note that, in this loss, embeddings for distance computation and producing embeddings with no data points are not normalized.

N-Pair Loss [9]. N-Pair loss extends the triplet loss by considering all embeddings of different classes during inter-class distance maximization. Specifically, the index set of the sampled embeddings is $\mathcal{P} = \{(a, p, \mathcal{R})\}$, where $y_a = y_p \neq y_r, r \in \mathcal{R}$, and the pair-wise distance is calculated as $\mathbf{D}_{ij} = \mathbf{v}_i^T \mathbf{v}_j$. Then the formulation of N-Pair loss is as follows

$$\mathcal{L}_{\text{N-Pair}} = \sum_{(a,p,\mathcal{R})\in\mathcal{P}} \log\left(1 + \sum_{r\in\mathcal{R}} \exp\left(\mathbf{D}_{ar} - \mathbf{D}_{ap}\right)\right) + \nu \left\|\mathbf{v}_a\right\|^2, \tag{V}$$

where ν (set to 5e^{-3} in this paper) controls the optimization strength on the embedding regularization. Note that, in this loss, embeddings for distance computation and producing embeddings with no data points are also not normalized.

Multi-similarity Loss [10]. Apart from considering simple anchor-positive, anchor-negative relationships, multi-similarity loss better leverages all embeddings in a batch by additionally considering positive-positive and negative-negative relationship. Also, to save computation cost, each embedding in a batch will only be used as anchor once. For a anchor a, let $\mathcal{P}_a$ and $\mathcal{N}_a$ denote its corresponding positive and negative embedding index sets, respectively. Given the pair-wise distance computed by $\mathbf{D}_{ij} = \mathbf{v}_i^T\mathbf{v}_j$, the sampled embedding index set is constructed as $\mathcal{P} = \{(a, \mathcal{Q}, \mathcal{R})\}$, where $a \notin \mathcal{Q}, \mathcal{R},\ \ \mathcal{Q} = \{q \ \mid \ y_q = y_a,\ \ \mathbf{D}_{aq} > \min_{i\in\mathcal{P}_a}\left(\mathbf{D}_{ai} - \epsilon\right)\}$ and $\mathcal{R} = \{r \ \mid \ y_r \neq y_a,\ \ \mathbf{D}_{ar} < \max_{j\in\mathcal{N}_a}\left(\mathbf{D}_{aj} + \epsilon\right)\}$. Then the formulation of multi-similarity loss is as follows

$$\begin{aligned} \mathcal{L}_{\text{MS}} = \sum_{(a,\mathcal{Q},\mathcal{R})\in\mathcal{P}} & \frac{1}{\alpha}\log\left[1 + \sum_{q\in\mathcal{Q}} \exp\left(-\alpha\left(\mathbf{D}_{aq} - \lambda\right)\right)\right] \\ & + \frac{1}{\beta}\log\left[1 + \sum_{r\in\mathcal{R}} \exp\left(\beta\left(\mathbf{D}_{ar} - \lambda\right)\right)\right], \end{aligned} \tag{VI}$$

where $\alpha, \beta, \lambda, \epsilon$ are hyper-parameters to be set. In this paper, we set $\alpha = 2, \beta = 40, \lambda = 5\text{e}^{-1}, \epsilon = 1\text{e}^{-1}$.

A.2 Proxy-based Loss Function

Softmax Loss [12]. Different from the pair-based loss function, softmax loss1 introduces a proxy *i.e.,* classifier for each class and optimizes the embedding by pulling it close to its proxy. The formulation of softmax loss is as follows:

$$\mathcal{L}_{\text{Softmax}} = -\sum_i \log \frac{\exp\left(\mathbf{W}_{y_i}^T\mathbf{v}_i \ / \ T\right)}{\sum_{c\in C}\exp\left(\mathbf{W}_c^T\mathbf{v}_i \ / \ T\right)}, \tag{VII}$$

where $\mathbf{W} \in \mathbb{R}^{C\times d}$ is the classifier weight for all training classes. Since the embedding $\mathbf{v}_i$ is normalized, a temperature T (set to 5e^{-2} in this paper) is used to boost the gradient. Moreover, the learning rate of $\mathbf{W}$ is set to 1e^{-5} for CARS and CUB, 2e^{-3} for SOP.

ArcFace Loss [1]. ArcFace loss improves the vanilla softmax by adding an angular margin into embedding and its corresponding proxy to achieve more compact intra-class representation. The formulation of ArcFace loss is as follows

$$\mathcal{L}_{\text{ArcFace}} = -\sum_i \log \frac{\exp\left(s \cdot \cos\left(\mathbf{W}_{y_i}^T \mathbf{v}_i + \gamma\right)\right)}{\exp\left(s \cdot \cos\left(\mathbf{W}_{y_i}^T \mathbf{v}_i + \gamma\right)\right) + \sum_{c \neq y_i} \exp\left(s \cdot \cos\left(\mathbf{W}_c^T \mathbf{v}_i\right)\right)}, \tag{VIII}$$

where $\mathbf{W} \in \mathbb{R}^{C \times d}$ is the classifier weight for all training classes and γ, s are hyperparameters to be set. In this paper, we set $\gamma = 5\text{e}^{-1}$ and $s = 16$. Moreover, the learning rate of $\mathbf{W}$ is set to $5\text{e} - 4$ for all datasets.

B Detailed Formulations of Sampling Method for DML

Random Sampling [4]. Random sampling simply selects the index of positive pair or negative pair in a most trivial way *i.e.,* randomly selecting. To be specific, given an embedding $\mathbf{v}_i$, its index of positive is randomly draw from $\{j \mid y_i = y_j, i \neq j\}$ and its index of negative is randomly draw from $\{k \mid y_i \neq y_k, i \neq k\}$.

Semi-hard Sampling [8]. Semi-hard sampling is proposed to effectively sample embedding triplets that grows cubically to batch size. In the training process, most of the triplets satisfy the objective function and they provide limited (or no) training signal to train the model, thereby impeding the model learning [8]. Thus, given an anchor $\mathbf{v}_a$ and its positive $\mathbf{v}_p$ (randomly sampled), semi-hard sampling carefully choose negative embedding's index as follows

$$n \sim \{i \mid y_i \neq y_a, \|\mathbf{v}_a - \mathbf{v}_p\|^2 < \|\mathbf{v}_a - \mathbf{v}_i\|^2\}. \tag{IX}$$

Soft-hard Sampling [7] To avoid selecting "hard" embeddings that impedes model training, semi-hard sampling chooses embeddings that are relatively close to the anchor. Soft-hard triplet sampling shows that a probabilistic (soft) selection of potentially hard embeddings is actually beneficial. Given an anchor embedding $\mathbf{v}_a$, soft-hard sampling attain the indexes of positive and negative embedding as follows

$$p \sim \{i \mid y_i = y_a, \|\mathbf{v}_a - \mathbf{v}_i\|^2 > \operatorname*{arg\,min}_{q \in \mathcal{Q}_a} \|\mathbf{v}_a - \mathbf{v}_q\|^2\}, \tag{X}$$

$$n \sim \{j \mid y_j \neq y_a, \|\mathbf{v}_a - \mathbf{v}_j\|^2 < \operatorname*{arg\,max}_{r \in \mathcal{R}_a} \|\mathbf{v}_a - \mathbf{v}_r\|^2\}, \tag{XI}$$

where $\mathcal{R}_a = \{r \mid y_r \neq y_a\}$, $\mathcal{Q}_a = \{q \mid y_q \neq y_a\}$ are the positive and negative index sets w.r.t. the anchor a, respectively. In this way, soft-hard sampling explores more triplets than semi-hard sampling to improve the model training.

Distance-weighted Sampling [11] . Different from other sampling strategy that considers a certain distance range of embeddings, distance-weighted sampling considers a wide range of embeddings in a probabilistic way. Since the embedding space is typically a d-dimensional hypersphere $\mathbb{S}^{d-1}$, the analytical distribution of pairwise distance on a hypersphere obeys

$$q\left(\mathbf{D}_{ij}\right) \propto \mathbf{D}_{ij}^{d-2}[1-\frac{1}{4}\mathbf{D}_{ij}]^{\frac{d-3}{2}}, \tag{XII}$$

and $\mathbf{D}_{ij} = \|\mathbf{v}_i - \mathbf{v}_j\|$ for any embedding pairs $\mathbf{v}_i, \mathbf{v}_j \in \mathbb{S}^{d-1}$. To obtain a wide range of negative embeddings that are able to improve the embedding diversity as well as model training, distance-weighted sampling acquires the index of negative embedding based on the inversed distance distribution

$$P(n \mid a) \propto \min\left(\lambda, q^{-1}(\mathbf{D}_{an})\right). \tag{XIII}$$

In this paper, we set $\lambda = 5\mathrm{e}^{-1}$ and the largest distance to 1.4.

C More Implementation Details

In this section, we provide more implementation details. As for image augmentation process, random crop (image size 224×224) with random horizontal flip ($p = 0.5$) is applied during training and single center crop (image size 256×256) is used for testing. In terms of training strategy, the number of training epochs is 300. We use Adam [5] as the optimizer. The initial learning rate is $1\mathrm{e}^{-5}$, which is reduced by a factor of 0.3 in 200th and 250th epoch, respectively. The weight decay is $4\mathrm{e}^{-4}$. For batch preparation, SPC-2 construction [7] is used (2 samples per category). The batch size is set to 112.

D Efficiency and Overhead Analysis

DAS takes extra cost only in the training stage. Specifically, w/ and w/o DAS, the training time cost for [11] are 1.15s *vs.* 0.70s per batch, which includes the cost of DAS and using more embeddings for sampling and loss computation. Moreover, DAS only consumes 13% of the total time, which is efficient compared to the whole training procedure.

E Effectiveness of DAS on Proxy-based Loss

Although DAS is developed for pair-based loss, we perform experiments to evaluate the generalization ability of DAS on classic and widely used proxy-based losses *i.e.,* Softmax and ArcFace. The results are presented in Table I. The improvements are still observed when equipped with DAS for Softmax and ArcFace across different datasets.

F DAS w/o Image Augmentation

We further perform experiments without image augmentation using triplet loss and distance weighted sampling on CARS. The results are shown in Table II. DAS boosts all metrics considerably, showing that DAS is complementary to image augmentation technique.

Table I: Comparisons with proxy-based approaches on various datasets

Method	CUB			CARS			SOP		
	R@1	F1	NMI	R@1	F1	NMI	R@1	F1	NMI
Softmax [12]	61.58	36.12	66.73	79.07	37.11	67.01	77.92	37.20	90.05
Softmax + DAS	**62.02**	**36.24**	**67.42**	**81.23**	**39.95**	**68.91**	**79.36**	**38.72**	**90.40**
ArcFace [1]	61.56	35.73	66.83	79.50	37.75	67.82	78.08	37.79	90.18
ArcFace + DAS	**62.80**	**37.63**	**67.80**	**82.22**	**40.82**	**69.82**	**78.12**	**38.08**	**90.26**

Table II: Comparisons without image augmentation on CARS

DAS	R@1	F1	NMI
	61.47	22.06	53.88
✓	65.13 (+**3.66**)	23.77 (+**1.71**)	55.89 (+**2.01**)

G Effect of Batch Size

In this section, we investigate the effect of batch size on the proposed DAS. The results are presented in Fig. I. The loss function and sampling method are margin loss [11] and distance-weighted sampling [11], respectively. From Fig. I, we have the following observations: **First**, under various batch size and image retrieval evaluation metrics, when equipped with DAS, the model is able to consistently obtain better results than the one trained without DAS. **Second**, we observe that the model trained with DAS and batch size = 32 outperforms the one trained without DAS and batch size = 224 in terms of R@1. It shows that producing effective embeddings without datapoints by DAS is as equally important as providing more data points in a batch to achieve improved performance. These results well prove the rationality of our motivation and the efficacy of DAS.

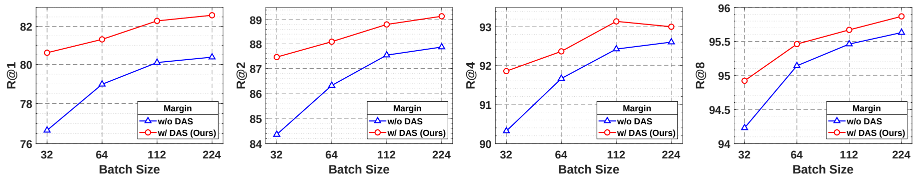
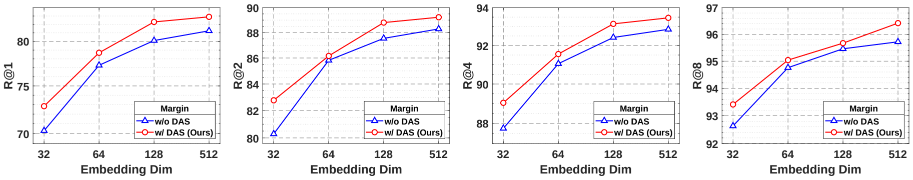


Fig. I: The test set R@$\{1, 2, 4, 8\}$ on CARS with different batch size

H Effect of Embedding Dimension

In this section, we evaluate the proposed method on different embedding dimensions. The results are shown in Fig. II. We use the margin loss [11] as the loss function while leveraging the distance-weighted sampling as the sampling method [11]. From Fig. II, we obtain the following results: **First**, under different embedding dimension, DAS consistently reaches the best performance for all image retrieval metrics. **Second**, the model that trained with DAS and embedding dimension = 64 obtains a comparable result like the one trained without DAS and embedding dimension = 128 regarding R@1. It shows that the produced embeddings by DAS are able to force the model to better leverage the model capacity. And covering the barren area in embedding space is important to get improved performance when model capacity is low. These results demonstrate the effectiveness of DAS.

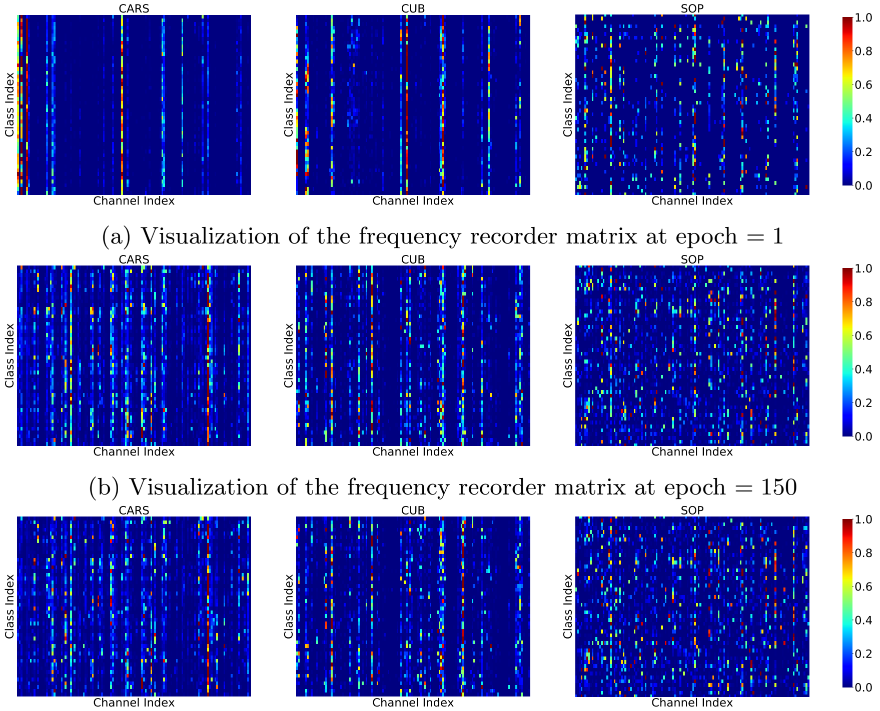


(a) Experiments using different embedding dim.

Fig. II: Test set R@$\{1, 2, 4, 8\}$ on CARS with different embedding dimension d

I Visualization Results on Frequency Recorder Matrix

In this section, we visualize the Frequency Recorder Matrix (FRM) **P** introduced in the DFS module. The FRM serves as a stable and effective identifier for semantic scaling by considering the top activated features for one class as the effective semantics instead of noises. The loss function and sampling method we used here are triplet loss [8] and distance-weighted sampling [11], respectively. We perform experiments on all three datasets (*i.e.,* CARS, CUB and SOP). The results are depicted in Fig. III, from which, we have the following observations: **First**, for different training stages (*i.e.,* epoch $= 1, 150, 300$), the number of top activated features for embeddings of the same classes are limited (*i.e.,*around $4 \sim 8$) across all datasets. **Second**, as the training process proceeds, more features are likely to be the top activated features. **Third**, for the large scale dataset SOP, more features are likely to be the top activated ones due to the rich semantics covered by adequate data points. In this sense, with the proposed FRM, we are able to figure out channels with more discriminative power to achieve effective semantic scaling. These results demonstrate the rationality of the proposed FRM.



(a) Visualization of the frequency recorder matrix at epoch = 1

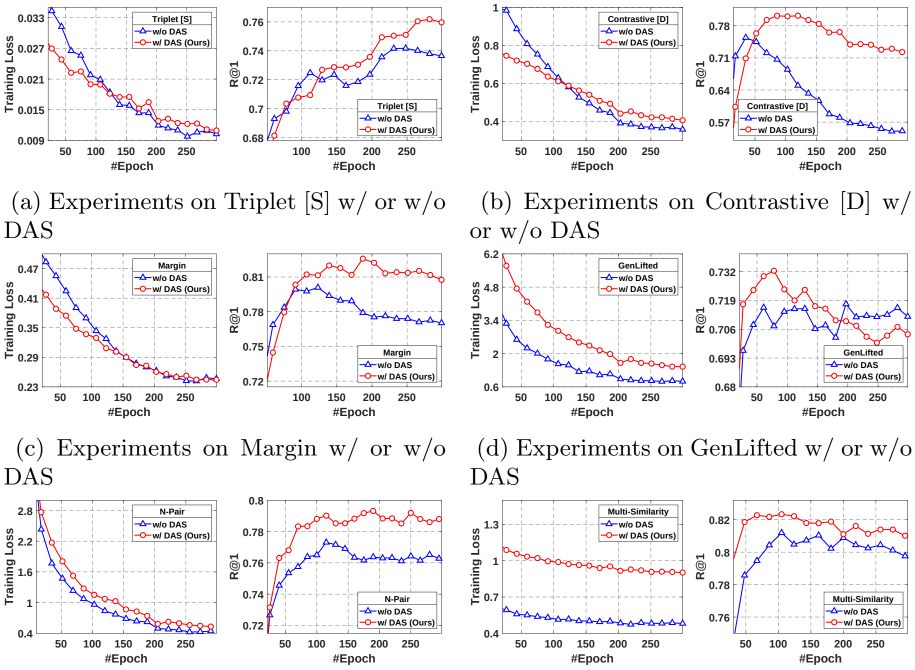
(b) Visualization of the frequency recorder matrix at epoch = 150

(c) Visualization of the frequency recorder matrix at epoch = 300

Fig. III: From left to right, the visualized FRM on CARS, CUB and SOP, respectively. Each element in $\mathbf{P}$ is normalized (*i.e.,* divided by the maximum value in its row). Only the first 48 classes are presented due to page limit

J Evolution of Training Process w.r.t. Different Losses

In this section, we provide the evolution of training loss and test set R@1 w.r.t. different losses in the training process. The results are depicted in Fig. IV. We have the following observations: **First**, when training with DAS, the training losses are generally higher and decrease smoother than the baseline, which demonstrates that producing more embeddings by DAS is able to consistently provide training signal to train the model. **Second**, when equipped with DAS, the test set R@1s are higher than the baseline. **Third**, for some loss functions that face severe overfitting problems such as contrastive loss and generalized lifted structure loss, DAS is able to ease the overfitting problem. These results verify the effectiveness of the proposed method across different loss functions and sampling methods.



(a) Experiments on Triplet [S] w/ or w/o DAS

(b) Experiments on Contrastive [D] w/ or w/o DAS

(c) Experiments on Margin w/ or w/o DAS

(d) Experiments on GenLifted w/ or w/o DAS

(e) Experiments on N-Pair w/ or w/o DAS

(f) Experiments on MS w/ or w/o DAS

Fig. IV: The training loss and test set R@1 on CARS with different losses

K Ablation Studies on Hyper-parameters

In this section, we investigate the effect of the hyper-parameters r_s, r_b, T in DAS. The loss function and sampling method are margin loss [11] and distance-weighted sampling [11], respectively. The default hyper-parameters' settings are $(r_s, r_b, T) = (1\text{e}^{-2}, 1\text{e}^{-2}, 3)$.

Random scale in DFS (r_s). The results on different random scale in DFS are shown in Table III (a). Our method is insensitive to a wide range of random scale, showing that scaling the discriminative features is able to provide effective semantics of different strength.

Semantic shifting scale in MTS (r_b). r_b is to provide the flexibility of controlling the strength of adding intra-class semantic differences. The results on different semantic shifting scales in MTS are shown in Table III (b). Our method obtains similar results under different r_b and reaches the best results when $r_b = 1$, which suggests that larger r_b is able to cover more barren area in the embedding space to improve the model training.

Number of the produced embeddings T. The results on different numbers of the produced embeddings are shown in Table III (c). As T increases from 1 to 5, the proposed DAS achieves better results and reaches the best result at $T = 5$. When $T = 10$, the performance is worse than $T = 5$, which indicates that too many produced embeddings with no data points will dominate the optimization direction and impair the learning of embeddings with data points.

Table III: Experiments on different hyper-parameters on CARS

(a) Effect of the random scale r_s in DFS

r_s	1e^{-2}	1e^{-1}	2e^{-1}	5e^{-1}
R@1	82.29	82.25	82.30	**82.43**

(b) Effect of the scale r_b in MTS

r_b	1e^{-3}	1e^{-2}	1e^{-1}	1
R@1	82.19	82.29	82.07	**82.55**

(c) Effect of the number of produced embedding (T) in DAS

T	1	3	5	10
R@1	80.78	82.29	**83.40**	81.75

L Qualitative Results of DFS and MTS

In this section, we investigate the effectiveness of the proposed DFS and MTS. Specifically, we apply DFS and MTS in the test phase and compare results from the model trained with or without them. Since the training and test classes are different, the DFS and MTS modules used for training are unavailable here.

Thus, the semantic scaling is implemented as randomly scaling the top K features in an embedding; Whilst we perform semantic shifting by adding the transformation (obtained from another two embeddings of the same class) to the embedding. The loss function and sampling method we used here are contrastive loss [2] and distance-weighted sampling [11], respectively. The results for semantic scaling and shifting are shown in Fig. V and Fig. VI, respectively. We have the following observations: **1)** When we apply different semantic scaling to the query, the model trained with DAS consistently retrieves correct results, which is not the case for the baseline. **2)** The model trained with DAS is able to retrieve expected results even with the semantic shifted embedding while the baseline fails to do so. These results show that DAS is able to produce embeddings with effective semantics to train the model, which is insensitive to the semantic differences and consistently achieves good generalization ability after training.

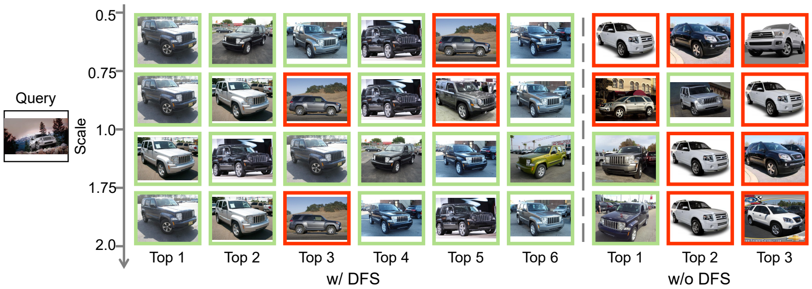


Fig. V: Top 6 retrieved results with different scales on CARS. The expected and unexpected results are framed by green and red rectangles, respectively

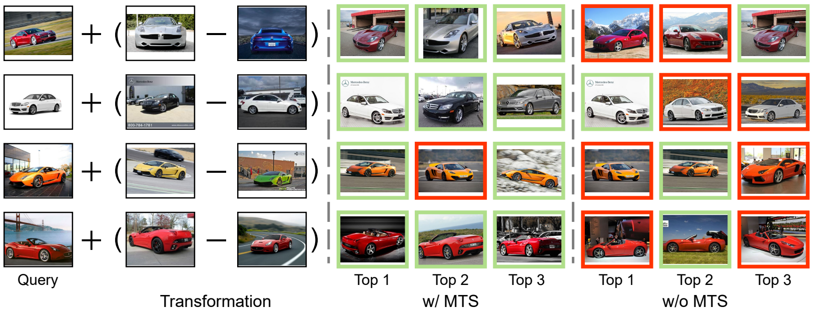
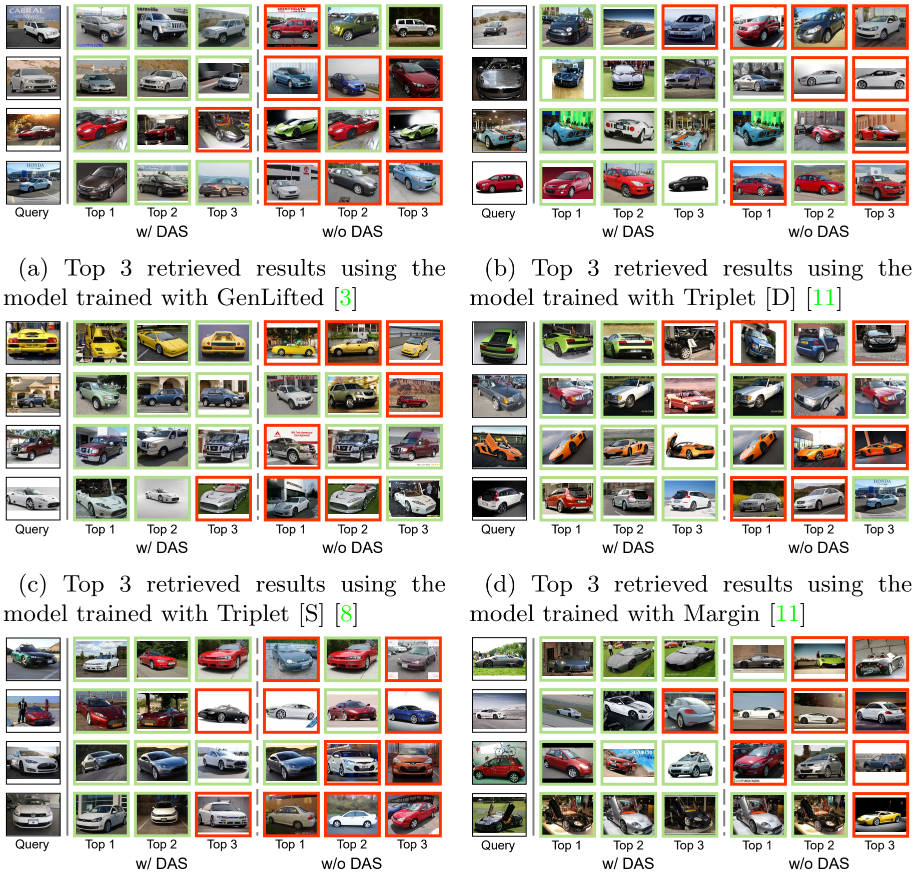


Fig. VI: Top 3 retrieved results with MTS on CARS. The expected and unexpected results are framed by green and red rectangles, respectively

M More Qualitative Results

In this section, we provide qualitative results on different losses w/ or w/o DAS. The results for CARS and SOP are in Fig. VII and Fig. VIII, respectively. From those results, we can see that the proposed DAS can enforce the model to focus on real semantics despite the background noises and other semantics' interference such as car's colors, drastic viewpoint changes *etc.* These results show the generalization ability and robustness of the proposed DAS.



(a) Top 3 retrieved results using the model trained with GenLifted [3]

(b) Top 3 retrieved results using the model trained with Triplet [D] [11]

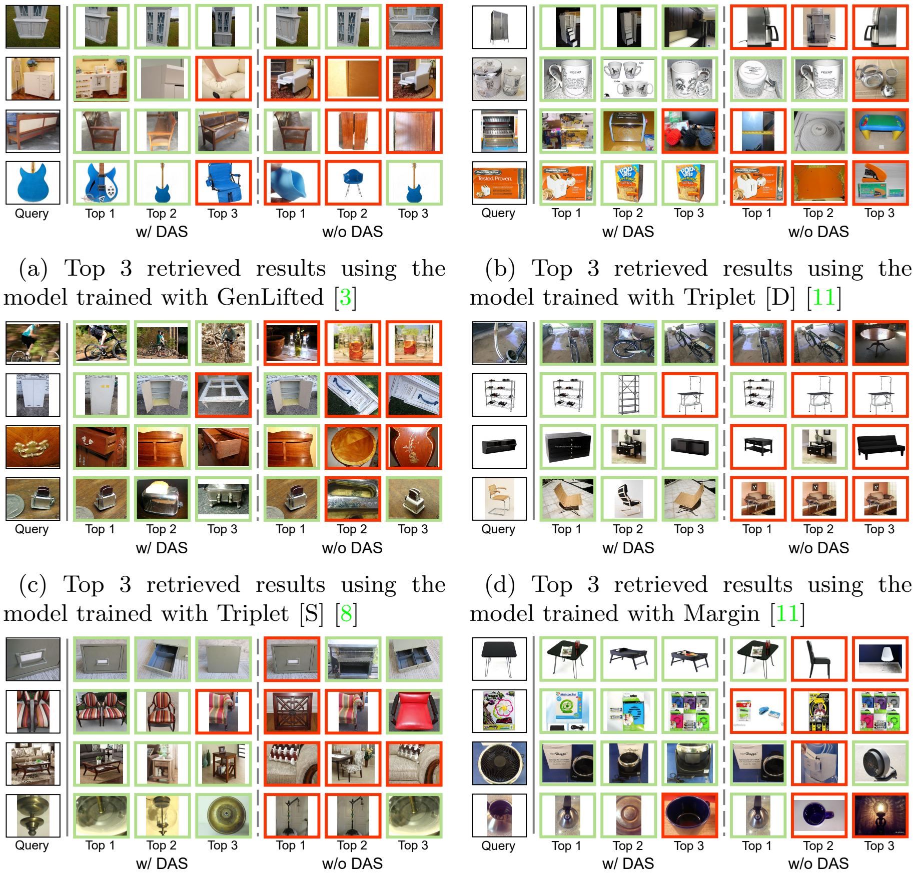
(c) Top 3 retrieved results using the model trained with Triplet [S] [8]

(d) Top 3 retrieved results using the model trained with Margin [11]

(e) Top 3 retrieved results using the model trained with N-Pair [9]

(f) Top 3 retrieved results using the model trained with MS [10]

Fig. VII: Top 3 retrieved results using the model trained by different loss functions that are equipped w/ or w/o on CARS. The expected and unexpected results are framed by green and red rectangles, respectively



(a) Top 3 retrieved results using the model trained with GenLifted [3]

(b) Top 3 retrieved results using the model trained with Triplet [D] [11]

(c) Top 3 retrieved results using the model trained with Triplet [S] [8]

(d) Top 3 retrieved results using the model trained with Margin [11]

(e) Top 3 retrieved results using the model trained with N-Pair [9]

(f) Top 3 retrieved results using the model trained with MS [10]

Fig. VIII: Top 3 retrieved results using the model trained by different loss functions that are equipped w/ or w/o on SOP. The expected and unexpected results are framed by green and red rectangles, respectively

References

1. Deng, J., Guo, J., Xue, N., Zafeiriou, S.: Arcface: Additive angular margin loss for deep face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4690–4699 (2019) 3, 6
2. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. vol. 2, pp. 1735–1742. IEEE (2006) 1, 11
3. Hermans, A., Beyer, L., Leibe, B.: In defense of the triplet loss for person reidentification. arXiv preprint arXiv:1703.07737 (2017) 2, 12, 13
4. Hu, J., Lu, J., Tan, Y.P.: Discriminative deep metric learning for face verification in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1875–1882 (2014) 4
5. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Proceedings of the International Conference on Learning Representations (2015) 5
6. Oh Song, H., Xiang, Y., Jegelka, S., Savarese, S.: Deep metric learning via lifted structured feature embedding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4004–4012 (2016) 2
7. Roth, K., Milbich, T., Sinha, S., Gupta, P., Ommer, B., Cohen, J.P.: Revisiting training strategies and generalization performance in deep metric learning. In: International Conference on Machine Learning. pp. 8242–8252. PMLR (2020) 4, 5
8. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 815–823 (2015) 1, 4, 7, 12, 13
9. Sohn, K.: Improved deep metric learning with multi-class n-pair loss objective. In: Proceedings of the 30th International Conference on Neural Information Processing Systems. pp. 1857–1865 (2016) 2, 12, 13
10. Wang, X., Han, X., Huang, W., Dong, D., Scott, M.R.: Multi-similarity loss with general pair weighting for deep metric learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5022–5030 (2019) 3, 12, 13
11. Wu, C.Y., Manmatha, R., Smola, A.J., Krahenbuhl, P.: Sampling matters in deep embedding learning. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2840–2848 (2017) 2, 4, 5, 6, 7, 10, 11, 12, 13
12. Zhai, A., Wu, H.Y.: Classification is a strong baseline for deep metric learning (2019) 3, 6